Online Learning of Tire Behaviour combined with MPC for Autonomous Racing

Kunal Iyer

TU Delft

Online Learning of Tire Behaviour combined with MPC for Autonomous Racing

by

Kunal Iyer

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended on Wednesday August 26, 2020 at 02:00 PM.

Student number:	4794397	
Project duration:	September 1 , 2019 – August 26, 2020	
Thesis committee:	Prof. Dr. ir. Martijn Wisse,	TU Delft, Chair, Cognitive Robotics
	Dr. Barys Shyrokau,	TU Delft, Supervisor, Cognitive Robotics
	Ir. Yanngu Zheng,	TU Delft, Daily Supervisor, Cognitive Robotics
	Dr. Wei Pan,	TU Delft, External Member
	Dr. Valentin Ivanov,	Technische Universität Ilmenau, Guest

TU Delft

Preface

सरस्वती नमस्तुभ्यं वरदे काममरूपिणी।
विद्यारम्भं करिष्यामि सिद्धिर्भवतु मे सदा॥ १॥

This masters thesis marks the end of an amazing two years at the Technical University of Delft. I have always been passionate about vehicle dynamics and I tried to focus my masters towards the application of control in this domain. My master studies at TU Delft introduced me to a whole new dimension, filled with opportunities for innovation specifically with regards to the application of control systems in the automotive scenario. Having previously worked in student racing teams, and with a growing affinity towards working with Model Predictive Control, I contacted a few professors showing my interest in this field. I received a prompt response from Dr. Barys Shyrokau with the idea of applying a learning algorithm with MPC for autonomous racing. This was the start of a tedious yet interesting research journey, which is soon coming to an end.

Firstly, I would like to thank Dr. Barys Shyrokau for giving me the opportunity to work on such an exciting research project, and it is a privilege for me to make a small contribution in this field, by implementing a unique algorithm for a new application. I would also like to thank Yanngu Zheng for providing his valuable insight.

I would like to thank my parents for their unconditional love, and constant support through all the ups and downs over the past two years. Lastly, I would also like to thank my friends in Delft, for giving me a lot of good memories in the last two years.

Kunal Iyer
Delft, August 2020

Contents

List of Figures

List of Tables

1 Introduction

The motivation behind this research project along with some background is given in section 1.1. Next, some of the previous research work is briefly discussed in section 1.2. In section 1.3.1, the research objective is formulated. Finally, the outline of the thesis is given in section 1.4.

1.1. Motivation

The recent trend of shifting towards autonomous mobility has given birth to autonomous racing. Racing involves the vehicle being at the limits of handling to extract maximum performance. During a race, the vehicle performs extreme maneuvers which push the tires to the non-linear regime. An autonomous race-car hence needs a system to control the vehicle at the limits of handling.

Current research in the domain of intelligent vehicles portray that in order to keep the vehicle stable in adverse driving or environmental conditions, these systems need to be equipped to monitor current vehicle states, learn changes in operating conditions, and to adapt their behaviour depending on the scenario.

In order to make use of highly accurate vehicle models, it is necessary to encapsulate the non-linear vehicle dynamic behaviour which is well exploited in a racing environment. The proposed control framework is that of model predictive control (MPC). MPC plays a crucial role in advanced control of vehicles due to its ability over other control techniques, to handle constraints, provide optimal control as well as capture non-linearities. The aggressive nature of racing has a lot of effects on the tires which causes dynamic changes in the tire properties. In contrast to traditional racing, where professional racing drivers adapt their behavior based on the current vehicle state, the conventional controllers in an autonomous race-car have a non-adaptive tire model and thus fail to adapt to the varying dynamics of the tires.

Since it is difficult to predict the tire behavior during the race, online learning of the tire properties is considered. A potential solution is locally weighted projection regression (LWPR), a non-linear function approximation technique, used to learn the tire dynamics incrementally. This learning defines a notion of experience for the controller as it captures the complete input-output relationship for all previous control actions. In order to utilize online learning, the application must involve repeatability in terms of the working range of the tire. This is prevalent in a racing environment where the vehicle operates on multiple laps of the same circuit.

In this project, a combined framework of LWPR and non-linear model predictive control (nMPC) is proposed. The LWPR algorithm updates the tire model constantly online with sensor data to encapsulate the complete vehicle dynamic behavior. A distinct feature of the suggested technique is that along with its ability to adapt to dynamically changing tire behavior, the controller can also handle abrupt disturbances such as gust. This paper focuses on the design of an nMPC controller with online learning of tire properties, intending to improve performance in autonomous racing.

1.2. Previous Research Work

Over the past years, a variety of control techniques have been researched upon for the autonomous racing application. The controller to be designed, requires a reference trajectory to be followed. Once a reference trajectory is known, a control scheme must be put in place to make sure that the vehicle follows the generated trajectory.

1.2.1. Trajectory tracking control

One of the most basic control methods involves a **proportional feedback controller** that helps control the path error, by the use of evasive steering [67]. The future path information (y_{road}), deviation in position at the current instant (y) and the lateral error based on heading angle ($y_{projected}$) are used to compute the path error. The lateral path error is defined as,

$$d = y + y_{projected} + y_{road} \tag{1.1}$$

A speed varying proportional feedback gain (K_p) is calculated at each iteration step. This gain has a physical meaning and is computed using "curb following" concept where, at every time instant, a circular path of radius R is defined using the rear axle, heading angle and future projection in the reference trajectory. From vehicle dynamic analysis, the gain is computed as a function of the understeer gradient (K_{us}), wheelbase (L) and longitudinal velocity (u). The control law is then given by equation (1.3).

$$K_p = \frac{L + K_{us} u^2}{R} \tag{1.2}$$

$$\delta = K_p d \tag{1.3}$$

The vehicle's trajectory depends hugely on the look-ahead time. The look-ahead time must be chosen carefully. A trade-off must be made between phase lag and overshoot in the trajectory, and the vehicle cutting corners. It was also observed that this controller cannot perform at high speeds which is crucial for competitive racing scenarios.

In order to deal with the phase delays that come with feedback control, a feedforward control law is introduced. This results in a **feedforward and feedback controller** [60]. The feedback control action is designed to be a PID to follow the reference yaw rate and heading angle.

$$\delta_{fb} = k_p e_\psi + k_i \int_{t_0}^{t_f} e_\psi + k_d \dot{e}_\psi \tag{1.4}$$

The gains k_p, k_i and k_d are chosen by gain scheduling at varying velocities. The feedforward control law based on steady state steering corresponding to bicycle model.

$$\delta_{ff} = \frac{L}{R} + K_{us} a_y \tag{1.5}$$

The use of **linear quadratic regulator (LQR)** to compute the optimal feedback control action has been presented in [62]. The distance of the centre of gravity from the reference (e_1) and the difference in yaw angle with the reference yaw angle (e_2) are the controlled states. The state-space model with the state vector $\epsilon = [e_1, \dot{e}_1, e_2, \dot{e}_2]$ and input vector δ make up the following cost function,

$$J = \frac{1}{2} \int_0^\infty [\epsilon^T Q \epsilon + \delta^T R \delta] \, dt \tag{1.6}$$

where Q and R are penalty matrices on the states and input respectively. In addition, with the use of a state observer to estimate the lateral velocity, this technique shows good performance using steering. The limitation of such a technique however, is that it cannot handle constraints.

It was observed that optimal distribution of lateral and longitudinal forces can be computed to match the required yaw moment and total lateral force [41]. Other control strategies that have been researched upon include sliding mode control (SMC) [6], friction based control techniques [3] and non-linear back-stepping with adaptive fuzzy sliding mode control [27]. It can thus be concluded that, *an integrated control scheme that uses optimal control methods designed in the non-linear range* provides best performance.

1.2.2. Path following control

An issue faced with trajectory tracking methods as explained in [26], is that the generation of feasible trajectories in real-time is difficult especially at the limits of handling. Moreover, oversimplified models yield inaccurate trajectories and highly complex models prove to be computationally demanding. This is the main motivation to look into path following control techniques.

As defined in [20], path following control is "*the tracking of a geometric reference with high precision, whereby the timing to move along the reference is of secondary interest, and can be considered as an additional degree of freedom*". Since the reference is parameterized, the control structure influences both the evolution of the reference as well as the vehicle dynamic parameters. Most control methods employing path following make use of model predictive control framework. Further elaboration of this control technique is done in chapter 1.

1.2.3. Learning control

As stated earlier, racing exploits the non-linear regime of the tire resulting in non-linear vehicle dynamics which is difficult to model accurately. As a result, it is difficult to achieve accurate control when the vehicle is subjected to such aggressive maneuvers. A promising approach of using iterative learning control (ILC) is elaborated in [34]. This technique is used to gradually determine appropriate steering inputs for a transient maneuver by using information from repeated instances to improve performance.

Race tires are crucial to performance and undergo significant changes during the course of a race. Conventional controllers however, use fixed tire models that are unable to adapt to changing tire properties during the race, and thus do not capture the complete vehicle dynamic behaviour. This ultimately contributes to sub-optimal performance. An inviting solution is to employ online learning of tire properties, which shall further be discussed in part II.

1.3. Research formulation

In order to complete this project, a research framework was developed, such that the effectiveness of this unique combination of LWPR and MPC provides improved performance in autonomous racing.

1.3.1. Research objective

The main research objective is formulated as follows:

Research Objective : *To design an MPC for autonomous racing which utilizes the online learning of tire properties to improve performance and minimize lap time*

Since model accuracy is an integral part of MPC design and tires are crucial to vehicle performance during a race, a challenge to address is the fast changing system dynamics in such an environment involving aggressive maneuvers. This served as a motivation to look into online learning methods to help the system adapt to the changing dynamics and to account for unmodeled non-linearities. Research work on learning methods portray LWPR as a suitable alternative to neural networks in this domain. The result of combining the MPC framework with online learning using LWPR is an **advanced autonomous system that is capable of learning the varying dynamics online as well as ensuring vehicle motion control even at the limits of handling**. Learning based control techniques are still an area of active research in the industry. A gap was observed in the implementation of such techniques in real-time.

Autonomous racing is an ideal application for such learning based control techniques, because the vehicle is subjected to repeated instances of the same circuit and hence can utilize learning to enhance lap-by-lap performance. Finally, the designed controller shall be compared in terms of performance with some benchmark controllers.

1.4. Thesis Outline

The report comprises of three parts. The scientific article pertaining to the application of LWPR with MPC for autonomous racing is presented in part I. Following it, part II provides the necessary background information and consists of four chapters. Chapter 2 describes the design of the MPC. A brief section is devoted to the theory behind MPC, along with the motivation to design a non-linear MPC (nMPC). Some challenges regarding MPC design are also highlighted. Finally this chapter reviews previous research on MPC implementations for the autonomous racing scenario. Next, chapter 3 of part II discusses some of the tire models used. Apart from the standard tire models, an incremental learning algorithm is introduced in an attempt to learn tire force behaviour online. Some theory is provided along with the applicability of this algorithm. This chapter also highlights some of the previous research work done on the use of this algorithm in real-time. More detail regarding the benchmark controllers used for comparison can be found in chapter 4 of this document. This comparison will try to highlight some possible benefits of using this control scheme and stress the importance of learning based MPC, especially for scenarios like autonomous racing. Finally, chapter 5 provides a conclusion on the master thesis along with some recommendations. The author's previous work on the application of LWPR for tire force reconstruction is also made available in part III.

I

Scientific Article

Article

Online Learning of Tire Properties Combined With Non-linear Model Predictive Control for Autonomous Racing

Abstract: In this paper, a unique method of combining online learning with model predictive control is applied to autonomous racing. A concern in autonomous racing is that accurate models that encapsulate the dynamics of the vehicle, are complex, nonlinear, and difficult to identify. In order to make this more practical for control purposes, the controller is initialized with a nominal tire model, which then learns tire properties online using locally weighted projection regression during the course of the race. This makes it more practical for control purposes while maintaining model accuracy. Focus is placed on learning the tire properties which in reality, keep varying due to wear, temperature and pressure fluctuations, etc. The main objective is to minimize lap times by allowing the controller to "learn" its varying tire behaviour while on the track.

Keywords: Autonomous racing, vehicle dynamics, online learning, locally weighted projection regression, model predictive control.)

---

1. Introduction

The evolution of autonomous driving has sparked the growth of autonomous racing [1]. With recent technological advancements working towards full autonomy, the birth of autonomous racing and its growth was inevitable. Though relatively new, autonomous racing technology is rapidly increasing. Roborace is the company which started the world's first motorsports series for driver-less cars [2]. The aim is to pit autonomous cars against each other on track.

A lot of innovations in the automotive industry has come from racing. For example, research work in Formula One on "racing-line" driver models [3], have found applications in path-following scenarios and have helped develop ADAS functions such as Lane Keeping Assist (LKA). Similarly, the big picture regarding autonomous racing, is that the innovations and research work that is done on the track will eventually find its way on road cars. The development of autonomous vehicles that are robust enough to race will push the boundaries of research and technology and eventually lead us to the age of driver-less cars. Autonomous racing thus, plays an vital role in the future of mobility, by providing a platform to test and develop new technologies. Organizations like Roborace are aimed at developing advanced platforms to test autonomous vehicle technology in highly competitive and demanding environments [4].

Professional drivers are very talented in knowing when to hit the throttle, when to brake and how to apply fast steering corrections, all with a limited preview horizon. Moreover, they are able to adapt their driving behaviour based on tire wear, tire temperatures, road irregularities and other environmental conditions. These professional drivers are thus, **analogous to extremely robust controllers** which can operate the vehicle safely at its handling limits and adapt to changing characteristics. Furthermore, this adaptation is carried out without having a fixed vehicle or tire model, but based on drivers sensory feedback and racing experience. The objective is that future control systems with modern sensors in autonomous vehicles must incorporate such adaptation and robustness characteristics.

Since this paper looks mainly into the dynamics of an autonomous race-car, focus shall be placed on the control system design. The main idea for designing a path following controller for an

autonomous race-car, is to compute a reference path and to follow it as accurately as possible. The generation of the reference path is such that it results in minimal lap-time. The controller on the other hand, enables path tracking by estimating the speed of the vehicle, the steering input, and other vehicle dynamic parameters. It is assumed that the reference is always available to the controller and attention is placed on the controller design itself. The following section briefly introduces a unique solution to the autonomous racing scenario.

1.1. Proposed solution

With developments being made to increase computational power, machine learning techniques and methods based on optimization have been studied [5] to tackle these problems. Model Predictive Control (MPC) is an advanced control method that is useful for path following and autonomous racing [6], [7], [8]. An iterative learning control (ILC) method has been proposed in [9] to reduce tracking error by improving lap-by-lap performance. The authors proved the effectiveness of the ILC experimentally on a vehicle. The authors in [10] have used a learning-based MPC to solve the lap time minimization problem.

MPC uses a *prediction model* and an optimization algorithm over a certain *horizon* to obtain the *optimal control sequence* that satisfy a set of *constraints*. The cost function can be shaped intuitively by analyzing the trade-offs to be made based on the control objectives. It is desired for the prediction model to be able to capture the dynamics of the system accurately while being simple enough to facilitate optimization online. Autonomous racing exploits this trade-off as it exposes the vehicle to its limits of handling. While a simple prediction model facilitates seamless optimization, it can result in reduced performance. Similarly, an accurate complex model results in a large computational burden. Additionally, during a race, the dynamics is constantly varying. The changes in dynamics can be attributed to variations in tire properties due to tire wear, tire temperature and pressure fluctuations or changes in the race environment. The model thus needs to adapt to this online during the race.

The concept of using a regressor to identify the varying dynamics of the prediction model within an MPC is a technique discussed in the literature and implemented in other domains [11], [12]. In [13], the authors present a solution to the aforementioned problems by using a nominal prediction model which is augmented online using Gaussian process regression (GPR). The authors have also presented experimental results by implementing their technique on a full-scale autonomous race car and have proved a reduction in lap times.

This paper presents the application of locally weighted projection regression (LWPR) in tandem with MPC for autonomous racing. LWPR is a non-linear function approximation algorithm that is used to learn the varying tire properties online, thus allowing the MPC to adapt to the fluctuating dynamics.

2. Locally Weighted Projection Regression

LWPR is a unique algorithm that supports non-linear function approximation in high dimensional spaces [14]. The non-linear system behaviour, can be accurately captured by using this technique.

The key use of this technique is to use piece-wise linear models to approximate non-linear functions. The characteristics of LWPR include being numerically robust, especially in high dimensional spaces and its capability to perform incremental online learning with the predefined learning rate.

2.1. Algorithm

The fundamental parts of the algorithm are discussed under this section. The entire algorithm is presented under chapter 3 in part II of the thesis report.

2.1.1. Activation

A weight $w_{k,i}$, also called the weighting kernel or "activation", is defined for every data point (x_i, y_i) corresponding to its distance to the kernel centre c_k within every local unit. This is used primarily to determine the locality. Usually, Gaussian kernels are chosen,

$$w_{k,i} = exp(-\frac{1}{2}(x_i - c_k)^T D_k (x_i - c_k)), \tag{1}$$

The shape of the receptive fields (RF) or the region of validity, is influenced by the distance metric D_k in (1). Assuming the prediction comprises of K locally linear models, a prediction y_k is computed from each linear model, given an input vector x. The net output is the weighted mean of all the linear models.

2.1.2. Partial Least Squares

Algorithm 2 shows how an incrementally locally weighted variant of partial least squares (PLS) is used to generate linear model parameters within the LWPR scheme. The PLS predictor adds linear projections in an incremental fashion until the point where adding further projections does not improve the accuracy.

2.1.3. Distance Metric

The distance metric D influences the shape and size of each RF and thus also influences the effectiveness of each local model. This distance metric is optimized separately for each RF using an incremental gradient descent based on stochastic leave-one-out cross validation criterion. This is shown in the algorithm 3.

2.1.4. Forgetting factor

In the algorithm 2, $\lambda \in [0, 1]$ is the forgetting factor. This decides how much of the old data of the parameters used in the regression will be forgotten. This parameter is useful especially in the case of online learning.

An incremental learning system which embeds the above update laws, and generates additional locally linear models as and when needed is shown in algorithm 4.

2.2. Current state-of-the-art

The application of this algorithm for real-time robot learning has been presented in [15]. The results shown in the aforementioned paper demonstrates the successful application of autonomous learning to complex robotic systems. It was also concluded that this technique, using its learning abilities outperforms traditional control techniques.

Previous literature shows that the LWPR algorithm has been used for dynamic model learning of a robotic manipulator [16]. The use of LWPR in the aforementioned application was aimed at assisting model-based control techniques by introducing adaptive learning of the dynamic model of the robot.

Offline and online learning performance of this algorithm, specifically pertaining to tire behaviour, is demonstrated in [17]. These results show the feasibility of this algorithm to reconstruct non-linearities in tire-force behaviour.

Keeping the above-mentioned advantages in mind, this method is an alluring candidate for learning dynamic tire behaviour. In order to utilize online learning, the application must involve repeatability in terms of the working range of the tire. This is prevalent in a racing environment where the vehicle operates on multiple laps of the same circuit.

2.3. Implementation

This section focuses on an online implementation of LWPR for autonomous racing on the Hockenheim circuit. It is to be noted that no control has been applied for these simulations.

The LWPR online learning module is deliberately initialized with a highly inaccurate tire model. The model is then updated online with the sensing force information based on the Delft-tire model for the simulations. The learnt tire force information is then compared with the bad model without adaptability to analyze the learning performance. To assess real-time capability, simulation was conducted using dSPACE real-time (DS1006) machine with IPG/CarMaker HIL. The simulation parameters used can be seen in Table 1.

Table 1. Parameters for online learning of lateral force

Parameter	Value	Unit
Initial distance metric	10e4 $\mathbf{I}_2$	[-]
Initial component-wise learning rate	0.05 $\mathbf{I}_2$	[-]
Pre-factor of smoothness penalty	0.5	[-]
Initial forgetting factor	0.55	[-]
Final forgetting factor	0.85	[-]
Annealing constant for forgetting factor	0.8	[-]

Figure 1 shows the lateral force learning from the poorly initialized model to the Delft tire model. The algorithm shows good learning performance and adaptability.

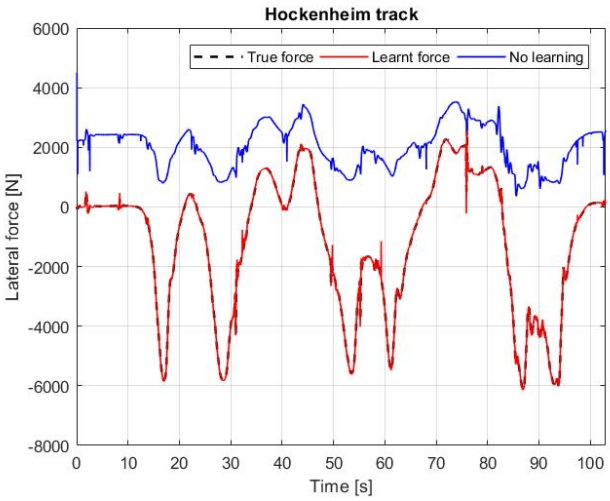

Figure 1. Online learning performance

3. Model Predictive Control

3.1. Motivation

A driver has the inherent ability of observing his surroundings and to predict the future state of the vehicle according to the current vehicle state and road information. The driver is then able to plan a feasible trajectory merely by comparing the future state with the desired one. Based on this, the driver supplies an input to the system in real time. When the vehicle achieves a new state, the driver must be able to comprehend this change and react to it. In this light, the driver behaviour goes hand-in-hand with the basic idea of MPC [18].

MPC's implicit predictive nature is useful in anticipating external disturbances or sudden changes in the system which can be incorporated into the prediction model [19]. This is highly advantageous and beneficial especially when it comes to aggressive manoeuvres, which are encountered during

racing. Any abrupt changes in surroundings causes a swift change in the dynamics of the system. Prediction of such changes in states results in more efficient computation of control actions. The ability of the controller to look ahead and realize any potential dynamical changes in the system helps it to take remedial action sooner.

The domain of vehicle motion control especially at the limits of handling comprises of a multi-variable, non-linear system subject to physical as well as operational constraints on its inputs and states. In contrast to the classical approaches, model predictive control seeks to bring a balance between the predictive nature of infinite-horizon control and reactive nature of traditional control techniques [20]. As seen in [21], the concept of non-linear model predictive control (nMPC) along with its underlying principle of optimal control is an inviting alternative. This is due to its ability to handle complex processes with a lot of inputs and states that simultaneously fulfill the respective constraints imposed on the system.

3.2. Theory

A detailed theoretical background is provided under chapter 2 of part II in the thesis report. As mentioned in [22], the important aspects of an MPC are the prediction model, the constraints imposed on the system and the designed cost function.

3.3. Prediction Model

For the designed MPC to be efficient, the prediction model in the MPC formulation must be as accurate as possible. The model equations must represent the vehicle's dynamic behaviour accurately. It is unwise to rely on the control actions from the MPC if the model is not well-defined.

This is the most crucial part of MPC design. The prediction model encapsulates the dynamics of the vehicle and thereby helps the MPC realise how the system behaves for a certain control action. A general representation of the prediction model is shown in equation (2).

$$x(k+1) = f(x(k), u(k)) \tag{2}$$

In the above equation, $x(k), u(k)$ represents the states of the system and control input to the system at the current instance respectively. Refer table 2 for the different variables and their significance in the prediction model.

Table 2. Parameters in the prediction model

Parameter	Symbol
Longitudinal velocity	v_x
Lateral velocity	v_y
Yaw angle	ψ
Yaw rate	r
Longitudinal position of centre of gravity	X_p
Lateral position of centre of gravity	Y_p
Steering angle	δ
Steering rate	$d\delta$
Front axle cornering stiffness	C_f
Rear axle cornering stiffness	C_r
Vehicle mass	m
Overall steering ratio	i_s
Body inertia around z-axis	I_{zz}
Distance from front axle to centre of gravity	l_f
Distance from rear axle to centre of gravity	l_r
Initial longitudinal velocity	V_0

The prediction model comprises of the states : $x = [v_x, v_y, r, X_p, Y_p, \psi, \delta]$ and the control input : $u = d\delta$. Equations (3),(4),(5),(6),(7),(8) and (9) represent the non-linear bicycle model. The controller is designed to track a lateral position and heading angle reference ($x_{ref} = [0, 0, 0, 0, y_{ref}, \psi_{ref}, 0]$).

$$\dot{v_x} = v_y\dot{\psi} \tag{3}$$

$$\dot{v_y} = -\frac{C_f + C_r}{mV_0}v_y + (\frac{l_r * C_r - l_f * C_f}{mV_0} - V_0) * r + \frac{C_f}{m}\delta \tag{4}$$

$$\dot{r} = \frac{l_r C_r - l_f C_f}{I_{zz} V_0}v_y - \frac{(l_r)^2 C_r + (l_f)^2 C_f}{I_{zz} V_0}r + \frac{l_f C_f}{I_{zz}}\delta \tag{5}$$

$$\dot{X}_p = v_x cos(\psi) - v_y sin(\psi) \tag{6}$$

$$\dot{Y}_p = v_x sin(\psi) + v_y cos(\psi) \tag{7}$$

$$\dot{\psi} = r \tag{8}$$

$$\dot{\delta} = d\delta \tag{9}$$

It must be noted that the cornering stiffness (C_f, C_r) values are obtained in a dynamic fashion as elaborated under section 4.2. This dynamic cornering stiffness along with the system dynamics contributes to the non-linearity in the prediction model. The non-linear prediction model is thus developed using the plant dynamics, and is further used for optimal control action calculation by solving an optimization problem.

3.4. Optimization problem

An optimization problem is set up which minimizes a cost function (V_N) subject to the system dynamics and constraints ($G(x, u, \Delta u)$), in order to obtain the optimal control sequence (u_{N_p}). The general structure of the optimal control problem is as shown in equation (10) subject to the system dynamics in equation (2) and constraints within a specified bound (ϵ) in equation (11).

$$min_{u_{\mathbf{N_p}}} V_N(x_0, u_{Np}) \tag{10}$$

$$G(x, u, \Delta u) \leq \epsilon \tag{11}$$

3.4.1. Cost function

The cost function is split into a stage cost ($l(x(k), u(k))$ and terminal cost (V_f) as per equation (12)

$$V_{Np}(x_0, u_{Np}) = \sum_{k=0}^{N_p-1} \underbrace{l(x(k), u(k))}_{\text{stage cost}} + \underbrace{V_f(x(N_p))}_{\text{terminal cost}} \tag{12}$$

The stage cost comprise of state penalty weight Q_{stage} and a weight for the control input R_{stage}. This is of the form shown in equation (13).

$$l(x, u) = (x - x_{ref})^T Q_{stage}(x - x_{ref}) + u^T R_{stage} u \tag{13}$$

The terminal cost is designed with a state penalty weight Q_{term} as shown in equation (14).

$$V_f(x(N_p)) = x(N_p)^T Q_{term} x(N_p) \tag{14}$$

The tuning weight Q is a diagonal matrix with the values on the main diagonal corresponding to the penalty on the respective states and the weight R penalises the control input. This results in the state

penalty matrix with individual penalty weights ($[W_{v_x}, W_{v_y}, W_r, W_{X_p}, W_{Y_p}, W_\psi, W_\delta]$) as per equation (15).

$$Q_{stage} = \begin{pmatrix} W_{v_x} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & W_\delta \end{pmatrix} \tag{15}$$

Similarly the control input penalty is assigned as per equation (16).

$$R_{stage} = W_{d\delta} \tag{16}$$

The terminal state penalty matrix Q_{term} is constructed in a similar manner to Q_{stage} as in equation (15) with weights ($[S_{v_x}, S_{v_y}, S_r, S_{X_p}, S_{Y_p}, S_\psi, S_\delta]$).

3.4.2. Constraints

As mentioned under section 3.1, one of the benefits of using MPC is its ability to handle constraints imposed on the system. In the autonomous racing scenario, the race-car is a mechanical system with various components working together. Its performance is thus directly impacted by working limits of the vehicle. Actuator limits which impose bounds on vehicle performance, can be expressed as constraints within the MPC.

$$-\frac{50\pi}{180} \leq r \leq \frac{50\pi}{180} \tag{17}$$

$$-\frac{330\pi}{180 i_s} \leq \delta \leq \frac{330\pi}{180 i_s} \tag{18}$$

$$-\frac{750\pi}{180 i_s} \leq \dot{\delta} \leq \frac{750\pi}{180 i_s} \tag{19}$$

Equation (17) represents the constraint on the yaw rate (rad/s). Equations (18) and (19) show the limits applied to the steering wheel angle (SWA) and steering rate. Here, the use of i_s is to express the constraint on wheel angle as a bound on the state. The SWA can turn a maximum of 330 degrees. This is converted into radians as shown in equation (18). The limits for steering angle and steering rate were chosen based on moose tests performed by Daimler [23]. The steering rate was limited to 750deg $/s$ and resulted in satisfactory performance. Similarly, the usage of i_s in order to obtain the bounds in wheel velocities is shown in equation (19).

4. Simulation setup

The simulation setup is elaborated under this section. Details regarding the setup of IPG CarMaker, MPC, LWPR and the benchmark controllers are provided in this section. Furthermore, some key performance indicators are defined in order to compare and evaluate the performance of the different controllers.

4.1. IPG CarMaker setup

This section elaborates the different settings chosen on IPG CarMaker for the simulations.

4.1.1. Vehicle and Tire

To ensure that the simulations are as close to the real-world implementation of control systems in an actual car, it is crucial that the vehicle chosen as the plant encapsulates all the important dynamics in all directions. This is achieved by choosing the Audi A6 multi-body vehicle modelled in the IPG CarMaker software with the Delft tire model. The vehicle parameters used are reported in table A1.

Tire property variation has been implemented by scaling the cornering stiffness coefficient. This tire degradation is simulated as a linearly deteriorating factor of the cornering stiffness coefficient over

time. Conventional controllers with fixed tire models will not be able to adapt accurately with this change in tire behaviour.

4.1.2. Sensor setup

The vehicle is equipped with two types of sensors. Slip angle sensors are placed at the middle of the front and rear axle in order to measure the front and rear slip angle. The second sensor is a road preview sensor with a preview distance of 1 meter. This sensor measures the lateral position and heading angle deviation from the path center line. This reference information is then passed on to the controller.

4.1.3. Race track

The Hockenheim race track is selected for all the simulations. The plot of the center-line of this circuit is shown in figure 2. All simulations are performed for 3 laps of this circuit. The width of the track is 12 meters.

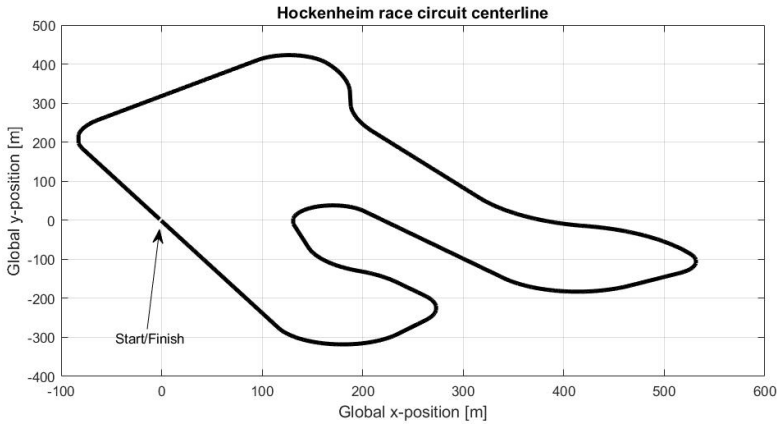

Figure 2. Hockenheim race track

4.1.4. Driving behaviour

An aggressive race driving behaviour was chosen on IPG CarMaker. The acceleration limits ($a_{y_{max}}, a_{x_{max}}, a_{x_{min}}$) of this setting is shown in table 3.

Table 3. Acceleration limits

$a_{y_{max}}$	$a_{x_{max}}$	$a_{x_{min}}$	**Units**
4	4	-6	m/s^2

4.2. MPC setup

As discussed under section 3, MPC computes an optimal control sequence from the predicted output. In this case, the MPC computes an optimal steering wheel rate which can in turn be used to obtain an optimal steering wheel angle. Therefore, the MPC provides lateral control to the vehicle. A suitable speed profile as shown in figure 3 is chosen and is constant throughout the simulation. This controls the vehicle in the longitudinal direction. These inputs are then fed to the IPG vehicle.

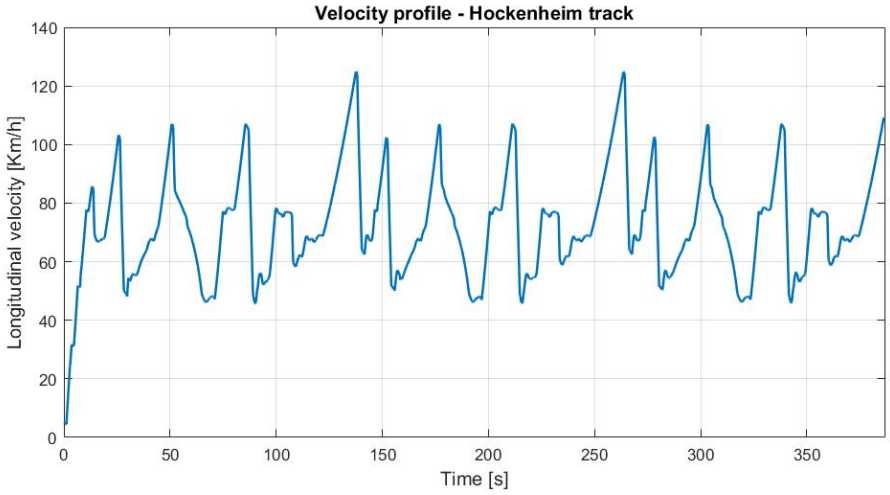

Figure 3. Velocity profile

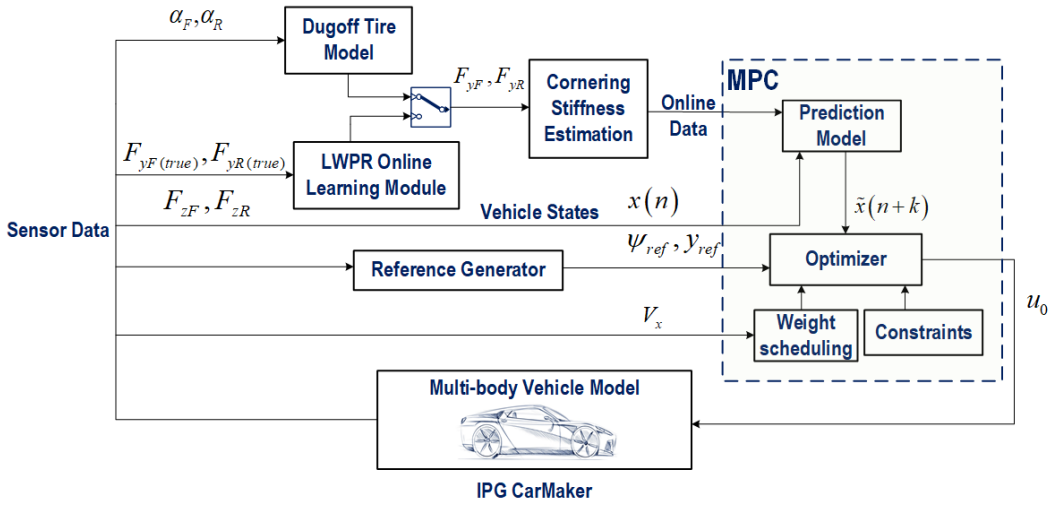

Figure 4. MPC overview

An overview of the work flow of the control setup can be seen in figure 4. The sensor data from the IPG vehicle is utilized as shown in the figure. The longitudinal velocity signal (V_x) is used to determine the penalty weight matrices for the cost function via a weight scheduling method. This, along with the designed constraints are fed to the optimizer. The current vehicle states ($x(n)$) are made available to the prediction model within the MPC. Online data variables consisting of the dynamic cornering stiffness of the front and rear axle are obtained either through online learning or a static Dugoff tire model. A set of predicted future states ($\tilde{x}(n+k)$) is then obtained and is fed to the optimizer. The reference values (ψ_{ref}, y_{ref}) along with the state prediction is utilized by the optimizer in order to calculate the optimal control sequence. As per the receding-horizon principle, only the first input from this sequence (u_0) is applied to the plant.

The wheel slip angle and force signals from IPG CarMaker are filtered before being accessed. A low-pass filter was designed by choosing the cutoff frequency after analyzing the power spectral density (PSD) of the individual signals. While choosing the cutoff frequency, it must be kept in mind to retain as much information as possible while eliminating noise. The PSD of the IPG CarMaker lateral force signals of front and rear axle can be seen in figure 5.

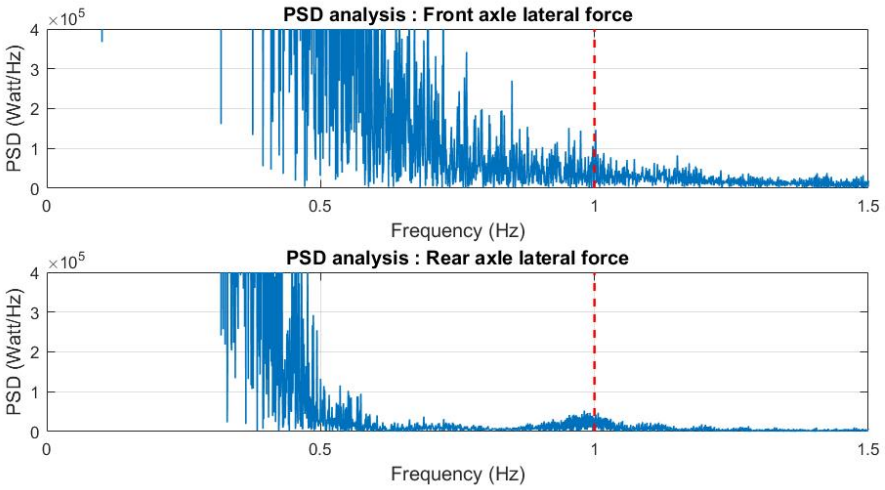

Figure 5. Power spectral density - Lateral Force

4.2.1. ACADO

In this project, ACADO toolkit is used to solve the optimal control problem (OCP). Other than its capability of working in the MATLAB and SIMULINK domain, ACADO can handle non-linear OCPs effectively.

4.2.2. Online data

Apart from the reference values (heading angle deviation and lateral position offset), the controller requires the updated information of all the states in the prediction model. This is vital to know the vehicle parameters at the current time-step. These values are measured using vehicle sensors and estimators.

The controller also requires additional data in order to compute the optimal control action. The dynamic cornering stiffness of the front and rear axle are computed online. These values are then used in the prediction model. In ACADO syntax, this is referred to as "Online Data".

The estimation of dynamic cornering stiffness was done by setting a threshold for wheel slip angle (α) defining the linear working range of the tire (± 0.03 rad $\approx \pm 2$ deg). In this range the static value of cornering stiffness is used with the assumption of linear tire model (20).

$$F_y = C_\alpha \alpha \tag{20}$$

Outside of this range, the dynamic cornering stiffness at instant k is computed as a slope between the lateral force values and wheel slip angle values as per (21).

$$C_{\alpha_k} = \frac{|F_{y_k} - F_{y_{k-1}}|}{|\alpha_k - \alpha_{k-1}|} \tag{21}$$

4.2.3. Tuning Parameters

Lastly, the controller requires its parameters to be tuned for best performance. An important parameter to be set while designing an MPC is how far ahead in the future can the controller predict. The performance of the controller is directly dependant on its future predictions. Ideally, in theory, MPC gives best performance when it can predict more. However, this comes at the cost of computational effort. The controller designed must have a low enough computational time to facilitate real-time computation of the optimal control action and to minimize the effect of delays due to the computational burden.

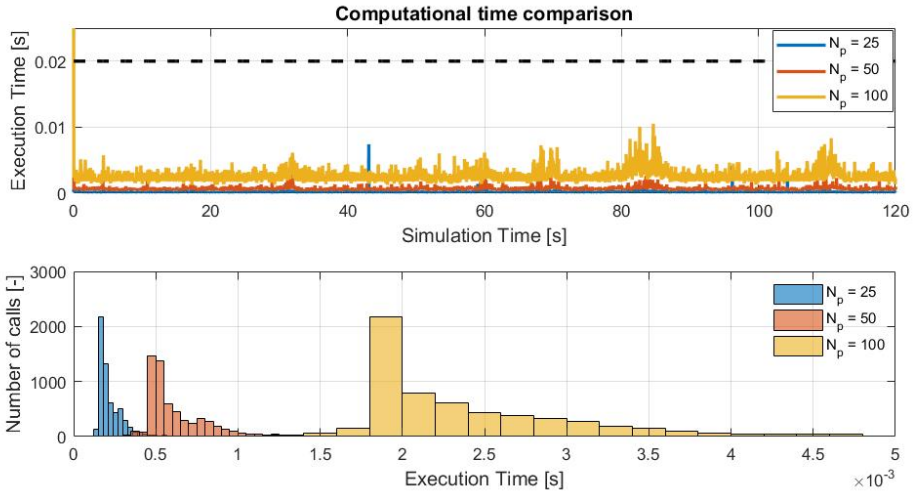

Figure 6. Computational time for varying prediction horizon with $T_s = 0.02s$

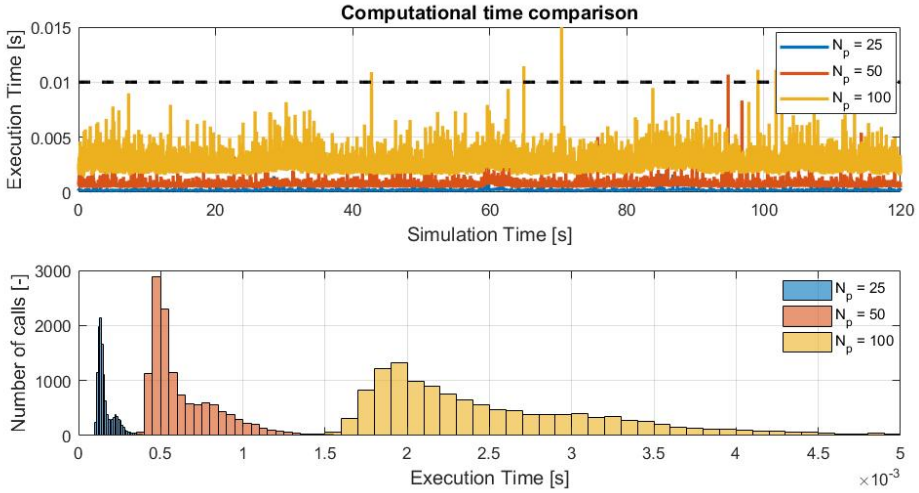

Figure 7. Computational time for varying prediction horizon with $T_s = 0.01s$

Figures 6 and 7 show a clear comparison for varying prediction horizon for the MPC running at 50Hz and 100Hz respectively. It is clearly evident that large prediction horizon shows peaks which reach execution times larger than the respective MPC sampling times. This is a potential stumbling block for real-time implementation. Furthermore, $N_p = 25$ shows that the execution times are well below the sample time. This shows the real-time applicability of the designed controller. This comparison validates the choice of the sampling time for the MPC as $T_s = 0.01s$ with a prediction horizon of $N_p = 25$. These values also ensure that the LWPR module receives data at 100 Hz. Previous studies [14],[15],[24],[25] show that an update rate of 20-200Hz is required for LWPR for real-time online learning. A controller frequency of 100 Hz is also feasible in real-time. In order to refrain from compromising on MPC performance and its predictions, the control horizon was set to be the same as the prediction horizon ($N_c = N_p$).

Apart from these parameters, the MPC also requires the stage and terminal cost weight matrices to be defined. These weights depend on the instantaneous longitudinal velocity of the vehicle. Hence, a weight scheduling method has been applied for varying velocity set points ($[30, 50, 70, 100, 120, 150] Km/h$). These weights are then linearly interpolated for intermediate velocities.

The idea behind choosing the initial weights for each velocity set-point was adapted from [22]. Firstly, the states to be controlled were decided. Only these states are penalized in the cost function.

Hence, the states to be omitted from the cost function, is done by setting their penalty to zero. The states to be controlled were chosen to be $[Y_p, \psi, \dot{\delta}]$. Since only lateral control is implemented in the MPC, it does not make sense to minimize the error for states in the longitudinal direction (v_x, X_p). Apart from the aforementioned states to be controlled, the wheel angle rate ($d\delta$) was also chosen. As elaborated in [22], ideally the SWA needs to have minimum variation to achieve the desired performance. This is done by penalizing the SWA by penalizing the control action $d\delta$. Hence, the stage cost weights are constructed as shown in equations (22) and (23).

$$Q_{stage} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & W_{Y_p} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & W_{\psi} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{22}$$

$$R_{stage} = W_{d\delta} \tag{23}$$

Similarly, the terminal cost weight matrices are constructed using terminal weights ($[S_{Y_p}, S_{\psi}]$) as shown in equation (24).

$$Q_{term} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & S_{Y_p} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & S_{\psi} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{24}$$

Each of the tuning weights (stage and terminal) W_{ϵ} are initially computed based on the equation (25).

$$W_{\epsilon} = \frac{1}{\epsilon_{error}} \tag{25}$$

Here, the smaller the desired error (ϵ_{error}), the larger the tuning weight (W_{ϵ}). Additionally, a state with a high penalty weight means that more emphasis is placed on controlling that respective state relative to the other states. It is thus a general practice to set a high value for the tuning weight to minimize the error leading to best achievable performance.

Once the ballpark values have been established, the controller is fine tuned via simulations where the tuning weights are varied and the corresponding performance is analyzed. By doing so, it is possible to understand the effect of varying the tuning parameters on vehicle performance. This process was repeated for all the velocity set-points and then linear interpolation was applied.

During the fine-tuning of the controller for different velocities and different scenarios, few key trends emerged. It must be noted that the chosen weights for each state to be controlled, correspond to the relative importance of the respective state in the cost function.

- An increase in $W_{d\delta}$, led to less chatter in the control input. If this value was set to be too large, it led to understeer.
- An increase in S_{Y_p} led to increased corner cutting.
- At high speeds (>120 Km/h), the weights W_{Y_p}, S_{Y_p} are increased to ensure proper tracking on straights.
- In general, smaller stage weights led to smoother convergence of the state to its desired value and small terminal weights led to delayed convergence. However, on the other hand, stage weights

that were too small led to delayed convergence and terminal weights which were too large failed to converge.

It must be noted that the states being controlled are vastly different in their order of magnitude. Hence, these weights have been normalized before being applied to the cost function. It is also important to realize that each of the weights are designed in the relative scale. This means that in order to increase the importance of a particular state in the cost function, increasing the penalty weight of the desired state or decreasing the penalty weight of the other states have the same effect. The tuned weights for the racing scenario have been reported in table A3 in appendix B.

4.3. LWPR setup

In order to demonstrate the working of MPC in combination with LWPR, the simulation setup shown in figure 4 was used. True force data from the IPG vehicle sensors are fed to the LWPR learning module. This includes the true lateral force and normal force per axle. Using this information along with data from the slip angle sensors, the LWPR module "learns" the transient tire behaviour and adapts the online data variables (cornering stiffness) within the prediction model of the MPC. This ensures that the MPC is always able to adapt to the varying tire behaviour online during the race. An upper limit is applied on the memory allocated for the algorithm. This is done by limiting the maximum number of receptive fields to 500. The learning module is initialized with the Delft tire model.

The training data for the initialization of the learning module was chosen as expressed in table 4. These values were chosen after carefully analyzing the range of values of wheel slip angle and normal load variation for the vehicle with a race driving behaviour (without control). The general parameters for the learning module for the simulation can be found under table 5.

Table 4. Training data for initialization of learning module

Input	Range	Unit
Wheel slip angle (front and rear)	± 5	[deg]
Normal load	$5000 - 16000$	[N]

Table 5. LWPR parameters for the autonomous racing scenario

Parameter	Value	Unit
Initial distance metric	4e4 $\mathbf{I}_2$	[-]
Initial component-wise learning rate	30 $\mathbf{I}_2$	[-]
Pre-factor of smoothness penalty	0.01	[-]
Initial forgetting factor	0.55	[-]
Final forgetting factor	0.80	[-]
Annealing constant for forgetting factor	0.90	[-]

4.4. Other controllers

Apart from analysing the performance of the MPC with a fixed Dugoff tire model, with the vehicle sensor data and with online learning, some other benchmark controllers, namely, the stanley controller [26] and path control with preview [27], were also used for comparison in this autonomous racing scenario. More details regarding these controllers can be found under chapter 4 of part II in the thesis report.

4.5. Performance indicators

Some key performance indicators have been defined to help compare the performance of the designed controllers.

4.5.1. Laptime

Since this is a racing scenario, an indicator of performance is the time it takes for the vehicle to complete a lap of the circuit. This value is directly obtained from the IPG software.

4.5.2. Control effort

In order to compare the control effort of the different controllers, the root-mean-squared (RMS) tool is used. The RMS of the control action gives an idea about how much control effort is required. For a data set x with n data points, the RMS is calculated as per equation (26).

$$RMS = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}(x_i)^2} \tag{26}$$

For the autonomous racing scenario, the RMS of steering wheel angle (δ_{swa}) is given by equation (27).

$$\delta_{swa_{rms}} = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}(\delta_{swa_i})^2} \tag{27}$$

4.5.3. Tire utilization

The lateral and longitudinal acceleration limits give an idea on how much of the tire is utilized during the race. This can be pictorially represented using a g-g diagram. This is an important metric to compare how much of the tire is used with each controller. The mean absolute lateral acceleration can be computed and used to compare with other controllers as shown in equation (28).

$$a_{y_{mean}} = \frac{1}{n}\Sigma_{i=1}^{n}|a_{y_i}| \tag{28}$$

A controller that utilizes the tire to its maximum potential in the lateral direction will have the largest mean absolute value of lateral acceleration.

5. Results

All simulations were carried out for peak acceleration values as per section 4.1.4. These values were chosen based on the fact that the benchmark controllers could not handle a more aggressive driving behaviour along with tire degradation. Thus, an appropriate driving behaviour was chosen so as to perform comparative analysis of the designed MPC-based controllers with the benchmark controllers.

As mentioned in section 4.1.1, tire degradation was implemented by scaling the cornering stiffness factor of each tire continuously such that there is a 5 percent degradation per lap for 3 laps. This can be seen from the tire force behaviour in figure 8.

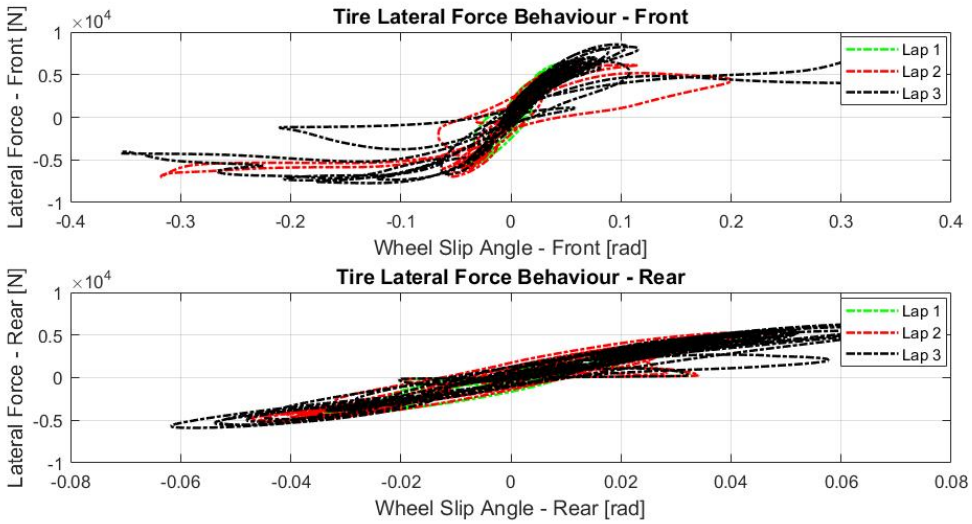

Figure 8. Tire degradation trend

5.1. Performance indicators

As per the performance indicators defined under section 4.5, the results are as shown in table 6.

Table 6. KPI comparison - 3 Laps with tire degradation

	MPC				
KPI	**Dugoff**	**Sensor data**	**LWPR**	**PCwP**	**Stanley**
Total Lap time [s]	372.210	372.370	372.160	374.420	373.910
Final Lap time [s]	121.821	121.510	121.300	122.386	122.189
$\delta_{swa_{rms}}$ [rad]	0.660	0.661	0.658	0.663	0.672
$a_{y_{mean}}$ [g]	0.228	0.228	0.230	0.222	0.224

5.2. Other results

Some other results have been reported in figures 9,10,11,12 and 13.

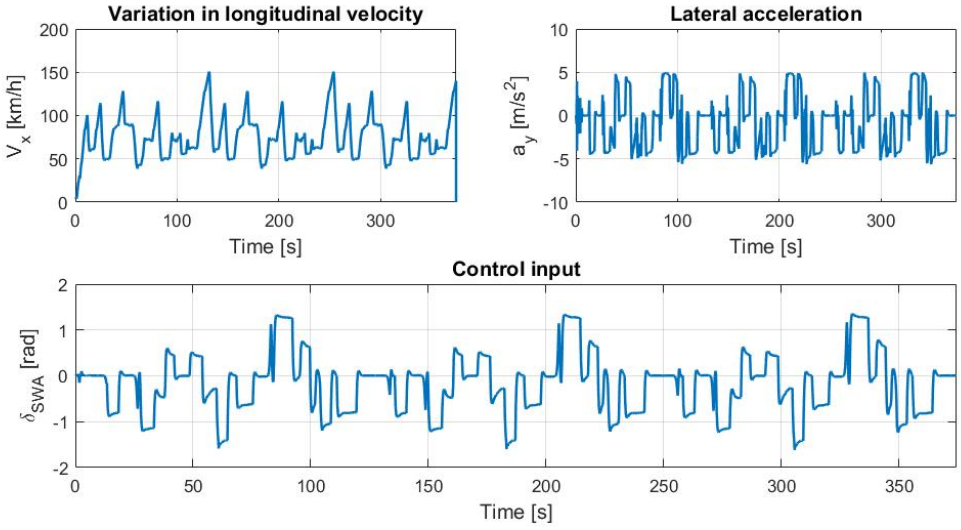

Figure 9. Results : PCwP

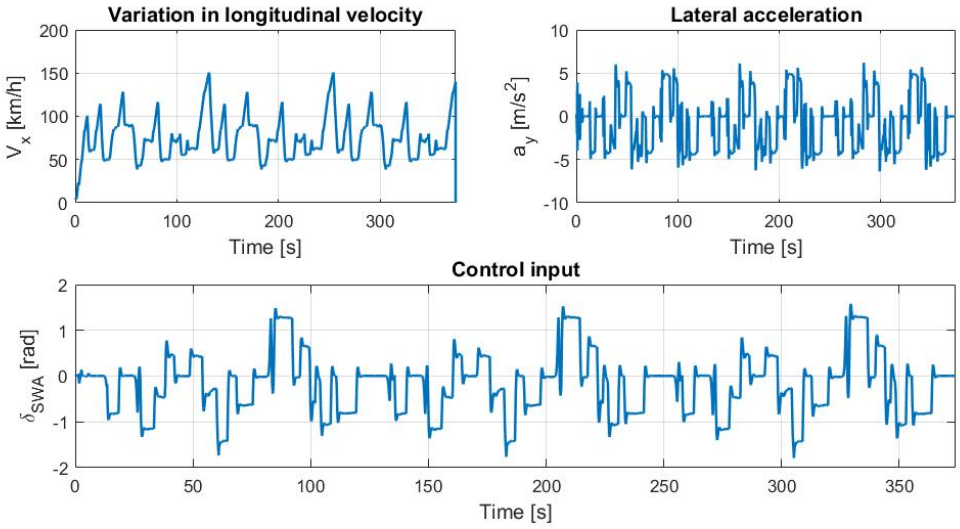

Figure 10. Results : Stanley

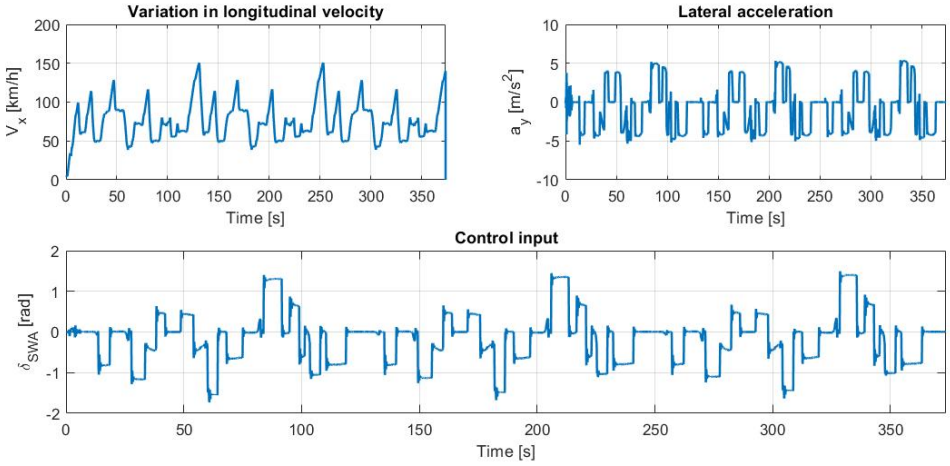

Figure 11. Results : MPC with Dugoff tire model

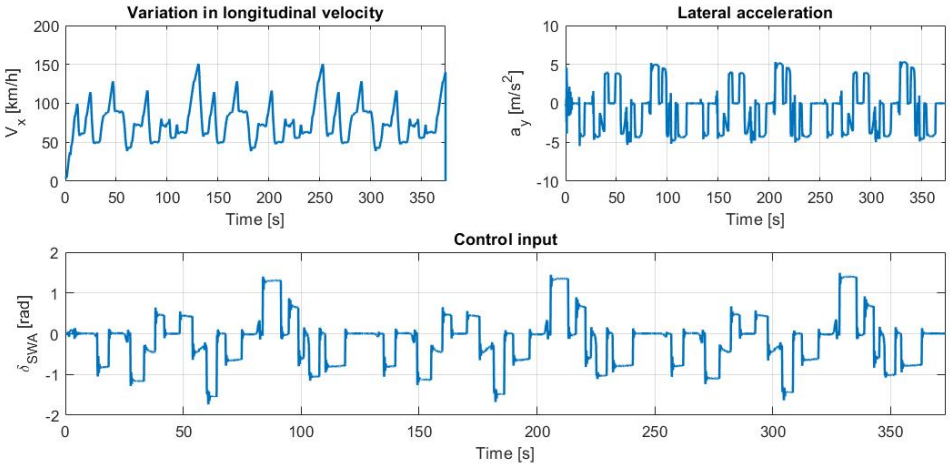

Figure 12. Results : MPC with sensor data

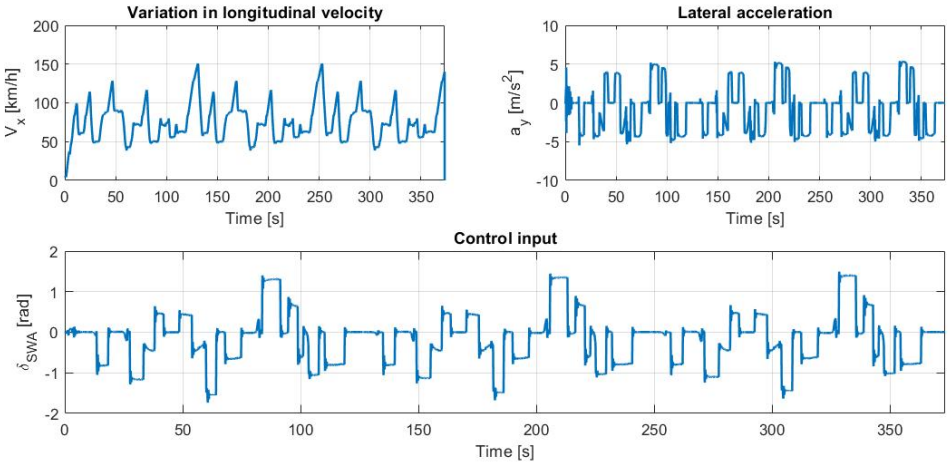

Figure 13. Results : MPC with LWPR learning

5.3. Real-time implementation

The real-time feasibility of the proposed design as verified in section 4.2.3, was also validated on the SCALEXIO real-time system. SCALEXIO is based on a state-of-the-art technological setup of real-time modular systems for hardware-in-the-loop applications. It also offers a high degree of flexibility in its configurations. The task-turnaround time for the controller was found to be 50 times lesser than the sample time.

6. Conclusion

The computed performance indicators in each case were normalized and compared as shown in figure 14.

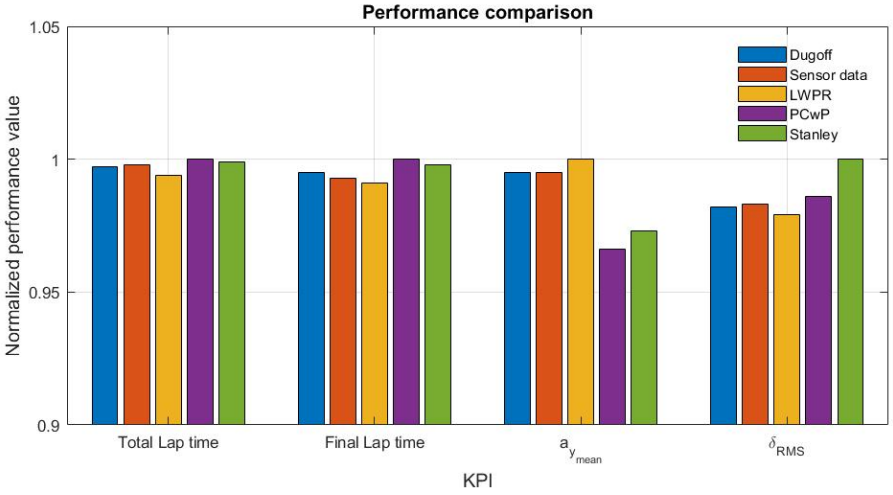

Figure 14. Comparison of controller performance

6.1. Lap time

It can be inferred from the above figure, that in terms of total lap time as well as the final lap time, the MPC with learning shows marginally better performance than the other MPC based controllers as well as the benchmark controllers.

6.2. Mean lateral acceleration

Among the discussed control techniques, the combination of MPC with online learning shows higher values of mean lateral acceleration ($a_{y_{mean}}$). This shows the better utilization of the tire by learning based MPC control.

6.3. Control effort

The metric δ_{RMS} gives an idea regarding the control effort of each controller. It can be seen that the benchmark controllers apply larger control actions as compared to the MPC-based techniques. This can be attributed to the fact that MPC can handle constraints on the vehicle states such as δ. However, the benchmark controllers do not have this provision of imposing constraints on the steering angle, leading to a large control action.

6.4. Summary

In summary, it can be seen that the benchmark controllers apply larger control actions to the vehicle as compared to the MPC based techniques. If the control effort is taken into account, as it is by imposing constraints on the MPC based control techniques, it can further be inferred that MPC with online learning shows best performance. **The learning based MPC, is able to adapt to the continuously degrading tire and show better lap times as well as higher mean lateral accelerations, while simultaneously keeping the control effort in an acceptable range.** The MPC based on the fixed Dugoff tire model, however, is not able to adapt to varying tire properties.

It must also be noted that this marginal improvement in performance is seen only within 3 laps of the same circuit. Over multiple laps, it is fair to say that this improvement will be scaled up. Furthermore, as mentioned in section 5, the driving behaviour was chosen such that a comparative study was possible with the benchmark controllers in mind. However, it was observed that the MPC-based learning controller was able to run on the SCALEXIO real-time machine with the same controller settings, even for more aggressive peak accelerations.

7. Future work

Since the designed MPC only controls the vehicle laterally, it makes sense to extend this implementation to compute an optimal velocity profile dynamically for longitudinal control instead of using a pre-defined velocity profile. Apart from this, steering actuator dynamics can be included within the prediction model of the MPC which takes into account the delay between steering actuation and its realization on the tire. Furthermore, a planar prediction model which takes into account load transfer can be used instead of a bicycle model.

It must be noted that the current implementation makes use of fixed constraints within the MPC. The inclusion of adaptive constraints is necessary in the case of tire degradation. The capabilities of the tire dictate the maximum permissible lateral acceleration and yaw rates. Thus, the constraints need to be varying in correspondence with the dynamic tire behaviour.

In order to accurately replicate a racing scenario, the reference must consist of the optimal racing line which needs to be calculated in a dynamic fashion. The current implementation can directly be used with the optimal racing line as a reference to provide better performance in terms of racing.

From the point of view of learning, the LWPR algorithm can be extended for multiple inputs other than normal load and wheel slip angle. This needs to be done in order to fully learn the tire behaviour.

An accurate simulation of tire degradation due to wear, changes in temperature and pressure needs to be implemented. This will be more realistic than linearly degrading the tire cornering stiffness coefficient over multiple laps.

Appendix A Simulation Parameters

Table A1. Vehicle and tire parameters

Parameter	Value	Unit
Vehicle mass	1659	[Kg]
Body inertia around z-axis	2916.6	[Kgm^2]
Wheelbase	2.91	[m]
Distance from front axle to center of gravity	1.2966	[m]
Half of front track width	0.808	[m]
Half of rear track width	0.802	[m]
Overall steering ratio	17	[-]
Half of rear track width	0.802	[m]
Front axle static cornering stiffness	1.65e5	[N/rad]
Rear axle static cornering stiffness	1.5e5	[N/rad]

Table A2. ACADO solver settings

Parameter	Setting
Hessian approximation	Gauss Newton
Discretization type	Multiple Shooting
Quadratic Programming (QP) solver	QP OASES
Levenberg-Marquardt regularization of the QP	1e-4
Integration method to discretize the time-continuous model formulation in the OCP	Explicit Euler
Number of integrator steps along the prediction horizon	3*N_p

Appendix B Tuning parameters

Table A3. MPC tuning - Race driving behaviour with tire degradation

	Velocity set-points [Km/h]					
Tuning weight	**30**	**50**	**70**	**100**	**120**	**150**
W_{Y_p}	3e2	4e2	4.5e2	4.5e2	3e3	3.5e3
W_{ψ}	7e9	7.5e9	7e9	7e9	6.5e9	7.5e9
$W_{d\delta}$	2e6	2e6	2.5e6	2e6	1e4	1e3
S_{Y_p}	7e1	7.5e1	7e1	7e1	3.5e3	4.5e3
S_{ψ}	4e12	1.5e12	2.5e12	2.5e12	1e11	1e11

References

1. Ahlberg, M. Optimization Based Trajectory Planning for Autonomous Racing. PhD thesis, 2018. doi:10.13140/RG.2.2.23680.38402.
2. Stewart, J. Meet the Self-Driving Star of the World's First Human-Free Car Race, 2018.
3. Casanova, D. On minimum time vehicle manoeuvring : the theoretical optimal lap. 2000.
4. Acosta Reche, M.; Ivanov, V.; Malygin, S. On Highly-Skilled Autonomous Competition Vehicles: An FSM for Autonomous Rallycross. 2019. doi:10.1109/ICMECH.2019.8722840.
5. Paden, B.; Cap, M.; Yong, S.Z.; Yershov, D.; Frazzoli, E. A Survey of Motion Planning and Control Techniques for Self-driving Urban Vehicles, 2016, [arXiv:cs.RO/1604.07446].
6. Rosolia, U.; Zhang, X.; Borrelli, F. Data-Driven Predictive Control for Autonomous Systems. *Annual Review of Control, Robotics, and Autonomous Systems* **2018**, *1*, 259–286. doi:10.1146/annurev-control-060117-105215.

7. Williams, G.; Drews, P.; Goldfain, B.; Rehg, J.M.; Theodorou, E.A. Aggressive driving with model predictive path integral control. 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 1433–1440.
8. Liniger, A.; Domahidi, A.; Morari, M. Optimization-based autonomous racing of 1:43 scale RC cars. *Optimal Control Applications and Methods* **2014**, *36*, 628–647. doi:10.1002/oca.2123.
9. Kapania, N.R.; Gerdes, J.C. Path tracking of highly dynamic autonomous vehicle trajectories via iterative learning control. *2015 American Control Conference (ACC)* **2015**. doi:10.1109/acc.2015.7171151.
10. Rosolia, U.; Carvalho, A.; Borrelli, F. Autonomous Racing using Learning Model Predictive Control, 2016, [arXiv:cs.LG/1610.06534].
11. Aswani, A.; Gonzalez, H.; Sastry, S.S.; Tomlin, C. Provably Safe and Robust Learning-Based Model Predictive Control, 2011, [arXiv:math.OC/1107.2487].
12. Ostafew, C.; Schoellig, A.; Barfoot, T. Learning-based nonlinear model predictive control to improve vision-based mobile robot path-tracking in challenging outdoor environments. 2014, pp. 4029–4036. doi:10.1109/ICRA.2014.6907444.
13. Kabzan, J.; Hewing, L.; Liniger, A.; Zeilinger, M.N. Learning-Based Model Predictive Control for Autonomous Racing. *IEEE Robotics and Automation Letters* **2019**, *4*, 3363–3370.
14. Vijayakumar, S.; Schaal, S. Locally Weighted Projection Regression : An O(n) Algorithm for Incremental Real Time Learning in High Dimensional Space **2016**. pp. 1–14. doi:10.1007/978-1-4899-7502-7_493-1.
15. Schaal, S.; Atkeson, C.; Vijayakumar, S. Scalable Techniques from Nonparametric Statistics for Real Time Robot Learning. *Applied Intelligence* **2000**, *17*. doi:10.1023/A:1015727715131.
16. Cruz, J.S.D.L.; Calisgan, E.; Kulić, D.; Owen, W.; Croft, E.A. On-line Dynamic Model Learning for Manipulator Control. *IFAC Proceedings Volumes* **2012**, *45*, 869–874. doi:10.3182/20120905-3-hr-2030.00149.
17. Iyer, K.; Shyrokau, B.; Ivanov, V. Offline and online tyre model reconstruction by locally weighted projection regression.
18. Hong Chen, Shuyou Yu, X.L.F.X.T.Q.; Wang, F. Applying Model Predictive Control in Automotive **2012**. doi:10.1109/wcica.2012.6357828.
19. Bruyne, S.D. Model-based control of mechatronic systems **2013**. *16*, 1425–1436. doi:10.1002/asjc.863.
20. Desaraju, V.R.; Michael, N. Leveraging Experience for Computationally Efficient Adaptive Nonlinear Model Predictive Control **2017**. doi:10.1109/icra.2017.7989625.
21. Johansen, T.A. Chapter 1: Introduction to nonlinear model predictive control and moving horizon estimation **2018**. *39*, 904–918. doi:10.1002/oca.2388.
22. Chowdhri, N. Model Predictive Control for Automated Driving and Collision Avoidance. Master's thesis, 2019.
23. Breuer, J.J. Analysis Of Driver-Vehicle-Interaction In An Evasive Manoeuvre - Results of "Moose Test" Studies **no. 98-S2-W-35, 1998**.
24. Asadi, F.; Mollakazemi, M.J.; Ghafouri, A. Influence of Parameters of Modeling and Data Distribution for Optimal Condition on Locally Weighted Projection Regression Method. *World Academy of Science, Engineering and Technology, International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering* **2015**, *8*, 1800–1807.
25. Vijayakumar, S. Incremental Online Learning in High Dimensions **2005**. *17*, 2602–2634. doi:10.1162/089976605774320557.
26. Thrun, S.; Montemerlo, M.; Dahlkamp, H.; Stavens, D.; Aron, A.; Diebel, J.; Fong, P.; Gale, J.; Halpenny, M.; Hoffmann, G.; Lau, K.; Oakley, C.; Palatucci, M.; Pratt, V.; Stang, P.; Strohband, S.; Dupont, C.; Jendrossek, L.E.; Koelen, C.; Markey, C.; Rummel, C.; van Niekerk, J.; Jensen, E.; Alessandrini, P.; Bradski, G.; Davies, B.; Ettinger, S.; Kaehler, A.; Nefian, A.; Mahoney, P. Stanley: The Robot That Won the DARPA Grand Challenge: Research Articles. *J. Robot. Syst.* **2006**, *23*, 661–692.
27. Lu, Z.; Shyrokau, B.; Boulkroune, B.; van Aalst, S.; Happee, R. Performance Benchmark of State-of-the-art Lateral Path-following Controllers. 2018. doi:10.1109/AMC.2019.8371151.

II

Background

2

Model Predictive Control

This chapter provides a background on model predictive control (MPC). Firstly, section 2.1 introduces the reader to MPC along with a motivation behind exploring non-linear model predictive control (nMPC). Section 2.2 briefly discusses the theory behind MPC. Following this, section 2.3 the fundamentals of MPC design is presented. The challenges that come with MPC are given in section 2.4. Finally, the implementation of MPC for autonomous racing is presented in section 2.5.

2.1. Introduction

With the advancement in science and technology over recent years, the requirement of efficient control techniques is ever increasing. Instead of classical techniques that satisfy a stable control design, optimization is integrated in the design of the controller in order to attain better control performance.

The need for nMPC

The domain of vehicle motion control especially at the limits of handling comprises of a multiple non-linearities in a system which is bound to physical as well as operational constraints on its inputs and states. Previous research shows that the prominent non-linear control techniques such as feedback linearization [2] [50] [28] and Lyapunov based methods [36] [59] that result in effective solutions. These techniques, however, depend on complicated procedures that cannot be scaled to larger systems and cannot handle constraints in an orderly manner. High-rate adaptive control techniques [44] [32], even though simple fail yo satisfy constraints and are merely reactive in nature in order to eliminate unmodeled dynamics.

In contrast to the aforementioned approaches, model predictive control seeks to bring a balance between the predictive nature of infinite-horizon control and reactive nature of traditional control techniques [16]. As seen in [33], the concept of non-linear model predictive control (nMPC) along with its underlying principle of optimal control is an inviting alternative. This is due to its ability to handle complex processes with a lot of inputs and states that simultaneously fulfill the respective constraints imposed on the system.

2.2. Theory

MPC comprises of defining an optimal control problem (OCP) within a finite horizon at every sampling instance. The states of the plant are updated at every sample, this results in a new optimization problem to be solved at each sample time. This is the underlying concept of the receding-horizon approach.

MPC uses a prediction model which makes use of the open loop dynamics of the system to predict its future behaviour over a finite time interval called the *prediction horizon*. The corresponding control actions required to drive the predicted response to its reference optimally, are computed by solving an OCP over the prediction horizon. The OCP consists of minimizing a cost function based on the control objectives of the MPC while satisfying various constraints which are imposed on the system.

On solving the OCP at each sampling instant over the prediction horizon, results in an optimal control "sequence". Only the first control input from this "sequence" of control actions, is provided to the system. This is the main idea behind the *receding horizon principle*. The new system states and measurements are then used to formulate the new OCP to be solved at the next sampling instant. This procedure is continued over the entire horizon, thus giving optimal control at each and every sampling instant [11].

The authors in [14] show the fundamental concept of MPC applied to a SISO system. It can be noted that, we predict the evolution of the output for the next P steps, **parametrically** on the initial condition and the control sequence. The receding horizon principle can be seen here, where the past control action sequence is completely ignored and the future "optimal" control sequence is obtained by solving the OCP at that particular sampling instance. At the current instance k, the MPC framework computes a set of M values of the input sequence $u(k+i-1), i = 1,2,...M$. The number of control actions M is called the *control horizon*. The control action is held constant after the stipulated M control inputs. These control inputs are computed so that the set of P predicted outputs $\hat{y}(k+1), i = 1,2...P$ reaches its respective set-point or reference in an optimal fashion. Here, the number of predictions P represents the *prediction horizon*.

The receding horizon principle allows the utilization of new information immediately. Otherwise, the predictions and subsequently computed control actions will be purely dependant on past data, thereby adversely affecting the system dynamics on account of unmeasured disturbances. A driver, who is not able to predict the behaviour of the system over a long prediction horizon will naturally find it difficult to react when sudden evasive action is required. This is what sets MPC apart from the skills of a driver.

However, for MPC to be efficient, the prediction model in the MPC formulation must be as accurate as possible. The model equations must represent the vehicle's dynamic behaviour accurately. It is unwise to rely on the control actions from the MPC if the model is not well-defined.

2.3. Design

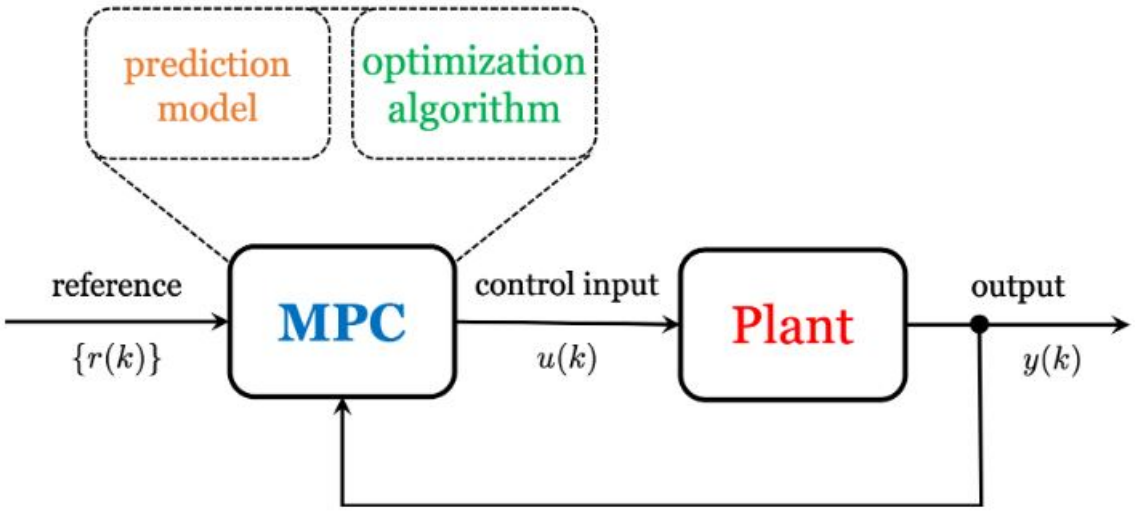

Figure 2.1: MPC control scheme [25]

As seen in figure 2.1, there are two parts that form the core of a model predictive controller,

- A prediction model : to predict the future evolution of vehicle states
- An optimization algorithm : to obtain the best possible control action at the present instant

2.3.1. The prediction model

The system model can be given in general in the form,

$$x(k+1) = f(x(k), u(k)) \tag{2.1}$$

$x(k)$ represents the states of the system, $u(k)$ is the control action and f is the function representing the system dynamics. Generally, f can be either linear or non-linear, both of which can be handled well within the MPC framework.

The prediction model is derived from the system model which is then used to compute the optimal control action. Considering a linear time invariant (LTI) system, the model equation (2.2) is given by,

$$x(k+1) = Ax(k) + Bu(k) \tag{2.2}$$

By propagating the above equation further in time, the j^{th} system state prediction is given by (2.3),

$$x(k+j) = A^j x(k) + \sum_{i=0}^{j-1} (A^i B u(k+j-1-i)) \tag{2.3}$$

Considering the output $y(k)$ of the LTI system as,

$$y(k) = Cx(k) \tag{2.4}$$

From (2.4) and (2.2), the prediction equation for the system output is obtained,

$$y(k+j) = CA^j x(k) + \sum_{i=0}^{j-1} (CA^i B u(k+j-1-i)) \tag{2.5}$$

For a prediction horizon N_p, the prediction model (2.6) is obtained as,

$$Y_p(k) = A_p x(k) + B_p U(k) \tag{2.6}$$

where,

$$Y_p(k) = \begin{bmatrix} y(k+1) \\ y(k+2) \\ . \\ . \\ . \\ y(k+N_p) \end{bmatrix}, A_p = \begin{bmatrix} CA \\ CA^2 \\ . \\ . \\ . \\ CA^{N_p} \end{bmatrix}, B_p = \begin{bmatrix} CB & 0 & 0 & . & . & . & 0 \\ CAB & CB & 0 & . & . & . & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ CA^{N_p-1}B & 0 & 0 & . & . & . & CB \end{bmatrix}, U(k) = \begin{bmatrix} u(k) \\ u(k+1) \\ . \\ . \\ . \\ u(k+N_p-1) \end{bmatrix}$$

As seen above, Y_p is the predicted output at instance k, U is the predicted control action at the k^{th} instance. A_p is the extended observability matrix and B_p is the Toeplitz matrix. The prediction model is thus developed using the plant dynamics and this is further used for optimal control action calculation by solving an optimization problem.

2.3.2. The optimization problem

The general structure of the optimal control problem is as shown,

$$\begin{gathered} min_{u_{\mathbf{Np}}} V_N(x_0, u_{Np}) \\ \text{s.t } \textit{system dynamics} \\ \text{s.t } \textit{system constraints} \\ x(N_p) \in \mathbf{X}_f \\ \text{where, } x(t) = x_0 \end{gathered}$$

$\mathbf{X_f}$ represents the terminal set. It is defined such that an admissible control sequence must be able to drive the state from its initial value x_0 to $\mathbf{X_f}$ within N_p steps. The system dynamics is as shown in (2.1). The cost function is defined as,

$$V_{Np}(x_0, u_{Np}) = \sum_{k=0}^{N_p-1} \underbrace{l(x(k), u(k))}_{\text{stage cost}} + \underbrace{V_f(x(N_p))}_{\text{terminal cost}} \tag{2.7}$$

The cost function is designed to be in two parts. Firstly, the stage cost comprises of penalties on states and inputs (Q and R positive definite matrices). This helps keep the error between the reference and actual output to be low while, simultaneously minimizing the control action energy.

$$J_{stage}(k) = \sum_{i=1}^{N_p-1} ((y(k+i) - r(k+i))^T Q_i (y(k+i) - r(k+i)) + u(k+i-1)^T R_i u(k+i-1)) \tag{2.8}$$

Secondly, the terminal cost is important for asymptotic stability. This cost function helps ensure that the plant is near the desired trajectory at the end of the prediction horizon. It also makes sure that the optimal control action keeps the plant in the stable region.

$$J_{terminal}(k) = V_f(x(N_p)) = (y(k+N_p) - r(k+N_p))^T P (y(k+N_p) - r(k+N_p)) \tag{2.9}$$

where, P is the penalty for the terminal cost. The future state values comes from the prediction model as in (2.6) for linear systems.

As touched upon earlier in sections, constraint handling makes MPC a very powerful control technique. There may be equality or inequality constraints on states, control action or their rates. It can be represented as,

$$G(x, u, \Delta u) \leq \epsilon \tag{2.10}$$

where, ϵ is the bound value. An equality constraint that forms the definition of the control horizon (N_c) is as follows,

$$\Delta u(k+j) = 0, \quad j \geq N_c \tag{2.11}$$

The control horizon is defined such that after N_c the control input is assumed to be held at the same value ($u(k+N_c-1)$) for remaining predictions. Usually, $N_c \leq N_p$. This helps reduce the computational effort of the controller.

2.3.3. Outline

The general MPC law can be described using the following algorithm,

Algorithm 1: Basic MPC law

1. Obtain the new set of vehicle states x(t)
2. Obtain optimal control sequence by solving the OCP (2.3.2)
3. Use the first control action from the optimal sequence, $u(t) = u(t+0|t)$
4. t $\longleftarrow t+1$

Repeat from step 1

2.4. Challenges

The framework of an MPC, however well designed has its own problems.

Computational Effort

One such issue that comes with implementing MPC in control problems is the heavy computational demand specifically when the controller deals with non-linearities. The optimization problem along with its constraints should be solved online. At each iteration, the controller makes use of standard algorithms which results in heavy computational burden and thus large computational time. Due to its computational complexity, MPC was deemed suitable, mainly for processes with slow dynamics and with larger sampling periods [74].

However, a vehicle dynamic system is the exact opposite. It involves fast changing dynamics at small sampling rates. In areas such as manufacturing, mechanical, electrical and aerospace, where fast dynamical systems are prevalent, conventional algorithms used for optimization, usually fail to satisfy the demands of real-time computing with small sampling periods. Highly efficient MPC design is thus required in order to achieve real-time computing.

The need of developing nMPC is increasingly recognized by both the control society and industry. Nowadays, high performance computing is readily available with the growth of the process-building industry. This allows nMPC to process fast dynamics and the OCP can be solved at small sampling times.

Feasibility

Ensuring the feasibility of the optimization problem shown in section 2.3.2 at each time instance t is another stumbling block in the MPC formulation. As stated in [9], the problem is assumed to be feasible at the initial instant ($t = 0$), while the cost function (2.7) is chosen with the idea of conserving feasibility at later time instances. One way to do this is by ensuring that the shifted optimal control sequence $\{u(t+1|t), ..., u(t+N_p|t)\}$ is feasible at $t+1$.

Additionally, constraints involving the state components are treated as "soft" constraints with the use of slack

variables (s) as follows,

$$G(x,u) \leq \epsilon + s \begin{bmatrix} 1 \\ . \\ . \\ 1 \end{bmatrix} \tag{2.12}$$

Pure input constraints are kept "hard", since the inputs are obtained after the optimization. By relaxing the state constraints, the issue regarding feasibility can be eliminated in case of stable systems.

Stability

As seen in [42], it is not straightforward as to the conditions under which the closed loop system is stable in either the infinite or finite horizon constrained MPC. Some of the techniques used in the literature to "enforce" stability are reviewed in [9].

End (Terminal) Constraint

The stability of the overall system in the linear case, can be ensured by specifically ensuring that the system converges to the region of feasibility at the end.

$$x(t+N_p) = 0 \tag{2.13}$$

Such constraints have the problem of resulting in high control actions that may be needed to drive the state to its reference, particularly for a small prediction horizon. The region of attraction of the closed loop system (which includes MPC and the plant) is restricted to the set of initial states x_0 that can be steered to origin within N_p steps.

A modification of the idea of the terminal constraint has been proposed in [51], where only the unstable modes are driven to the origin at the end of the horizon.

Terminal Weight Matrix

The terminal cost function (2.9) has a weight matrix P. This can be chosen as the solution of a Riccati inequality in order to guarantee stability for the linear case without the addition of separate constraints.

Robustness

Another point of concern regarding MPC is its robustness to model uncertainties and noise. As stated in [9], a control system is deemed "robust" if it is possible to ensure its stability and its performance meets the necessary specifications for a particular uncertainty range. There is more scope of research regarding controlling systems with constraints while ensuring robustness.

2.5. Implementation in autonomous racing

Two approaches of tackling the autonomous racing problem have been addressed in [37]. The first approach employs a high level MPC to create a suitable reference for the low level MPC to track. The next technique makes use of Model Predictive Contouring Control (MPCC) to strike a balance between track progress and contouring error. A comparison of two applications of MPC in autonomous racing is presented in [68]. The authors compare a tracking based MPC with an MPC that is designed to minimize lap time.

In general, designing an MPC is challenging in the presence of model uncertainty. In [7], a nominal model is used to validate constraints and a learned model to enhance performance. It is however difficult to predefine a feasible model for the racing problem. An alternative proposed in [47], involves an MPC framework using a simplified vehicle model in combination with a learned disturbance model. The authors have highlighted the effectiveness of using this technique for simple path following. For the racing problem however, it is desired to have a vehicle model that can capture the limit handling behaviour. The authors in [66], have formulated a solution to the racing problem which is based on the linearization of the dynamics as well as the reference trajectories. The aforementioned paper pays close awareness to the cost function and constraints in order to reduce the effect of linearization.

In [52], the authors investigate a learning nMPC for autonomous racing. The proposed technique utilizes previous lap information to improve the closed loop performance on track. Furthermore, this is made to work hand-in-hand with a system identification algorithm that is able to recognize the vehicle dynamic behaviour at the handling limits. It is proved through simulations that the designed system allows the vehicle to operate

at its limits. The authors, in [10], have experimentally verified that the aforementioned repetitive formulation of MPC improves the track performance on a 1:10 scaled setup.

It must be noted that, there is a mismatch between the tire model within the MPC and real-world tire behaviour. This therefore poses a limitation in terms of robustness of the designed nMPC. An attempt to overcome this, is to somehow "learn" the tire properties using sensor data from the vehicle. This shall be further discussed in the following chapter.

3

Tire Models

This chapter provides some context regarding tire models. Section 3.2 and section 3.3 introduce the reader to the Dugoff and Delft tire models respectively. Finally, section 3.4, starts off with the motivation behind using learning based techniques. This is followed by the theory behind the LWPR algorithm, its applicability and some possible challenges. Finally, some implementations of this algorithm have been presented to the reader.

3.1. Introduction

Tires are one of the most crucial components to be considered while designing a vehicle control system. Accurate representation of the capabilities of a tire to generate force both laterally and longitudinally is crucial in racing. Extensive research work has been done on tire modelling. This can be classified into two fundamental approaches - physical methods and empirical methods.

The physical methods are centered around the understanding of the physical mechanism which gives rise to tire forces. This ranges from the fundamentals of mechanics to complex finite element methods to try to comprehend tire behaviour. Usual practice is to model the tire contact patch as a spring-mass-damper system. Under this classification, different models such as LuGre, Dugoff and brush tire model have been developed.

The empirical methods involve interpolating or fitting data in which a set of parameters are designed to fit a curve that encapsulates as much of the tire characteristics as possible. This type of tire modeling is heavily reliant on tire data. The more data available, the more accurate the fit. The most accurate empirical tire model is the Magic Formula (MF) tire model [48].

As elaborated under section 2.3, the prediction model within the MPC includes tire force behaviour which comes from either a static Dugoff tire model or via online learning. Furthermore, the IPG vehicle used as the plant comprises of the Delft tire model. This chapter discusses the different tire models used as well as the algorithm used for online learning of tire force behaviour.

3.2. Dugoff Model

The Dugoff model [17] is a widely accepted mathematical tire model that includes the rotational degrees of freedom of the wheel as well as the influence of longitudinal and lateral tire forces (F_x, F_y) as a function of normal load on the tire (F_z), coefficient of friction (μ), wheel slip angle (α) and longitudinal slip (κ).

The following equations (3.1) to (3.5) represent the Dugoff tire model. Equation (3.1) represents the friction coefficient computation based on the peak friction coefficient (μ_0), the friction reduction coefficient (e_r), the longitudinal velocity (V_x), κ and α. Equations (3.2) and (3.3) represent the weighting coefficient (λ) and the corresponding weighting function ($f(\lambda)$). The equations (3.4) and (3.5) show the calculation of F_x and F_y which are a function of the longitudinal and lateral tire stiffness ($C_{x\kappa}, C_{x\kappa}$).

$$\mu = \mu_0(1 - e_r V_x \sqrt{\kappa^2 + tan^2(\alpha)}) \tag{3.1}$$

$$\lambda = \frac{\mu F_z(1-\kappa)}{2\sqrt{(C_{x\kappa}\kappa)^2 + (C_{y\alpha} tan(\alpha))^2}} \tag{3.2}$$

$$f(\lambda) = \begin{cases} \lambda(2-\lambda), & \text{if } \lambda < 1 \\ 1, & \text{otherwise} \end{cases} \tag{3.3}$$

$$F_x = \frac{C_{x\kappa}\kappa}{1-\kappa} f(\lambda) \tag{3.4}$$

$$F_y = \frac{C_{y\alpha} tan(\alpha)}{1-\kappa} f(\lambda) \tag{3.5}$$

3.3. Delft Model

The Delft tyre model [1] was used in order to implement tyre degradation within the IPG CarMaker vehicle. A scaling factor was applied to the cornering stiffness coefficient, degrading the tire throughout the simulation.

3.4. LWPR learning

3.4.1. Motivation

Autonomous vehicles require precise control to satisfy safety and comfort constraints. In some cases, the accuracy of models are not enough to ensure effective control performance due to unmodeled non-linearities. As a result, accurate dynamic models are required. A major obstacle to overcome is the continuously varying system dynamics. This may be due to external factors (e.g. road surface) or internal vehicle dynamic parameters, whose estimation may be difficult (e.g. cornering stiffness, slip angle changes etc).

In order to capture the entire vehicle dynamic behaviour a complex non-linear model is required. Even for relatively simple maneuvers in which the tire predominantly resides in the linear regime, tire stiffness changes with changes in vehicle velocity and environment [12]. As stated in [13], the authors used adaptive control techniques to estimate dynamic model parameters online. However, despite such advancements, these methods still rely on adequate knowledge of system model and are thus prone to modelling uncertainties. An alternative to this is sliding mode control (SMC) which, even though robust to modelling errors, is susceptible to chattering.

While adaptive techniques assume an inherent dynamic model structure, model learning controllers attempt to "learn" the model. Conventional control techniques follow the trend of considering known tire parameters while modelling for a fixed condition. In this sense, there exists a research gap in the development of systems that can adapt to varying tire behaviour. Online learning is beneficial to adapt to fast changing dynamics. Therefore, in order to further enhance the tracking performance of the above elaborated nMPC, an attempt is made to infuse online learning within the control structure.

The concept of making use of a regressor to determine the dynamics of a system used in the MPC has been well documented in different applications [7]. Non-linear regression techniques come with high computational complexity and hence, usually dynamic models are only learned offline for pre-sampled trajectories [19][54]. In order to utilize a learning approach to its fullest, online learning becomes necessary to help adapt due to changes in dynamics. When it comes to vehicle dynamics at the limits of handling, a training data set used for offline learning will not suffice. The model needs to be updated constantly via online learning with sensor measurements in order to encapsulate the complete behaviour of the vehicle.

Online learning using neural networks has been portrayed to be effective in high performance vehicle control [23] and in robotics control [43] [30] [21] [29]. As stated in [24], online adaptation of neural networks comes with the problem of "catastrophic forgetting", which can be described as "*the tendency of neural networks to forget old data when fed new data from another distribution*". This may lead to controller failure. In an attempt to overcome this issue, attention is turned to a technique that is immune to catastrophic forgetting, namely locally weighted projection regression (LWPR).

3.4.2. Theory

LWPR is a relatively new algorithm that is used to approximate non-linearities in high dimensions in the presence of redundant input dimensions [71]. Recent studies shows that non-linear function approximation while dealing with input data of high dimensions is still an area of active research, especially for real-time applications. As proposed in [70] an algorithm is needed that

- avoids numerical problems due to redundant inputs
- eliminates irrelevant input dimensions

- maintains a low computational complexity in its learning updates
- allows online learning
- accurately approximates functions with adequate generalization

There are two main classes of techniques used for function approximation .

Global fitting

Firstly, the technique of fitting non-linear functions globally. This technique is in line with current research trends. For example, Gaussian process regression (GPR) [73], variational Bayes for mixture models (VBM) [22] and support vector machine regression (SVMR) [57] come under this category. Even though these techniques have a sound theoretical base in terms of convergence, they are not suited for high dimensional real-time applications. This is because,

- They require appropriate determination of kernels or basis functions in the case of SVMR and GPR and proper initialization in case of VBM
- They are not suited for incrementally arriving data. Adding new data points is computationally costly in case of SVMR and GPR, whereas VBM needs storing data when new mixture components are being added.
- They are computationally expensive for real-time learning

Previous work [46], comparing these methods report that although SVMR and GPR may lead to higher accuracy than LWPR, their heavy computational loads does not allow for incremental online learning.

Local fitting

In contrast to this, the second class of function approximation techniques involve fitting non-linear functions with spatially localized models. This is suitable for real-time incremental learning. In this domain, locally weighted learning (LWL) [8] methods are handy when there is little known regarding model complexity. As elaborated in [55], with LWL, it is possible to increase the model resources in a data-driven and incremental way. However, with increasing number of input dimensions, in order to maintain accuracy, these techniques suffer from an exponential increase in the number of local models, also known as the "curse of dimensionality" [58].

Projection regression

Additionally, projection regression (PR) techniques are able to cope with high dimensional inputs. The multivariate regressions are dissolved to form a cluster of single-variate regressions along selected input directions. A problem encountered is the selection of efficient projections to achieve accurate fitting.

Locally weighted projection regression

LWPR, as suggested by its full form, is an algorithm that extends the useful properties of localized learning to non-linear function approximation. As seen in [70], a prerequisite for this approach is that the high-dimensional problem has locally low dimensional distributions. This assumption is seen to cover a broad class of real-world data [64] [53] [72] [18].

With this assumption in mind, local models are allocated within the locally low dimensional distributions. Thus, the local models only occupy a small part of the high dimensional space. This eliminates the problem caused by the "curse of dimensionality" of LWL. Under these conditions, an alternate method of PR can be used which finds efficient local projections. The result is a learning algorithm that fuses the speed, efficiency, and incremental capabilities of LWL techniques without having to face the problems due to high dimensional input spaces with the help of local projections.

3.4.3. Algorithm

The LWPR algorithm involves two main steps namely, the incremental partial least squares and the distance metric update. These are elaborated under this section.

Algorithm 2: Incremental PLS

Given : A training point (x, y)

Update the means of input and output :

$x_0^{n+1} = \frac{\lambda W^n x_0^n + wx}{W^{n+1}}$

$\beta_0^{n+1} = \frac{\lambda W^n \beta_0^n + wy}{W^{n+1}}$

where $W^{n+1} = \lambda W^n + w$ and $x_0^0 = u_i^0 = \beta_0^0 = W^0 = 0$

Update the local model :

1. Initialize : $\mathbf{z} = x, res_1 = y - \beta_0^{n+1}$

2. For $i = 1 : r$

(i) $u_i^{n+1} = \lambda u_i + wzres_i$

(ii) $s = z^T u_i^{n+1}$

(iii) $SS_i^{n+1} = \lambda SS_i^n + ws^2$

(iv) $SR_i^{n+1} = \lambda SR_i^n + wsres_i$

(v) $SZ_i^{n+1} = \lambda SZ_i^n + wzs$

(vi) $\beta_i^{n+1} = \frac{SR_i^{n+1}}{SS_i^{n+1}}$

(vii) $p_i^{n+1} = \frac{SZ_i^{n+1}}{SS_i^{n+1}}$

(viii) $z = z - sp_i^{n+1}$

(ix) $res_{i+1} = res_i - s\beta_i^{n+1}$

(x) $MSE = \lambda MSE_i^n + wres_{i+1}^2$

Predicting with novel data :

Initialize : $y = \beta_0, z = x - x_0$

For i = 1:k

(i) $s = u_i^T z$

(ii) $y = y + \beta_i s$

(iii) $z = z - sp_i^n$

SS,*SR* and *SZ* are memory terms that help perform uni-variate regression using recursive least squares as shown in step (vi) in the model update. Step (vii) helps regress the projection from the current input data $\mathbf{z}$ and the current projected data s. This ensures that u_{i+1} is orthogonal to u_i. There are two important properties of the local projection scheme. Firstly, if we have statistically independent input variables, PLS takes only a single iteration to find the optimal projection direction u_i. This corresponds to the gradient of the locally linear function to be approximated. Secondly, step (i) in the model update in algorithm 2 ensures that the projection direction is chosen by correlating the input and output data. This results in the automatic exclusion of input dimensions that do not contribute to the output. Finally, since the uni-variate regressions will never be singular, there is no concern of numerical problems in PLS.

Algorithm 3: Distance metric update

$D = M^T M$, where M is upper triangular

$M^{n+1} = M^n - \alpha \frac{\delta J}{\delta M}$ where the cost function to be minimized is chosen to be,

$J = \frac{1}{W} \sum_{i=1}^{M} \sum_{k=1}^{r} \frac{w_i res_{k+1,i}^2}{1 - w_i \frac{s_{k,i}}{s_k^T W s_k}} + \gamma \sum_{i,j=1}^{N} D_{ij}^2$

The first term in the cost function represents the mean leave-one-out cross-validation error of the local model. The second term is a penalty term which ensures that the receptive fields do not shrink in case of

huge amounts of training data.

Algorithm 4: LWPR Outline

1. Initialize the LWPR with no receptive fields
2. **For** every new training sample (x, y)
   - **For** $k = 1 : RF$
     (i) Calculate the activation from eq. (**??**)
     (ii) Update according to algorithms 2 and 3
   - **End**
   - **If** no linear model was activated by more than w_{gen},
     create a new RF with $r = 2, c = x, D = D_{def}$
   - **End**

End

The major assignment within the LWPR framework consists of determining the number of local models k, computing the regression coefficient β_k and the weight w_k for the k^{th} locally linear model. Additionally, it consists of regulating the local model's receptive field. In the algorithm 4, a threshold w_{gen} is predefined. This determines when to create new receptive fields. The closer this value is to 1, the more overlap local models will have, but will be more costly to compute. The distance metric D is initialized to D_{def} and is usually diagonal.

3.4.4. Applicability of LWPR

As stated in [35] the proposed LWPR technique is most applicable for the following problems

- A **non-linear** function needs to be learnt. Otherwise there is no justification of having multiple locally linear models.
- There is **large amount of training data**.
- The application requires **online and incremental learning**. In comparison with global function approximators such as multi-layer neural networks, this algorithm has a benefit that its local models are able to *learn independently*.
- The model requires **adaptation** to changes in the target mapping with time. LWPR excels in such cases because a tune-able built-in forgetting factor is available. This can be used to match the expected time scale of the transience. The adaptation is quick because all other parameters such as the receptive field placements and sizes, and local projection directions of a well trained model can be retained while the forgetting factor and regression parameters are adjusted.

3.4.5. Previous research

Apart from the state of the art mentioned in the attached paper, the authors in [31] have analyzed the online learning performance of this algorithm specific to tire force reconstruction. The results were a motivation to implement this algorithm for autonomous racing in combination with MPC. The research paper concerning this is made available to the reader in part III.

3.4.6. Challenges

An obstacle faced while utilizing locally linear methods is the computational cost. For this method to be accurate, usually tens of thousands of local models are needed. Considering that tire input data is received at high frequencies, the algorithm needs to produce a lot of predictions per second. In contrast to this, even less intensive MPC algorithms need an order of tens of thousands of predictions per second. As stated in [24], the computational demands on the model using LWPR is 3 orders of magnitude less than the that used for model predictive control. This essentially means that, when used in tandem with MPC, it is possible to make use of LWPR models that demand more computationally, which are suitable for online adaptation.

The authors in [39] state that, the accuracy of this algorithm is affected by the linearity of each projected subset. Subsets with more linearity will result in better accuracy. However, a common misconception is that if there are more projection directions, the prediction will be more accurate. The final reconstructed model

may deviate a lot from the true value if too many projections are used. This is because there is some deviation between the prediction and true values on each projection direction which may be compounded if too many projections are taken.

As mentioned in [45], the main stumbling block of this algorithm is the manual tuning of highly data-specific parameters. Compared to other techniques it is difficult to determine the suitable learning rates and a appropriate initial size for the receptive fields [63]. Therefore, parameter sensitivity is a concern while employing this technique.

3.4.7. Implementation

The racing environment is an ideal area of application of learning based control because the race-cars are exposed to repeated laps of the same circuit while operating at the limits of handling. This creates an scenario where the vehicles are repeatedly subjected to non-linear varying dynamics and road conditions from lap to lap. In this project, online learning is attempted with focus on the tires. The tires are the most complex part of a vehicle dealing with a lot of non-linearities especially when considering vehicle motion control at handling limits. The tire model is updated online using LWPR in order to capture non-linearities. This "learning" helps the controller develop a notion of experience while it fully encompasses the input-output behaviour for all previous control actions [15].

Problems faced

Even though the MATLAB library for this algorithm provided in [69] is useful, it is important to note that it can be used only for offline learning. As explained in [49], the provided code is restricted to offline usage where the update of receptive field parameters needs to be done offline first using a fixed training data set. Model approximation is then done using the trained parameters. This is because the MATLAB code uses a slightly different function library than SIMULINK. Since C code is used to compile SIMULINK programs running in real-time, it cannot access any MATLAB files or functions from the base workspace. Furthermore, since the structure of the code in the provided library is not suitable to implement online learning in SIMULINK, a solution was to rewrite the code provided.

Similar to the approach used in [24], three different types of tests will be used to validate this algorithm.

- Offline learning : Model adaptation is not allowed in the simulation.
- Online learning : Model adaptation is allowed during simulations, but no control is applied.
- Active learning : Model adaptation is allowed and the vehicle is controlled with the adapted model.

1. **Offline learning**

   (a) *Non-linear function approximation*

   This section shall demonstrate the function approximation capabilities of the LWPR algorithm. LWPR was run on 500 noisy training data points from the two dimensional cross function,

   $$y = max(e^{10x_1^2}, e^{50x_2^2}, 1.25e^{5(x_1^2+x_2^2)}) + N(0, 0.01) \tag{3.6}$$

   This function was chosen since it has a combination of areas with varying curvature [70], which makes it suitable to test the algorithm's learning capabilities. Figure 3.1 shows the fitted function with a normalized mean square error of 0.045. The table 3.1 shows the values of the tuning parameters used.

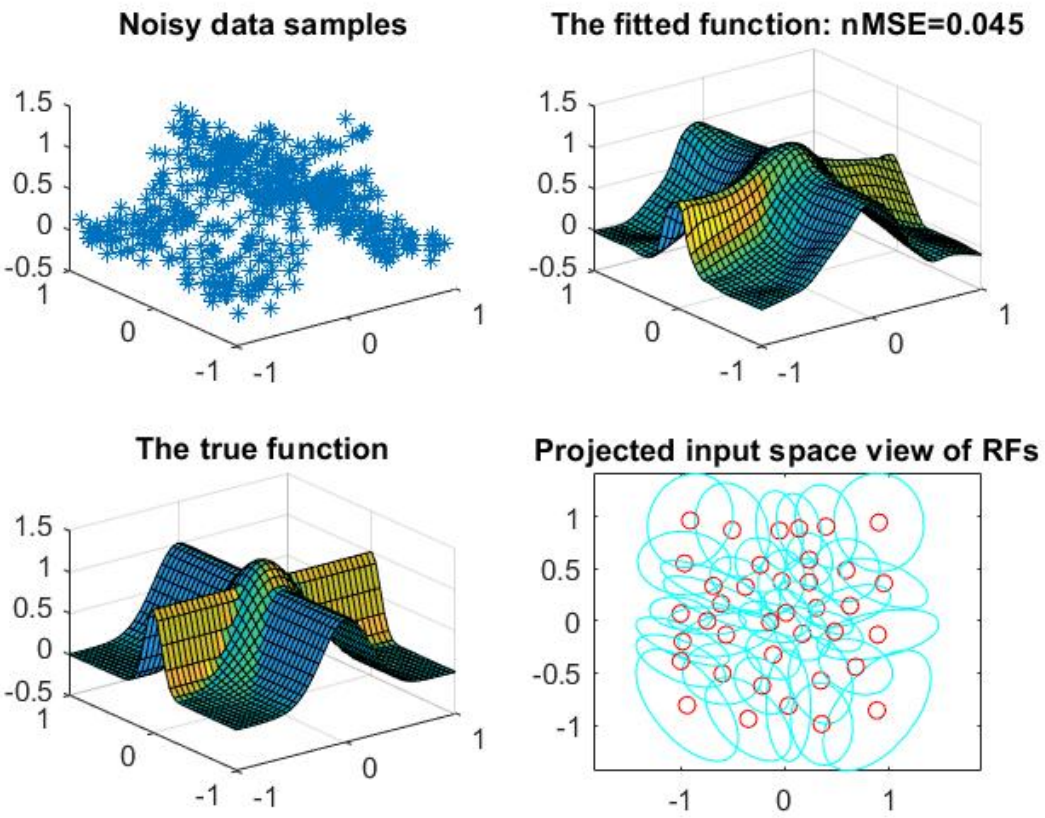

Figure 3.1: Non-linear function approximation using LWPR

Parameter	Value
Initial distance metric	0.25**I**
Initial element-wise learning rate	250
w_{gen}	0.2
w_{prune}	0.7
Learning rate for 2^{nd} order distance metric updates	250
Kernel	Gaussian

Table 3.1: LWPR parameters for non-linear function approximation

The results show that LWPR is an excellent non-linear function approximation technique.

(b) *Offline learning of tire model - Lateral force vs slip angle*

The implementation of LWPR scheme to estimate tire properties are shown in this section. First, simple learning of pure lateral force is shown. The LWPR algorithm is presented with training data comprising of lateral force F_y versus wheel slip angle α. A constant normal force is assumed and the slip angle is varied between -20 and 20 degrees. The camber is set to zero. The algorithm then "learns" the given model and is presented with test data which lies in the same range as the training data.

The figures shown below represent four separate iterations of the learning procedure by increasing the initial distance metric by an order of 10 each iteration. It is evident that as the initial distance metric changes, there is a change in the number of receptive fields generated as well as the mean square error. Bad choices of the initial distance metric can affect the learning procedure. If the

initial distance metric is set to be too small, the receptive fields are larger and this may result in local minima and slow convergence. Moreover, if this metric is set to a large value, smaller receptive fields are generated which may lead to over-fitting. As seen in section 3.4.3, the algorithm takes care of updating the distance metric. However, a good initialization helps simplify the algorithm.

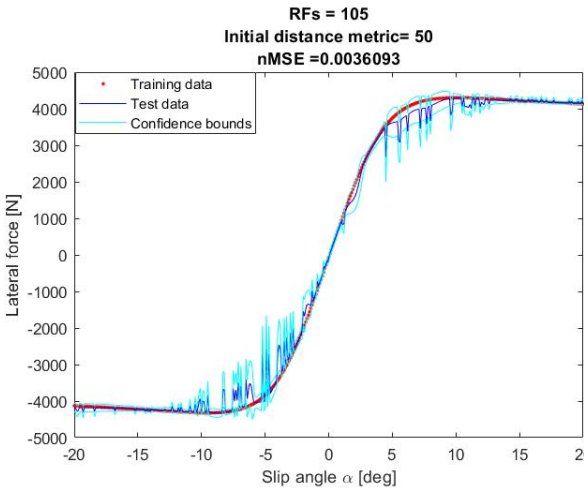

Figure 3.2: Iteration 1

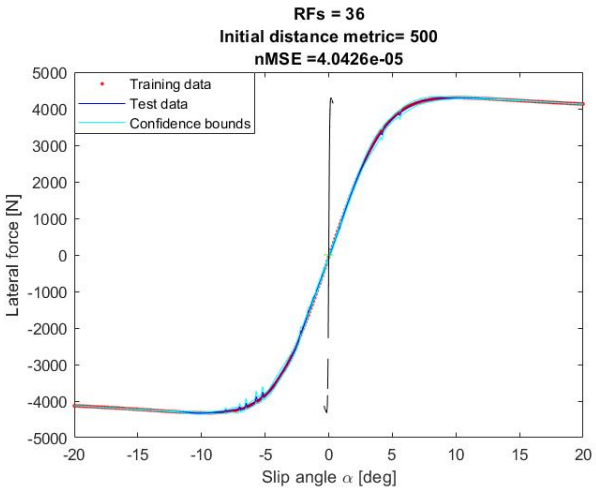

Figure 3.3: Iteration 2

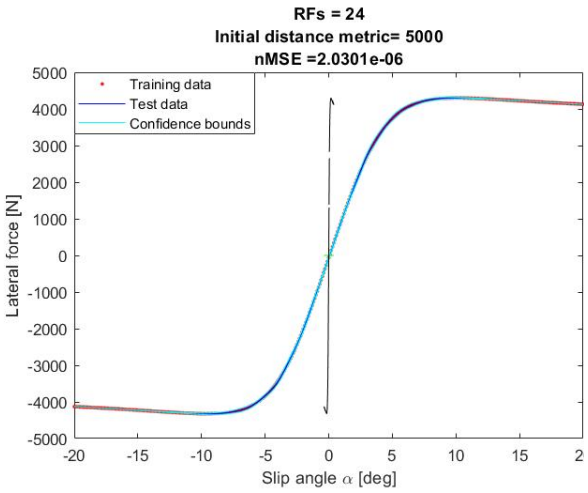

Figure 3.4: Iteration 3

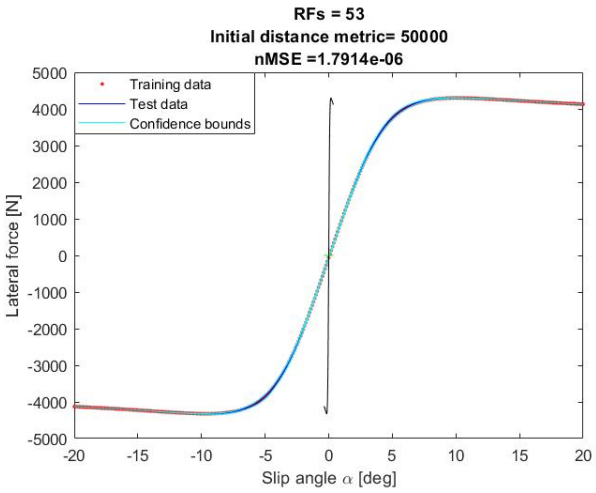

Figure 3.5: Iteration 4

Figure 3.5 has the least mean square error from the training data showing that the LWPR scheme can be used to successfully learn the tire characteristics presented to it. This can be seen by the accurate matching of test data on the training data set. However, it is important to note that with fewer RFs, figure 3.4 shows satisfactory matching with the training data set. The number of training and test data points input into the model was set to 400 data points each. The training data presented to the model was obtained using the TNO-Delft tire model. The parameters used for learning which are kept same for all iterations, are shown in table 3.2. Further tuning of the parameters to achieve best fit was done as per [35].

Parameter	Value
Pre-factor of smoothness penalty term	10^{-4}
Initial element-wise learning rate	40
Kernel	Gaussian

Table 3.2: LWPR parameters for offline learning of pure lateral force $F_y(\alpha)$

This offline learnt model, can then be used within the control scheme to compute the lateral forces for changing slip angles. The cornering stiffness can then be computed from this value. Further, the number of inputs to the model is increased by attempting to reconstruct combined effects of wheel slip (κ) and slip angle (α) on the longitudinal force (F_x).

(c) *Offline learning of tire model - Longitudinal force vs wheel slip and slip angle*

In order to reconstruct combined behaviour, the wheel slip (κ) is varied from -0.5 to 0.5 and the slip angle (α) is varied from -10 to 10 deg. The normal force is kept constant and camber is set to zero. The model is trained using the TNO-Delft tire model. Two separate iterations were performed by increasing the initial distance metric by an order of 10.

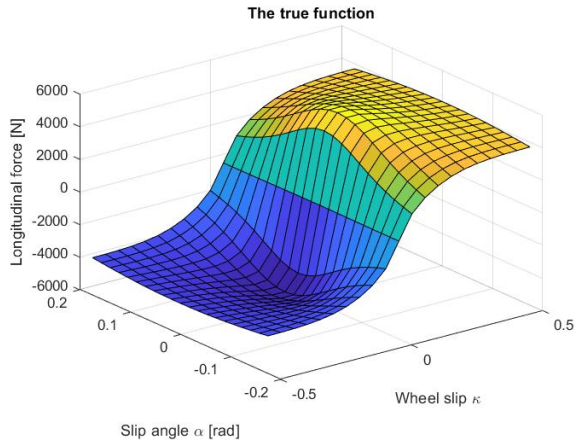

Figure 3.6: Delft-tire model

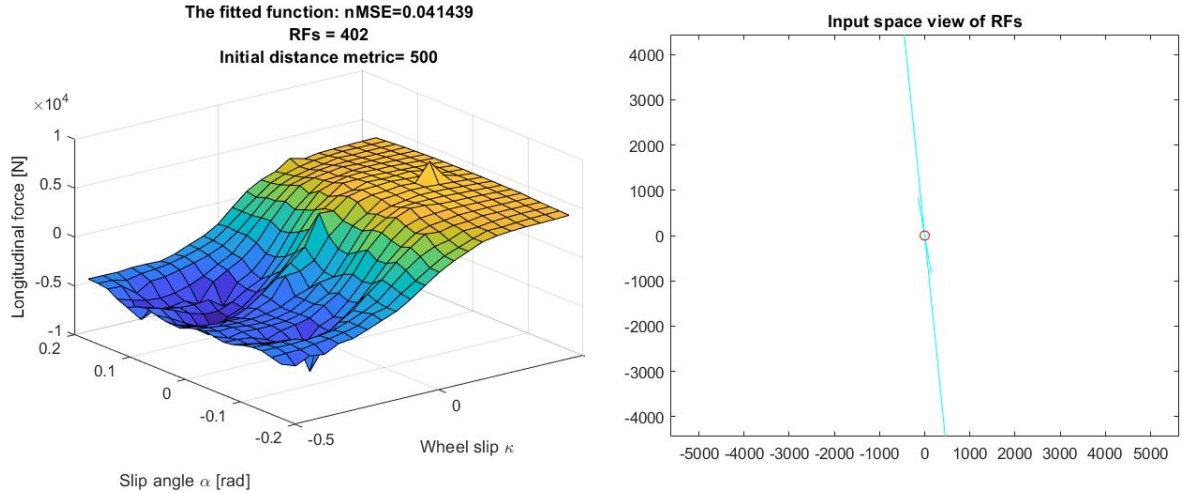

Figure 3.7: Iteration 1

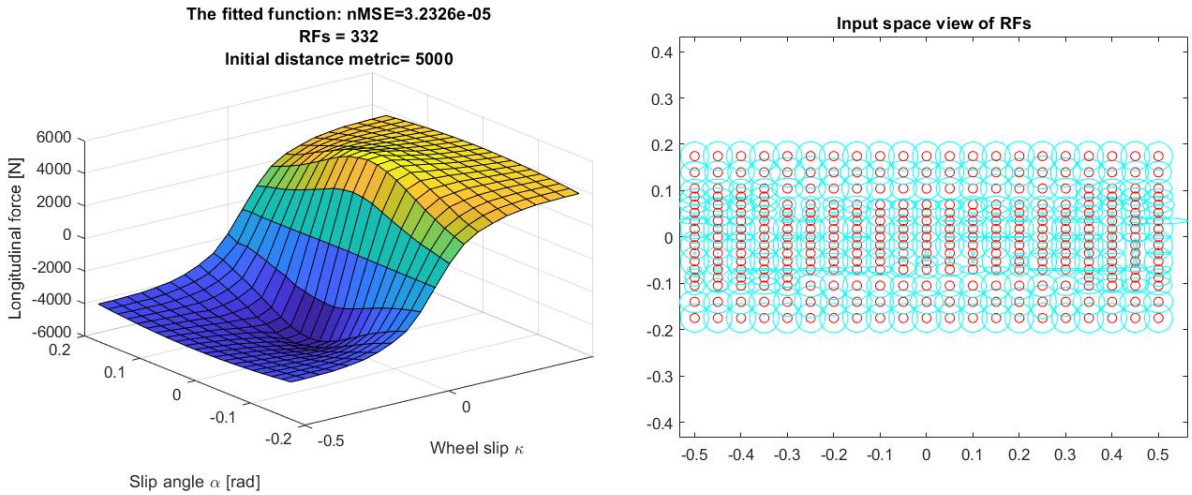

Figure 3.8: Iteration 2

Figures 3.6,3.7 and 3.8 demonstrate the capability of this algorithm to reconstruct complex non-linear tire behaviour. A drastic improvement in terms of mean square error can be seen with the increase in the value of the initial distance metric. 400 data points were fed to the algorithm while training. The parameters used for learning which are kept same for both iterations, are shown in the table 3.3. Further tuning of parameters to achieve better results was done as per [35].

Parameter	**Value**
Initial element-wise learning rate	$50\,\mathbf{I}_2$
w_{gen}	0.2
Learning rate for 2^{nd} order distance metric updates	250
Kernel	Gaussian

Table 3.3: LWPR parameters for offline learning of combined effect on longitudinal force $F_x(\kappa,\alpha)$

2. Online learning

In order to implement online learning, the offline trained model is updated constantly with new information which it needs to "learn". To analyse the online learning capabilities of this method, a badly trained model is first loaded. Further, more data points corresponding to the true lateral force are supplied to the badly trained model. The model is then updated online and the predicted result is compared to the badly trained model without adaptation or learning. This test is expected to portray the learning capabilities of the algorithm.

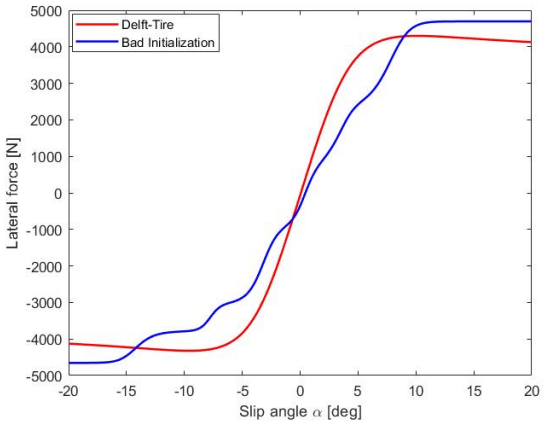

Figure 3.9: Badly trained model initialization

The simulations are carried out for the simple one-dimensional case of pure lateral force. The true model used in the simulations is the Delft tire model. A badly trained model as shown in figure 3.9 is used to initialize the algorithm before learning commences.

Since the model has been trained for a range of -20 to 20 degrees of wheel slip (α), the steering wheel angle input is set to vary sinusoidally within this range throughout the simulation to analyse the learning ability. The steering wheel angle input is set as : $y(t) = 30sin(2\pi ft)$ degrees. The frequency f is initially set to 0.2Hz. The rate at which data is obtained from the true model is set at 100Hz. The LWPR parameters are kept the same in all the simulations to have uniformity. These parameters are set as shown in table 3.4.

Parameter	**Value**
Initial distance metric	$50\,\mathbf{I}_2$
Initial element-wise learning rate	10^{-7}
Pre-factor of smoothness penalty term	10^{-4}
Initial forgetting factor	0.5
Final forgetting factor	0.55
Annealing constant for the forgetting factor	0.8
Kernel	Gaussian

Table 3.4: LWPR parameters for online learning of lateral force behaviour $F_y(\alpha)$

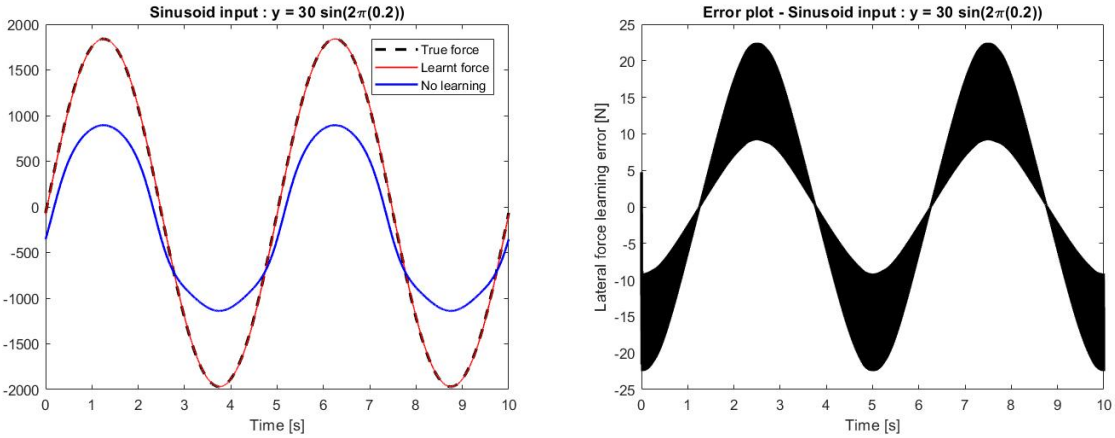

Figure 3.10: Sinusoid of frequency 0.2Hz, Noiseless sensor data at 100Hz

Figure 3.10 shows that LWPR is able to learn satisfactorily well even when initialized with a bad training model. As seen from the error plot, there is a marginal error in the learnt force measurements.

(a) *Effect of noise*

Noise is applied to the input data being made available to the learning module. Since the force data from sensors may contain noise, the effect of noise on sensor data is also evaluated. All other parameters are kept the same as the previous simulation.

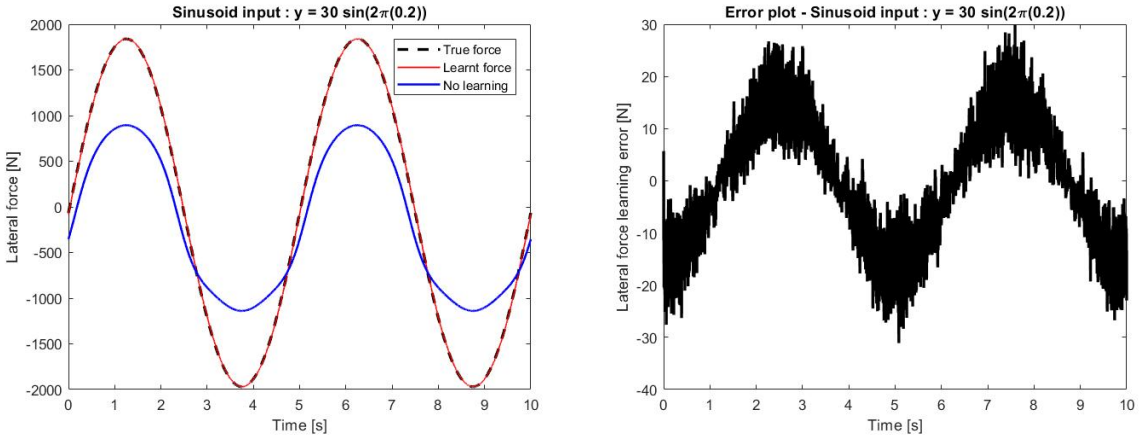

Figure 3.11: Sinusoid of frequency 0.2Hz, Noisy sensor data at 100Hz

Figure 3.11 shows that noise does not have a significant effect on learning performance. The algorithm is able to learn the true model online with reasonable accuracy and a similar marginal error to the noiseless case.

(b) *Effect of input frequency*

The sinusoidal input frequency is increased to 0.8Hz to examine its effect on learning. All other parameters are kept the same during the simulation.

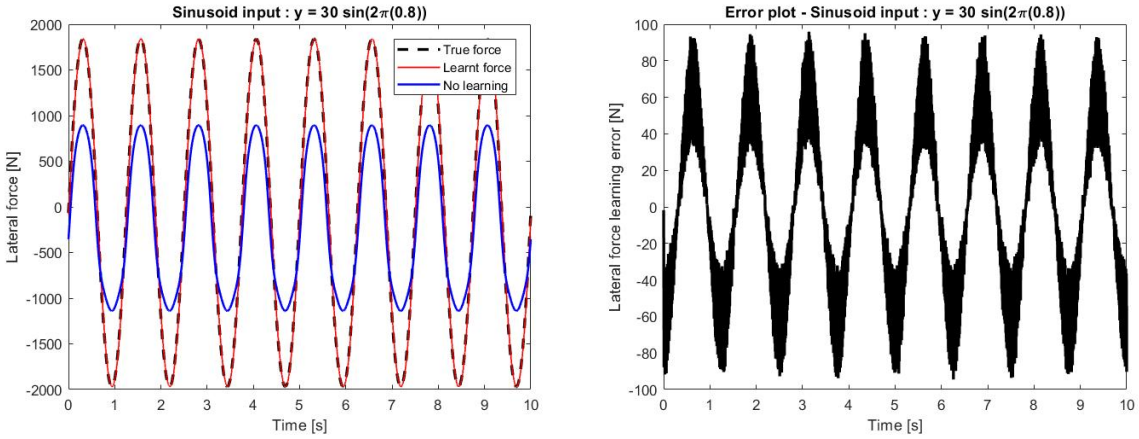

Figure 3.12: Sinusoid of frequency 0.8Hz, Noisy sensor data at 100Hz

The effect of the increased input frequency is clearly evident from the error plot in figure 3.12. The magnitude of error is seen to have increased significantly implying poorer learning performance.

(c) *Effect of data buffer*

The sensor data being made available to the is varied to 1KHz and its consequence on learning performance is analysed. All other parameters are kept the same as previous simulations.

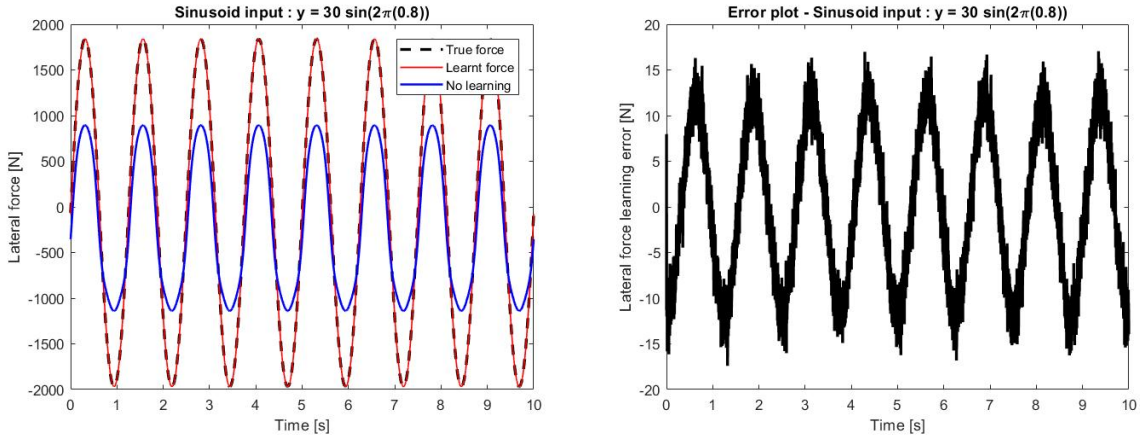

Figure 3.13: Sinusoid of frequency 0.8Hz, Noisy sensor data at 1KHz

Figure 3.13 shows a significant decrease in the magnitude of error enhancing tracking performance.

3. Active learning

For the application of autonomous racing, LWPR has been used with MPC in order to learn the changes in tire force behaviour online. This learnt model is then made available to the prediction model within the MPC. This helps the MPC to solve an optimization problem at each sampling instance and thereby obtaining the optimal control action, **while adapting to varying tire behaviour online**.

Some key trends that were observed while choosing the LWPR parameters [35] are reported below.

- Initial learning rate : A large value leads to instability in convergence, whereas if there is delayed convergence, this value needs to be larger. This learning rate gets updated by the algorithm causing the local models to adapt quickly to the data.
- Penalty term : A small value results in larger distance metrics, which corresponds to narrow receptive fields. This further implies functions that are not smooth. In order to obtain smooth function reconstruction, a larger value needs to be set.
- Initial distance metric : A large initialization may lead to over-fitting due to allotment of a large number of receptive fields. However, if this value is too small, it could result in delayed convergence and possible lead to local minima. Once initialized, this value gets updated by the algorithm.

4

Benchmark Controllers

Details regarding some of the benchmark controllers designed for comparative analysis are discussed in this chapter. Firstly section 4.1 gives the reader some idea behind the design of the Stanley controller for lateral path following. Following this, section 4.2 discusses the idea behind path control with preview. Finally a summary of the benchmark controllers can be found under section 4.3.

4.1. Stanley

The "Stanley" autonomous vehicle won the DARPA grand challenge using steering control based solely on a simple kinematic model [61]. This method is based on a non-linear feedback function of the lateral position error and the heading error. Exponential convergence has been shown for this method [65]. Figure 4.1 showing the error definitions for this method is adapted from previous research [61].

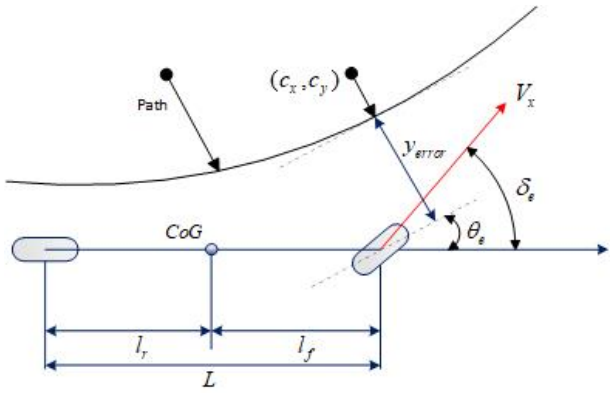

Figure 4.1: Error definitions : Stanley controller

The error metrics are the lateral position error (y_{error}) which is the distance from the centre of the front axle to the nearest point on the path (c_x, c_y) and the heading error (θ_e) which is the difference between the vehicle's actual heading angle and the heading angle of the nearest point on the path (c_x, c_y). Two gain parameters K_{ye} and $K_{\psi e}$ are tuned to get desirable results. The steering control action (δ_e) is given by equation (4.1).

$$\delta_e = arctan(K_{y_e} \frac{y_{error}}{V_x}) + K_{\psi_e} \theta_e \tag{4.1}$$

4.2. Path control with preview

A conclusion can be made based on previous research, that lateral path following control must include a feedback term that is either designed with a lateral position and heading error [5] or with the displacement in the lateral direction computed at a particular look-ahead distance [4] and a feedforward term in order to decrease the effect of disturbances. A generic combination of these two ideas is proposed in [56].

The control objective of this technique is to ensure that the vehicle centre of gravity follows the path while maintaining the heading angle of the vehicle along the tangent to the path [56]. The lateral position error (y_e) and the heading error (ψ_e) are defined as shown in figure 4.2, adapted from [56]. Some additional parameters include longitudinal velocity (V_x), understeer gradient (K_{us}), curvature ($\kappa = \frac{1}{R}$), wheelbase (L), distance from vehicle centre of gravity to rear axle (l_r) and look-ahead distance (x_{la}).

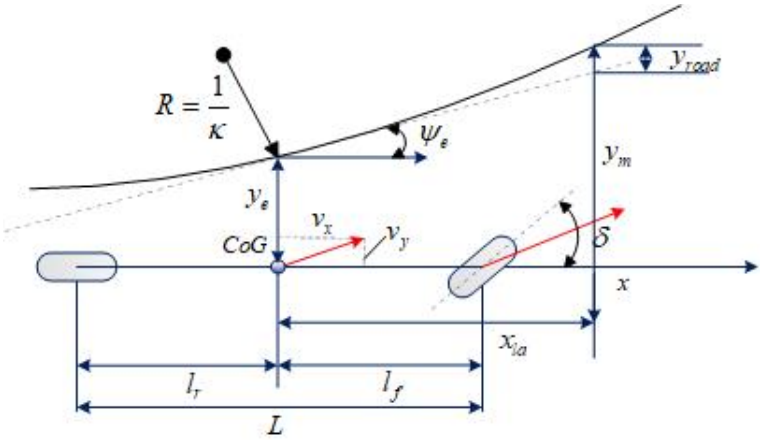

Figure 4.2: Error definitions : Path control with preview

The look-ahead distance can be expressed in terms of the look-ahead time (t_{la}) as shown in equation (4.2).

$$x_{la} = V_x t_{la} \tag{4.2}$$

The general steering control action (δ) is split into a feedforward (δ_{ff}) and a feedback (δ_{fb}) part as per equation (4.3).

$$\delta = \delta_{ff} + \delta fb \tag{4.3}$$

According to the concepts of gain determination reported in [40], the two parts of the control law is as per equations (4.4) and (4.5).

$$\delta_{ff} = \frac{(K_{us}V^2 + L)}{R} \tag{4.4}$$

$$\delta_{fb} = \frac{2(K_{us}V^2 + L)}{(V t_{la} + l_r)^2}(y_e + (V t_{la}\psi_e)) \tag{4.5}$$

4.3. Summary

Since this technique is based on geometric model, the Stanley controller is one of the simplest and intuitive control methods for lateral path following. This technique however is not suitable for aggressive maneuvers and is not robust in handling large deviations in the path. It must be kept in mind that this controller can be tuned specifically for a particular path by having high values for the gains which may prove to be unstable for other paths. Since this technique does not have a look-ahead, an over-tuned Stanley controller may overshoot during cornering.

The path control with preview takes into account the look-ahead distance as well as has information regarding the desired heading angle. This makes this technique robust for different paths at moderate velocities. Choosing the look-ahead distance while tuning is not very straightforward. This parameter is varied based on different longitudinal velocities, since the look-ahead distance is dependant on the velocity with which the vehicle is following the path. Over-tuning is an issue with this control technique also. Changing the look-ahead distance varies the turn radius and may lead to corner-cutting. A trade off has to be made between tuning this parameter to ensure stability (large look-ahead distance) while sacrificing on performance.

5

Conclusion and Recommendations

The results of this research project demonstrates that MPC with online learning using LWPR is an interesting and effective solution to the autonomous racing problem. A comparative study shown in part I, demonstrates the effectiveness of this combination in comparison to the other controllers. The scientific article also shows the real-time applicability of the learning based MPC, which is validated by running the simulations successfully on Scalexio.

Chapter 1 provides the reader with an introduction to the research project with the necessary context. Following this, chapter 2 gives some background regarding MPC design. Next, chapter 3 presents the tire models used and discusses the online learning algorithm proposed in this project. Finally, the benchmark controllers used for comparison are presented in chapter 4.

5.1. Future Work

The designed MPC can be extended to two control inputs, one for lateral direction and the other for the longitudinal. This can be made possible by dividing the entire design problem into two sub-problems. Firstly, a longitudinal control sub-problem where the MPC computes an optimal speed profile for a given circuit. Secondly, a lateral control sub-problem where the MPC provides lateral control in order to follow a dynamic racing line reference, while ensuring the vehicle remains within the circuit boundaries.

Since the racing scenario exploits the vehicle to its limits of handling, designing control techniques for autonomous racing could potentially be useful for autonomous passenger vehicles of the future. Apart from passenger vehicles, learning algorithms can be applied to other domains. With growing interest in automation, and the effectiveness of learning algorithms in repetitive scenarios, new avenues for applying this algorithm are emerging. Agricultural vehicles that are required to move in specified paths [38] could make use of such learning based path following algorithms. Apart from this, other applications could be for vehicles in warehouses or in an industrial setting. Shuttle services and other public modes of transport that take people to and from specified locations along fixed paths could also be a potential application for learning based path following.

III

Previous Work

Tyre Force Reconstruction: Methods and Case Study

Kunal Iyer
Department of Cognitive Robotics
Delft University of Technology
Delft, Netherlands
V.K.Iyer@student.tudelft.nl

Barys Shyrokau
Department of Cognitive Robotics
Delft University of Technology
Delft, Netherlands
b.shyrokau@tudelft.nl

Valentin Ivanov
Automotive Engineering Group
Technische Universität Ilmenau
Ilmenau, Germany
valentin.ivanov@tu-ilmenau.de

***Abstract*—This paper provides an analysis of methods used in automotive control applications for finding the tyre forces. An attention is given to three main classes of relevant methods: tyre-model-based, tyre-model-free, and sensor-based. After analysis of advantages and disadvantages of each class, an original application of the approach based on locally weighted projection regression (LWPR) is discussed. This approach can find combined use for both model-free and sensor-based tyre force reconstruction.**

***Index Terms*—tyre, vehicle dynamics, vehicle control, tyre model, locally weighted projection regression**

I. Overview of Methods for Tyre Force Reconstruction

A. Introduction

The problem of tyre force reconstruction belongs to one of the most important tasks by designing the vehicle motion control systems. To ensure proper control on the vehicle safety, comfort, driving efficiency and other functions, corresponding on-board systems should handle in real-time the information about longitudinal, lateral and vertical tyre forces. This can be done either by use of corresponding state observers or by direct measurement of tyre forces. The latest option is definitely more advantageous for the system design from practical viewpoint. However, sensor technologies for tyre forces and torques have still various technological obstacles for the use on mass-production vehicles. Available solutions in this area are therefore mainly implemented on experimental and test vehicles. As a result, the observation remains as the main tool for tyre force reconstruction in the automotive controllers. Here two main techniques can be identified: tyre-model-based and tyre-model-free. Next sections provide an overview for the most typical solutions for each class of the methods.

B. Tyre-model-based Force Reconstruction

The tyre-model-based force reconstruction represents a virtual tyre sensor, which observer is using one or another tyre models. Considering requirements to real-time operation of virtual tyre sensors, highly precise and complex tyre models are finding less application here, and the priority is given to semi-empirical and sufficiently simplified physical models of tyre-surface interaction. As applied to longitudinal and lateral tyre forces, widely-accepted techniques cover predominantly the Dugoff tyre model (linear and non-linear variants) [1], [2], [3] and the Magic Formula tyre model [4]. Few studies are also proposed the solutions with other models of different complexity as the Burckhardt model [5], the *arctan*-function-based model [6], and TMeasy [7].

In most cases the variations of the Kalman filter (extended, unscented and dual) are applied as the corresponding estimation tools. They require a priori knowledge about relevant tyre states as tyre longitudinal slip, tyre lateral slip, vertical load et al. This causes certain limitations of this method. Firstly, not all the states can be directly measured with the conventional automotive on-board systems and, as a result, extra estimators can be required, e.g. for the slip-related parameters. Secondly, tyre forces are influenced by many operational factors, for instance, by tyre inflation pressure, contact temperature, and road surface roughness. Despite the consideration of these and another factors reduces the tyre model uncertainty, this increases the model complexity.

C. Tyre-model-free Force Reconstruction

The limitations of the previous class have motivated many studies, where a tyre model is not required for the force reconstruction. One of the main ideas behind such tyre-model-free approach is to emulate the tyre forces as random variables. These variables can then be embedded into a model-based state estimator of the vehicle planar dynamics, which is used by the vehicle motion controller.

The work [8] proposes to identify three methods for building the tyre-model-free reconstruction of the longitudinal force: based on vehicle dynamics, based on wheel rotation dynamics, and stochastic. The methods based on vehicle dynamics use predominantly the vehicle acceleration sensor information and GPS signals to derive the forces from the longitudinal force balance equations with consideration of driving resistance parameters [9], [10]. The methods based on the wheel rotation dynamics require the information from the wheel speed sensors to reconstruct the longitudinal tyre force, for example, from wheel torque balance equations. The corresponding examples can be found in [11], [12] and [13]. Finally, by the stochastic methods, the forces are mainly interpreted as random-walk or "black-box" variables and can be derived using different vehicle planar models [14], [15]. As

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 734832.

for the lateral force reconstruction, both stochastic and vehicle-dynamics-based methods can be also applied.

The tyre-model-free force reconstruction is especially beneficial in the case of uncertain road friction parameters because an estimation of a corresponding friction scaling factor is not required to correct the reconstructed tyre forces. Nevertheless, this approach can have limitations in terms of complex tuning and computations costs.

D. Sensor-based Force Reconstruction

The force sensing technology has been intensively studied for last decades to develop accurate, robust and inexpensive solution. Due to the fact that transmission of forces and moments affects all components between tire road contact and vehicle body, all components carrying the load can be used to force/torque measurement and reconstruction, e.g. tyre, rim, bearing and suspension. Several approaches and their limitations are discussed below:

a) Suspension bushing deformation: the forces transmitted through the suspension bushing can be reconstructed via direct deformation measurement using eddy-current displacement sensors [16] or the estimation of the relative bushing deformation based on acceleration measurement [17]. The main drawbacks are complexity of the approach and the durability of the bushings.

b) Deformation between knuckle and brake calipers: this approach is based on the installation of a sensor bracket with strain resistance elements between knuckle and brake calipers to measure brake torque [18]. Only brake torque can be reconstructed in such approach and the method performance is temperature dependent.

c) Wheel force transducer: force/torque measurement is performed by strain measurements in the wheel rim [19]. Only this method is currently applied for commercial products (Kistler, MTS, Michigan Scientific Corporation , etc.) Although it provides high accuracy and bandwidth, the application of wheel force transducer as a standard vehicle sensor is too expensive even for premium class vehicles.

d) Tyre sidewall deformation: tyre deformation measurement can be used to reconstruct force using an optical position detection sensor [20], laser-based sensor system [21], a passive surface acoustic wave sensor [22] or by combination of a Hall sensor and magnet [16]. The common drawback of these approaches is the necessary adjustment and calibration after tyre replacement.

e) Tyre inner liner accelerometer: a MEMS accelerometer is located and fixed to the inner liner of the tyre [23]. The reconstruction of longitudinal and normal forces is demonstrated both in laboratory and road test conditions [24]. Furthermore also the measurement of lateral contact forces was recently demonstrated [25]. The common limitations are discontinuous measurement signal and durability of the approach due to the relatively short lifetime of tyres.

f) Bearing displacement or deformation based: Using the wheel-end bearing two principally different approaches can be applied. The first approach is displacement based (relative inner- to outer-ring displacement) using Hall effect [26], eddy-current sensors [27] or capacitive [28] sensors. Its limitation is that a limited number of loads can be reconstructed. The second approach is to measure outer-ring deformation using strain gauges [29], [30]. The measured strain should be translated to the bearing loading using empirical methods, e.g. least squares fitting or artificial neural networks. However, due to nonlinear behaviour, such translation is nontrivial and significantly affects accuracy of force reconstruction. Instead of empirical methods, the model-based approach for the estimation of bearing forces is proposed [31] using a cascaded extended and unscented Kalman filtering. An experimental study covering both laboratory and field tests showed that the model-based approach led to accurate load estimates in various conditions and outperforms the data-driven methods.

E. Summarising Remarks

The introduced short overview of basic approaches for the tyre force reconstruction demonstrates a variety of tools, which can be used in vehicle motion control systems. Taking into account such factors as uncertainties of tyre models as well as for demand on extensive test procedures for proper parameterisation of tyre models, the model-free reconstruction can be considered as a more advantageous candidate. An interesting advancement can be proposed in the case of development of hybrid approaches, where the same analytical base can be used both for the model-free and sensor-based reconstruction. Here it makes sense to apply not only conventional stochastic methods but also another variants of computational intelligence tools. One example of such an approach is discussed in next section.

II. Case study : Locally weighted projection regression

Locally weighted projection regression (LWPR) is an algorithm that supports non-linear function approximation in high dimensional spaces [32]. The nonlinear system behaviour, especially steady state, can be accurately captured by using this technique.

The key idea of this method is to approximate non-linear functions by using piece-wise linear models. The features of LWPR are numerical robustness in high dimensional spaces and the capability to perform incremental online learning with the predefined learning rate.

A. The LWPR algorithm [32]

A weighting kernel used to determine the locality is defined in the way that computes a weight $w_{k,i}$ for each data point (x_i, y_i) corresponding to the distance from the centre c_k of the kernel within each local unit. Usually, a gaussian kernel is chosen,

$$w_{k,i} = exp(-\frac{1}{2}(x_i - c_k)^T D_k (x_i - c_k)), \qquad (1)$$

where D_k is the distance metric that influences the size and shape of the region of validity or receptive field (RF). It is assumed that there are K locally linear models that are

combined to form the prediction. Each linear model calculates a prediction y_k given an input vector x. The net output is the weighted mean of all the linear models.

Algorithm 1 shows how an incrementally locally weighted variant of partial least squares (PLS) is used to generate linear model parameters within the LWPR scheme. In the algorithm 1, $\lambda \in [0, 1]$ is the forgetting factor that decides amount of the old data of the parameters used in the regression will be forgotten. The PLS predictor adds linear projections in an incremental fashion until the point where adding further projections does not improve the accuracy.

The distance metric D influences the shape and size of each RF and thus also influences the effectiveness of each local model. This distance metric is optimized separately for each RF using an incremental gradient descent based on stochastic leave-one-out cross validation criterion. This is shown in the algorithm 2.

An incremental learning system which embeds the above update laws, and generates additional locally linear models as and when needed is shown in algorithm 3.

B. Current state of the art

The LWPR algorithm has been used to learn the dynamic model of a robot manipulator [33]. In comparison to other classical learning controllers, it was reported that the LWPR provides best performance when there is no a-priori knowledge of the system dynamics. The application of this algorithm for real-time robot learning has been presented in [34]. The results demonstrates the successful application of autonomous learning to complex robotic system. It was also concluded that this technique, using its learning abilities outperforms traditional control techniques. Comparison of the LWPR with a few other regression techniques to estimate the inverse dynamic model of a robot from measured data is demonstrated in [35]. This is mainly done in order to capture non-linearities arising from dynamics of hydraulic cables, actuator dynamics or complex friction dynamics.

Regarding automotive domain, this method has been only applied for the scaled off-road vehicle [36]. Based on the authors' knowledge, this paper is the first application of LWPR for tyre force reconstruction. Taking into account the above-mentioned advantages, it can be an interesting candidate for tyre force reconstruction, especially, regarding steady-state tyre properties. To evaluate the LWPR capability for tyre force reconstruction, two main features should be discussed: (i) capability to reconstruct tyre force based on offline training similar to data-driven learning techniques required a large data set; (ii) capability to learn tyre characteristic online assuming a repeatable track, e.g. racing laps. The following discussion is organized according to these features.

C. Offline learning

1) Pure longitudinal force reconstruction: The algorithm is presented with training data comprising of the longitudinal force F_x behaviour with wheel slip κ. See table I for the parameters set during training. The wheel slip is set to $\kappa \in \begin{bmatrix} -0.5 & 0.5 \end{bmatrix}$ and the algorithm is trained with the corresponding longitudinal force behaviour obtained from the baseline Delft-tyre model. The LWPR model is presented with test data and the result of reconstruction is shown in Fig. 1.

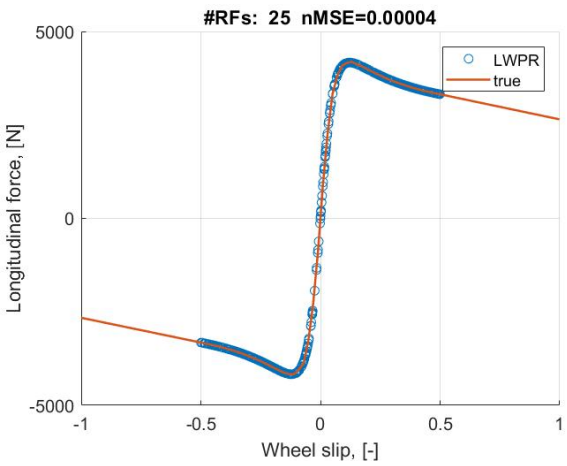

Fig. 1. Longitudinal force reconstruction $F_x(\kappa))$

TABLE I
PARAMETERS FOR LONGITUDINAL FORCE RECONSTRUCTION

Parameter	Value	Unit
Camber	0	rad
Longitudinal velocity	80	km/h
Turnslip	0	1/m
Normal load	4000	N
Initial distance metric (LWPR)	5000	[-]

2) Combined force reconstruction: The combined force behaviour with wheel slip κ, slip angle α and normal load F_z is presented as the training data set to the algorithm. The corresponding lateral and longitudinal force behaviour from the Delft-tyre model is used while training. The LWPR tuning parameters and the ranges for the three inputs (κ, α, F_z) are set as per table II. The results of reconstruction are shown in Fig. 2 and Fig. 3.

TABLE II
PARAMETERS FOR COMBINED FORCE RECONSTRUCTION

Parameter	Value	Unit
Camber	0	rad
Longitudinal velocity	80	km/h
Turnslip	0	1/m
Normal load	[2000 8000]	N
Wheel slip	[-0.5 0.5]	[-]
Slip angle	[-15 15]	deg
Initial distance metric (LWPR)	[1e+3 0 0; 0 1e+3 0; 0 0 5e+1]	[-]

D. Online learning

To analyse the online learning capabilities of the LWPR method for tyre force reconstruction, a badly trained model as

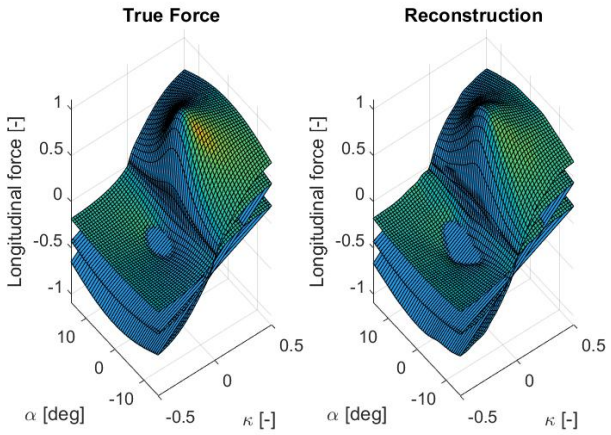

Fig. 2. Longitudinal force reconstruction $F_x(\kappa, \alpha, F_z)$

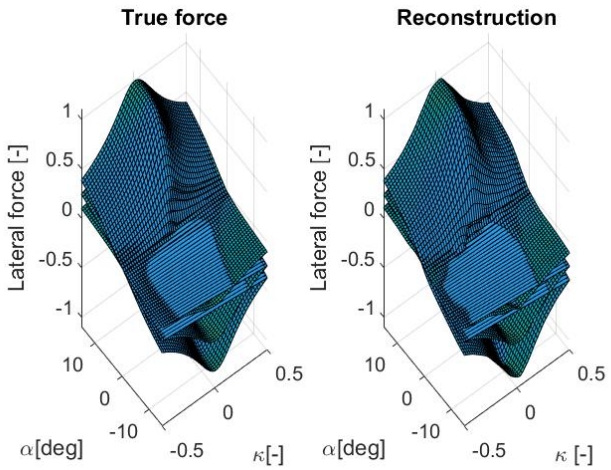

Fig. 3. Lateral force reconstruction $F_y(\kappa, \alpha, F_z))$

shown in Fig. 4 is used to train the LWPR learning module before learning commences. The model is then updated online with the sensing force information (based on the Delft-tyre model) corresponding to the slip angle range used for training and the predicted result is compared to the badly trained model without adaptation or learning. The simulation parameters used can be seen in table III. It can be inferred from Fig. 5 and Fig. 6 that the LWPR algorithm is able to learn the true tyre behaviour online when initialized with a poor tyre model.

A significant improvement in tyre force reconstruction, due to online learning, can be inferred from table IV.

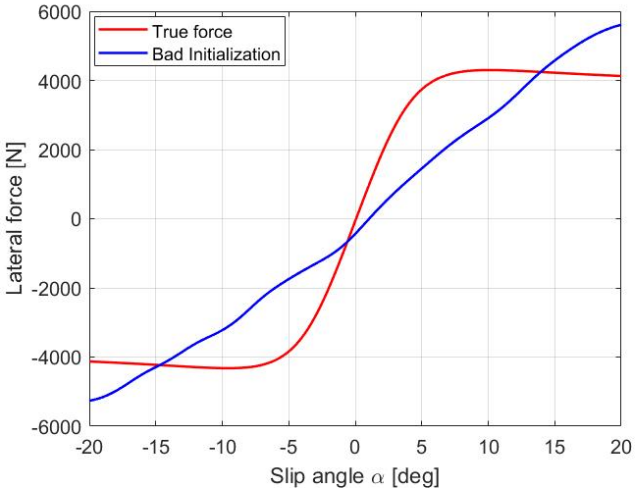

Fig. 4. Initialization for LWPR learning module

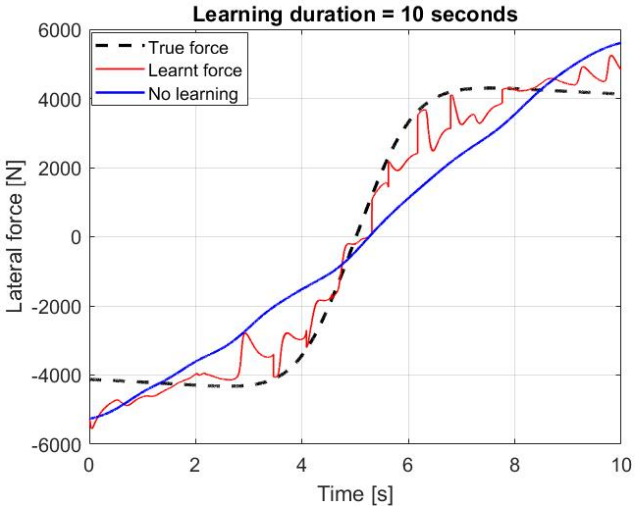

Fig. 5. Online learning of lateral force $F_y(\alpha)$ for 10 seconds

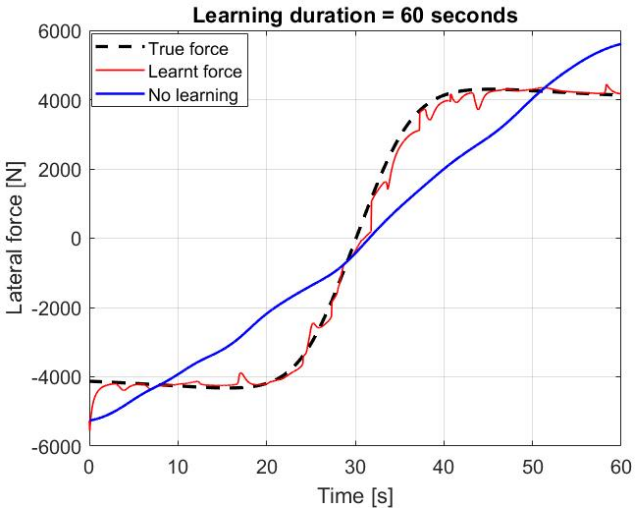

Fig. 6. Online learning of lateral force $F_y(\alpha)$ for 60 seconds

TABLE III
PARAMETERS FOR ONLINE LEARNING OF LATERAL FORCE

Parameter	Value	Unit
Camber	0	rad
Longitudinal velocity	80	km/h
Turnslip	0	1/m
Normal load	5000	N
Wheel slip	0	[-]
Slip angle	[-20 20]	deg
Initial distance metric (LWPR)	5000	[-]
Initial component-wise learning rate (LWPR)	40	[-]
Pre-factor of smoothness penalty (LWPR)	0.01	[-]
True force data input rate	100	Hz

TABLE IV
AVERAGE ERROR DURING ONLINE LEARNING OF LATERAL FORCE

Learning duration [s]	Average Error [N]
10	523.1
60	180.0

E. Future work

A possible application of this algorithm is in autonomous racing. Since it is difficult to predict the tire behavior during the race, online learning of the tyre properties is considered. Throughout the course of the race, the algorithm can continuously update the tyre model in the controller with sensor data, thereby making it adaptive to changing tire behaviour. This is a potential method of improving tracking performance and minimizing lap time for autonomous racing.

III. CONCLUSIONS

The paper discusses various methods for tyre force reconstruction and summarizes three major directions: tyre-model-based, tyre-model-free, and sensor-based approaches. Besides the state-of-the-art overview, the application of a new method named as locally weighted projection regression is considered for tyre force reconstruction. The main advantage of the considered method is a capability to perform learning through both offline and online training. The simulation results demonstrate that the method can be effectively used to learn steady-state tyre characteristics with a high accuracy.

REFERENCES

[1] M. Doumiati, A. Charara, A. Victorino, and D. Lechner, Vehicle Dynamics Estimation using Kalman Filtering, Wiley, 2012.

[2] J. Dakhlallah, S. Glaser, S. Mammar, and Y. Sebsadji, "Tire-road Forces Estimation Using extended Kalman Filter and Sideslip Angle Evaluation," American Control Conference, Seattle, WA, pp. 4597-4602, 2008.

[3] X. Jin X and G. Yin, "Estimation of lateral tire-road forces and sideslip angle for electric vehicles using interacting multiple model filter approach," Journal of the Franklin Institute, vol. 352(2), pp. 686–707, 2015.

[4] S. Antonov, A. Fehn, and A. Kugi, "Unscented Kalman filter for vehicle state estimation," Vehicle System Dynamics, vol. 49(9), pp. 1497–1520, 2011.

[5] G. Baffet, A. Charara, and G. Dherbomez, "An Observer of Tire-Road Forces and Friction for Active Security Vehicle Systems," IEEE/ASME Transactions on Mechatronics, vol. 12, pp. 651–661, 2007.

[6] X. Gao, "Nonlinear Estimation of Vehicle Sideslip Angle Based on Adaptive Extended Kalman filter," SAE Technical Papers, 2010-01-0117, 2010.

[7] T. Wenzel, K. Burnham, M. Blundell, and R.A. Williams, "Dual Extended Kalman Filter for Vehicle State and Parameter Estimation," Vehicle System Dynamics, vol. 44(2), pp: 153–171, 2006.

[8] M. Acosta, S. Kanarachos, and M. Blundell, "Virtual tyre force sensors: An overview of tyre model-based and tyre model-less state estimation techniques," Proc. of the IMechE, Part D: Journal of Automobile Engineering, vol. 232(14), pp. 1883–1930, 2018.

[9] C. R. Carlson and J. C. Gerdes, "Consistent nonlinear estimation of longitudinal tire stiffness and effective radius," IEEE Transactions on Control Systems Technology, vol. 13(6), pp. 1010-1020, 2005.

[10] C.S. Ahn, "Robust Estimation of road friction Coefficient," PhD Thessis, University of Michigan, USA, 2011.

[11] W. Cho, J. Yoon, S. Yim, B. Koo, and K. Yi, "Estimation of Tire Forces for Application to Vehicle Stability Control," IEEE Transactions on Vehicular Technology, vol. 59(2), pp. 638-649, 2010.

[12] A. Albinsson, F. Bruzelius, M. Jonasson, and B. Jacobson, "Tire Force Estimation Utilizing Wheel Torque Measurements and Validation in Simulations and Experiments," Proc. of the 12th International Symposium on Advanced Vehicle Control, p. 294–299, 2014.

[13] M. Acosta, S. Kanarachos, and M. Fitzpatrick, "A Virtual Sensor for Integral Tire Force Estimation using Tire Model-less Approaches and Adaptive Unscented Kalman Filter," Proc. of the 14th International Conference on Informatics in Control, Automation and Robotics, pp. 386-397, 2017.

[14] G. Baffet, A. Charara, and G. Dherbomez, "An Observer of Tire–Road Forces and Friction for Active Security Vehicle Systems," IEEE/ASME Transactions on Mechatronics, vol. 12(6), pp. 651-661, 2007.

[15] A. Turnip and H. Fakhrurroja, "Estimation of the Wheel-Ground Contact Tire Forces using Extended Kalman Filter," International Journal of Instrumentation Science, vol. 2(2), pp. 34–40, 2013.

[16] M. Suzuki, K. Nakano, A. Miyoshi, A. Katagiri, and M. Kunii, "Method for sensing tire force in three directional components and vehicle control using this method," SAE Technical Papers, 2007-01-0830, 2007.

[17] C. K. Chae, B. K. Bae, K. J. Kim, J. H. Park, and N. C. Choe, "A feasibility study on indirect identification of transmission forces through rubber bushing in vehicle suspension system by using vibration signals measured on links," Vehicle System Dynamics, vol. 33(5), pp. 327-349, 2000.

[18] N. Ohkubo, T. Horiuchi, O. Yamamoto, and H. Inagaki, "Brake torque sensing for enhancement of vehicle dynamics control systems," SAE Technical Papers, 2007-01-0867, 2007.

[19] W. Weiblen, D. Barz, W. Evers, M. Herrmann, P. Wolfer, and J. Reichel, "Integrated Wheel Dynamometer Technology for Vehicle and Bench Testing," SAE Technical Papers, 2003-01-0194, 2003.

[20] A. Tuononen, "On-board estimation of dynamic tyre forces from optically measured tyre carcass deflections," International Journal of Heavy Vehicle Systems, vol. 16(3), pp. 362-378, 2009.

[21] Y. Xiong, and A. Tuononen, "A laser-based sensor system for tire tread deformation measurement," Measurement Science and Technology, vol. 25(11), pp. 115103, 2014.

[22] A. Pohl, R. Steindl, and L. Reindl, "The intelligent tire utilizing passive SAW sensors measurement of tire friction," IEEE Transactions on Instrumentation and Measurement, vol. 48(6), pp. 1041-1046, 1999.

[23] H. Lee and S. Taheri, "Intelligent tires? A review of tire characterization literature," IEEE Intelligent Transportation Systems Magazine, vol. 9(2), pp. 114-135, 2017.

[24] F. Cheli, G. Audisio, M. Brusarosco, F. Mancosu, D. Cavaglieri, and S. Melzi, "Cyber Tyre: A Novel Sensor to Improve Vehicle's Safety," SAE Technical Papers, 2011-01-0990, 2011.

[25] F. Cheli, D. Ivone, and E. Sabbioni, "Smart Tyre Induced Benefits in Sideslip Angle and Friction Coefficient Estimation," Sensors and Instrumentation, vol. 5, pp. 73-83, 2015.

[26] K. Nam, "Application of novel lateral tire force sensors to vehicle parameter estimation of electric vehicles," Sensors, vol. 15(11), pp. 28385-28401, 2015.

[27] K. Ayandokun, P. Orton, N. Sherkat, and P. Thomas, "Smart bearings: developing a new technique for the condition monitoring of rotating machinery," Proc. of the IEEE Inter. Conference n Intelligent Engineering Systems, pp. 505-510, 1997.

[28] L. Rasolofondraibe, B. Pottier, P. Marconnet, and X. Chiementin, "Capacitive sensor device for measuring loads on bearings," IEEE Sensors Journal, vol. 12(6), pp. 2186-2191, 2012.
[29] K. Nishikawa, "Hub Bearing with Integrated Multi-axis Load Sensor," NTN Technical Review, vol. 79, 2011.
[30] S. Kerst, B. Shyrokau and E. Holweg, "Wheel force measurement for vehicle dynamics control using an Intelligent Bearing," Proc. of 13th Inter. Symposium on Advanced Vehicle Control, pp. 13-16, 2016.
[31] S. Kerst, B. Shyrokau and E. Holweg, "A Model-based approach for the estimation of bearing forces and moments using outer ring deformation," IEEE Transactions on Industrial Electronics, vol. 67(1), pp. 461-470, 2019.
[32] S. Vijayakumar and S. Schaal, "Locally Weighted Projection Regression: An O(n) Algorithm for Incremental Real Time Learning in High Dimensional Space," Proc. of the 17th Inter. Conference on Machine Learning, pp. 288-293, 2000.
[33] J. S. D. L. Cruz, E. Calisgan, D. Kulić, W. Owen, and E. A. Croft. "Online dynamic model learning for manipulator control," IFAC Proceedings, vol. 45(22, pp. 869-874, 2012.
[34] S. Schaal, C. Atkeson, and S. Vijayakumar, "Scalable techniques from non-parametric statistics for real time robot learning," Applied Intelligence, vol. 17(1), pp. 49-60, 2002.
[35] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Model learning with local gaussian process regression," Advanced Robotics, vol. 23(15), pp. 2015–2034, 2009.
[36] G. Williams, B. Goldfain, J. M. Rehg and E. A. Theodorou, "Locally Weighted Regression Pseudo-Rehearsal for Online Learning of Vehicle Dynamics," arXiv preprint arXiv:1905.05162, 2019.

APPENDIX

Algorithm 1 Incremental PLS

Given : A training point (x, y)
Update the means of input and output :
$x_0^{n+1} = \frac{\lambda W^n x_0^n + wx}{W^{n+1}}$
$\beta_0^{n+1} = \frac{\lambda W^n \beta_0^n + wy}{W^{n+1}}$
where $W^{n+1} = \lambda W^n + w$ and $x_0^0 = u_i^0 = \beta_0^0 = W^0 = 0$
Update the local model :
1. Initialize : $\mathbf{z} = x$,$res_1 = y - \beta_0^{n+1}$
2. For $i = 1 : r$
(i) $u_i^{n+1} = \lambda u_i + wzres_i$
(ii) $s = z^T u_i^{n+1}$
(iii) $SS_i^{n+1} = \lambda SS_i^n + ws^2$
(iv) $SR_i^{n+1} = \lambda SR_i^n + wsres_i$
(v) $SZ_i^{n+1} = \lambda SZ_i^n + wzs$
(vi) $\beta_i^{n+1} = \frac{SR_i^{n+1}}{SS_i^{n+1}}$
(vii) $p_i^{n+1} = \frac{SZ_i^{n+1}}{SS_i^{n+1}}$
(viii) $z = z - sp_i^{n+1}$
(ix) $res_{i+1} = res_i - s\beta_i^{n+1}$
(x) $MSE = \lambda MSE_i^n + wres_{i+1}^2$
Predicting with novel data :
Initialize : $y = \beta_0, z = x - x_0$
For i = 1:k
(i) $s = u_i^T z$
(ii) $y = y + \beta_i s$
(iii) $z = z - sp_i^n$

SS,SR and *SZ* are memory terms that help perform univariate regression using recursive least squares as shown in step (vi) in the model update. Step (vii) helps regress the projection from the current input data **z** and the current projected data s. This ensures that u_{i+1} is orthogonal to u_i. There are two important properties of the local projection scheme. Firstly, if we have statistically independent input variables, PLS takes only a single iteration to find the optimal projection direction u_i. This corresponds to the gradient of the locally linear function to be approximated. Secondly, step (i) in the model update in algorithm 1 ensures that the projection direction is chosen by correlating the input and output data. This results in the automatic exclusion of input dimensions that do not contribute to the output. Finally, since the univariate regressions will never be singular, there is no concern of numerical problems in PLS.

Algorithm 2 Distance metric update

$D = M^T M$, where M is upper triangular
$M^{n+1} = M^n - \alpha \frac{\delta J}{\delta M}$ where the cost function to be minimized is chosen to be,
$J = \frac{1}{W} \sum_{i=1}^{M} \sum_{k=1}^{r} \frac{w_i res_{k+1,i}^2}{1 - w_i \frac{s_{k,i}}{s_k^T W s_k}} + \gamma \sum_{i,j=1}^{N} D_{ij}^2$

The first term in the cost function represents the mean leave-one-out cross-validation error of the local model. The second term is a penalty term which ensures that the receptive fields do not shrink in case of huge amounts of training data.

Algorithm 3 LWPR Outline

1. Initialize the LWPR with no receptive fields
2. **For** every new training sample (x, y)
- **For** $k = 1 : RF$
(i) Calculate the activation from eq. (1)
(ii) Update according to algorithms 1 and 2
- **End**
- **If** no linear model was activated by more than w_{gen}, create a new RF with $r = 2, c = x, D = D_{def}$
- **End**
End

The major assignment within the LWPR framework consists of determining the number of local models k, computing the regression coefficient β_k and the weight w_k for the k^{th} locally linear model. Additionally, it consists of regulating the local model's receptive field. In the algorithm 3, a threshold w_{gen} is predefined. This determines when to create new receptive fields. The closer this value is to 1, the more overlap local models will have, but will be more costly to compute. The distance metric D is initialized to D_{def} and is usually diagonal.

Bibliography

[1] *Delft-Tyre manual, TASS International*, 2013.

[2] Isidori A. Nonlinear control systems. 26:2401–2417, 1989. ISSN 1049-8923. doi: 10.1002/rnc.3454.

[3] Carlo Ackermann, Jakob Bechtloff, and R. Isermann. Collision avoidance with combined braking and steering. *6th International Munich Chassis Symposium 2015 Proceedings*, page 199–213, 2015. doi: 10.1007/978-3-658-09711-0_16.

[4] M. Alirezaei, Matteo Corno, Ali Ghaffari, and Reza Kazemi. A new approach to the design of coordinated road departure avoidance systems. *Proceedings of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics (SAGE), (Impact factor: 0.566)*, 226:45–60, 08 2011. doi: 10.1177/1464419311421285.

[5] M. Alirezaei, Sven Jansen, A. Schmeitz, and A. K. Madhusudhanan. *Collision avoidance system using state dependent Riccati equation technique: An experimental robustness evaluation: Proceedings of the 13th International Symposium on Advanced Vehicle Control (AVEC' 16), Munich, Germany, 13–16 September 2016*, pages 127–132. 12 2016. ISBN 978-1-138-02992-7. doi: 10.1201/9781315265285-21.

[6] M.k. Aripina, Y. M. Sam, A. D. Kumeresan, M.f. Ismail, and Peng Kemao. A review on integrated active steering and braking control for vehicle yaw stability system. *Jurnal Teknologi*, 71(2), 2014. doi: 10.11113/jt.v71.3728.

[7] Anil Aswani, Humberto Gonzalez, S. Shankar Sastry, and Claire Tomlin. Provably safe and robust learning-based model predictive control, 2011.

[8] Moore Atkeson and Schaal. Locally weighted learning. pages 11–73, 1997. doi: 10.1007/978-94-017-2053-3_2.

[9] Alberto Bemporad and Manfred Morari. Robust model predictive control : A survey. September . doi: 10.1007/bfb0109870Âă.

[10] Maximilian Brunner, Ugo Rosolia, Jon Gonzales, and Francesco Borrelli. Repetitive learning model predictive control: An autonomous racing example. pages 2545–2550, 12 2017. doi: 10.1109/CDC.2017.8264027.

[11] S. D. Bruyne. Model-based control of mechatronic systems. 16:1425–1436, 2013. ISSN 1561-8625. doi: 10.1002/asjc.863.

[12] V. Cerone, D. Piga, and D. Regruto. Set-membership lpv model identification of vehicle lateral dynamics. *Automatica*, 47(8):1794–1799, 2011. ISSN 0005-1098. doi: 10.1016/j.automatica.2011.04.016.

[13] Joseph Sun De La Cruz, Dana Kulić, and William Owen. A comparison of classical and learning controllers. *IFAC Proceedings Volumes*, 44(1):1102–1107, 2011. doi: 10.3182/20110828-6-it-1002.03279.

[14] T. F. Edgar D. E. Seborg and D. A. Mellichamp. Process dynamics and control, ch. 20: Model predictive control,. 36:269–271, 2015. ISSN 0143-2087. doi: 10.1002/oca.2167.

[15] Vishnu R. Desaraju and Nathan Michael. Experience-driven predictive control. 2016. doi: 10.1109/icra.2017.7989625.

[16] Vishnu R. Desaraju and Nathan Michael. Leveraging experience for computationally efficient adaptive nonlinear model predictive control. 2017. doi: 10.1109/icra.2017.7989625.

[17] Howard Dugoff, P. S. Fancher, and Leonard Segel. An analysis of tire traction properties and their influence on vehicle dynamic performance. *SAE Transactions*, 79:1219–1243, 1970. ISSN 0096736X, 25771531. URL http://www.jstor.org/stable/44644491.

[18] Vijayakumar S. D'Souza, A. and S. Schaal. Are internal models of the entire body learnable? 408:2353–2363, 2010. ISSN 0035-8711. doi: 10.1111/j.1365-2966.2010.17285.x.

[19] B. Sprenger E. Burdet and A. Codourey. Experiments in nonlinear adaptive control. doi: 10.1109/robot.1997.620092.

[20] Timm Faulwasser, Janine Matschek, Pablo Zometa, and Rolf Findeisen. Predictive path-following control: Concept and implementation for an industrial robot. *2013 IEEE International Conference on Control Applications (CCA)*, 2013. doi: 10.1109/cca.2013.6662755.

[21] Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. 2017. doi: 10.1109/icra.2017.7989324.

[22] Z. Ghahramani and M Beal. Variational inference for bayesian mixtures of factor analysers. 1:793–831, 2006. ISSN 1936-0975. doi: 10.1214/06-ba126.

[23] Brian Goldfain James M Rehg Grady Williams, Paul Drews and Evangelos A Theodorou. Information-theoretic model predictive control: Theory and applications to autonomous driving. 34:1603–1622, 2018. ISSN 1552-3098. doi: 10.1109/tro.2018.2865891.

[24] James M. Rehg Grady Williams, Brian Goldfain and Evangelos A. Theodorou. Locally weighted regression pseudo-rehearsal for online learning of vehicle dynamics. 2017. doi: 10.1109/icra.2017.7989202.

[25] Sergio grammatico. Model predictive control (sc42125) lecture notes - lecture 01 : Introduction to mpc. *Delft University of Technology*, 2018-19.

[26] A. Gray, Yiqi Gao, T. Lin, J. K. Hedrick, H. E. Tseng, and F. Borrelli. Predictive control for agile semi-autonomous ground vehicles using motion primitives. *2012 American Control Conference (ACC)*, 2012. doi: 10.1109/acc.2012.6315303.

[27] Jinghua Guo, Ping Hu, and Rongben Wang. Nonlinear coordinated steering and braking control of vision-based autonomous vehicles in emergency obstacle avoidance. *IEEE Transactions on Intelligent Transportation Systems*, 17(11):3230–3240, 2016. doi: 10.1109/tits.2016.2544791.

[28] Nijmeijer H and van der Schaft AJ. Nonlinear dynamical control systems. 14:283–288, 1990. ISSN 1931-4973. doi: 10.1002/tee.22807.

[29] Ross A Knepper Ian Lenz and Ashutosh Saxena. Deepmpc: Learning deep latent features for model predictive control. 2015. doi: 10.15607/rss.2015.xi.012.

[30] Ronald S. Fearing Pieter Abbeel Sergey Levine Ignasi Clavera, Anusha Nagabandi and Chelsea Finn. Learning to adapt: Meta-learning for model-based control. 2018. doi: 10.1109/icra.2018.8463189.

[31] K. Iyer, B. Shyrokau, and V. Ivanov. Offline and online tyre model reconstruction by locally weighted projection regression. 2020.

[32] E. Xargay J. Wang, F. Holzapfel and N. Hovakimyan. Non-cascaded dynamic inversion design for quadrotor position control with l1 augmentation. 2013. doi: 10.2514/6.2013-4855.

[33] T. A. Johansen. Chapter 1: Introduction to nonlinear model predictive control and moving horizon estimation. 39:904–918, 2018. ISSN 0143-2087. doi: 10.1002/oca.2388.

[34] Nitin R. Kapania and J. Christian Gerdes. Path tracking of highly dynamic autonomous vehicle trajectories via iterative learning control. *2015 American Control Conference (ACC)*, Jul 2015. doi: 10.1109/acc.2015.7171151. URL http://dx.doi.org/10.1109/ACC.2015.7171151.

[35] Stefan Klanke and Sethu Vijayakumar. A library for locally weighted projection regression. supplementary documentation. 50:180–196, 2008. ISSN 0197-6729. doi: 10.1002/atr.1325.

[36] Kanellakopoulos I Krstic M and Kokotovic P. Nonlinear and adaptive control. 40:426–440, 1995. ISSN 0018-9286. doi: 10.1109/9.376055.

[37] Alexander Liniger, Alexander Domahidi, and Manfred Morari. Optimization-based autonomous racing of 1:43 scale rc cars. *Optimal Control Applications and Methods*, 36(5):628–647, Jul 2014. ISSN 0143-2087. doi: 10.1002/oca.2123. URL http://dx.doi.org/10.1002/oca.2123.

[38] N. Liu and A. G. Alleyne. Iterative learning identification for an automated off-highway vehicle. In *Proceedings of the 2011 American Control Conference*, pages 4299–4304, 2011.

[39] Yang Liu. The cointerpretation of flow rate and pressure data from permanent downhole gauges using wavelet and data mining approaches, 2009.

[40] Zhenji Lu, Barys Shyrokau, Boulaid Boulkroune, Sebastiaan van Aalst, and Riender Happee. Performance benchmark of state-of-the-art lateral path-following controllers. 03 2018. doi: 10.1109/AMC.2019.8371151.

[41] Ossama Mokhiamar and Masato Abe. Simultaneous optimal distribution of lateral and longitudinal tire forces for the model following control. *Journal of Dynamic Systems, Measurement, and Control*, 126(4):753–763, Jan 2004. doi: 10.1115/1.1850533.

[42] Manfred Morari and Jay H. Lee. Model predictive control: past, present and future. 23:667–682, 1999. ISSN 0098-1354. doi: 10.1016/s0098-1354(98)00301-9.

[43] Anusha Nagabandi, Gregory Kahn, Ronald S. Fearing, and Sergey Levine. Neural network dynamics for modelbased deep reinforcement learning with model-free finetuning. 2018. doi: 10.1109/icra.2018.8463189.

[44] K. S. Narendra and A. M. Annaswamy. Stable adaptive systems. 31:1754–1778, 1989. ISSN 0890-6327. doi: 10.1002/acs.2798.

[45] Duy Nguyen-Tuong and Jan Peters. Learning robot dynamics for computed torque control using local gaussian processes regression. 2008. doi: 10.1109/lab-rs.2008.16.

[46] Duy Nguyen-tuong, Jan Peters, Matthias Seeger, and Bernhard Schölkopf. Learning inverse dynamics: a comparison, 2008.

[47] Chris Ostafew, Angela Schoellig, and Timothy Barfoot. Learning-based nonlinear model predictive control to improve vision-based mobile robot path-tracking in challenging outdoor environments. pages 4029–4036, 05 2014. doi: 10.1109/ICRA.2014.6907444.

[48] Hans B. Pacejka, Hans B. Pacejka, and Hans B Pacejka. Preface. *Tire and Vehicle Dynamics*, pages xiii–xvi, 2012. doi: 10.1016/b978-0-08-097016-5.05001-4.

[49] Luká Palaj. Aplikace lokálních aproximátorů pro řízení reálného mechatronického systému. 2011.

[50] Marino R and Tomei P. Nonlinear control design: Geometric, adaptive and robust. 27:35–45, 1995. ISSN 0890-6327. doi: 10.1002/acs.2346.

[51] James B. Rawlings and Kenneth R. Muske. The stability of constrained receding horizon control. *IEEE Transactions on Automatic Control*, 38(10):1512–1516, 1993. doi: 10.1109/9.241565.

[52] Ugo Rosolia, Ashwin Carvalho, and Francesco Borrelli. Autonomous racing using learning model predictive control, 2016.

[53] Roweis S. and Saul L. Nonlinear dimensionality reduction by local linear embedding. 290:2323–2326, 2000. ISSN 0036-8075. doi: 10.1126/science.290.5500.2323.

[54] C. G. Atkeson S. Schaal and S. Vijayakumar. Real-time robot learning with locally weighted statistical learning. doi: 10.1109/robot.2000.844072.

[55] Stefan Schaal and Christopher G. Atkeson. Constructive incremental learning from only local information. 10:2047–2084, 1998. ISSN 0899-7667. doi: 10.1162/089976698300016963.

[56] A. Schmeitz, J. Zegers, J. Ploeg, and M. Alirezaei. Towards a generic lateral control concept for cooperative automated driving theoretical and experimental evaluation. In *2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, pages 134–139, 2017.

[57] Burgess C. & Smola A. J. Sch¨olkopf, B. Advances in kernel methods: Support vector learning. doi: 10.1002/0470011815.b2a14038.

[58] Scott. Multivariate density estimation. 4:108–116, 1992. ISSN 2049-1573. doi: 10.1002/sta4.81.

[59] Jankovic M Sepulchre R and Kokotovic P. Constructive nonlinear control. 1997. ISSN 0178-5354. doi: 10.1007/978-1-4471-0967-9.

[60] Jitendra Shah. Development and control of evasive steer assist using rear wheel steering. *SAE Technical Paper Series*, 2015. doi: 10.4271/2015-26-0004.

[61] Jarrod Snider. Automatic steering methods for autonomous automobile path tracking. 04 2011.

[62] Damoon Soudbakhsh and Azim Eskandarian. A collision avoidance steering controller using linear quadratic regulator. *SAE Technical Paper Series*, Dec 2010. doi: 10.4271/2010-01-0459.

[63] Patrick Stalph, Jérémie Rubinsztajn, Olivier Sigaud, and Martin Butz. Function approximation with lwpr and xcsf: A comparative study. volume 5, pages 1863–1870, 07 2010. doi: 10.1145/1830761.1830818.

[64] de Silva V Tenenbaum J and Langford J. A global geometric framework for nonlinear dimensionality reduction. 290:2319–2323, 2000. ISSN 0036-8075. doi: 10.1126/science.290.5500.2319.

[65] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, Kenny Lau, Celia Oakley, Mark Palatucci, Vaughan Pratt, Pascal Stang, Sven Strohband, Cedric Dupont, Lars-Erik Jendrossek, Christian Koelen, Charles Markey, Carlo Rummel, Joe van Niekerk, Eric Jensen, Philippe Alessandrini, Gary Bradski, Bob Davies, Scott Ettinger, Adrian Kaehler, Ara Nefian, and Pamela Mahoney. Stanley: The robot that won the darpa grand challenge: Research articles. *J. Robot. Syst.*, 23(9):661–692, September 2006. ISSN 0741-2223.

[66] Julian P. Timings and David J. Cole. Minimum Maneuver Time Calculation Using Convex Optimization. *Journal of Dynamic Systems, Measurement, and Control*, 135(3), 03 2013. ISSN 0022-0434. doi: 10.1115/1.4023400. URL https://doi.org/10.1115/1.4023400. 031015.

[67] H. Eric Tseng, Jahan Asgari, Davor Hrovat, Pim Van Der Jagt, Ann Cherry, and Steve Neads. Steering robot for evasive maneuvers - experiment and analysis. *IFAC Proceedings Volumes*, 35(2):79–86, 2002. doi: 10.1016/s1474-6670(17)33922-8.

[68] R. Verschueren, S. De Bruyne, M. Zanon, J. V. Frasch, and M. Diehl. Towards time-optimal race car driving using nonlinear mpc in real-time. In *53rd IEEE Conference on Decision and Control*, pages 2505–2510, Dec 2014. doi: 10.1109/CDC.2014.7039771.

[69] Sethu Vijayakumar. Developed software documentation, 2004. URL http://homepages.inf.ed.ac.uk/svijayak/software.1.html.

[70] Sethu Vijayakumar. Incremental online learning in high dimensions. 17:2602–2634, 2005. ISSN 0899-7667. doi: 10.1162/089976605774320557.

[71] Sethu Vijayakumar and Stefan Schaal. Locally weighted projection regression : An o(n) algorithm for incremental real time learning in high dimensional space. pages 1–14, 2016. doi: 10.1007/978-1-4899-7502-7_493-1.

[72] Motomura Y Vlassis N and Krose B. Supervised dimensionality reduction of intrinsically low-dimensional data. 14:191–215, 2002. ISSN 0899-7667. doi: 10.1162/089976602753284491.

[73] C. K. I. Williams and Rasmussen. Gaussian processes for regression. pages 378–382, 1996. ISSN 1387-666X. doi: 10.1007/978-1-4615-6099-9_66.

[74] D. W. Li Y. G. Xi and S. Lin. Model predictive control - status and challenges. *Acta Automatica Sinica, vol. 39, pp. 222-236*, 39:222–236, 2013. ISSN 0254-4156. doi: 10.3724/sp.j.1004.2013.00222.