UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

MASTER'S DEGREE IN CONTROL SYSTEMS ENGINEERING

LEARNING-BASED NONLINEAR MODEL PREDICTIVE CONTROL FOR A MOTORCYCLE VIRTUAL RIDER

ADVISOR: Prof. Mattia Bruschetta

STUDENT: Francesco Bianchin

ACADEMIC YEAR 2021/22

10 October 2022

ABSTRACT (EN)

Model predictive control (MPC) is an established technique for high-performance control that is used in many application fields, ranging from the automotive sector to the control of biological systems. In particular, the possibility of applying MPC in a real-time fashion has led to its successful implementation in demanding scenarios like car driving and motorcycle riding. However, the dissimilarity between the internal model and the real system often represents a crucial issue of MPC, especially when dealing with nonlinear systems. For that reason, new research in the field has been devoted towards the introduction of learning techniques into MPC frameworks, notably in order to reduce the mismatch between the model used by the controller and the controlled system itself. Such approaches belong to the general field of learning-based MPC (LbMPC). This thesis revolves around the previously cited themes of LbMPC. Specifically, learning techniques are explored in order to improve the nonlinear physics-based model of a motorcycle system to be used for Nonlinear Model Predictive Control. The objective is to design a Virtual Rider for high performance motorcycle riding. The main challenges are related to the determination of which parts of the dynamic model would benefit most from the addition of a learnt component and which learning strategy to pursue. Most of the effort has been devoted to Gaussian Process Regression strategies, with the exploration of both black-box and grey-box approaches and the investigation of sparse approximations in order to guarantee the real-time implementation of learning-based MPC.

ABSTRACT (IT)

Il controllo predittivo (MPC) si è imposto come tecnica avanzata di controllo ad alta prestazione, con applicazioni di vario tipo, a partire dal settore automotive fino al controllo di sistemi biologici. La possibilità di usare il controllo predittivo in real-time ne ha permesso l'applicazione in scenari complessi come la guida di auto e moto. Le differenze presenti tra il modello usato dal controllore e il sistema reale rappresentano tuttavia una forte criticità tipica di molte delle applicazioni del controllo predittivo, soprattutto quando il sistema in questione presenta nonlinearità significative. Di conseguenza, negli scorsi anni un interessante ambito di ricerca ha riguardato l'inclusione di tecniche di Machine Learning negli approcci di controllo predittivo al fine di ridurre le discrepanze del modello predittivo rispetto al sistema oggetto dell'azione di controllo. Approcci di questo tipo apprertongono al mondo del Learning-based MPC (LbMPC).
Questa tesi si pone quindilo scopo di esplorare tematiche proprie del controllo predittivo con applicazioni di learning. In particolare, le tecniche di Machine Learning sono state implementate al fine di migliorare il modello nonlineare usato dal controllore MPC nell'ambito della guida simulata di una motocicletta. Lo scopo è infatti quello di creare un pilota virtuale (rappresentato da un controllore predittivo) che sappia simulare una guida ad alte prestazioni. Le sfide principali affrontate nel corso della tesi sono state l'individuazione delle porzioni della dinamica di veicolo che possano beneficiare dall'applicazione di tecniche di Machine Learning e la scelta delle strategie di learning da usare. Gran parte dell'attenzione è stata rivolta alle tecniche di regressione con processi gaussiani, con le quali sono stati esplorati sia approcci di modellizzazione black-box che grey-box. Sono state inoltre esplorate le approsimazioni sparse dei processi gaussiani, che possono essere utilizzate per la creazione di modelli che siano implementabili in tempo reale nel controlo predittivo.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

BRT Bike Real-Time

CM Center of Mass

DoF Degree of Freedom

EoM Equation of Motion

FITC Fully Independent Training Conditional

GP Gaussian Process

GPR Gaussian Process Regression

i.i.d. Independent and Identically Distributed

LbMPC Learning-based Model Predictive Control

MI Mutual Information

MPC Model Predictive Control

NLL Negative Log Likelihood

NLP Nonlinear Programming

NMPC Nonlinear Model Predictive Control

pdf Probability Density Function

QP Quadratic Programming

RTI Real-Time Iteration

R.V. Random Variable

SE Squared Exponential (kernel)

SQP Sequential Quadratic Programming

SMPC Stochastic Model Predictive Control

VFE Variational Free Energy

VSGP Variational Sparse Gaussian Process

1

INTRODUCTION

The development of Virtual Riding strategies has been a core project at the University of Padua in the past years [1][2]. The project's objective is in line with the efforts towards virtual prototyping for the automotive industry. Virtual prototyping's advantages range from the reduced costs and time-to-market to the increased safety. Significant results have been attained exploiting MPC schemes, which allow for high-performance real-time control of the motorcycle system, while emulating a realistic riding behavior. As mentioned, efforts towards physics-based modeling of the motorcycle system for MPC control date back to [1], which set the foundation for an MPC-oriented and Lagrangian-based bicycle model of the motorcycle system. Further refinements came along the years, culminating with [8], which provided a comprehensive modeling of the the motorcycle system, avoiding non-holonomic constraints, in favor of a sliding plane approach. Additional modeling improvements included the lateral tire forces generation according to *Pacejka Magic formula* [9] and the incorporation of the wheel gyroscopic effects. Advancements in the modeling front were accompanied by significant progress on the MPC solving tools front. A notable endeavor was in fact the development of an efficient and open-source *MATLAB* toolbox, *MATMPC* [11], which allows for real-time solution of the NLP optimization problem associated to MPC. The final modeling adjustments came with [2], which enriched the physical model with a point-mass motorcycle rider and added the associated lateral movement to the set of system inputs. It is evident that riding solutions so far relied on internal modeling which was exclusively physics-based; however, significant limitations remained, as the accuracy of the physics-based motorcycle model was still questionable when compared to the real system behavior; for that reason, the advent of learning-based MPC (LbMPC) frameworks [3] represent an exciting new direction for the exploration of new data-driven modeling approaches, to be used for the improvement of the MPC internal accuracy.

LbMPC is a broad research field which generally refers to the implementation of learning techniques into MPC control schemes. In particular, it has developed into two main subfields: the *learning dynamics* one, which involves the data-based adaptation of the prediction model, and the *learning design* one, whose efforts entail the use of data-driven techniques for controller design and improvement. This thesis deals with the former, as learning techniques were employed in order to improve the internal model of the Virtual Rider. Gaussian Process Regression (GPR) was the prime candidate in order to construct learning-based dynamical models, in view of its success stories in

multiple MPC-related applications [4][5]. GPR is a probabilistic and non-parametric regression approach which has found large application in the world of nonlinear dynamics modeling due to its flexibility. A direct implementation of GP-based models into MPC schemes is generally not viable, especially in view of its computational requirements and due to the added optimization complexity. For that reason, a large amount of research revolved around the exploration of sparse GP approximations [20][18]; at the same time, additional research was directed into feature selection procedures [25], which are fundamental as they lead to computational savings and to better conditioned learning-based models.

The contributions of this thesis are thus related to the implementation of learning-based modeling techniques into the MPC framework, in the context of the Virtual Rider, which constitutes a relevant and challenging case study. The adopted solutions were tailored to guarantee the feasibility of the global strategy and a satisfactory performance of the MPC controller. In particular, the beneficial impact of learning-based solution w.r.t. to the physics-based one will be highlighted, proving the relevance of the novel LbMPC approaches and justifying further research in this direction.

The first part of the thesis deals with the most relevant methodologies for the purpose of the conducted research. Firstly, an overview of the state-of-the-art physics-based implementation of the Virtual Rider will be provided. This includes a review of the modeling approach, a presentation of the NMPC problem to be solved by the Virtual Rider and the introduction to the co-simulation environment which was used for the testing phase. After highlighting the issues faced by the physics-based model in the Virtual Riding task, the attention will be shifted towards a recap of the main GP methodologies of interest. This includes an overview of the general GP framework, which is followed by an investigation of scalable GP models [6], that can be used for the derivation of computationally efficient models to be implemented into the MPC framework.

The second part is centered on the experimental development of this thesis, including an overview of the research phases and the presentation of the main results and analyses obtained from the undertaken trials. A large part of research focused on the feature selection procedure, since only the relevant state-input information should be used for a given regression target. It will become clear that the choice of features has a deep impact on the performance of the models when implemented into the Virtual Rider. The devised feature selection procedures led to working solutions, but showed that additional refinements should be investigated, as the established selection criteria generally did not provide satisfactory guarantees for the closed-loop performance. The most interesting learning-based models were then thouroghly tested in the co-simulation environment; the obtained results will thus be compared with the corresponding outcomes of the physics-based solution. Model evaluation criteria were multiple and included the general tracking properties of the Virtual Rider, the system input management, the lap times and the computational requirements of each solution.

The third and last part is devoted to the conclusive remarks, with a discussion of the advantages and limitations of the devised strategies. A final overview of prospective research directions that could be undertaken in the LbMPC field concludes the thesis.

Part I

STATE OF THE ART AND METHODOLOGIES

This section is devoted to the introduction of the problem of Nonlinear Model Predictive Control and its application to the motorcycle Virtual Rider, which has been a core project at University of Padua. The learning-based MPC approaches that have been used are also detailed in order to delineate the background of the thesis project and to motivate the research directions that have been undertaken.

2

A VIRTUAL MOTORCYCLE RIDER FOR HIGH PERFORMANCE DRIVING

Virtual prototyping and simulative tools have become a pillar for many industries, in order to reduce both costs and the time-to-market. That is also the case for the automotive industry, and specifically for the motorcycle sector, with the development of virtual riding solutions that go back to [7], in which a simple PID control action was used, and to [1], in which the first MPC-based solutions to virtual riding were employed. As often highlighted in those works, a motorcycle is a complex dynamical system, with the added obstacle of instability, as it has to be maintained to the upright position while following a desired path. It is also characterized by complex dynamics, that include suspensions, chain pull effects and tire profiles, among the others.
Solutions based on MPC appear particularly appealing in order to simulate a human-like riding behavior, as the approach to input generation of a Model Predictive Controller is based on the prediction of how the system trajectory will evolve in the future on the basis of the internal model that is used. That is conceptually similar to how an actual rider would ride, i.e. by choosing the appropriate input (e.g. in terms of steering and throttling) based on future projections and the desired trajectory. MPC also allows for constraint enforcement, which may be used in order to guarantee that a human-like behavior (in terms of roll angle etc.) is maintained along the track.
The next sections will be devoted to an overview of the most recent developments in the design of a physics-based virtual rider at the University of Padua, whose results have been published in [8] and [2].

2.1 A PHYSICS-BASED VIRTUAL RIDER

This section is devoted to an overview of the later developments of a physics based Virtual Rider for high performance motorcycle driving [2]. The rider is constituted by a real-time capable NMPC controller, which is based on a nonlinear internal model derived through Lagrangian modeling. In particular, the motorcycle is represented through a sliding plane model with two contact points on the ground, while the rider is modeled as a point mass that can move in the lateral direction. Lateral slipping is regarded through Pacejka Magic Formula and additional gyroscopic effects along with the roll and caster effect on the steering angle on the ground have been considered.

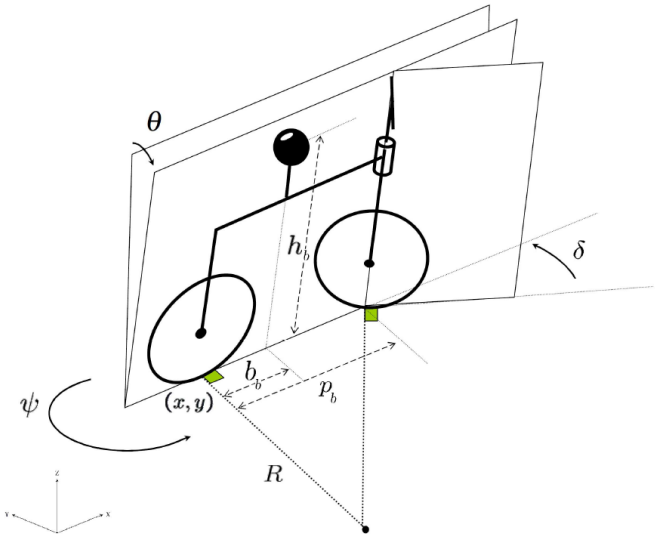


Figure 2.1: Scheme and reference frame for the motorcycle model. The main parameters of the model and the generalized coordinates are highlighted.

2.1.1 *Model for Control Synthesis*

The nonlinear internal model used in the NMPC algorithm is based on a sliding plane representation, in which the plane is allowed to roll and slide both in the x- and y-directions; a point mass with a lateral movement DoF represents the rider (Figure 2.1).

2.1.1.1 *Dynamics*

Dynamic equations for the combined motorcycle-rider system have been derived through the Euler-Lagrangian method, using $q = [x, y, \psi, \theta]$ as the generalized coordinates for the system, with (x, y) being the rear wheel contact position coordinates, ψ the yaw angle and θ the roll angle (see Figure 2.1). The rider lateral freedom of movement is exemplified by the frontal and sagittal plane views in Figure 2.2.
The first step is to derive the Lagrangian $L = K_b + K_r - U_b - U_r$, which includes the kinetic energy components of both the motorcycle body (K_b) and the rider (K_r):

$$K_b = \frac{1}{2} m_b (\dot{x}_{cm,b}^2 + \dot{y}_{cm,b}^2 + \dot{z}_{cm,b}^2) + \frac{1}{2} \omega_b^T I \omega_b \tag{2.1}$$

$$K_r = \frac{1}{2} m_r (\dot{x}_{cm,r}^2 + \dot{y}_{cm,r}^2 + \dot{z}_{cm,r}^2) \tag{2.2}$$

and their potential energy (U_b and U_r respectively):

$$U_b = m_b g h_b c_\theta \tag{2.3}$$

$$U_r = m_r g h_r c_\theta \tag{2.4}$$

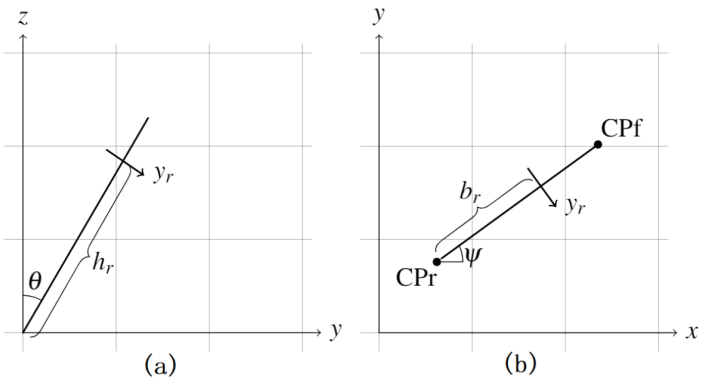


Figure 2.2: Frontal and sagittal views of the motorcycle highlighting the rider movement direction and the sign convention from behind (a) and from above (b).

The center of mass (CM) and the angular velocity of the motorcycle body are derived as:

$$x_{cm,b} = x + b_b c_\psi + h_b s_\theta s_\psi \tag{2.5}$$
$$y_{cm,b} = y + b_b s_\psi - h_b s_\theta c_\psi \tag{2.6}$$
$$z_{cm,b} = h_b c_\theta \tag{2.7}$$
$$\omega_b = [\dot{\theta}, \dot{\psi} s_\theta, \dot{\psi} c_\theta]^T \tag{2.8}$$

while the rider CM is defined as:

$$x_{cm,r} = x + b_r c_\psi + (h_r s_\theta + y_r c_\theta) s_\psi \tag{2.9}$$
$$y_{cm,r} = y + b_r s_\psi - (h_r s_\theta + y_r c_\theta) c_\psi \tag{2.10}$$
$$z_{cm,r} = h_r c_\theta - y_r s_\theta \tag{2.11}$$

The parameters involved are the following:

1. $m_{b/r}$, mass of the motorcycle body/rider;
2. $b_{b/r}$, longitudinal distance from the rear wheel contact position to the motorcycle body/rider CM;
3. $h_{b/r}$, height of the motorcycle body/rider CM;
4. p_b, rear to front wheel contact positions' distance;
5. $I = \text{diag}([I_{xx}, I_{yy}, I_{zz}])$, inertia matrix of the motorcycle body.

Using the generalized forces formulation, it is possible to derive the effect of the forces on the minimal coordinates' system.

$$Q_j = \sum_{k=1}^{N} F_k \cdot \frac{\partial r_k}{\partial q_j}, \quad j \in \{1, ..., 4\}, \quad k \in \{1, ..., 5\} \tag{2.12}$$

where F_k is the force vector at point k and r_k is the position vector of point k according to the inertial reference frame. The generalized forces $Q_j \quad j \in \{1, ..., 4\}$ can be expressed as $B(q) \cdot w$, where w is the vector of forces that are applied to the system, i.e.

$$w = [F_{xf}, F_{yf}, F_{xr}, F_{yr}, F_d]^T \tag{2.13}$$

with the subscripts assigned according to the following convention:

1. x/y: longitudinal/lateral wheel forces
2. f/r: front/rear wheel forces
3. F_d: longitudinal drag force

The frontal tire forces are defined w.r.t. the motorcycle longitudinal direction according to the steering angle on the ground δ_G, which is derived from the handlebar angle δ according to

$$\delta_G = \text{atan}\left(\frac{c_\epsilon s_\delta}{c_\theta c_\delta - s_\theta s_\epsilon s_\delta}\right) \tag{2.14}$$

with ϵ being the caster angle, namely the angular displacement of the steering axis w.r.t. the vertical axis of the frontal wheel. One may thus obtain the typical equation of motion (EoM)

$$M(q)\ddot{q} + C(q, \dot{q}) + G(q) = B(q)w \tag{2.15}$$

where $M(q)$ is the inertia matrix, $C(q, \dot{q})$ includes the centrifugal and the Coriolis components and $G(q)$ the gravitational components. The equation is then reformulated, according to a reference frame that rotates jointly with the motorcycle around the global $Z - axis$, using a velocity transformation T that leads to the following velocity coordinates:

$$\dot{\tilde{q}} = \begin{bmatrix} v_x \\ v_y \\ \dot{\psi} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} c_\psi & s_\psi & 0 & 0 \\ -s_\psi & c_\psi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{\theta} \end{bmatrix} = T^T \dot{q} \tag{2.16}$$

where v_x and v_y are the longitudinal and lateral velocities in the new frame. One may now find the relationship between accelerations in the two frames by differentiating (2.16), yielding:

$$\ddot{q} = T\ddot{\tilde{q}} + \dot{T}\dot{\tilde{q}} \tag{2.17}$$

which, when substituted in (2.15) and after premultiplying by T^T leads to the modified EoM:

$$T^T M(q) T\ddot{\tilde{q}} + T^T[M(q)\dot{T}\dot{\tilde{q}} + C(q, T\dot{\tilde{q}})] + T^T G(q) = T^T B(q)w \tag{2.18}$$

which may be rewritten as

$$\tilde{M}(q)\ddot{\tilde{q}} + \tilde{C}(q, \dot{\tilde{q}}) + \tilde{G}(q) = \tilde{B}(q)w \tag{2.19}$$

using the following substitutions:

$$\tilde{M} = T^T M T, \quad \tilde{G} = T^T G, \quad \tilde{C} = T^T(M\dot{T}\dot{\tilde{q}} + C(q, T\dot{\tilde{q}})), \quad \tilde{B} = T^T B \tag{2.20}$$

2.1.1.2 *Forces*

The forces which are applied to the combined motorcycle/rider system are defined as follows.
The longitudinal forces are computed as

$$F_{xf} = -\gamma_B \lambda \frac{\tau_b^f}{r_{wf}} \tag{2.21}$$

$$F_{xr} = -\gamma_b (1-\lambda) \frac{\tau_b^r}{r_{wr}} + \gamma_t \frac{\tau_t^M}{r_{wr}} + (1-\gamma_t) \frac{\tau_t^m}{r_{wr}} \tag{2.22}$$

with $\gamma_t \in [0,1]$ being the normalized throttle input, $\gamma_b \in [0,1]$ the normalized brake input, $\lambda \in [0,1]$ the front-rear brake bias, $\tau_b^{f/r}$ the maximum front rear torque, τ_t^M the maximum positive transmission torque, τ_t^m the maximum negative torque given by engine braking, $r_{wf/r}$ the front and rear wheel radii. The maximum transmission torque parameters τ_t^M τ_t^m are varied online in order to model the effect of the gearshift.
The lateral forces' formulas are instead derived from the *Pacejka Magic Formula* for motorcycles [9]:

$$\begin{aligned} F_{y,i} =& D_y^i \sin[C_y^i \text{atan}(B_y^i - E_y^i(B_y^i \alpha_i - \text{atan}(B_y^i \alpha_i))) \\ &+ C_\theta^i \text{atan}(B_\theta^i \theta - E_\theta^i(B_\theta^i \theta - \text{atan}(B_\theta^i \theta)))] \quad i \in f, r \end{aligned} \tag{2.23}$$

where α_i are the frontal/rear side-slip angles of the tires, which are defined as

$$\alpha_r = \text{atan}\left(\frac{v_y}{v_x}\right) \tag{2.24}$$

$$\alpha_f = \text{atan}\left(\frac{v_y + p_b \dot{\psi}}{v_x}\right) - \delta_G \tag{2.25}$$

Additionally, the normal forces are computed starting from an estimation of the load transfer torque as

$$F_{zf} = \frac{m_T b_T g - \tau_{LT}}{p_b} \tag{2.26}$$

$$F_{zr} = \frac{m_T (p_b - b_T) g + \tau_{LT}}{p_b} \tag{2.27}$$

where the load transfer torque is

$$\tau_{LT} = (F_{xf} + F_{xr}) h_T \cos(\theta) \tag{2.28}$$

and the total inertial and geometric parameters are

$$\begin{aligned} m_T &= m_b + m_r \\ b_T &= \frac{b_b m_b + b_r m_r}{m_T} \\ h_T &= \frac{h_b m_b + h_r m_r}{m_T} \end{aligned} \tag{2.29}$$

Finally, the longitudinal drag force is modeled through the simplified formula

$$F_d = \frac{1}{2} \rho C_d A v_x^2 \tag{2.30}$$

where ρ is the air density, C_d is the drag coefficient and A is the frontal section area.

2.1.1.3 *Auxiliary Dynamics*

The model accuracy is finally enhanced through the addition of the gyroscopic torques that are generated by the motion of the wheels to the yaw and roll components of the damping matrix as

$$\bar{C} = \tilde{C} + \begin{bmatrix} 0 \\ 0 \\ C_{gyro}^{yaw} \\ C_{gyro}^{roll} \end{bmatrix} \tag{2.31}$$

where the added components are defined as

$$C_{gyro}^{yaw} = (I_{wf}\omega_{wf} + I_{wr}\omega_{wr})\dot{\theta} \tag{2.32}$$

$$C_{gyro}^{roll} = -(I_{wf}\omega_{wf} + I_{wr}\omega_{wr})\dot{\psi}c_{\theta} \tag{2.33}$$

with $\omega_{wi} = (v_x/r_{wi})$, $i \in \{f, r\}$ being the front and rear wheels' rotational velocities and I_{wi}, $i \in \{f, r\}$ being the front and rear wheel rotational inertias. With this auxiliary components the EoM may be revised to the final formulation:

$$\ddot{\tilde{q}} = \tilde{M}^{-1}(\tilde{B}w - \bar{C} - \tilde{G}) \tag{2.34}$$

2.1.1.4 *Dynamics Spatial Reformulation*

Another key ingredient in the system dynamics' formulation to be employed for the NMPC solution is their conversion into spatial coordinates, such that s, the arc length along the track, is the independent variable to be integrated over, rather than time. This is done so that the position reference that is fed to the MPC is not dependent on the velocity. To this end, the reference trajectory σ is parameterized on s and the curvature $\zeta = (1/\rho)$, with ρ being the instantaneous curvature radius, which can be computed as:

$$\zeta = \frac{x'y'' - y'x''}{((x')^2 + (y')^2)^{\frac{3}{2}}} \tag{2.35}$$

where $f' = (df(s)/ds)$ and $f'' = (d^2f(s)/ds^2)$ indicate first and second order derivation w.r.t. the arc length. One may thus define the tracking errors geometrically according to the scheme shown in Figure 2.3:

$$e_{\psi} = \psi - \psi_s \tag{2.36}$$

$$e_y = c_{\psi_s}(Y_s - Y) + s_{\psi_s}(X_s - X) \tag{2.37}$$

where ψ_s is the reference yaw angle, (X_s, Y_s) is the absolute position reference at the current spatial coordinates and (X, Y) is the current absolute position.
One may now derive the tracking error dynamics:

$$\dot{e}_{\psi} = \dot{\psi} - \zeta\dot{s} \tag{2.38}$$

$$\dot{e}_y = v_x s_{e_{\psi}} + v_y c_{e_{\psi}} \tag{2.39}$$

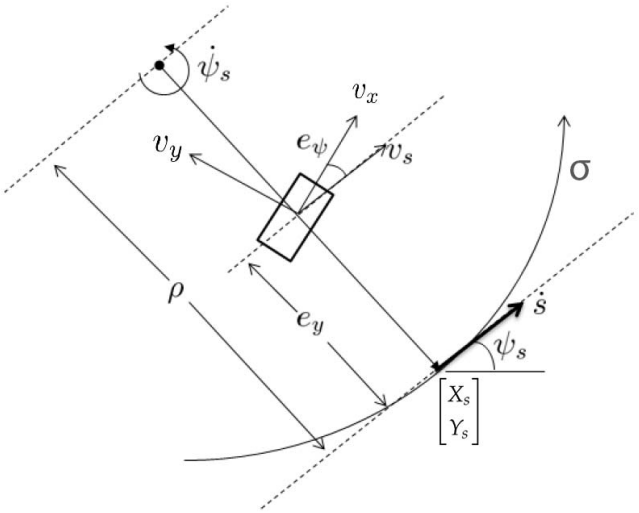


Figure 2.3: Lateral and heading errors definition according to the spatial coordinates system

Using the chain rule, it is possible to compute the derivative of any time-dependent function $\Psi(t)$ w.r.t. the curvilinear abscissa s, i.e.

$$\Psi' = \frac{d\Psi}{ds} = \frac{d\Psi}{dt}\frac{dt}{ds} = \frac{d\Psi}{dt}\frac{1}{\dot{s}} = \frac{\dot{\Psi}}{\dot{s}} \tag{2.40}$$

where the arc length time derivative si given by

$$\dot{s} = \frac{v_x c_{e_\psi} - v_y s_{e_\psi}}{1 - \zeta e_y} \neq 0 \quad \forall t \tag{2.41}$$

2.1.1.5 *Complete Model*

Using the obtained dynamics model, the following state vector may be defined

$$\xi = \begin{bmatrix} e_\psi \\ e_y \\ \theta \\ v_x \\ v_y \\ \dot{\psi} \\ \dot{\theta} \\ \delta \\ \gamma_t \\ \gamma_b \\ y_r \end{bmatrix} \begin{array}{lll} \Rightarrow & \text{Yaw trajectory error} \\ \Rightarrow & \text{Lateral trajectory error} \\ \Rightarrow & \text{Roll angle} \\ \Rightarrow & \text{Longitudinal velocity} \\ \Rightarrow & \text{Lateral velocity} \\ \Rightarrow & \text{Yaw rate of change} \\ \Rightarrow & \text{Roll rate of change} \\ \Rightarrow & \text{Steering angle} \\ \Rightarrow & \text{Normalized throttle input} \\ \Rightarrow & \text{Normalized brake input} \\ \Rightarrow & \text{Rider lateral movement} \end{array} \tag{2.42}$$

and the input vector

$$u = \begin{bmatrix} \dot{\delta} \\ \dot{\gamma}_t \\ \dot{\gamma}_b \\ \dot{y}_r \end{bmatrix} \quad \begin{array}{ll} \Rightarrow & \text{Steering angle rate of change} \\ \Rightarrow & \text{Throttle input rate of change} \\ \Rightarrow & \text{Brake input rate of change} \\ \Rightarrow & \text{Lateral rider movement rate of change} \end{array} \tag{2.43}$$

One can see that the input used in the MPC formulation is composed by the derivatives of the actual motorcycle inputs. This is intentional, as it has been proven to allow for a smoother action of the controller, leading to less aggressive and more realistic riding behavior.
The final spatial-based system dynamics may now be expressed as:

$$\xi' = \phi(\xi(s), u(s); \sigma(s)) \tag{2.44}$$

2.1.2 *NMPC formulation*

After defining and tuning the internal physics-based model to be used for prediction and optimization, the nonlinear Model Predictive Control problem to be solved can now be formulated, in order to generate the desired input quantities (namely the steering angle, the throttle and brake commands and the rider lateral displacement) to be fed to the simulated motorcycle system at each control instant.
The NMPC problem is initially formulated in its continuous variant (in terms of dynamics and cost function) in order to be discretized according to the OCP (Optimal Control Problem) framework. This is done considering a prediction horizon $T = [t_0, t_f]$, to be subdivided into N shooting intervals $[t_0, t_1, ..., t_N]$. The resulting optimization problem will be a NLP (Nonlinear Programming Problem) that can be summarized as follows:

$$\min_{\xi_{\cdot|i}, u_{\cdot|i}} \sum_{k=0}^{N-1} \frac{1}{2} ||h_{k|i}(\xi_{k|i}, u_{k|i})||^2_{W_k} + \frac{1}{2} ||h_{N|i}(\xi_{N|i})||^2_{W_N} \tag{2.45a}$$

$$s.t.\ 0 = \xi_0 - \hat{\xi}_0, \tag{2.45b}$$

$$0 = \xi_{k+1|i} - \phi_k(\xi_{k|i}, u_{k|i}; \sigma_{k|i}), \quad k = 0, 1, \ldots, N-1, \tag{2.45c}$$

$$\underline{r}_{k|i} \leq r_k(\xi_{k|i}, u_{k|i}) \leq \overline{r}_{k|i}, \quad k = 0, 1, \ldots, N-1, \tag{2.45d}$$

$$\underline{r}_{N|i} \leq r_N(\xi_{N|i}) \leq \overline{r}_{N|i} \tag{2.45e}$$

in which the following decision variables are involved:

$$\begin{aligned} \xi_{\cdot|i} &= \left[\xi_{0|i}^\top, \xi_{1|i}^\top, \ldots, \xi_{N|i}^\top\right]^\top, \\ u_{\cdot|i} &= \left[u_{0|i}^\top, u_{1|i}^\top, \ldots, u_{N-1|i}^\top\right]^\top \end{aligned} \tag{2.46}$$

- $\xi_{k|i} \in \mathbb{R}^{n_\xi}$ are the system states at the discrete time instants t_k for $k = 0, ..., N$, while $u_{k|i}$ are the piecewise continuous inputs for $k = 0, ..., N-1$.

- $h_{k|i} : \mathbb{R}^{n_\xi} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_y}, h_{N|i} : \mathbb{R}^{n_\xi} \to \mathbb{R}^{n_{y_N}}$ are vector functions of the state and the control input (ξ, u) that define the cost function terms along the prediction horizon.
- The cost function terms are weighted according to W_k, W_N.
- $\hat{\xi}_0$ is the measurement/estimation of the current state, which is used as the starting point for the numerical integration process and for the predicted trajectory.
- The functions $r(\xi_{k|i}, u_{k|i}) : \mathbb{R}^{n_\xi} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_c}$ and $r(\xi_{N|i}) : \mathbb{R}^{n_\xi} \to \mathbb{R}^{n_{c,N}}$ are used in order to enforce user-defined constraints along the trajectory. They can be linear or nonlinear and are restricted by the lower and upper bounds $\underline{r}_k, \overline{r}_k$.
- $\phi_k(\xi_{k|i}, u_{k|i})$ is a numerical integration operator that solves the initial value problem (IVP) obtained by imposing $\xi(0) = \xi_{0|i}$ in $\xi' = \phi(\xi(s), u(s); \sigma(s))$ (2.44) and returns the solution at t_{k+1}. Intuitively, $0 = \xi_{k+1|i} - \phi_k(\xi_{k|i}, u_{k|i}; \sigma_{k|i})$ represents the continuity constraint, guaranteeing the continuity of the predicted trajectory.

In the specific case of the motorcycle system, the cost function should be employed in order to guarantee the reference trajectory's tracking and a realistic riding behaviour, neither too aggressive or too laid back. These requirements have led to the choice of the following vector functions to be employed in the NLP cost formulation:

$$h_k(\xi_k, u_k) = \begin{bmatrix} e_\psi + \alpha \\ e_y \\ \dot{e}_\psi \\ \dot{e}_y \\ e_v \\ \alpha \\ y_r \\ \gamma_t \cdot \gamma_b \\ \dot{\delta} \\ \dot{\gamma}_t \\ \dot{\gamma}_b \\ \dot{y}_r \end{bmatrix} \qquad h_N(\xi_N, u_N) = \begin{bmatrix} e_\psi + \alpha \\ e_y \\ \dot{e}_\psi \\ \dot{e}_y \\ e_v \\ \alpha \\ y_r \\ \gamma_t \cdot \gamma_b \end{bmatrix} \tag{2.47}$$

where $e_v = v - v_{ref}$ [1] is the vehicle velocity error, which is defined w.r.t. the velocity reference which is fed to the controller and $\alpha = \text{atan}((v_y + b\psi)/v_x)$ is the motorcycle side slip angle.
The role and the meaning of the various cost function components can be summarized as follows:

- e_ψ, e_y and e_v are employed in order to guarantee the reference tracking (both in terms of position and velocity);

1 $v = \sqrt{v_x^2 + v_y^2}$ is the instantaneous velocity along the plane

- $\dot{e}_\psi$, $\dot{e}_y$ (the derivatives of tracking errors) are introduced to guarantee smoother tracking;
- the α penalty can be used to tone down the aggressiveness of the controller, as large values of the side slip angle are related to a riskier riding behaviour;
- y_r is used to guarantee that the rider returns to the vertical position; namely, the vertical position is imposed as the rider's base reference;
- $\gamma_t \cdot \gamma_b$ is used to avoid concurrent throttling and braking, which would be a 'waste', as it would represent an atypical and inefficient riding behaviour.

The constraints are instead imposed on the following state and input quantities:

$$r_k = \begin{bmatrix} \delta \\ \gamma_t \\ \gamma_b \\ y_r \\ \dot{\delta} \\ \dot{\gamma}_t \\ \dot{\gamma}_b \\ \dot{y}_r \end{bmatrix} \qquad r_N = \begin{bmatrix} \delta \\ \gamma_t \\ \gamma_b \\ y_r \end{bmatrix} \tag{2.48}$$

with the following purposes:

- constraints on δ, γ_t, γ_b and y_r are intrinsic control input bounds
- constraints on $\dot{\delta}$, $\dot{\gamma}_t$, $\dot{\gamma}_b$ and $\dot{y}_r$ are added to enforce a smoother input generation.

According to the general MPC framework, the NLP is solved at each control time step (the specifics are detailed in the next section) in order to obtain the optimal control sequence $u^*_{\cdot|i}$ along the horizon. Only the first element of the sequence $u^*_i = u^*_{0|i}$ is applied to the controlled system. Note that the input employed in the NMPC formulation is constituted by the derivatives of the actual system's input, meaning that integration is required in order to obtain the actual input to be applied, i.e.

$$\begin{bmatrix} \delta \\ \gamma_t \\ \gamma_b \\ y_r \end{bmatrix}^*_i = \begin{bmatrix} \delta \\ \gamma_t \\ \gamma_b \\ y_r \end{bmatrix}^*_{i-1} + \Delta t \cdot u^*_i = \begin{bmatrix} \delta \\ \gamma_t \\ \gamma_b \\ y_r \end{bmatrix}^*_{i-1} + \Delta t \cdot \begin{bmatrix} \dot{\delta} \\ \dot{\gamma}_t \\ \dot{\gamma}_b \\ \dot{y}_r \end{bmatrix} \tag{2.49}$$

2.1.3 *NLP solution strategy: Sequential Quadratic Programming*

After setting up the Nonlinear Programming problem as shown in the previous section, one may choose among different solver algorithms in order to find a solution to the

optimization problem. The two most prominent approaches are Interior Point Methods (IPM) and Sequential Quadratic Programming (SQP). They have been both analysed extensively in terms of performance and properties. A comparison of their effectiveness as solvers for optimization problems arising from optimal control schemes is available in [10]. In this section, only the SQP scheme is reviewed, as it is the algorithm that has been employed in all the trials that were attempted as part of this thesis. The general reference is [11], where the NMPC solving toolbox *MATMPC*, which will also be referenced in the next section, is detailed.

2.1.3.1 *Sequential Quadratic Programming*

The SQP is the algorithm that has been employed to solve problem (2.45). The basic idea at the basis of SQP is to solve the NLP by iteratively reformulating it into a Quadratic Programming (QP) problem, for which many high-performance commercial and open-source solvers have been developed. In order to retrieve the QP problem, the objective functions appearing in (2.45a) are replaced by their local quadratic approximation; also, the constraints appearing in (2.45c-2.45e) are replaced by their local affine approximations. At each iteration l of SQP, the following QP problem will be obtained:

$$\min_{\Delta\xi,\Delta u} \sum_{k=0}^{N-1} \frac{1}{2} \begin{bmatrix} \Delta\xi_k & \Delta u_k \end{bmatrix} H_k^l \begin{bmatrix} \Delta\xi_k \\ \Delta u_k \end{bmatrix} + {g_k^l}^T \begin{bmatrix} \Delta\xi_k \\ \Delta u_k \end{bmatrix} + \frac{1}{2}\Delta\xi_N^T H_N^l \Delta\xi_N + {g_N^l}^T \Delta\xi_N \tag{2.50a}$$

$$s.t.\ \Delta\xi_0 = \hat{\xi}_0 - \xi_0, \tag{2.50b}$$

$$\Delta\xi_{k+1} = A_k^l \Delta\xi_k + B_k^l \Delta u_k + a_k^l, \tag{2.50c}$$

$$\underline{c}_k^l \leq C_k^l \Delta\xi_k + D_k^l \Delta u_k \leq \overline{c}_k^l, \tag{2.50d}$$

$$\underline{c}_N^l \leq C_N^l \Delta\xi_N \leq \overline{c}_N^l, \tag{2.50e}$$

where $\Delta\xi = \xi - \xi^l$, $\Delta u = u - u^l$ are the optimization variables resulting from the compact notation $\Delta\xi = [\xi_0^T, \xi_1^T, ..., \xi_N^T]^T$, $u = [u_0^T, u_1^T, ..., u_N^T]^T$ of the discrete state and control variables. Note that ξ^l and u^l represent the guess for the state and control trajectories derived from the previous SQP iteration. The solution $(\Delta\xi^l, \Delta u^l)$ to the QP (2.50) is then used to update the solution to the NLP (2.45) according to:

$$\xi^{l+1} = \Delta\xi^l + \beta^l \Delta\xi^l, \tag{2.51}$$

$$u^{l+1} = \Delta u^l + \beta^l \Delta u^l \tag{2.52}$$

where β^l is the step length which is determined by the globalization strategy employed. For completeness' sake, the linearization matrices involved in (2.50) are now reported:

$$\begin{aligned} &A_k^l = \frac{\partial\hat{\phi}}{\partial\xi_k}, \qquad B_k^l = \frac{\partial\hat{\phi}}{\partial u_k}, \qquad a_k^l = \hat{\phi}(\xi_k^l, u_k^l) - \xi_{k+1}^l, \\ &C_k^l = \frac{\partial r_k}{\partial\xi_k}, \qquad D_k^l = \frac{\partial r_k}{\partial u_k}, \qquad C_N^l = \frac{\partial r_N}{\partial\xi_N}, \\ &\overline{c}_k^l = \overline{r}_k - r_k(\xi_k^l, u_k^l), \qquad \underline{c}_k^l = \underline{r}_k - r_k(\xi_k^l, u_k^l), \\ &\overline{c}_N^l = \overline{r}_N - r_N(\xi_N^l), \qquad \underline{c}_N^l = \underline{r}_N - r_N(\xi_N^l) \end{aligned} \tag{2.53}$$

The Hessian matrices H_k^l, H_N^l appearing in the modified cost function (2.50a) are instead approximated by means of the Gauss-Newton method, i.e. as

$$H_k^l = \frac{\partial h_k^l}{\partial(\boldsymbol{\xi}_k, \boldsymbol{u}_k)}^T \frac{\partial h_k^l}{\partial(\boldsymbol{\xi}_k, \boldsymbol{u}_k)} \tag{2.54}$$

which is guaranteed to be always positive semi-definite.

2.1.3.2 *QP iteration solution*

The iterative QP problems can then be solved according to different strategies, among which sparse solvers like *HPIPM*, *OSQP* and *Ipopt*, which may be used to exploit the generally sparse structure arising from the linearization of the NMPC problem. An alternative is to instead condense problem (2.50) (and integrating the state variables in the process) in order to obtain a dense QP problem of the type

$$\min_{\Delta \boldsymbol{u}} \frac{1}{2} \Delta \boldsymbol{u}^T H_c \Delta \boldsymbol{u} + g_c^T \Delta \boldsymbol{u} \tag{2.55}$$

$$s.t. \, \underline{c}_c \leq C_c \Delta \boldsymbol{u} \leq \overline{c}_c \tag{2.56}$$

In this case solvers like *qpOASES* may be employed. A final alternative may be to instead perform partial condensing, which leads to a smaller but still sparse QP problem. *HPIPM* was the preferred solver for the NLP problem arising from the Virtual Rider's NMPC.

2.2 SIMULATIVE RESULTS AND DRAWBACKS OF THE PHYSICS-BASED VIRTUAL RIDER

This section is devoted to a brief overview of how the Virtual Rider/NMPC controller is tested in simulation and how the NMPC problem is set up and solved in the *MATLAB* environment. Finally, when discussing the performance of the physics-based Virtual Rider, the main drawbacks of this approach will be highlighted.

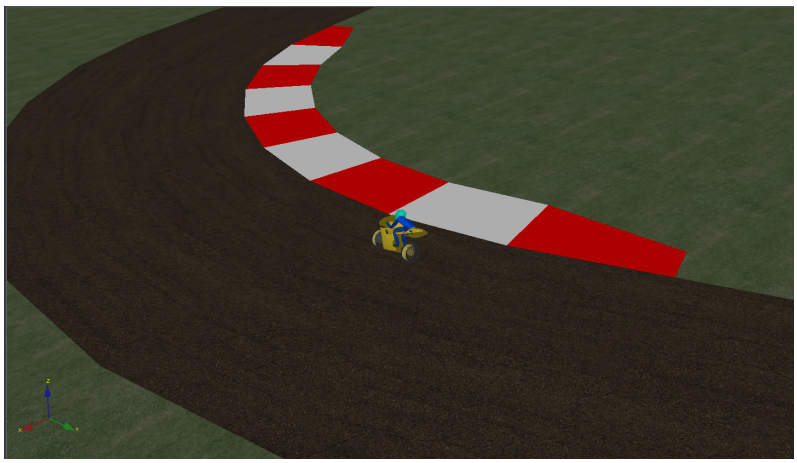


Figure 2.4: A still taken from the *VI-BRT Animator* software, providing animations of the simulated track traversal using the NMPC-based Virtual Rider

2.2.1 *Simulative Environment*

The Virtual Rider is tested in a co-simulation environment: the NMPC controller is in fact tested on a high-fidelity simulative environment called *VI-BikeRealTime*, provided by VI-Grade in its 20.0 version [12]. The general simulation scheme is reported in Figure 2.5, showing how the different components are interconnected in the general *Simulink* environment.
As it can be seen, there is a specific block named as $VI-BRT$, which refers to the simulation model of the motorcycle, which is integrated into the *Simulink* framework by means of an *s-function*, having both inputs (produced by the Virtual Rider/NMPC controller) and outputs (to be used e.g. for state/trajectory errors' estimation). The simulation model in question counts 11 DoFs, including 6 for the sprung mass, 2 for each wheel-suspension and 1 for the rotation between pinion and chassis. It also includes comprehensive dynamics of the tires, suspensions, brakes, driveline and gyroscopic effects. Additionally, a moving mass with lateral DoF simulates the rider. It is clear that significant deviations between the simulation model and the internal model of the Virtual Rider are to be expected.
The simulation block is connected to the designed NMPC controller, which solves the NLP problem at each control instant according to the strategy outlined in the previous section. The solution itself is attained through the open-source *MATLAB* toolbox *MATMPC* [11].
The *MATMPC*-computed controls (which are the derivatives of the actual motorcycle inputs) are then integrated using a simple Euler scheme, in order to be fed to the simulation block. In particular, the trials performed with the physics-based Virtual Rider entailed a *VI-BRT* simulation frequency $f_{sim} = 1000Hz$ of the vehicle dynamics, while the controls are updated at a $f_c = 100Hz$ frequency.

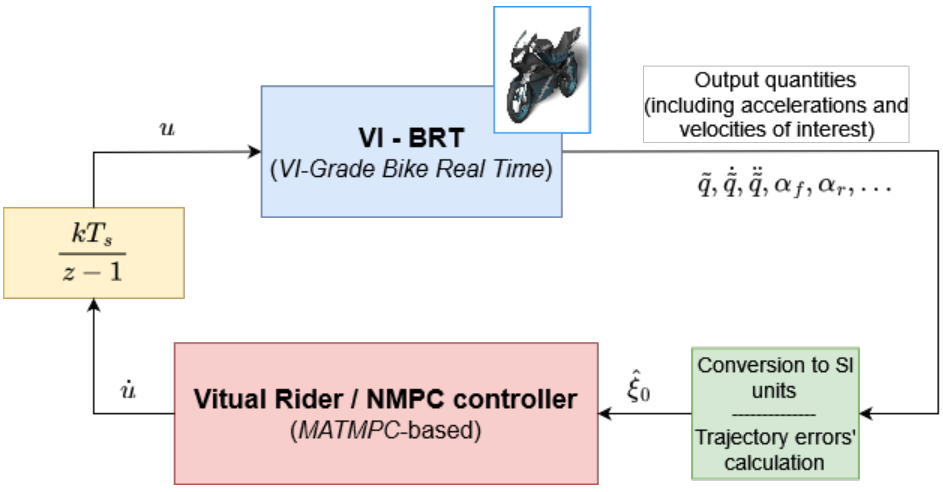


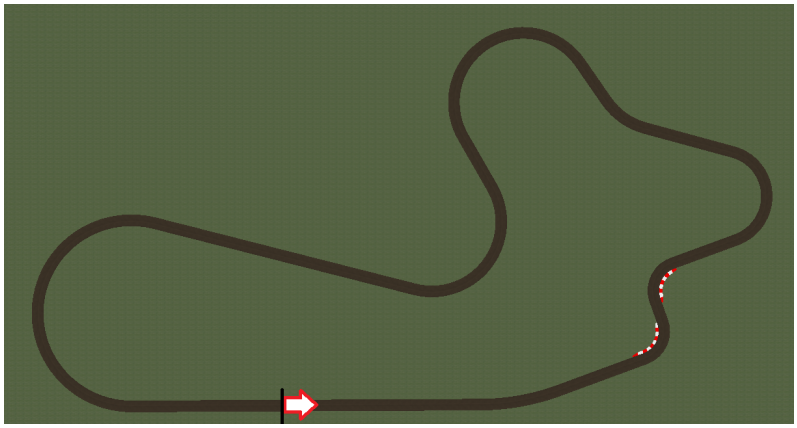
Figure 2.5: Scheme of the Co-Simulation Environment

2.2.2 *Performance of the physics-based Virtual Rider*

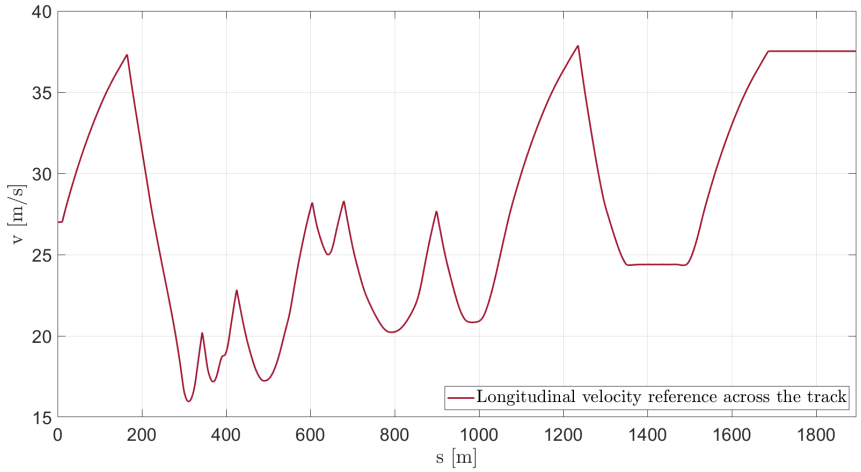
Once the general simulation has been set up according to the scheme of Figure 2.5, it is possible to perform a simulation, using a given internal model and the chosen NMPC configuration. In particular, a suitable weight configuration may be found through a tuning procedure. An interesting approach is the one provided by the Non-dominated Sorting Genetic Algorithm NSGA-II, as described in [13]. Using such approach, the NMPC weights may be tuned in order to find the desired configuration, in a range that goes from performance-oriented to stability-inducing. The tuning procedure must be performed w.r.t. a given task, i.e. a given track to be completed. In particular, the configuration which was adopted for the main comparisons of this thesis was a mid-range one (in the *performance* $\leftrightarrow$ *stability* spectrum). More specifically the weights along the prediction horizon and at the the terminal step (according to the weighting approach already presented in (2.47)) are:

$$h_k(\xi_k, u_k) = \begin{bmatrix} 0.951 \\ 0.130 \\ 0.900 \\ 10^{-4} \\ 0.103 \\ 2.084 \\ 10^{-3} \\ 10^{5} \\ 0.500 \\ 10^{-4} \\ 8 \cdot 10^{-4} \\ 0.600 \end{bmatrix} \qquad h_N(\xi_N, u_N) = \begin{bmatrix} 0.951 \\ 0.130 \\ 0.900 \\ 10^{-4} \\ 0.103 \\ 2.084 \\ 10^{-3} \\ 10^{5} \end{bmatrix} \tag{2.57}$$

As previously mentioned, the weight configuration has to be chosen according to the driving task. For the purposes of this thesis the general framework for comparative analysis is the so called *VI-Track*, which is a test track provided by *VI-BRT*, which is shown in Figure 2.6a. The track is supposed to be completed according to a corner-cutting route that is computed off-line. A suitable velocity profile to be tracked is also generated off-line using the tools provided by *VI-Road*. A sufficiently conservative velocity profile is chosen in order to guarantee a comparative framework that is sufficiently easy to approach. The velocity profile to be tracked is shown in Figure 2.6b.



(a) *VI-Track*, the track to be used as the testbed for comparisons



(b) The longitudinal velocity reference to be tracked

Figure 2.6: General testbed for trials and comparisons of the devised Virtual Rider control strategies

In terms of constraints to be satisfied along the trajectory, the chosen configuration (according to framework shown in (2.48)) was the following:

$$\underline{r}_k = \begin{bmatrix} -\pi/20 \\ 0 \\ 0 \\ -0.15 \\ -1 \\ -10 \\ -10 \\ -1 \end{bmatrix} \quad \bar{r}_k = \begin{bmatrix} \pi/20 \\ 1 \\ 1 \\ +0.15 \\ 1 \\ 10 \\ 10 \\ 1 \end{bmatrix} \qquad \underline{r}_N = \begin{bmatrix} -\pi/20 \\ 0 \\ 0 \\ -0.15 \end{bmatrix} \quad \bar{r}_N = \begin{bmatrix} \pi/20 \\ 1 \\ 1 \\ +0.15 \end{bmatrix} \tag{2.58}$$

Using this base configuration with the nominal (physics-base) model as the internal model for the Virtual Rider satisfactory results can be attained, with a lap time of 66.94s when using a uniform grid for integration, consisting of $N = 80$ shooting intervals of $T_{st} = 1m$ and including two integration steps per shooting interval.

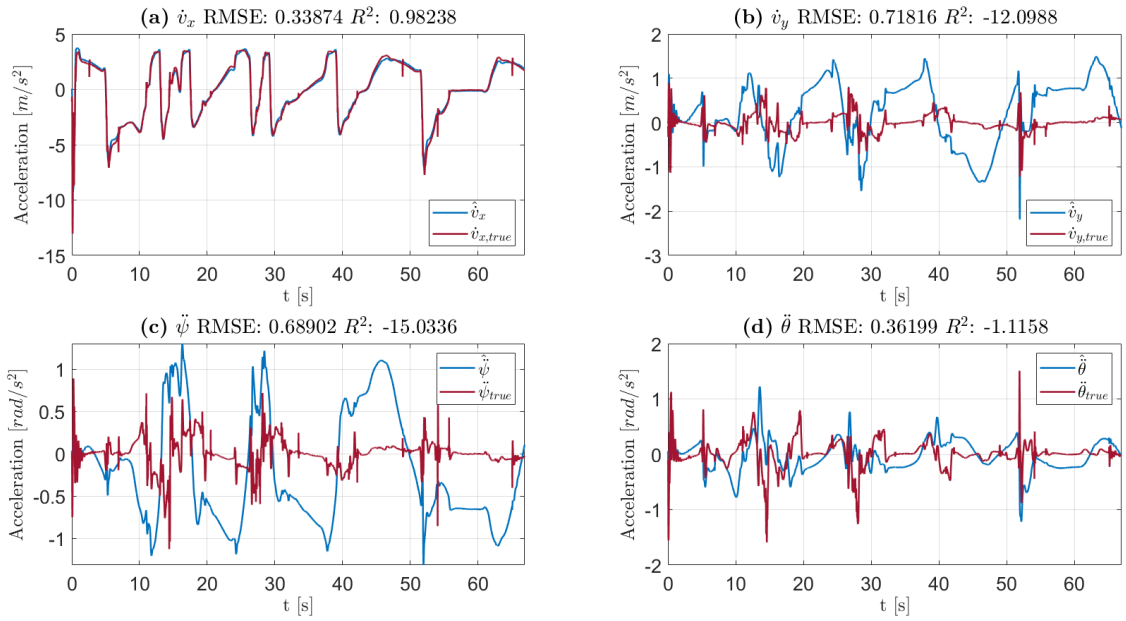


Figure 2.7: Plots showing a comparison between the real accelerations (shown in red) and the accelerations estimated by the internal physics-based model. Specifically, (a) shows the longitudinal acceleration $\dot{v}_x$, (b) shows the lateral acceleration $\dot{v}_y$, (c) shows the yaw acceleration $\ddot{\psi}$, (d) shows the roll acceleration $\ddot{\theta}$

At the same time the physics-based model presents some drawbacks, which are mainly related to the significant mismatch that exists w.r.t. the real motorcycle system. This is exemplified by the plots of the 4 accelerations of the system reported in Figure 2.7: it can be seen how the accelerations that are estimated by the internal white-box model are generally very different w.r.t. the real system accelerations. Barring the longitudinal acceleration ($\dot{v}_x$), for which there is a close similarity, it is clear that for the others (namely the lateral acceleration $\dot{v}_y$, the yaw acceleration $\ddot{\psi}$ and the roll acceleration $\ddot{\theta}$) the internal model is only able to capture the general tendencies, while it often greatly overestimates accelerations. This does not appear to affect the general efficacy of the NMPC controller, which seems to cope with such discrepancies when it is appropriately tuned. Still, it begs the question as to whether improving the internal modelling capabilities of the NMPC controller may lead to improvements in the riding performance. For the task at hand, performance may formulated in terms of lap time and trajectory tracking capabilities, but also in terms of accuracy of the NMPC internal model.

This is where this thesis work comes into play, as the main focus was placed into exploring learning paradigms that may employed in order to augment the white-box dynamics with additional learning-based ones or alternatively to substitute the physics-based dynamics with black-box models to be derived by learning from data. This is done with the stated objective of improving the general performance. Such research direction falls under the general framework of *learning dynamics*, inside the broader world of Learning-based MPC (LbMPC), whose main objective is to improve the internal model accuracy in order to elicit improvements to the general control performance.

The next chapter is thus devoted to an overview of the main learning techniques (mostly related to Gaussian Process Regression) that were explored and tested for this thesis in order to model dynamics.

3

GAUSSIAN PROCESS REGRESSION FOR DYNAMICS LEARNING

This chapter is devoted to the topic of Gaussian Process Regression, which has been the main methodology used in order to move towards the world of Learning-based MPC. Gaussian Process Regression has been demonstrated to be a prime approach to tackle the problem of *learning dynamics* to be employed in the NMPC's internal model, as highlighted in reviews on LbMPC [3], according to which GP regression is the most commonly employed technique, as it provides a flexible stochastic non-parametric approach to modeling. It is known for not being prone to overfitting and its probabilistic nature lends itself to active learning strategies [14] and to advanced frameworks like stochastic MPC (SMPC), where the stochastic state distributions are propagated over the prediction horizon.
GP-based strategies have already led to various success stories, also in the applicative field, including the automotive world [4] and the robotics field [5]. Their usefulness has also led to the development of dedicated toolboxes, including the one developed at the University of Padua, namely *LbMATMPC* [15], whose framework has been at the basis of this thesis with proper alterations.
This chapter is thus devoted to introducing the mathematical tools and the general concepts related to Gaussian Process Regression, with a particular focus on the model approximations that are found in literature and that are best suited for application in a demanding scenario like the one posed by NMPC, especially when applied to high performance motorcycle riding.
Finally, an overview of how GPR is integrated in the general framework introduced in Chapter 2 will be presented.

3.1 GAUSSIAN PROCESS REGRESSION

The reference used to introduce the general framework of Gaussian Processes and their application to the (dynamics) regression problem is the seminal work by Rasmussen and Williams [16]. In particular, an insightful understanding of GP inference may be gained by taking the so-called *function-space view*, according to which a GP may be used to describe a distribution over function. In such sense, it must be recalled that a *Gaussian Process* is a collection of random variables, any finite number of which have a joint Gaussian distribution. A GP $f(\boldsymbol{x})$ (in our case the learnt dynamics) may be

completely defined through a mean function $m(\boldsymbol{x})$ and a covariance function $k(\boldsymbol{x}, \boldsymbol{x'})$, which take the form:

$$m(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})] \tag{3.1a}$$

$$k(\boldsymbol{x}, \boldsymbol{x'}) = \mathbb{E}[(f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\boldsymbol{x'}) - m(\boldsymbol{x'}))] \tag{3.1b}$$

leading to the general notation

$$f(\boldsymbol{x}) \sim GP(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x'})) \tag{3.2}$$

The random variables represent the value of the function $f(\boldsymbol{x})$ at the location $\boldsymbol{x}$, the input for regression, which for the purpose of this thesis can be considered as a direct transformation of the state and control quantities of the motorcycle system ($\boldsymbol{\xi}$ and $\boldsymbol{u}$ respectively).
Starting from this general framework, the simple formulas to be used for prediction may be derived by incorporating the knowledge that the training data provides about the function. The simplest case to be considered is when the observations in the training dataset are noise-free, i.e. the training dataset is of the type $\{(\boldsymbol{x}_i, f_i)|i = 1, ..., n\}$. In this case, it is possible to derive the joint distribution of the training outputs, denoted as $\mathbf{f}$, and the test outputs, denoted as $\mathbf{f}_*$. According to the prior and assuming the mean function to be $m(\boldsymbol{x}) \equiv 0$, the joint distribution is

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(X, X) & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right) \tag{3.3}$$

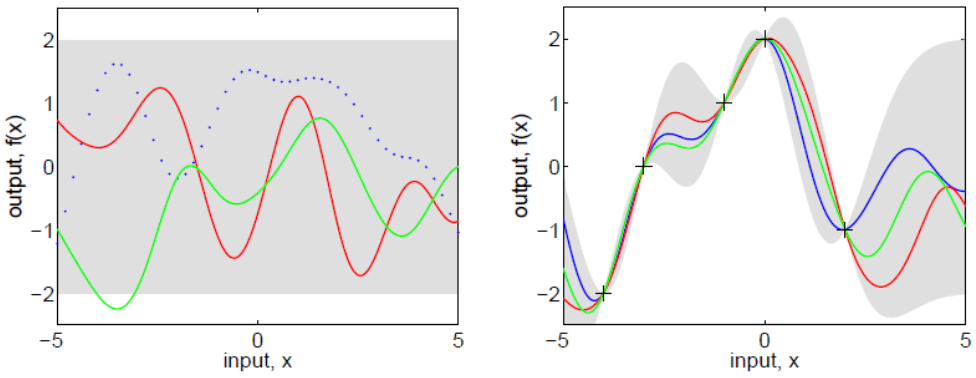
Assuming there are n training points and n_* test points $K(X, X_*)$ denotes the $n \times n_*$ matrix comprising the covariances evaluated for all pairs of training and test points; in particular, given the training points $\{\boldsymbol{x}_i, \quad i = 1, ..., n\}$ and the test points $\{\boldsymbol{x}_{*,j}, \quad j = 1, ..., n_*\}$, the covariance matrix $K(X, X_*)$ will be

$$K(X, X_*) = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_{*,1}) & \dots & k(\boldsymbol{x}_1, \boldsymbol{x}_{*,n_*}) \\ \dots & \dots & \dots \\ k(\boldsymbol{x}_n, \boldsymbol{x}_{*,1}) & \dots & k(\boldsymbol{x}_n, \boldsymbol{x}_{*,n_*}) \end{bmatrix} \tag{3.4}$$

The other covariance entries $K(X, X)$, $K(X_*, X_*)$ and $K(X_*, X)$ are similarly defined. What is done next is to retrieve the posterior distribution over functions by restricting the prior distribution represented by (3.2) to contain only the functions that agree with the observed data points; this process may be graphically represented as in Figure 3.1. Fortunately, this may be done easily by exploiting the Gaussianity assumption on the R.V.s $\mathbf{f}$ and $\mathbf{f}_*$, which makes it easy to *condition* the joint prior distribution on the observations to give

$$\begin{aligned} \mathbf{f}_*|X_*, X, \mathbf{f} \sim \mathcal{N}(&K(X_*, X)K(X, X)^{-1}\mathbf{f}, \\ &K(X_*, X_*) - K(X_*, X)K(X, X)^{-1}K(X, X_*)) \end{aligned} \tag{3.5}$$

The next step is to move towards a more realistic modelling setup, by assuming that we do not have direct access to function values themselves, but to their noisy versions,



(a), prior (b), posterior

Figure 3.1: Panel (a) shows 3 functions sampled from a GP prior, i.e. from $f(\boldsymbol{x}) \sim GP(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'))$. Panel (b) shows 3 random functions drawn from the posterior, i.e. the prior conditioned on the five noise-free observations. The shaded area indicates the 95% confidence region. Figure taken from [16]

denoted as $y = f(\boldsymbol{x}) + \epsilon$. One generally makes the usual assumption that ϵ is additive i.i.d. Gaussian noise having variance σ_n^2, which leads to the modified prior of the noisy observations

$$\begin{aligned} cov(y_p, y_q) &= k(\boldsymbol{x}_p, \boldsymbol{x}_q) + \sigma_n^2 \delta_{pq} \\ \text{or} \qquad cov(\boldsymbol{y}) &= K(X, X) + \sigma_n^2 I \qquad \text{more compactly} \end{aligned} \tag{3.6}$$

where δ_{pq} is the Kronecker delta. The joint distribution of the observed noisy target values and the function at the test locations thus becomes:

$$\begin{bmatrix} \boldsymbol{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X, X) + \sigma_n^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right) \tag{3.7}$$

The analogous posterior distribution of (3.5) will thus be

$$\mathbf{f}_* | X, \boldsymbol{y}, X_* \sim \mathcal{N}(\boldsymbol{\mu}_*, \Sigma_*) \tag{3.8}$$

where

$$\boldsymbol{\mu}_* \triangleq \mathbb{E}[\mathbf{f} | X, \boldsymbol{y}, X_*] = K(X_*, X)[K(X, X) + \sigma_n^2 I]^{-1} \boldsymbol{y} \tag{3.9a}$$

$$\Sigma_* = K(X_*, X_*) - K(X_*, X)[K(X, X) + \sigma_n^2 I]^{-1} K(X, X_*) \tag{3.9b}$$

A softer notation may be derived by defining $K = K(X, X)$ $K_* = K(X, X_*)$ and $\boldsymbol{k}_* = \boldsymbol{k}(\boldsymbol{x}_*, X)$ to denote the vector of covariances between a single test point and the n training points, so that equations (3.9a) and (3.9b) become

$$\boldsymbol{\mu}_* = \boldsymbol{k}_* (K + \sigma_n^2 I)^{-1} \boldsymbol{y} \tag{3.10a}$$

$$\mathcal{V}[f_*] = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_* (K + \sigma_n^2 I)^{-1} \boldsymbol{k}_*{}^T \tag{3.10b}$$

This notation makes it clear that the mean prediction can be seen as a linear combination of n kernel functions, each one centered on a training point:

$$\boldsymbol{\mu}_* = \sum_{i=1}^{n} k(\boldsymbol{x}_*, x_i) \alpha_i = \boldsymbol{k}_* \boldsymbol{\alpha} \tag{3.11}$$

where $\alpha = (K + \sigma_n^2)^{-1}\boldsymbol{y}$. This is a notation that will be used recurrently, as it relates to an efficient way to compute predictions. It also highlights how a GP may be represented in terms of (possibly infinite) basis functions, which is a manifestation of the *representer theorem* [17].
Through a simple analysis of eq. (3.9b) it can be noted how the predictive variance does not depend on the observed targets but only on the inputs, which is a property of the Gaussian distribution. One can also see how it is the difference between two terms, with $K(X_*, X_*)$ being the prior covariance and with $K(X_*, X)[K(X, X) + \sigma_n^2 I]^{-1}K(X, X_*)$ being the information that the observations give about the function.

3.1.1 *Covariance functions*

As anticipated in the previous section, a Gaussian Process is defined by two main elements: the mean function, which in many cases is chosen to be equal to 0, and the covariance function, which is the most crucial ingredient, as it encodes the assumptions on the function that has to be learnt. The basic idea is that it encodes a measure of nearness or similarity under the GP view. It is interesting to consider a pair of covariance functions that are of interest for the purpose of dynamics learning.

3.1.1.1 *The Squared Exponential kernel*

The most commonly used covariance function is the *squared exponential* (SE), which has the general form

$$\begin{aligned} k_{\text{SE}}(\boldsymbol{x}, \boldsymbol{x'}) &= \sigma_f^2 \cdot exp\left(-\frac{1}{2}||\boldsymbol{x} - \boldsymbol{x'}||^2_{\Gamma^{-1}}\right) \\ &= \sigma_f^2 \cdot exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{x'})^T\Gamma^{-1}(\boldsymbol{x} - \boldsymbol{x'})\right) \\ &= \sigma_f^2 \cdot exp\left(-\sum_{i=1}^{D}\frac{([\boldsymbol{x}]_i - [\boldsymbol{x'}]_i)^2}{2l_i^2}\right) \end{aligned} \tag{3.12}$$

where the matrix

$$\Gamma = \begin{bmatrix} l_1^2 & 0 & 0 & \dots & 0 \\ 0 & l_2^2 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & l_{D-1}^2 & 0 \\ 0 & \dots & 0 & 0 & l_D^2 \end{bmatrix} \tag{3.13}$$

has as its diagonal elements the so-called length-scales, which determine the influence of each input feature ($i = 1, ..., D$) on the similarity measure expressed by the covariance. σ_f^2 instead represents the overall GP variance and is also known as amplitude. An exponential kernel results in a smooth prior and is generally preferred when no specific knowledge on the modelled function properties is available.

Other kernels of interest exist. Among those that are useful for dynamics modeling, the *Matérn* one must certainly mentioned, as it is able to gauge high-frequency dynamics that the SE kernel is not always able to interpret.
A focused research into kernels different from the SE kernel was avoided as other directions were favored. Still, future research should be concerned with the study of covariance functions as the kernel's choice is of the uttermost importance in determining the properties of the GP model. One can use simple kernels and also their combination (e.g. the sum or the product of kernels, which is still a valid kernel) in order to embed prior knowledge on the properties of the (dynamics) function that needs to be learnt.

3.1.2 *GP Hyperparameters*

As can be seen from the introductory section on GP covariance functions, there are some free parameters that need to be set. One can take as example the previously introduced SE kernel, which has also been the main covariance function of interest for this thesis in view of its ease of use. Looking at kernel (3.12), rewritten for the noisy targets y rather than for the underlying function f, i.e.

$$k_{\text{SE}}(\boldsymbol{x}, \boldsymbol{x'}) = \sigma_f^2 \cdot exp\left(-\sum_{i=1}^{D} \frac{([\boldsymbol{x}]_i - [\boldsymbol{x'}]_i)^2}{2l_i^2}\right) + \sigma_n^2 \delta_{\boldsymbol{x}\boldsymbol{x'}} \tag{3.14}$$

the following adjustable parameters can be identified

1. the length-scales l_i
2. the signal variance σ_f^2
3. the noise variance σ_n^2

which are referred to as *hyperparameters*. They are fundamental as they directly influence the properties of the predictive mean and variance structure of the GP model.

A simple example of how they influence the nature of the modelled function is shown in Figure 3.2, where a 1-dimensional toy example is shown. In particular, data generated through a SE kernel with $(l, \sigma_f, \sigma_n) = (1, 1, 0.1)$ is fit by using GP models using different hyperparameters sets, specifically:

(a) $(l, \sigma_f, \sigma_n) = (1, 1, 0.1)$

(b) $(l, \sigma_f, \sigma_n) = (0.3, 1.08, 5 \cdot 10^{-5})$

(c) $(l, \sigma_f, \sigma_n) = (3, 1.16, 0.89)$

When using the right hyperparameters (i.e. Figure 3.2(a)) it is possible to retrieve a proper predictive model, characterised by a good fit of the data and by low variance when sufficiently close to the regions explored by training data.
If instead predictions are made using a process having a lowered length-scale $l = 0.3$ the results are of the type shown in Figure 3.2(b). A lower length-scale is associated to

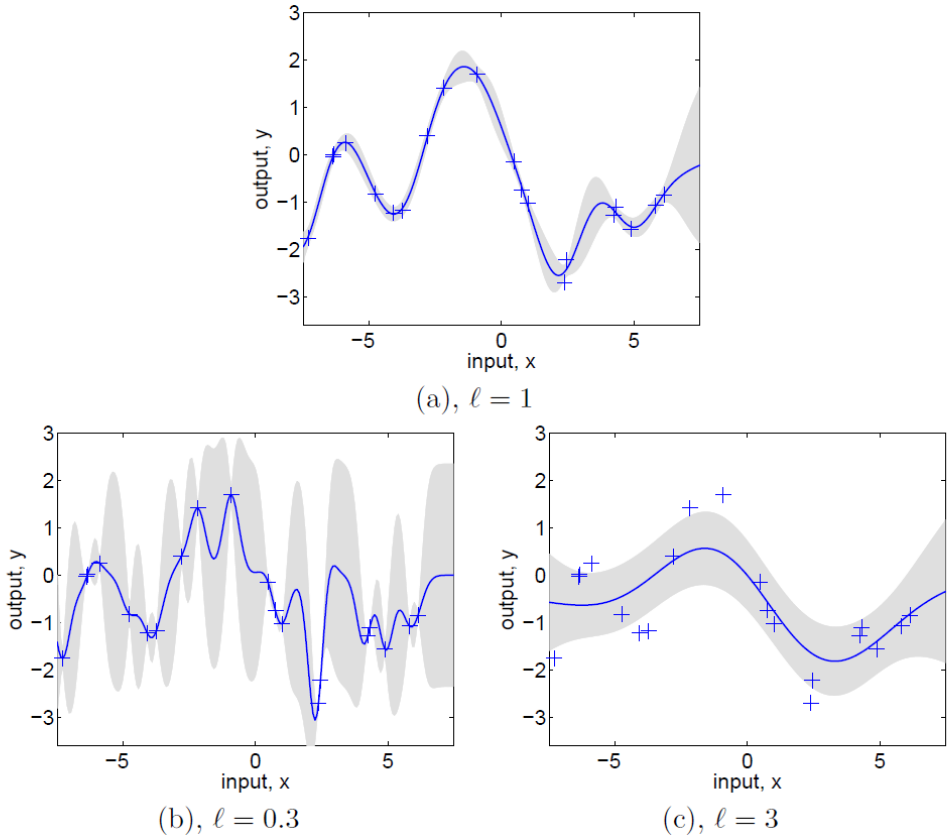


Figure 3.2: A figure (from [16]) showing the effect of varying the hyperparameters used in the GP model employed for prediction. The training data are shown as + and have been generated themselves by a GP with $(l, \sigma_f, \sigma_n) = (1, 1, 0.1)$. Panel (a) shows the predictive mean $\pm$ 2 standard deviations resulting from a prediction GP model having the exact hyperparameters $(l, \sigma_f, \sigma_n) = (1, 1, 0.1)$. Panels (b) and (c) instead show the results deriving from a suboptimal hyperparameters choice, i.e. $(l, \sigma_f, \sigma_n) = (0.3, 1.08, 5 \cdot 10^{-5})$ and $(l, \sigma_f, \sigma_n) = (3, 1.16, 0.89)$ respectively

a larger flexibility of the model, which may even fit the training data perfectly. At the same time, the resulting model may suffer from low generalizability, as shown by the exploding predictive variance in the same example, even in the vicinity of the training data.

A higher than needed length-scale $l = 3$ is associated to a smoother and slower-varying predicted function affected by a higher amount of noise, which cannot fit the presented training data properly, as shown in Figure 3.2(c).

Finding the right hyperparameters to fit data is probably the most fundamental step when using GPs for regression tasks and is thus the main requirement when training a GP for prediction. To that end, the most elegant methods that can be employed involve marginal likelihood maximisation.

3.1.3 *Model Selection (Hyperparameter Tuning) for GPR through Marginal Likelihood Optimization*

A brief overview of how hyperparameters are optimized when considering the full GP model is now presented. In particular, the general framework of marginal likelihood maximization is prioritized, as it was the method employed in the context of this thesis. Other approaches for hyper parameter tuning, like cross validation, are available.
A preliminary step is the derivation of the *marginal likelihood* (also called *evidence*) $p(\boldsymbol{y}|X))$. Mathematically speaking, it is the integral of the likelihood times the prior, i.e.

$$p(\boldsymbol{y}|X) = \int p(\boldsymbol{y}|\mathbf{f}, X)P(\mathbf{f}|X)d\mathbf{f} \tag{3.15}$$

In particular, under the Gaussian process model, the prior is Gaussian, $\mathbf{f}|X \sim \mathcal{N}(\mathbf{0}, K)$. In log terms, this corresponds to

$$\log p(\mathbf{f}|X) = -\frac{1}{2}\mathbf{f}^T K^{-1}\mathbf{f} - \frac{1}{2}\log|K| - \frac{n}{2}\log 2\pi \tag{3.16}$$

The likelihood is instead a factorized Gaussian $\boldsymbol{y}|\mathbf{f} \sim \mathcal{N}(\mathbf{f}, \sigma_n^2 I)$. By making use of formulas (A.2-A.4), it is possible to perform the integration in (3.15), yielding the log marginal likelihood

$$\begin{aligned}\log p(\boldsymbol{y}|X, \boldsymbol{\theta}) &= -\frac{1}{2}\boldsymbol{y}^T (K_y)^{-1}\boldsymbol{y} - \frac{1}{2}\log|K_y| - \frac{n}{2}\log 2\pi \\ &= -\frac{1}{2}\boldsymbol{y}^T (K + \sigma_n^2 I)^{-1}\boldsymbol{y} - \frac{1}{2}\log|K + \sigma_n^2 I| - \frac{n}{2}\log 2\pi\end{aligned} \tag{3.17}$$

where the conditioning on the hyperparameters $\boldsymbol{\theta}$ has been highlighted. It can be noted that (3.17) is also a direct consequence of $\boldsymbol{y}|X \sim \mathcal{N}(0, K + \sigma_n^2 I)$.
It is now possible to outline a model selection (hyperparameter tuning) procedure based on Bayesian principles, which provide a consistent framework for inference. Before delving deeper into that, it is interesting to highlight the roles of the three terms of the marginal likelihood in (3.17):

1. $-\frac{1}{2}\boldsymbol{y}^T K_y^{-1}\boldsymbol{y}$, which is the only term involving the observed targets, is the *data-fit* term
2. $\frac{1}{2}\log|K_y|$, which depends only on the covariance function and the inputs, is the *complexity* term
3. $\frac{n}{2}\log 2\pi$ is a normalization constant

In particular, the *data-fit* term decreases monotonically with the length-scales as the model becomes less and less flexible. The negative complexity term instead increases with the length-scales as the model becomes less complex with growing length-scales. What needs to be found is a hyperparameter configuration that accomodates both criteria, that is neither too complex (leading to overfitting) or too simple (causing underfitting).

In order to find a suitable configuration, one thus aims to maximise the marginal likelihood (equivalently minimize the negative log marginal likelihood). In order to do so, a gradient-based approach may be employed, with the gradient of (3.17) w.r.t. each hyperparameter being

$$\begin{aligned}\frac{\partial}{\partial\theta_j}\log p(\boldsymbol{y}|X,\boldsymbol{\theta}) &= \frac{1}{2}\boldsymbol{y}^T K^{-1}\frac{\partial K}{\partial\theta_j}K^{-1}\boldsymbol{y} - \frac{1}{2}\text{ tr }\left(K^{-1}\frac{\partial K}{\partial\theta_j}\right)\\ &= \frac{1}{2}\text{ tr }\left((\boldsymbol{\alpha}\boldsymbol{\alpha}^T - K^{-1})\frac{\partial K}{\partial\theta_j}\right) \quad \text{where} \quad \boldsymbol{\alpha} = K^{-1}\boldsymbol{y}\end{aligned} \tag{3.18}$$

What is done next is to use a gradient-based algorithm to retrieve the marginal likelihood maximum (the negative log likelihood minimum). It must be emphasized that the marginal likelihood may suffer from multiple local optima (each representing a different interpretation of data), even though they are not too much of a problem when employing simple covariance functions

3.2 SPARSE GP APPROXIMATIONS

Even though GPs provide a powerful Bayesian Regression framework, that is nonparametric and interpretable, they suffer from scalability problems, considering the cubic complexity of the complete model w.r.t. data size. That is why a lot of research has been devoted towards sparse GP model approximations to reduce the related computational complexity. This is especially important for online applications like the Virtual Rider, which require learning solutions that are scalable and that do not compromise the real-time feasibility of the NMPC controller. Scalable GP techniques have been reviewed in [6] and include both global approximations, which distillate the entire data, and local approximations, which divide the data for subspace learning. Through this thesis the focus has been on global approximators, while simple local approximations have been used only in the online context (see Section 3.4). In the next sections the basics of two global approximators, generally known as *FITC* (Fully Independent Training Conditional) and *VFE* (Variational Free Energy), are introduced. What they have in common is the hypothesis that the information stored in the training dataset can actually be synthesized through a set of inducing points (the so called *inducing point assumption*).

3.2.1 *The inducing input assumption*

The idea of this assumption is to use a set of m inducing points X_u having as corresponding function values $\mathbf{f_u}$. One then assumes that $\mathbf{f}$ and $\mathbf{f}_*$ are conditionally independent given $\mathbf{f_u}$, namely that

$$p(\mathbf{f},\mathbf{f}_*|\mathbf{f_u}) = p(\mathbf{f}|\mathbf{f_u})p(\mathbf{f}_*|\mathbf{f_u}) \tag{3.19}$$

The choice of the inducing points is itself a research subject; different procedures have been followed, as will become clear in the rest of the thesis work. Using the inducing point assumption, the prior distribution may be adjusted using the relation

$$p(\mathbf{f}, \mathbf{f}_*, \mathbf{f}_\mathbf{u}) = p(\mathbf{f}|\mathbf{f}_\mathbf{u})p(\mathbf{f}_*|\mathbf{f}_\mathbf{u})p(\mathbf{f}_\mathbf{u}) \tag{3.20}$$

All the probabilities appearing in (3.20) are Gaussian. This allows to formulate the following complete prior

$$p(\mathbf{f}, \mathbf{f}_*, \mathbf{f}_\mathbf{u}) \sim \mathcal{N}\left(\begin{bmatrix} m(X) \\ m(X_*) \\ m(X_u) \end{bmatrix}, \begin{bmatrix} K_{ff} & Q_{f*} & K_{fu} \\ Q_{*f} & K_{**} & K_{*u} \\ K_{uf} & K_{u*} & K_{uu} \end{bmatrix}\right) \tag{3.21}$$

where the notation $Q_{ab} = K_{au}K_{uu}^{-1}K_{ub}$ (which refers to the Nyström approximation) has been used. One can see how K_{f*} has now been substituted by Q_{f*}, which highlights how $\mathbf{f}$ (the observations) and $\mathbf{f}_*$ (the targets) communicate through $\mathbf{f}_\mathbf{u}$ (the function values at the inducing inputs' locations). The new posterior of $\mathbf{f}_*$, given the new prior, will be

$$\begin{aligned} p(\mathbf{f}_*|\mathbf{f}) &\sim \mathcal{N}(\boldsymbol{\mu}_*, \Sigma_*) \\ \boldsymbol{\mu}_* &= m(X_*) + Q_{*f}K_{ff}^{-1}(\mathbf{f} - m(X)) && \text{(3.22a)} \\ \Sigma_* &= K_{**} - Q_{*f}K_{ff}^{-1}Q_{f*} && \text{(3.22b)} \end{aligned}$$

Alternatively, by first finding the posterior distribution of $\mathbf{f}_\mathbf{u}$

$$\begin{aligned} p(\mathbf{f}_\mathbf{u}|\mathbf{f}) &\sim \mathcal{N}(\boldsymbol{\mu}_u, \Sigma_u) \\ \boldsymbol{\mu}_u &= m(X_u) + K_{uf}K_{ff}^{-1}(\mathbf{f} - m(X)) && \text{(3.23a)} \\ \Sigma_u &= K_{uu} - K_{uf}K_{ff}^{-1}K_{fu} && \text{(3.23b)} \end{aligned}$$

it is possible to obtain an alternative but equivalent formulation of (3.22a) and (3.22b)

$$\begin{aligned} \boldsymbol{\mu}_* &= m(X_*) + K_{*u}K_{uu}^{-1}(\boldsymbol{\mu}_u - m(X)) && \text{(3.24a)} \\ \Sigma_* &= K_{**} - K_{*u}K_{uu}^{-1}(K_{uu} - \Sigma_u)K_{uu}^{-1}K_{u*} && \text{(3.24b)} \end{aligned}$$

3.2.2 *The FITC GP approximation*

The rationale behind the *FITC* approximation has been developed by Snelson and Ghahramani in [18] and is well summarized in [19]. In particular, starting from the base *inducing input assumtpion*, a stronger one must be added: given $\mathbf{f}_\mathbf{u}$, also all other function values $\mathbf{f}$ are independent w.r.t. each other:

$$p(f_i, f_j|\mathbf{f}_\mathbf{u}) = p(f_i|\mathbf{f}_\mathbf{u})p(f_j|\mathbf{f}_\mathbf{u}) \tag{3.25}$$

which is the *FITC* assumption. A new prior of the form

$$p(\mathbf{f}, \mathbf{f}_*, \mathbf{f}_\mathbf{u}) \sim \mathcal{N}\left(\begin{bmatrix} m(x_1) \\ \vdots \\ m(x_n) \\ m(X_*) \\ m(X_u) \end{bmatrix}, \begin{bmatrix} K_{f_1 f_1} & \dots & Q_{f_1 f_n} & Q_{f_1 *} & K_{f_1 u} \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ Q_{f_n f_1} & \dots & K_{f_n f_n} & Q_{f_n *} & K_{f_n u} \\ Q_{* f_1} & \dots & Q_{* f_n} & K_{**} & k_{* u} \\ K_{u f_1} & \dots & K_{u f_n} & K_{u *} & K_{uu} \end{bmatrix}\right) \tag{3.26}$$

is thus derived. It is important to note that the assumption is made only for the elements of $\mathbf{f}$ and not for $\mathbf{f}_*$, as K_{**} does not need to be inverted. The covariance matrix K_{ff} from (3.21) is thus reformulated as

$$\tilde{K}_{ff} = Q_{ff} + \text{diag}(K_{ff} - Q_{ff}) = Q_{ff} + \Lambda_{ff} \tag{3.27}$$

$$\Lambda_{ff} = \text{diag}(K_{ff} - Q_{ff}) \tag{3.28}$$

while K_{**} remains unchanged. After additional manipulations (that can be found in Section A.2) the final posterior formulas are finally obtained:

$$\tilde{\boldsymbol{\mu}}_* = m(X_*) + k_{*u}\Sigma K_{uf}\Lambda^{-1}(\mathbf{f} - m(X)) \tag{3.29a}$$

$$\tilde{\Sigma}_* = K_{**} - Q_{**} + K_{*u}\Sigma K_{u*} \tag{3.29b}$$

where

$$\Sigma = (K_{uu} + K_{uf}\Lambda^{-1}K_{fu})^{-1} \tag{3.30}$$

$$\Lambda = \text{diag}(K_{ff} - Q_{ff}) \tag{3.31}$$

Note that when considering noisy observations $\boldsymbol{y}$ rather than $\mathbf{f}$, the posterior is modified by simply tweaking Λ, which becomes $\Lambda = \text{diag}(K_{ff} - Q_{ff} + \sigma_n^2 I_n)$.
It is now easy to note the significant advantage provided by the *FITC* assumption, as the only $n \times n$ matrix to be inverted is Λ, which is diagonal. The other matrices to be inverted are $m \times m$, which leads to meaningful reductions in computational complexity when choosing $m << n$ inducing points. This may be used effectively when applying GPR in an online setting like the one constituted by the virtual rider, especially when online matrix inversions are required.

3.2.3 *The VFE GP approximation*

The approach behind the *VFE* approximation (introduced in [20]) is conceptually different w.r.t. the *FITC* one as it is based on a posterior approximation (inference is approximated rather than the prior itself). In particular *VFE* inference aims at minimizing a distance between the exact posterior GP and a variational approximation. In doing so, the inducing inputs X_u become variational parameters which are rigorously selected as to minimize the distance. As a consequence, this approach inherently considers the inducing input selection as part of the GP training, which generally aims to determine

the optimal GP hyperparameters through marginal likelihood maximisation.
The first step is to approximate the posterior GP (3.9), which can also be described by means of the predictive Gaussian $p(\mathbf{f}_*|\boldsymbol{y}) = \int p(\mathbf{f}_*)p(\mathbf{f}|\boldsymbol{y})d\mathbf{f}$, where $p(\mathbf{f}_*|\mathbf{f})$ denotes the conditional prior over any finite set of function points $\mathbf{f}_*$. As done before, the objective is to approximate such Bayesian integral by using a small set of m auxiliary inducing variables $\mathbf{f_u}$ evaluated at the pseudo-inputs X_u, that are independent from the training inputs. By using the augmented joint model $p(\boldsymbol{y}|\mathbf{f})p(\mathbf{f}_*, \mathbf{f_u}, \mathbf{f})$, the predictive Gaussian may be rewritten as

$$p(\mathbf{f}_*|\boldsymbol{y}) = \int p(\mathbf{f}_*|\mathbf{f_u}, \mathbf{f})p(\mathbf{f}|\mathbf{f_u}, \boldsymbol{y})p(\mathbf{f_u}|\boldsymbol{y})d\mathbf{f}d\mathbf{f_u} \tag{3.32}$$

As a next step, it is supposed that $\mathbf{f_u}$ is a sufficient statistic for $\mathbf{f}$, namely that $\mathbf{f}_*$ and $\mathbf{f}$ are independent given $\mathbf{f_u}$, i.e. $p(\mathbf{f}_*|\mathbf{f_u}, \mathbf{f}) = p(\mathbf{f}_*|\mathbf{f_u})$, leading to the reformulation of (3.32) as

$$q(\mathbf{f}_*) = \int p(\mathbf{f}_*|\mathbf{f_u})p(\mathbf{f}|\mathbf{f_u})\phi(\mathbf{f_u})d\mathbf{f}d\mathbf{f_u} = \int p(\mathbf{f}_*|\mathbf{f_u})\phi(\mathbf{f_u})d\mathbf{f_u} = \int q(\mathbf{f}_*, \mathbf{f_u})d\mathbf{f}_u \tag{3.33}$$

where $q(\mathbf{f}_*) = p(\mathbf{f}_*|\boldsymbol{y})$ and $\phi(\mathbf{f_u}) = p(\mathbf{f}_u|\boldsymbol{y})$. In particular, the fact that $p(\mathbf{f}|\mathbf{f}_u) = p(\mathbf{f}|\mathbf{f}_u, \boldsymbol{y})$ has been used. It is true since $\boldsymbol{y}$ is just a noisy version of $\mathbf{f}$ and it has been assumed that $\mathbf{f}_*$ is conditionally independent from $\mathbf{f}$ given $\mathbf{f}_u$.
The approximation is evident by observing that in practice it is hard to find inducing variables $\mathbf{f}_u$ that are sufficient statistics, meaning that $q(\mathbf{f}_*)$ itself will always be an approximation of $p(\mathbf{f}_*|\boldsymbol{y})$. Consequently, one takes the additional step of choosing $\phi(\mathbf{f}_u)$ as a *free* variational Gaussian distribution, such that in general $\phi(\mathbf{f}_u) \neq p(\mathbf{f}_u|\boldsymbol{y})$. The variational distribution $\phi(\mathbf{f}_u)$ will depend on a mean vector $\boldsymbol{\mu}$ and a covariance matrix A, that will have to be selected. Through (3.33) it is finally possible to obtain the approximate posterior GP mean and covariance functions, which are of the type:

$$\boldsymbol{\mu}_*{}^q = \boldsymbol{\mu}^q(\boldsymbol{x}_*) = K_{*u}k_{uu}^{-1}\boldsymbol{\mu}_u \tag{3.34a}$$

$$\Sigma_*^q = K_{**} - K_{*u}K_{uu}^{-1}K_{u*} + K_{*u}BK_{u*} \tag{3.34b}$$

where $B = K_{uu}^{-1}AK_{uu}^{-1}$. This is a general sparse posterior GP that may be computed in $O(nm^2)$. The important question that *VFE* answers is how to select the ϕ distribution (namely $(\boldsymbol{\mu}_u, A)$) and the inducing inputs X_u. This is done through a variational method that allows to jointly specify these quantities and to treat X_u themselves as a variational parameter to be selected by minimizing the KL divergence.
What is done in practice is to minimize a distance between the augmented true posterior $p(\mathbf{f}, \mathbf{f}_u|\boldsymbol{y})$ and the augmented variational posterior $q(\mathbf{f}, \mathbf{f}_u)$, keeping into account that $q(\mathbf{f}, \mathbf{f}_u) = p(\mathbf{f}|\mathbf{f}_u)\phi(\mathbf{f}_u)$ from (3.33).
To determine the variational quantities (X_u, ϕ), the KL divergence $\mathbb{KL}(q(f)||p(\mathbf{f}, \mathbf{f}_u|\boldsymbol{y}))$ has to be minimized. Such minimization is equivalent to the maximization of the following variational lower bound of the true log marginal likelihood:

$$F_V(X_u, \phi) = \int p(\mathbf{f}|\mathbf{f}_u)\phi(\mathbf{f}_u) log \frac{p(\boldsymbol{y}|\mathbf{f})\cancel{p(\mathbf{f}|\mathbf{f}_u)}p(\mathbf{f}_u)}{\cancel{p(\mathbf{f}|\mathbf{f}_u)}\phi(\mathbf{f}_u)} d\mathbf{f}d\mathbf{f}_u \leq log p(\boldsymbol{y}) \tag{3.35}$$

The optimal choice for the variational distribution ϕ can be derived analytically and leads to the following revised bound:

$$F_V(X_u) = \underbrace{log[\mathcal{N}(\boldsymbol{y}|\mathbf{0}, \sigma_n^2 I + Q_{ff})]}_{\text{data fit + complexity penalty}} - \underbrace{\frac{1}{2\sigma_n^2} tr(\tilde{K})}_{\text{trace term}} \tag{3.36}$$

where

1. $Q_{ff} = K_{fu} K_{uu}^{-1} K_{uf}$ according to the previously introduced notation
2. $\tilde{K} = Cov(\mathbf{f}|\mathbf{f_u}) = K_{ff} - K_{fu} K_{uu}^{-1} K_{uf}$

The real novelty of this objective function is the regularization trace term $-\frac{1}{2\sigma_n^2} tr(\tilde{K})$. Further maximization of the bound can be achieved by optimizing over X_u and optionally over the number of inducing points, which determines the flexibility of the variational distribution $q(\mathbf{f}, \mathbf{f_u}) = p(\mathbf{f}|\mathbf{f_u})\phi(\mathbf{f_u})$ as both $p(\mathbf{f}|\mathbf{f_u})$ and the underlying optimal distribution ϕ^* are adapted when tuning X_u. In order to obtain the optimal ϕ^*, which will be needed for prediction, it is sufficient to differentiate (3.35) without constraints to obtain

$$\begin{aligned} \phi^*(\mathbf{f_u}) &\sim \mathcal{N}(\boldsymbol{\mu_u}, A) \\ \boldsymbol{\mu_u} &= \sigma_n^{-2} K_{uu} (K_{uu} + \sigma_n^{-2} K_{uf} K_{fu})^{-1} K_{uf} \boldsymbol{y} \qquad (3.37a) \\ A &= K_{uu} (K_{uu} + \sigma_n^{-2} K_{uf} K_{fu})^{-1} K_{uu} \qquad (3.37b) \end{aligned}$$

Along with (3.34), this now fully specifies the variational GP and allows for predictions at unseen input points. The predictive distribution is not novel per se, as it is exactly the one resulting from the projected process (PP) formulation [21]. In fact, it must be stressed again that the real novelty is given by how the inducing inputs and the kernel hyperparameters are selected. The additional trace term in the objective function penalises the sum of the conditional variances at the training inputs, conditioned on the inducing inputs. Intuitively, this ensures that *VFE* not only models the specific training dataset $\boldsymbol{y}$, but also approximates the covariance structure of the full GP K_{ff}. Additionally, when the variational lower bound is maximized, the hyperparameters $(\sigma_n^2, \boldsymbol{\theta})$ (in the case of the SE kernel $(\sigma_n^2, \sigma_f^2, l_1, \ldots, l_D)$) are regularized (e.g. the noise variance will tend to be higher w.r.t. other sparse approximations).

3.2.4 *The advantages of VFE for inducing input selection*

A comparison between sparse approximation methods has been conducted in [22], where the main differences between the two main approaches, namely *FITC* and *VFE*, are highlighted and will now be briefly summarized:

1. *FITC* may severely underestimate the noise variance, while *VFE* tends to overestimate it due to their respective objective functions

2. *VFE* tends to improve with additional inducing inputs while *FITC* may ignore them. This is due to the fact that adding an inducing input always leads to an improvement of the *VFE* bound while it may cause an additional penalty with the *FITC* cost function. When increasing the number of inducing variables, it will happen that *FITC* tends to place some of them on top of each other, whereas *VFE* spreads them out, tending to recover the full GP; this is highlighted by the example shown in Figure 3.3, which shows the inducing input positioning following the optimization procedure on a toy example.

3. *FITC* does not recover the full GP posterior when adding inducing inputs while *VFE* does, meaning that *FITC* may not utilise additional resources to model the data, as highlighted by the clumping effect.

4. *VFE* may be hindered by local optima and may cause underfitting. One has to put care when optimizing.

Because of the listed *FITC* pathologies, it is generally recommended to use *VFE* when looking for inducing inputs. One still has to pay attention to optimisation difficulties, which may be mitigated by careful initialisation, random restarts and other optimisation tricks.
The *FITC* approximation still provides a very useful framework that may be used in the online implementations of GPR, as will be demonstrated in Part ii.

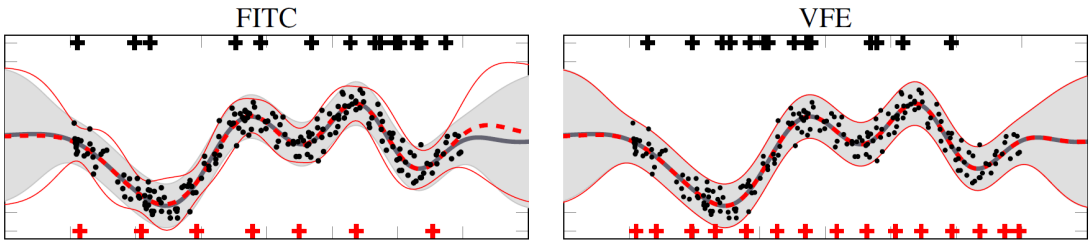


Figure 3.3: Fits for 15 inducing inputs using *FITC* and *VFE* (the black crosses being the initial configuration and the red ones the optimised one). One can see how *FITC* avoids the penalty of added inducing inputs by clumping some of them on top of each other (appearing as single red crosses) while *VFE* spreads out the inducing inputs to get closer to the true full GP posterior. The approximated models' predictive means and variances (in terms of 2 stds) are shown in red, while the true full GP posterior's ones are depicted in grey. The figure is taken from [22].

3.3 GP-BASED MPC

This section is devoted to an overview of how GP models may be used to augment or partially substitute the physics-based internal dynamics model in order to provide higher-fidelity representations of the actual motorcycle dynamics, leading to improvements in the predictions and in the virtual rider's driving behavior. While technical specifications of the derived data-based models will be deferred to the next chapter, the general approach to model augmentation is now presented.
Considering that NMPC modeling is done starting from the continuous-time model

(see Chapter 2), with the derivation of the accelerations w.r.t. the chosen minimal coordinates (denoted as $\ddot{q}$), the chosen approach is to employ continuous-time acceleration modeling also when using GPR, according to the procedure outlined in [23].
In particular, it is assumed that the real accelerations of the motorcycle system $\ddot{q}$ can be modeled by an augmented model, where the physics-based dynamics are combined with the GP-based ones and an additional i.i.d process noise, which is placed in order to explain the unmodeled dynamics. More formally, the general dynamics structure will be of the type:

$$\begin{aligned} \ddot{q}(t) &= \ddot{\tilde{q}} + \phi_{\ddot{q}}(t) + e(t) \\ \begin{bmatrix} \dot{v}_x \\ \dot{v}_y \\ \ddot{\psi} \\ \ddot{\theta} \end{bmatrix} &= \begin{bmatrix} \ddot{\tilde{v}}_x(t) \\ \ddot{\tilde{v}}_y(t) \\ \ddot{\tilde{\psi}}(t) \\ \ddot{\tilde{\theta}}(t) \end{bmatrix} + \begin{bmatrix} \phi_{\dot{v}_x}(t) \\ \phi_{\dot{v}_y}(t) \\ \phi_{\ddot{\psi}}(t) \\ \phi_{\ddot{\theta}}(t) \end{bmatrix} + \begin{bmatrix} e_{\dot{v}_x}(t) \\ e_{\dot{v}_y}(t) \\ e_{\ddot{\psi}}(t) \\ e_{\ddot{\theta}}(t) \end{bmatrix} \end{aligned} \tag{3.38}$$

where

1. $\ddot{\tilde{q}}$ represents the physics-based (white-box) dynamics
2. $\phi_{\ddot{q}}(t)$ represents the GP-modeled dynamics (grey-box if used in combination with the physics based dynamics, black-box otherwise).
3. $e(t)$ represents the process error, which for the simulated motorcycle system can describe both unmodeled dynamics and other mismatches between the true model and the Virtual Rider's one.

Note that each additional dynamics component (i.e. $\phi_{\dot{v}_x}(t)$, $\phi_{\dot{v}_y}(t)$, $\phi_{\ddot{\psi}}(t)$, $\phi_{\ddot{\theta}}(t)$) is modeled through a different GP model, as it is assumed that they are independent. This choice leads to a significant simplification of the models to be trained and allows for the possibility of augmenting just part of the dynamics (setting the other GP components to 0). Moreover, it is easy to see that, when aiming for black-box modeling of a given acceleration, it is possible to just set the corresponding white-box dynamic component (among $\ddot{\tilde{v}}_x(t)$, $\ddot{\tilde{v}}_y(t)$, $\ddot{\tilde{\psi}}(t)$, $\ddot{\tilde{\theta}}(t)$) to 0.

3.4 ONLINE LOCAL APPROXIMATIONS FOR GP-BASED MPC

This section is devoted to the exploration of local approximations, possibly based on the sparse models presented in the previous section, that may be employed in an online setting (e.g. for the Virtual Rider) by exploiting the nature of the NMPC problem. These approaches may allow to speed up computations further than sparse approximations, as they exploit a subset of the inducing variables that are selected beforehand. The strategies that were tested as part of this thesis may be referred to as the *nearest neighbor* approach and the *transductive learning* approach. This second name follows the naming convention established in [4].

3.4.1 *The nearest neighbor approach*

The *nearest neighbor* approach is quite simple in nature and has been proposed in different works, among which [24]. It uses a predictive model for the query point x_* that is based just on its nearest neighbors from a given GP model. In order to find the nearest neighbors, distances w.r.t. x_* have to be computed taking into account the length-scales of the employed GP model; more precisely, given a point x_i of the full GP model, its distance from the query point x_* will be

$$d_{*i} = ||x_* - x'||_{\Gamma^{-1}} = (x_* - x')^T \Gamma^{-1} (x_* - x') = \sqrt{\sum_{i=1}^{D} \frac{([x_*]_i - [x']_i)^2}{l_i^2}} \tag{3.39}$$

After finding the k nearest neighbors of x_* within a given set (e.g. the set of inducing points), it is possible to provide a prediction at the query location, using the full GP formulas on the neighbors' subset, i.e.

$$\mu_* = k_{*k} \boldsymbol{\alpha_k} \tag{3.40}$$

$$\mathcal{V}[f_*] = k_{**} - k_{*k}(K_{kk} + \sigma_n^2 I)^{-1} k_{k*} \tag{3.41}$$

where $\boldsymbol{\alpha_k} = (K_{kk} + \sigma_n^2 I)^{-1} \boldsymbol{y_k}$, with k representing the subset of the k nearest neighbors. It can be noted that the matrix of covariances K_{kk} may be quickly derived from the (possibly precomputed) K_{uu} matrix, by selecting the elements corresponding to the chosen neighbors.

Regarding the application to the NMPC problem, the main problem is related to the choice of of the neighbors at each control step, as the necessary query points (which are defined over the prediction horizon) are not known a priori. The nearest neighbors may be instead chosen w.r.t. the points belonging to the N-step prediction of the previous NMPC iteration or alternatively w.r.t. the current state estimation. Both choices will lead to a neighbor subset that will be closely related to the trajectory to be computed by the NMPC problem.

The advantages are apparent, as lowering the number of GP points (e.g. using 10-20 neighbors) to be employed for prediction, the complexity of the optimization problem to be solved as part of the NMPC may be significantly reduced. What remains unclear is whether such solution is sufficiently robust to guarantee a good closed-loop performance of the Virtual Rider. This will be partly addressed in Chapter 5.

3.4.2 *The transductive learning approach*

The usefulness of the so-called *transductive learning* approach in the context of GP-aided MPC is testified by the successful implementation obtained in [4]. Its main point of strength is given by the fact that it exploits the structure of the NMPC problem in order to simplify the predictive model.

The *transductive* property refers to the fact that the approximation is adjusted according to the available information on the test points (similarly to the *nearest neighbor* approach).

Information on plausible test points is provided by the approximate trajectories computed by the NMPC at the previous steps. In particular, the previous state-input trajectories may be used in order to place informative inducing variables directly. More formally, given

$$\xi_{\cdot|i-1} = \left[\xi_{0|i-1}^\top, \xi_{1|i-1}^\top, \dots, \xi_{N|i-1}^\top\right]^\top, \tag{3.42}$$

$$u_{\cdot|i-1} = \left[u_{0|i-1}^\top, u_{1|i-1}^\top, \dots, u_{N-1|i-1}^\top\right]^\top \tag{3.43}$$

which are the state-input predicted trajectory from the previous NMPC iteration, the corresponding evolution of the features to be used for regression (see (4.2)) may be summarized as

$$x_{\cdot|i-1} = \left[x_{0|i-1}^\top, x_{1|i-1}^\top, \dots, x_{N|i-1}^\top\right]^\top \tag{3.44}$$

According to the *transductive* approach, inducing variables may be placed directly along the previous iteration trajectory $x_{\cdot|i-1}$, as it is highly probable that the next NMPC trajectory (which is computed just $T_s = 10$ms later) will be in the same region. To do so, a *transduction ratio* may be defined, in order to determine how may points from the previous trajectory will be employed for *transduction*. More precisely, given a *transduction ratio* $h \in \mathbb{N}$ and $s = \lfloor N/h \rfloor$, points may be sampled from the previous trajectory as:

$$X_s = \left[x_{0|i-1}^\top, x_{h|i-1}^\top, x_{2h|i-1}^\top, \dots, x_{(s-1)\cdot h|i-1}^\top\right]^\top \tag{3.45}$$

The sampled points X_s may now be used to provide a local GP approximation to be employed for the current NMPC control action. This is done using a general (possibly already sparse) GP model based on dataset X_u and the *FITC* approximating formulas, i.e.:

$$\tilde{\boldsymbol{\mu}}_* = k_{*s}\Sigma K_{su}\Lambda^{-1}\mathbf{f} \tag{3.46}$$

$$\tilde{\Sigma}_* = K_{**} - Q_{**} + K_{*s}\Sigma K_{s*} \tag{3.47}$$

with

$$\Sigma = (K_{ss} + K_{su}\Lambda^{-1}K_{us})^{-1} \tag{3.48}$$

$$\Lambda = \operatorname{diag}(K_{uu} - Q_{uu}) \tag{3.49}$$

Using the online approximation approaches it is thus possible to provide 2 subsequent dataset reductions:

- an offline reduction (to be obtained through *VFE*) to determine the most informative points from the original dataset to be used for prediction;
- an online reduction (based on the *FITC* formulas when employing the *transductive* approach) that uses information from the previous NMPC iteration to further reduce the computational load.

Part II

EXPERIMENTAL DEVELOPMENT AND RESULTS

This part is devoted to the presentation of the main directions of research that were followed and to the analyses that were undertaken during the thesis work. The most relevant results are also shown in order to highlight the most promising lines of investigation. Thorough comparisons are then conducted in order to highlight the main differences and the advantages/ drawbacks of learning-aided control schemes w.r.t. the base white-box Virtual Rider.

4

INPUT ANALYSIS AND DYNAMICS MODEL LEARNING

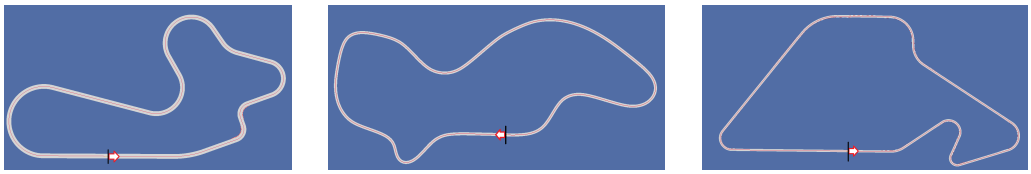
This chapter is devoted to describing the main approaches that were used in order to create and train GP-based learning models to represent unmodeled dynamics w.r.t. to the white-box ones (according to a grey-box strategy) or to represent the whole dynamics (according to a black-box strategy). The first step in this direction was to devise criteria to be used for model selection, as a fundamental objective was to construct models by selecting the portion of data (in terms of features and inducing points) that is most beneficial for the subsequent implementation into the NMPC scheme.
Formally speaking, the available dataset is of the type

$$z_{ij} = (x_i, y_{ij}) \quad x_i \in \mathbb{R}^D \quad y_{ij} \in \mathbb{R} \quad i \in 1, \dots, N \quad j \in 1, \dots, 4 \tag{4.1}$$

where

- N is the cardinality of the dataset
- x_i represent the regression input (*features*), which are originally D-dimensional
- y_{ij} represent the regression targets, which are 1-dimensional
- z_{ij} represent the whole datapoints, comprising the features x_i and the targets y_{ij}
- index j refers to the 4 different targets of interest, which are modelled as independent and, as such, require separate learning
  - $j = 1$: $\dot{v}_x$ for the black-box model, $\phi_{\dot{v}_x} = \dot{v}_x - \dot{\tilde{v}}_x$ for the grey-box one
  - $j = 2$: $\dot{v}_y$ for the black-box model, $\phi_{\dot{v}_y} = \dot{v}_y - \dot{\tilde{v}}_y$ for the grey-box one
  - $j = 3$: $\ddot{\psi}$ for the black-box model, $\phi_{\ddot{\psi}} = \ddot{\psi} - \ddot{\tilde{\psi}}$ for the grey-box one
  - $j = 4$: $\ddot{\theta}$ for the black-box model, $\phi_{\ddot{\theta}} = \ddot{\theta} - \ddot{\tilde{\theta}}$ for the grey-box one

In order to perform learning, an ample dataset was gathered. In particular, data was retrieved undertaking riding tasks along different test tracks (shown in Figure 4.1), including the *VI-track* used as the final testbed. This was done using the nominal controller presented in Chapter 2. The data that was gathered is related to the accelerations' evolution along the tracks, as well as the corresponding mismatch of the physics-based model's estimates. Data collection was performed according to the Virtual Rider control frequency $f_c = 100Hz$, using the output of the $VI - BRT$ simulation block, previously shown in Figure 2.5. This procedure led to a dataset of 24580 points, to be used for both



(a) *VI-Track* **(b)** Additional track #1 **(c)** Additional track #2

Figure 4.1: Tracks used for data generation

training and testing of grey-box and black-box regression models.
Data was gathered in order to provide continuous-time modelling for the accelerations, according to the framework already developed in [23]. This means that the features x_i and the corresponding targets y_{ij} were collected at the same time-step t.
After data collection, the following step was an *input analysis* for each regression target, as not all features are relevant for the prediction of each acceleration and the ones that do not appear as important should be discarded in order to avoid harmful or misleading correlations. Such attempt towards *model reduction* is also significative since the computational complexity associated to the NMPC problem is closely related to the complexity of the internal model itself. In such sense, the complexity of a GPR model directly depends on the dimension of the input space.
The first section of this chapter is thus devoted to an overview of the *input analysis* that was undertaken in order to find grey/black-box *minimal* models for the system accelerations.

4.1 INPUT ANALYSIS FOR THE GPR MODELS

As previously mentioned, the objective is to select the most significative features for each regression task, starting from a general set of features, given by:

$$
x = \begin{bmatrix}
\theta & \Rightarrow & \text{1. Roll angle} \\
v_x & \Rightarrow & \text{2. Longitudinal velocity} \\
v_y & \Rightarrow & \text{3. Lateral velocity} \\
\dot{\psi} & \Rightarrow & \text{4. Yaw rate of change} \\
\dot{\theta} & \Rightarrow & \text{5. Roll rate of change} \\
\delta & \Rightarrow & \text{6. Steering angle} \\
\gamma_t & \Rightarrow & \text{7. Normalized throttle input} \\
\gamma_b & \Rightarrow & \text{8. Normalized brake input} \\
y_r & \Rightarrow & \text{9. Rider lateral movement} \\
\dot{\delta} & \Rightarrow & \text{10. Steering angle rate of change} \\
\dot{\gamma}_t & \Rightarrow & \text{11. Throttle input rate of change} \\
\dot{\gamma}_b & \Rightarrow & \text{12. Brake input rate of change} \\
\dot{y}_r & \Rightarrow & \text{13. Lateral rider movement rate of change} \\
\alpha_r & \Rightarrow & \text{14. Rear slip angle} \\
\alpha_f & \Rightarrow & \text{15. Frontal slip angle} \\
c_\theta & \Rightarrow & \text{16. Cosine of the roll angle} \\
s_\theta & \Rightarrow & \text{17. Sine of the roll angle}
\end{bmatrix} \tag{4.2}
$$

The feature set includes:

- Part of the system state (θ, v_x, v_y, $\dot{\psi}$, $\dot{\theta}$, δ, γ_t, γ_b, y_r)
- The system input ($\dot{\delta}$, $\dot{\gamma}_t$, $\dot{\gamma}_b$, $\dot{y}_r$)
- Additional composite features (α_r, α_f, c_θ, s_θ)

Since the regression problem deals with nonlinear models, classical approaches to feature selection that are typical of linear regression are not suitable. Not surprisingly, it appears that a general consensus on which criteria to use for feature selection in the nonlinear context does not exist in the literature.
As a consequence, the attempts to *model reduction* were mainly related to two different strategies: one that is based on a mutual information criterion used to link each acceleration target with the most relevant features and the other one that is based on a *quality-of-fit* measure.

4.1.1 *Input analysis through mutual information criteria*

The idea for this type of analysis stems from the work presented in [25], which deals with the problem of selecting the most relevant independent variables to be used for nonlinear modelling of a given dependent variable. In particular, it highlights the difficulty of trying to force a dimensionality reduction of the input space through projection methods, as linear projections will generally not be appropriate and will lead to an interpretability loss, especially in the nonlinear context. As a consequence, a simple selection among the available variables should instead be preferred. To that end, the proposed method is a selection procedure based on the information theory concept of *mutual information*, with the stated objective of measuring the amount of information contained in a variable or a group of variables in order to predict the dependent one. The main advantages of this approach is that it is *model-independent* (no assumption is made about the model to be used for regression) and *nonlinear* (it measures nonlinear relationships, contrarily to correlation, which only measures linear ones).

4.1.1.1 *Mutual information*

The general concept of uncertainty of random variables is given by Shannon's information theory [26]. Given the random variables X and Y, their joint probability density function may be denoted as $\mu_{X,Y}$. The relative marginal density functions are given by

$$\mu_X(x) = \int \mu_{X,Y}(x,y)dy \tag{4.3a}$$

$$\mu_Y(y) = \int \mu_{X,Y}(x,y)dx \tag{4.3b}$$

The uncertainty on the variable Y is given by its entropy, which is defined as

$$H(Y) = -\int \mu_Y(y)\log\mu_{X,Y}(x,y)dy \tag{4.4}$$

If knowing X provides indirect knowledge on Y, then the uncertainty on Y knowing X is given by the conditional entropy, which may be formulated as

$$H(X|Y) = -\int \mu_X(x)\int \mu_Y(y|X=x)\log\mu_Y(y|X=x)dydx \tag{4.5}$$

Finally, there is the joint uncertainty of the (X,Y) pair, which is given by the joint entropy

$$H(X,Y) = -\int \mu_{X,Y}(x,y)\log\mu_{X,Y}(x,y)dxdy \tag{4.6}$$

Now, the mutual information between X and Y is essentially a measure of the amount of knowledge that X provides on Y. It is thus expressed as

$$I(X,Y) = H(Y) - H(Y|X) \tag{4.7}$$

namely the reduction of the uncertainty on Y when X is known. When Y is the dependent variable in a prediction context, $I(X,Y)$ measures the pertinence of X in a model employed for the regression of Y. Through the properties of entropy, the mutual information may be finally rewritten as

$$I(X,Y) = H(X) + H(Y) - H(X,Y) = \int \mu_{X,Y}(x,y)\log\frac{\mu_{X,Y}(x,y)}{\mu_X(x)\mu_Y(y)}dxdy \tag{4.8}$$

By (4.3) and (4.8), it can be concluded that estimating $\mu_{X,Y}$ is sufficient in order to estimate the mutual information between X and Y.

4.1.1.2 *Estimation of the mutual information*

As previously mentioned, the estimate of the mutual information (MI) relies on the estimation of the joint pdf of (X,Y), which may be conducted on the available dataset. Since the mutual information index must be computed on a high-dimensional feature space, histogram and kernel-based pdf estimations are not convenient, as they suffer from the curse of dimensionality. Due to this reason, the paper in question [25] suggests an MI estimate based on a k-nearest neighbor statistic. For the presentation of the associated algorithm, the set of real-valued independent variables will now denoted through a unique vector-valued variable X.

The algorithm for MI estimation is based on the available dataset, composed of the input-output pairs $z_i = (x_i, y_i)$, not to be confused with the ones introduced in (4.1), as x_i now indicates a subset of features of the whole feature vector (4.2) to be used for the prediction of the target y (among the targets of interest introduced at the beginning of the chapter). Basically, index j, indicating a given regression target, has been dropped for a lighter notation. The procedure that is described below will then be performed independently for each target of interest. Note that datapoints $z_i = (x_i, y_i)$ are assumed

to be i.i.d. realizations of a R.V. $Z(X,Y)$ with pdf $\mu_{X,Y}$.
Input-output pairs are to be compared through the maximum norm: given $z_i = (x_i, y_i)$ and $z_l = (x_l, y_l)$, such norm is defined as

$$||z_i - z_l||_\infty = max(||x_i - x_l||, ||y_i - y_l||) \tag{4.9}$$

where $||\cdot||$ indicates a suitable norm for the input and output space (with the euclidean one being the most immediate, even though others may be chosen based on the available domain knowledge). The idea is then to estimate $I(X,Y)$ using the average distance of z_i from its k-nearest neighbors in the X, Y and Z spaces, to be averaged over all z_*.
We may now consider the point z_i and the k-th nearest neighbor $z_{k(i)} = (x_{k(i)}, y_{k(i)})$ (according to the maximum norm in (4.9)). It can be noted that $x_{k(i)}$ and $y_{k(i)}$ are the input and output parts of $z_{k(i)}$ and thus not necessarily the k^{th} nearest neighbors of x_i and y_i. Finally, the following distances are defined:

- $\epsilon_i = ||z_i - z_{k(i)}||_\infty = max(\epsilon_i^X, \epsilon_i^Y)$
- $\epsilon_i^X = ||x_i - x_{k(i)}||$
- $\epsilon_i^Y = ||y_i - y_{k(i)}||$

Using such distances one may count

- the number n_i^X of points x_l whose distance from x_i is strictly less than ϵ_i
- the number n_i^Y of points y_l whose distance from y_i is strictly less than ϵ_i

It has been proven in [27] that $I(X,Y)$ may now be accurately estimated as

$$\hat{I}(X,Y) = \Psi(k) - \frac{1}{N}\sum_{i=1}^{N}\left[\Psi(n_i^X + 1) + \Psi(n_i^Y + 1)\right] + \Psi(N) \tag{4.10}$$

where Ψ is the digamma function given by

$$\Psi(t) = \frac{\Gamma'(t)}{\Gamma(t)} = \frac{d}{dt}ln\Gamma(t) \tag{4.11}$$

with

$$\Gamma(t) = \int_0^\infty u^{t-1}e^{-u}du \tag{4.12}$$

The quality of the estimator $\hat{I}(X,Y)$ is linked to the value chosen for k, with small values of k leading to large variance and small bias and large values of k causing small variance and large bias. A suggested mid-range value is $k = 6$.

4.1.1.3 *Application of the information-based criterion to the motorcycle dataset*

The information criterion presented in the previous section has been employed in order to evaluate the significance of the features to be used for each regression task. Since the objective is to find the subsets of features that best represent a given regression target, it makes sense to employ the mutual information criterion in order to establish

the most informative subset of x to be used for the corresponding black-box/grey-box target y_j. A general search among all possible input features combinations is generally unfeasible, as it requires the computation of the mutual information between each target and 2^{17} possible feature subsets. Nevertheless, since the subsets of interest should have a sufficiently low cardinality in order to be successfully applied in the NMPC scenario without aggravating the computational load disproportionately, it may be advisable to perform the search for the most informative features among combinations of $m \leq D$ features.

Consequently, the estimate $\hat{I}(x, y)$ (see (4.10)) was computed for all combinations (x) of less than 6 features and w.r.t. each regression target y (grey/black-box $\dot{v}_x$ dynamics, grey/black-box $\dot{v}_y$ dynamics, grey/black-box $\ddot{\psi}$ dynamics, grey/black-box $\ddot{\theta}$ dynamics). Note that the final closed-loop implementation of the learning-based models made use of just those related to $\ddot{\psi}$ and $\ddot{\theta}$, as will be further discussed in Chapter 5. For this reason, only the results associated to those accelerations will be shown from here on, while the analyses associated to $\dot{v}_x$ and $\dot{v}_y$ will be deferred to Appendix C.

The combinations were ordered according to the corresponding mutual information content, in order to determine the best feature subset of a given cardinality for each acceleration target. The results are summarized in Table 4.1 (for black-box targets) and Table 4.2 (for grey-box targets). For each target, the 3 best feature combinations of a given cardinality (from 1 to 5) are shown, together with the corresponding mutual information value.

In particular, features are denoted according to a reference number, in accord with the convention introduced in (4.2).

A quick look at the summary tables for the mutual information analysis leads to some straightforward observations:

- A single input feature is not generally very informative for a given acceleration target, which could be expected considering the complexity of dynamics learning.
- In terms of mutual information content, combinations made of 2-3 features seem to be preferable, with information content slightly reducing with a higher number of features.
- For each feature cardinality (from 1 to 5), there generally are multiple "competitive" combinations, having close mutual information indexes.
- There generally appears to be a consensus on some of the features that are most useful for a given regression target. An example is provided by features [1,6] (roll angle and steering angle respectively) appearing in all of the most informative feature combinations for the grey-box target $\phi_{\ddot{\psi}}$.

The most informative combinations pertaining to each feature number have been selected in order to be tested in terms of actual fitting capabilities, as will be shown later in the chapter.

Table 4.1: Feature combinations with the highest information content for the **black-box** targets. For each black-box target ($\ddot{\psi}$, $\ddot{\theta}$), the 3 best combinations of 1, 2, 3, 4 and 5 features are shown, along with the corresponding mutual information index.

TARGETS	$\ddot{\psi}$			$\ddot{\theta}$		
	1st	2nd	3rd	1st	2nd	3rd
1 FEAT	[6]	[5]	[15]	[4]	[6]	[14]
	1.1931	1.1691	1.1301	1.1275	1.0933	1.0587
2 FEAT	[5, 6]	[4, 5]	[2, 5]	[5, 6]	[4, 5]	[3, 5]
	2.321	2.2822	2.2575	2.1997	2.1986	2.1975
3 FEAT	[2, 5, 6]	[2, 5, 15]	[1, 2, 5]	[2, 3, 5]	[2, 5, 14]	[2, 4, 5]
	2.2105	2.1978	2.1901	2.1767	2.1717	2.1602
4 FEAT	[2, 5, 6, 14]	[1, 2, 5, 7]	[2, 5, 8, 14]	[2, 3, 4, 5]	[2, 5, 6, 14]	[2, 3, 5, 6]
	2.0149	2.013	2.0109	1.9669	1.964	1.9639
5 FEAT	[4, 5, 8, 11, 15]	[4, 5, 11, 12, 15]	[4, 5, 6, 8, 11]	[2, 3, 5, 6, 12]	[2, 5, 6, 12, 14]	[2, 3, 4, 5, 12]
	1.8934	1.8879	1.8821	1.8195	1.818	1.8171

Table 4.2: Feature combinations with the highest information content for the **grey-box** targets. For each grey-box target ($\phi_{\ddot{\psi}}$, $\phi_{\ddot{\theta}}$), the 3 best combinations of 1, 2, 3, 4 and 5 features are shown, along with the corresponding mutual information index.

TARGETS	$\phi_{\ddot{\psi}}$			$\phi_{\ddot{\theta}}$		
	1st	2nd	3rd	1st	2nd	3rd
1 FEAT	[1]	[4]	[15]	[4]	[6]	[1]
	2.3659	2.1769	2.1348	1.6752	1.6495	1.6309
2 FEAT	[1, 6]	[1, 3]	[1, 2]	[2, 15]	[4, 5]	[2, 3]
	3.4268	3.4046	3.3634	2.9226	2.9105	2.9096
3 FEAT	[1, 6, 14]	[1, 3, 15]	[1, 6, 15]	[2, 3, 6]	[2, 4, 5]	[2, 6, 14]
	3.3465	3.3339	3.3325	2.8589	2.8536	2.847
4 FEAT	[1, 6, 14, 15]	[1, 4, 14, 15]	[1, 6, 7, 15]	[2, 4, 8, 15]	[2, 4, 6, 8]	[1, 2, 6, 8]
	3.1314	3.119	3.1154	2.6904	2.6835	2.6739
5 FEAT	[1, 4, 6, 14, 15]	[1, 6, 8, 14, 15]	[1, 4, 8, 14, 15]	[1, 2, 4, 8, 15]	[2, 3, 4, 6, 8]	[2, 3, 6, 8, 9]
	2.9273	2.9224	2.9217	2.5067	2.5062	2.5027

4.1.2 *Input analysis through a fitting quality measure*

A simpler method was also used to find alternative model frameworks in terms of input features. In particular, in this second solution full GP models were trained on a subset of the complete dataset, comprising 2000 points, by performing exact inference, according to the framework of Section 3.1.3. In order to do so, the *MATLAB* toolbox *gpml*, provided by Rasmussen [16], was employed. In terms of inference, the standard Squared Exponential kernel and the Gaussian likelihood function were used.
In order to perform feature selection, the following pipeline is followed. Any given model to be evaluated is trained on the 2000 random points in order to obtain a suitable hyperparameters' configuration. The trained model is then tested on the complete dataset, by evaluating its prediction capabilities on possibly unseen data. The fit quality is finally measured through the average R^2 index (also known as *coefficient of determination*) across the data resulting from each of the 3 tracks. In particular, given a track's data, the R^2 index is computed comparing the real target evolution y and the predicted evolution $\hat{y}$ as follows:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \tag{4.13}$$

where

$$SS_{res} = \sum_i (y_i - \hat{y}_i)^2 = \sum_i e_i^2 \qquad \text{(residual sum of squares)} \tag{4.14a}$$

$$SS_{tot} = \sum_i (y_i - \bar{y})^2 \qquad \text{(total sum of squares)} \tag{4.14b}$$

$$\bar{y} = \frac{1}{N} \sum_{i=1}^{N} y_i \tag{4.14c}$$

As is known, the best case scenario is when the predicted values exactly match the observed ones, corresponding to $SS_{res} = 0$ and $R^2 = 1$. Baseline models, always correctly predicting $\bar{y}$ will have $R^2 = 0$, while worse models will have a negative R^2. Since this evaluation procedure would be unfeasible on all the 2^{17} possible feature subsets, it is advisable to instead employ an iterative approach. In order to construct subsets that provide a good general fit, it is possible to start by selecting the best single feature and then continue by adding the features that provide the biggest improvement to the fitting properties of the model. This study is repeated for all regression targets, in order to determine the sequence of significant features, while monitoring the possibly beneficial impact of newly added features. The results are reported in Table 4.3 (for the black-box models) and Table 4.4 (for the grey-box models), where the optimal choice for the feature subset of a given cardinality is shown along with the corresponding fit measure. The fit measure of a model is given by $\overline{R^2}$, the average R^2 index that the GP model based on the 2000-point training set attains on the data obtained from the 3 tracks.

Again, since the main objective is to obtain models that may be easily implemented in an online scenario, meaning that they do not add too much to the optimization problem complexity, the search is limited to models including a low number of features.
Only models consisting of $m \leq 7$ features are thus reported, even though the most interesting (from an implementability point of view) are the ones employing less than 5/6 features. The feature reference numbers are again related to the convention shown in (4.2). Note that the corresponding results for $\dot{v}_x$ and $\dot{v}_x$ can be found in Table C.3 and Table C.4.

Table 4.3: Feature combinations providing the best fit for the **black-box** targets on the whole dataset after training on a subset of 2000 points. The fit measure is given by $\overline{R^2}$, the average R^2 index over the 3 tracks' data, which is reported for each feature combination. For each black-box target ($\ddot{\psi}$, $\ddot{\theta}$), combinations of up to 7 features are shown.

TARGETS	$\ddot{\psi}$	$\ddot{\theta}$
1 Feat	[5]	[4]
	0.3660	0.1838
2 Feat	[5,4]	[4,2]
	0.5018	0.5622
3 Feat	[5,4,12]	[4,2,9]
	0.6192	0.7377
4 Feat	[5,4,12,1]	[4,2,9,1]
	0.7006	0.7283
5 Feat	[5,4,12,1,7]	[4,2,9,1,12]
	0.6238	0.7999
6 Feat	[5,4,12,1,7,14]	[4,2,9,1,12,8]
	0.6443	0.8152
7 Feat	[5,4,12,1,7,14,13]	[4,2,9,1,12,8,6]
	0.8192	0.8484

Table 4.4: Feature combinations providing the best fit for the **grey-box** targets on the whole dataset after training on a subset of 2000 points. The fit measure is given by $\overline{R^2}$, the average R^2 index over the 3 tracks' data, which is reported for each feature combination. For each grey-box target ($\ddot{\psi}$, $\ddot{\theta}$), combinations of up to 7 features are shown.

TARGETS	$\phi_{\ddot{\psi}}$	$\phi_{\ddot{\theta}}$
1 Feat	[1]	[1]
	-0.0194	-0.6903

2 Feat	[1,5]	[1,8]
	0.3459	0.5156
3 Feat	[1,5,2]	[1,8,2]
	0.6280	0.6998
4 Feat	[1,5,2,9]	[1,8,2,6]
	0.6588	0.7857
5 Feat	[1,5,2,9,17]	[1,8,2,6,4]
	0.6970	0.8425
6 Feat	[1,5,2,9,17,4]	[1,8,2,6,4,13]
	0.7142	0.8740
7 Feat	[1,5,2,9,17,4,8]	[1,8,2,6,4,13,7]
	0.7332	0.8819

A brief analysis of the tables reported above may lead to the following considerations:

- Contrary to the information criterion, according to which the preferable feature combinations should include $2-4$ features, the fit criterion suggests that adding more relevant features generally leads to improvements in the fit measure. This is true for both grey-box and black-box targets.
- The addition of the first features to the GP model leads to the biggest increases in the fit measure, while later additions appear as less significant, with the fit measure tending to plateau. It should thus be possible to assume that models employing less than 6 features are sufficiently representative for the acceleration targets.
- Grey-box models with a low number of features (e.g. 1-2) do not compare well with the corresponding black-box models. This may be related to a more demanding regression task, as grey-box targets may include both unmodeled dynamics and inaccuracies of the white-box model. Conversely, grey-box and black-box models based on a higher number of features lead to a similar fit measure.

4.2 COMPARATIVE ANALYSIS OF THE DYNAMICS MODELS

Two separate feature analyses have been conducted, one based on a mutual information criterion and the other based on a fit measure computed on the available dataset. The preliminary results of those analyses have been reported in the previous sections. It is now useful to conduct a comparative analysis between the found models on a common framework. In particular, considering the final objective of applying the GP models in the context of the Virtual Rider, it makes sense to move towards the sparse approximate

models that will be used in practice. In such sense, the first step is to distill the information contained in the available dataset into GP model approximations that employ a low amount of inducing points (e.g. 50, 100). The general framework is the *VFE* (Variational Free Energy) one that has been presented in Section 3.2.3, meaning that a simpler GP model representation is obtained by means of inference approximation, while the prior is kept unchanged. In particular, the inference procedure proposed in [20] considers the selection of the inducing inputs as part of the optimization procedure and leads to an approximation of the full GP model that is in general superior to other approximation methods (see Section 3.2.4).
In order to apply the *VFE* inference procedure, the *MATLAB* toolbox provided by Titsias [20] was employed.[1] The provided toolbox allows to create a Variational Sparse Gaussian Process (VSGP) model object, with the possibility of specifying:

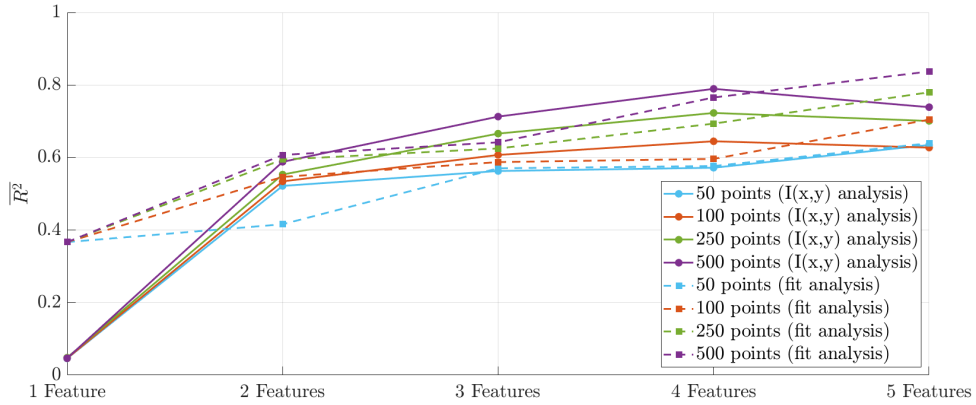
- the objective function, which can be either the variational objective from (3.36) or a custom alteration;
- the likelihood function (the Gaussian one was used as it is preferable when performing regression);
- the number m of inducing inputs to be placed through inference;
- the kernel to be used (with the SE kernel being the preferred choice for the scope of this thesis).

It is then possible to proceed with model optimization, after choosing:

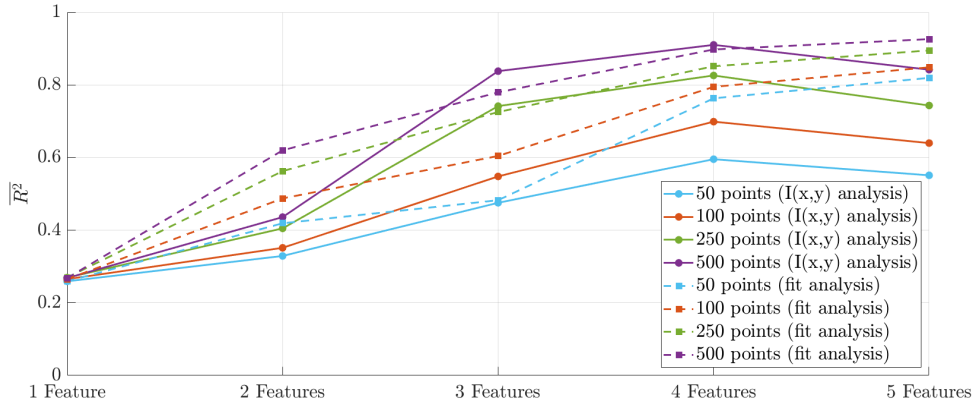
- the number of objective function evaluations to be performed;
- the variables to be optimized, according to 3 possible options:
  - joint optimization of model hyperparameters (for the kernel and the likelihood) and inducing variables. This is the main case of interest.
  - optimization of just the inducing variables.
  - optimization of just the hyperparameters.

For the comparative analysis and the creation of the desired VSGP models, joint optimization was employed. In particular, the best feature combinations from the previous analyses were considered and compared. At the same time, VSGP models with a variable amount of inducing inputs were trained in order to assess their influence on the reduced model performance. Models based on 50, 100, 250 and 500 pseudo-inputs were tested for each regression task and for each candidate feature combination. The results are reported in Figure 4.2 (black-box) and in Figure 4.3 (grey-box), where the average fit ($\overline{R^2}$) attained by the sparse GP models over the three tracks' data is reported.
Note that y-axis scaling may vary in order to guarantee better visibility of the graphs. The corresponding plots for $\dot{v}_x$ and $\dot{v}_y$ can be found in the appendix (Figure C.1 and Figure C.2).

1 The toolbox in question is available @ https://www2.aueb.gr/users/mtitsias/code/varsgp.tar.gz

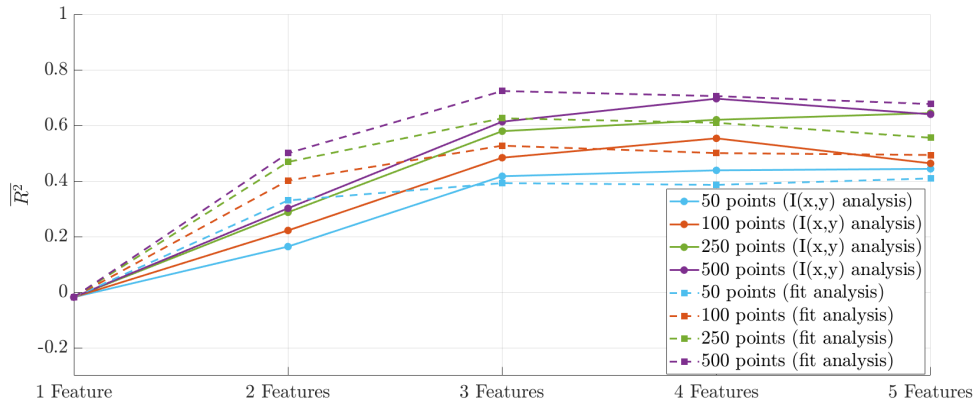


(a) $\overline{R^2}$ index for $\ddot{\psi}$

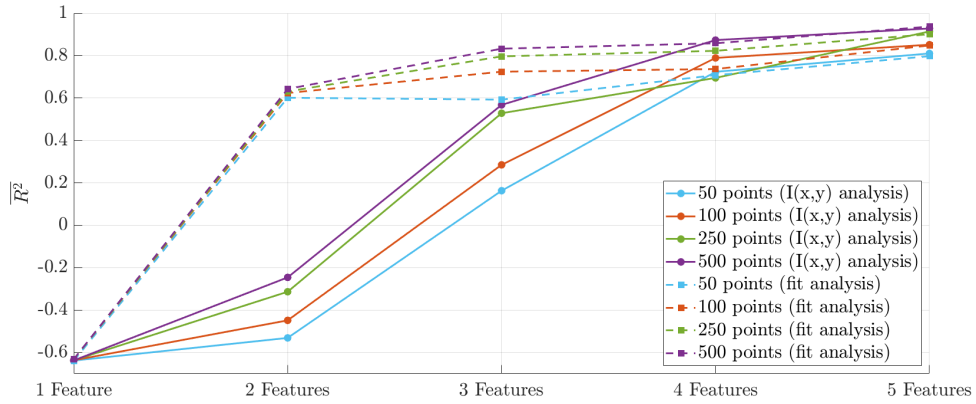


(b) $\overline{R^2}$ index for $\ddot{\theta}$

Figure 4.2: Results of the comparative analysis of the VSGP models for the **black-box** targets



(a) $\overline{R^2}$ index for $\phi_{\ddot{\psi}}$



(b) $\overline{R^2}$ index for $\phi_{\ddot{\theta}}$

Figure 4.3: Results of the comparative analysis of the VSGP models for the **grey-box** targets

Some considerations on the results of the comparative analysis of the VSGP models trained on the best feature combinations are now in order:

- Feature combinations with a low numerosity (e.g. 1-2) resulting from the information analysis generally do not compare well with the corresponding feature combinations resulting from the fit analysis. Feature combinations derived from the information criterion are instead very competitive when the numerosity is higher, often outperforming the fit-based counterparts (especially in terms of grey-box targets). This seems to confirm the validity of the information criterion as a feature selection approach.

- The addition of features generally leads to better performing models, with some exceptions, like $\phi_{\ddot{\psi}}$ and the information-based models for $\dot{\psi}$ and $\ddot{\theta}$. In general, it must be said that the impact of a higher number of features tends to be progressively less significant.

- The number of inducing variables has an impact on the performance of the VSGP models, as it is known that when their number is increased, *VFE* inference tends to reconstruct the full GP model. Still, it is interesting to observe that the best VSGP models employing 50, 100 points are generally quite close to the best models based on 250, 500 points in terms of fit quality. This is of the utmost importance when considering the importance of finding inexpensive models to be implemented in the Virtual Rider.

- Comparing grey-box and black-box strategies, it generally appears that the latter provides an all-around better performance, as the grey-box models for $\ddot{\psi}$ do not reach a comparable fit. This does not mean that grey-box models should be discarded, as they provide an inherent robustness that the black-box ones lack, since the latter are exclusively data-based. This may be relevant when applying learning models in the Virtual Rider.

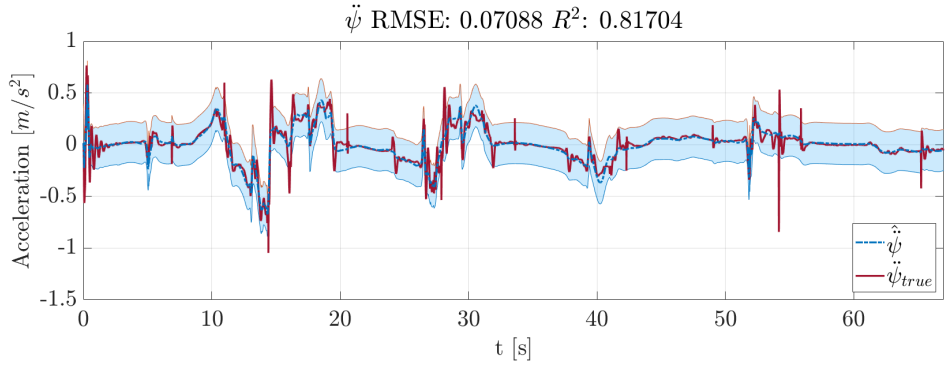
After training the VSGP models, it is possible to derive their more implementable version, according to the formulation $\boldsymbol{\mu}_* = \sum_{i=1}^{n} k(\boldsymbol{x}_*, x_i)\alpha_i = \boldsymbol{k}_*\boldsymbol{\alpha}$ for the predictive mean at a query point, which was already introduced in (3.11). In particular, when using the VSGP models (see Section 3.2.3) $\boldsymbol{\alpha}$ takes the form

$$\boldsymbol{\alpha} = \sigma_n^{-2}\Sigma K_{uf}\boldsymbol{y} \tag{4.15}$$

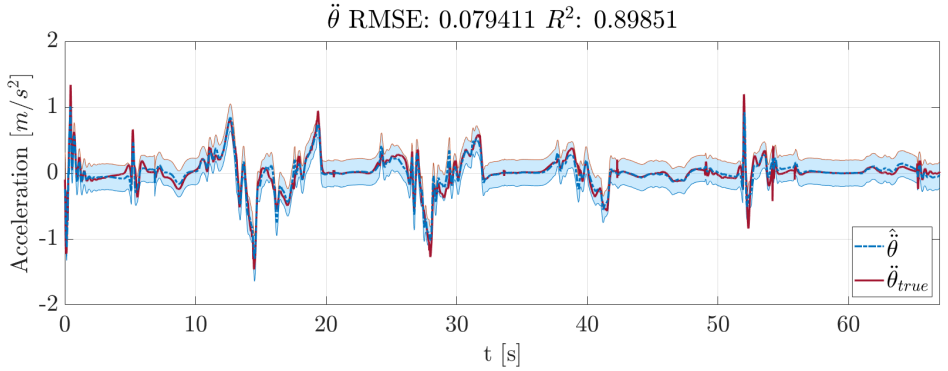
where $\Sigma = (K_u u + \sigma_n^2 K_{uf}K_{fu})^{-1}$. The predictive variance formula instead has the form

$$\mathcal{V}[f_*] = k_{**} - K_{*u}K_{uu}^{-1}K_{u*} + K_{*u}\Sigma K_{u*} \tag{4.16}$$

Pre-computation of $\boldsymbol{\alpha}$ and Σ leads to a more efficient implementation of the VSGP models, as online matrix inversions are no longer required. The simplified formulas were finally tested in order to provide a graphical analysis of the performance of the devised VSGP models. Figure 4.4 (black-box) and Figure 4.5 (grey-box) show the fit of the best 50-point VSGP models on the data from *VI-Track*.

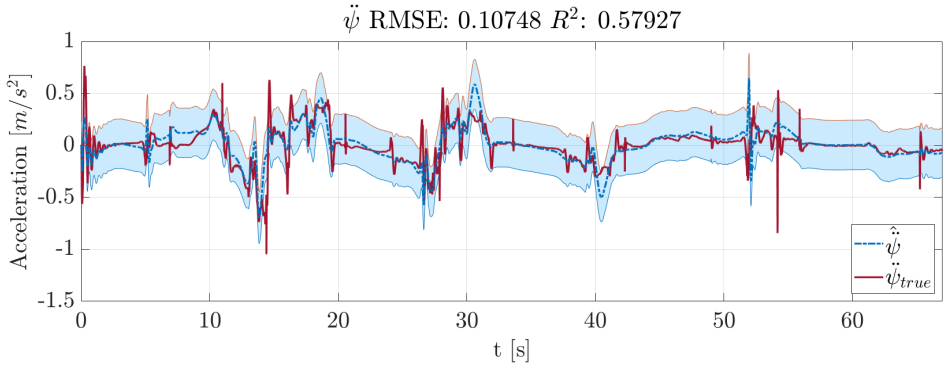


(a) Fit of the best 50-point $\ddot{\psi}$ model

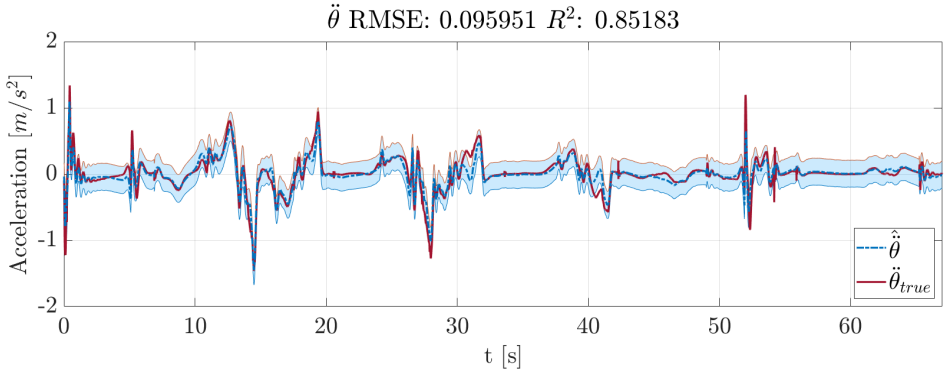


(b) Fit of the best 50-point $\ddot{\theta}$ model

Figure 4.4: The fit of the best 50-points **black-box** VSGP model configurations (with μ_* shown with a dashed blue line and the light blue area representing $[\mu_* - 2\sigma, \mu_* + 2\sigma]$), compared with the target acceleration (shown in red).



(a) Fit using the best 50-point $\phi_{\ddot{\psi}}$ model



(b) Fit using the best 50-point $\phi_{\ddot{\theta}}$ model

Figure 4.5: The fit of the best 50-points **grey-box** VSGP model configurations (with $\ddot{\tilde{q}} + \mu_*$ shown with a dashed blue line and the light blue area representing $[\ddot{\tilde{q}} + \mu_* - 2\sigma, \ddot{\tilde{q}} + \mu_* + 2\sigma]$), compared with the target acceleration (shown in red).

The plots shown in Figure 4.4 and Figure 4.5 (with Figure C.3 and Figure C.3 completing the picture for $\dot{v}_x$ and $\dot{v}_y$) highlight the general efficacy of the trained VSGP models on the data retrieved from the trial performed on *VI-Track*, which will be the general testbed for the closed-loop implementation. In particular, they demonstrate that even a low number of inducing variables (e.g. 50) may lead to satisfactory model fits. This is especially true for the black-box models (which outperform the grey-box models for the lateral and yaw acceleration targets).
While these base models are promising, as they fit the dynamics data well, it is still unclear whether this is a sufficient condition for a successful implementation into the Virtual Rider. The results shown up to now were all related to trials performed on data that was first collected and then analysed in an offline fashion. This means that there is not a general guarantee that the devised models will behave as well in the online closed-loop context, in which the modified internal model may have a direct impact on the explored dynamics and may even cause the failure of the combined strategy.
The next chapters are thus devoted to exploring how the devised GP strategies may be included in the Virtual Rider framework. An analysis of which dynamics model augmentations provide improvements to the performance of the Virtual Rider is conducted, along with some observations on why some of the learnt models may fail in the online context.

5

PRELIMINARY CLOSED-LOOP TRIALS OF THE GP-BASED VIRTUAL RIDER

Chapter 4 showed the main techniques that were employed in order to perform feature selection, with the objective of determining a small subset that is sufficiently informative for a given regression target. The reason behind this was twofold:

- unnecessary input information may worsen the prediction capabilities of the GP models and lead to undesired phenomena in the closed-loop implementation and should thus be discarded;
- in order to maintain the real-time feasibility of the NMPC problem, the implemented GP models should be sufficiently simple.

The second point was also addressed through sparse GP approximations and in particular through the *VFE* one (see Section 4.2), as it allows to summarize the information contained in ample datasets using few inducing variables.
Using the devised reduced models (both in terms of features and inducing points), the impact of learning may finally be tested in the closed-loop scenario, by modifying the internal model of the Virtual Rider accordingly (Section 3.3).
This chapter is thus devoted to the description of the results obtained in the final Virtual Rider scenario. Such trials led to some significant insights into the advantages and the limitations of the *learning dynamics* strategy. In particular, it soon became clear that some acceleration models required additional training procedures.
All of the trials and analyses that are shown in the next sections were performed on *VI-track*, the test track that was presented in Section 2.2.2.

5.1 PRELIMINARY TRIALS ON THE VIRTUAL RIDER

The first trials that were attempted in the closed-loop scenario were realized by applying the models developed in Chapter 4 through the feature selection and the sparse approximation procedures. In particular, models were evaluated according to the comparative analysis in Section 4.2. Both black-box and grey-box strategies were considered, using a variable amount of input features and of inducing variables. The most promising models from that first feature analysis were thus tested in the closed-loop scenario. While some results of interest were obtained, the need for additional feature analyses

was soon highlighted. Before delving into that, some key observations resulting from the first trials are in order:

A. Using a low amount of features, grey-box solutions appear as more robust than the black-box counterparts; even when the learnt grey-box model is of "low quality", meaning that it does not provide the hoped for performance in the closed-loop implementation, the Virtual Rider is able to complete the track trial. This is in general not true for black-box models, which work only if they also provide a good fit of the accelerations in the closed-loop trials (the fit quality shown in the offline trials is not a sufficient guarantee in general). This may attributed to the fact that the physics-based part of the grey-box models allows to ride even in conditions that are unknown to the learning portion of the models.

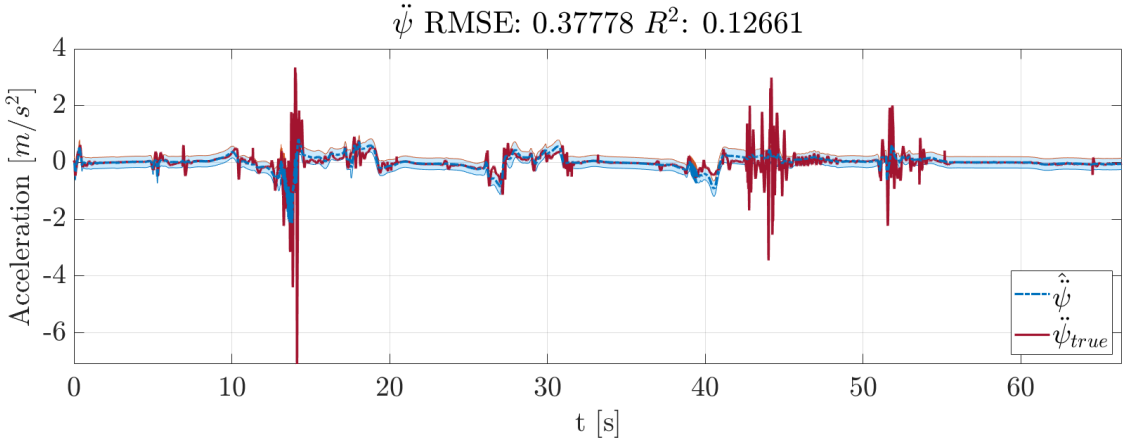
B. The addition of learning for the lateral acceleration's modeling did not provide significant improvements to the general performance of the Virtual Rider (in terms of tracking), while causing the rise of oscillatory behaviors in the accelerations' evolutions. While no conclusive evidence was collected in order to prove the ineffectiveness of having a GP-modeled $\dot{v}_y$, efforts towards $\dot{v}_y$ modeling were avoided in order to favor in-depth research into the more promising $\ddot{\psi}$ and $\ddot{\theta}$

C. Likewise, learning improvements to the longitudinal acceleration $\dot{v}_x$ appear avoidable. In this case, the main reason is related to the fact that the physics-based modeling of this acceleration was already top notch, as already shown in Figure 2.7. Adding a learning component does not provide any clear benefit, as it would barely improve the prediction capabilities while adding an unnecessary computational load.

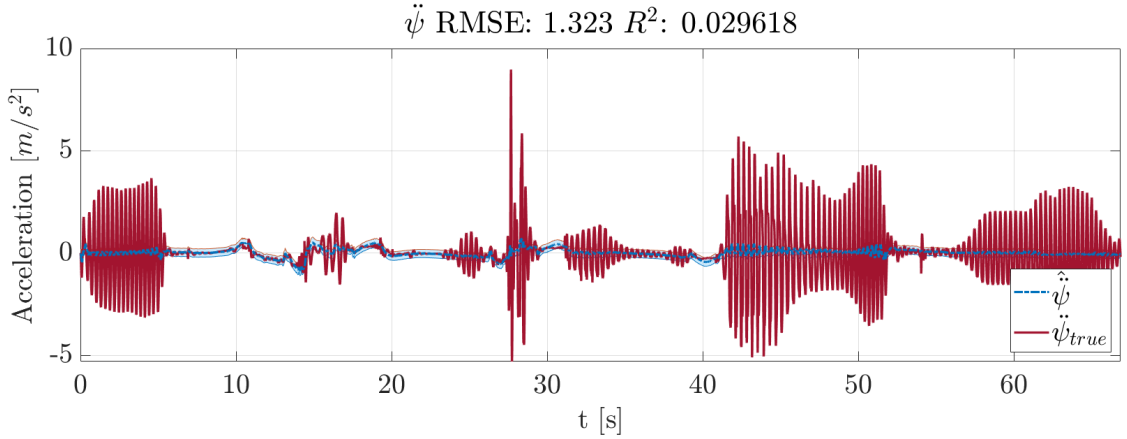
D. The most important takeaway from the first trials was that some feature combinations, while successful in the offline tests, were not sufficient in order to provide a satisfactory fit of the data obtained in the closed-loop scenario; this may be interpreted as a consequence of an inadequate variety in the dataset that was employed in the first feature analysis (see Chapter 4). Specifically, it is easy to show that some black-box models provide an unsatisfactory fit on data that is collected from the new closed-loop trials undertaken with the addition of GP models.
In such sense, Figure 5.1 provides a couple of examples for the black-box target $\ddot{\psi}$. The figure in question shows the fit provided by the "best" $\ddot{\psi}$ black-box model on newly collected data. Such model was deemed satisfactory in the original feature analysis, showing an impressive fit value of $R^2 = 0.817$ on the *VI-track* data belonging to the original dataset, but it is clear that it fails at representing some dynamics. Figure 5.1(a) shows that the model itself is not completely inadequate, as it correctly detects the general tendencies of the yaw acceleration, but if fails when significant acceleration oscillations are present (e.g. between 10-20s and between 40-60s). This is further highlighted by Figure 5.1(b), showing a trial characterised by large oscillations, which are completely undetected by the model. Such considerations may lead to the conclusion that the models obtained until now, while not

bad per se, may not be representative of all dynamics. This is the most probable reason behind their failure in the closed-loop scenario.

Point (D) in particular motivates a new analysis of the features, in light of the new data that was collected from the first closed-loop trials. This is especially true for the black-box target $\ddot{\psi}$, but a new analysis was performed for all of the different targets of interest, in order to verify whether improvements could be made. The next section is devoted to the description of the repeated feature analysis, which mirrors the fit approach from Chapter 4, with some tweaks related to the dataset and the selection algorithm.



(a) Fit of the current best 50-point $\ddot{\psi}$ model on new closed-loop data



(b) Fit of the current best 50-point $\ddot{\psi}$ model on new closed-loop data

Figure 5.1: Fit of the best 50-point **black-box** VSGP model configuration for $\ddot{\boldsymbol{\psi}}$ (see Section 4.2) on additional track data resulting from the first closed-loop trials. μ_* is shown with a dashed blue line while the light blue area represents $[+\mu_* - 2\sigma, \mu_* + 2\sigma]$); the real yaw acceleration is shown in red.

5.2 REVISED FEATURE ANALYSIS

Before delving into the revised feature analysis, some considerations are in order. In particular, it must be stressed that the feature selection procedure was a particularly delicate phase of research, whose importance cannot be overstated. The original approach from Chapter 4 showed some limitations, as some of the devised models did not provide satisfactory closed-loop results. The revised analysis presented below fixed some of the problems through a couple of significant changes. Still, it soon became clear that it was not optimal either, as it did not provide sufficient guarantees for the closed-loop performance. As a consequence, the choice of features for each target of interest (the black-box $\ddot{\psi}$, $\ddot{\theta}$ and the grey-box $\phi_{\ddot{\psi}}$, $\phi_{\ddot{\theta}}$) was done according to the following scheme:

- The models originating from the original feature analysis (Chapter 4) were singularly tested in the closed-loop scenario in order to verify which ones worked well already (e.g. $\ddot{\theta}$) and which ones required an additional feature analysis (e.g. $\ddot{\psi}$, which did not work at all, and $\phi_{\ddot{\psi}}$, $\phi_{\ddot{\theta}}$ to a lesser extent, as they caused oscillatory closed-loop behaviors).

- If the additional feature analysis led to more satisfactory results for a given target, the corresponding model was assumed to be the gold standard; otherwise, the model obtained from the previous analysis was kept.

According to to this work scheme, it resulted that the $\ddot{\theta}$ model could not be improved through the revised analysis, while improvements were attained for the $\ddot{\psi}$, $\phi_{\ddot{\psi}}$ and $\phi_{\ddot{\theta}}$ targets. For this reason, the revised analysis for the former will be omitted, while the analyses related to the latter will be reported. Before doing that, the revised feature selection approach will be described in more detail.
The revised feature analysis takes inspiration in large part from the fit-based analysis described in Section 4.1.2, with two major tweaks:

- The new analysis is performed on an enlarged dataset, that contains additional data w.r.t. the original one (which comprised data from the 3 physics-based trials performed on the corresponding tracks shown in Figure 4.1). The new data results from 4 additional trials that were performed on *VI-Track* with different learning augmentations (both grey-box and black-box). Such results were specifically chosen as they highlighted the limitations of some of the previously developed models, meaning that they could be used in order to perform a more informed feature selection. The newer dataset was thus composed by 51180 points (w.r.t. 24580 of the original one). The approach used for the fit evaluation was instead kept unchanged: a full GP model is trained (in terms of hyperparameter tuning) on a limited subset of 2000 points and its prediction capabilities are evaluated on the whole dataset, computing $\overline{R^2}$, the average R^2 index (see (4.13)) across the 7 trials' data.

- The feature selection procedure is slightly modified. It is still incremental in nature, in the sense that at each iteration the feature leading to the highest $\overline{R^2}$ index is added to the mix of features, but a backtracking step has been added, in accord with a principle also highlighted in [25]. In particular, at each iteration, after adding a new feature a backwards analysis is performed in order to verify whether the removal of previously added features leads to an improvement of the fit index. This is done in order to remove features that become redundant or harmful along the iterative procedure. The general procedure is schematized in Listing 5.1. It was employed for the different targets and specifically to revise the feature selection to be applied for the black-box target $\ddot{\psi}$ (the model for $\ddot{\theta}$ obtained from the original analysis proved to be already optimal) and the grey-box targets $\phi_{\ddot{\psi}}$ and $\phi_{\ddot{\theta}}$.

It must be stressed that an analogous procedure could have been followed also using the mutual information criterion, as it is not specific to any quality measure. This may have led to alternative candidate models, which can be a great help, since the fit-based criterion does not always lead to the optimal solution, as will soon become clear.
The procedure from Listing 5.1 was undertaken on the different targets, performing 10 search iterations for each. The results for the black-box target $\ddot{\psi}$ are shown in Figure 5.2 and led to the feature combinations shown in Table 5.1 (to be interpreted according to (4.2)). One can see that finding suitable feature combinations that can fit data belonging to the larger dataset is no easy task: no combinations having few features provide a satisfactory fit, countering the parsimony assumption that was made in Chapter 4. Starting from iteration 6, the quality of fit finally starts increasing and continues to do so in the following iterations, with the last being the best one. It is interesting to note how the new feature selection algorithm is different w.r.t. the one from Chapter 4: subsequent iterations do not necessarily lead to a larger feature combination, as it can be seen that a couple of iterations are characterized by a feature removal. This appears to be useful, as it allows to discard input information that appears as useful at the beginning of the selection procedure, before becoming unnecessary as new additions are made.

Listing 5.1: Revised feature analysis

```
for acceleration target
  good_features = [];
  for iter = 1:N
    for feature not in good_features
      simulate addition of feature to good_features;
      evaluate average R^2;
    end
    add feature leading to max(average R^2) to good_features;
    for feature in good_features
      simulate removal of feature from good_features;
      evaluate average R^2;
      if average R^2 increases
        remove feature;
      end
    end
  end
end
```

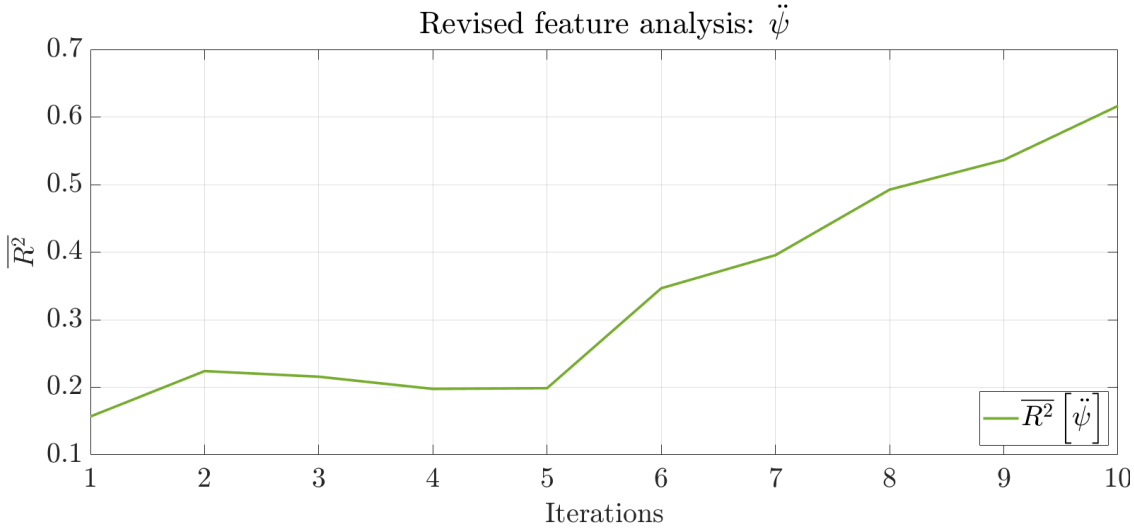


Figure 5.2: The $\overline{R^2}$ index obtained from the full GP models trained according to the revised feature analysis on the extended dataset.

Table 5.1: Feature combinations resulting from the revised input analysis for the **black-box** target $\ddot{\psi}$

ITER	1	2	3	4	5
	[5]	[5,16]	[5,16,17]	[5,16,12]	[5,16,12,2]
ITER	6	7	8	9	10
	[5,16,2,6]	[5,16,2,6,3]	[5,16,2,6,3,7]	[5,16,2,6,3,7,13]	[5,16,2,6,3,7,13,4]

In order to verify the validity of the newly developed black-box $\ddot{\psi}$ model, it comes as natural to check whether it actually absolves the task of representing the dynamics that were neglected by the previous $\ddot{\psi}$ models (see Figure 5.1). After training a sparse GP model based on the features obtained at iteration 10 ([5,16,2,6,3,7,13,4]) and 50 inducing points (through the usual *VFE* approach), a quick test was done on the same data from Figure 5.1(b), resulting in the predictions which are shown in Figure 5.3. It is clear that the revised feature selection procedure has managed to find a sufficiently expressive $\ddot{\psi}$ model, also thanks to the enlarged dataset, as the large oscillations that were neglected by earlier models are now fit perfectly.

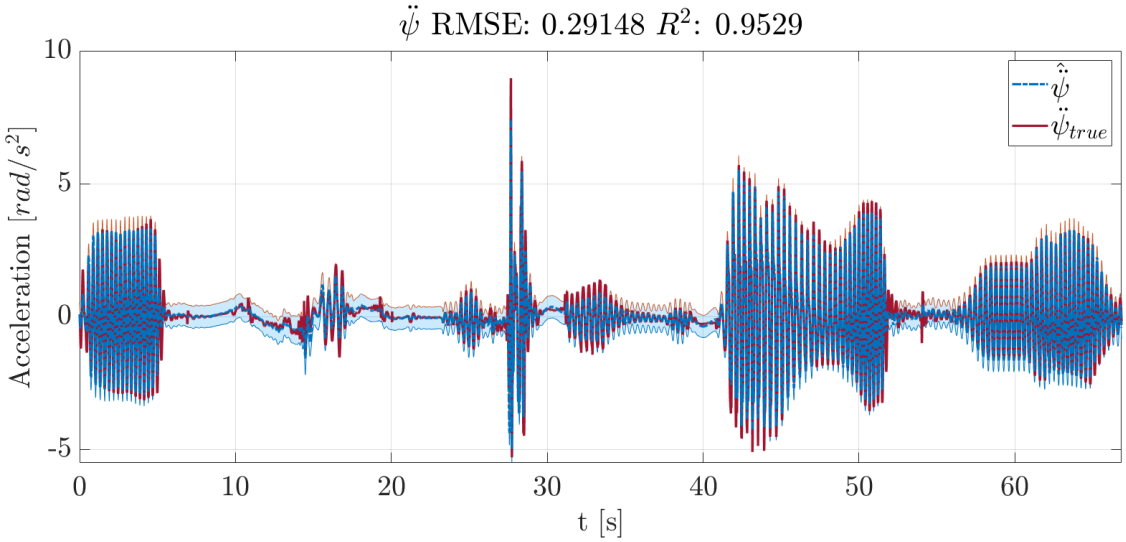


Figure 5.3: Fit provided by the revised 50-point 8-feature $\ddot{\psi}$ model.

As previously mentioned, the same revised procedure was undertaken on the grey-box targets $\phi_{\ddot{\psi}}$ and $\phi_{\ddot{\theta}}$, in order to gain more clarity on which features are most useful also in the grey-box context. This was also done as to provide a fair comparison between the strategies, with the end goal of determining the gold standard among the black-box and the grey-box implementations of the yaw and roll accelerations. The revised analysis for the aforementioned targets led to the results shown in Figure 5.4.
In the case of the grey-box models, good feature combinations appear already in the initial iterations. Increases in the $\overline{R^2}$ index continue up to the 9$^{\text{th}}$ iteration for the $\phi_{\ddot{\theta}}$ target and to the 6$^{\text{th}}$ iteration for the $\phi_{\ddot{\theta}}$ target. The feature combinations corresponding to the various iterations are reported in Table 5.2 ($\phi_{\ddot{\psi}}$) and Table 5.3 ($\phi_{\ddot{\theta}}$).

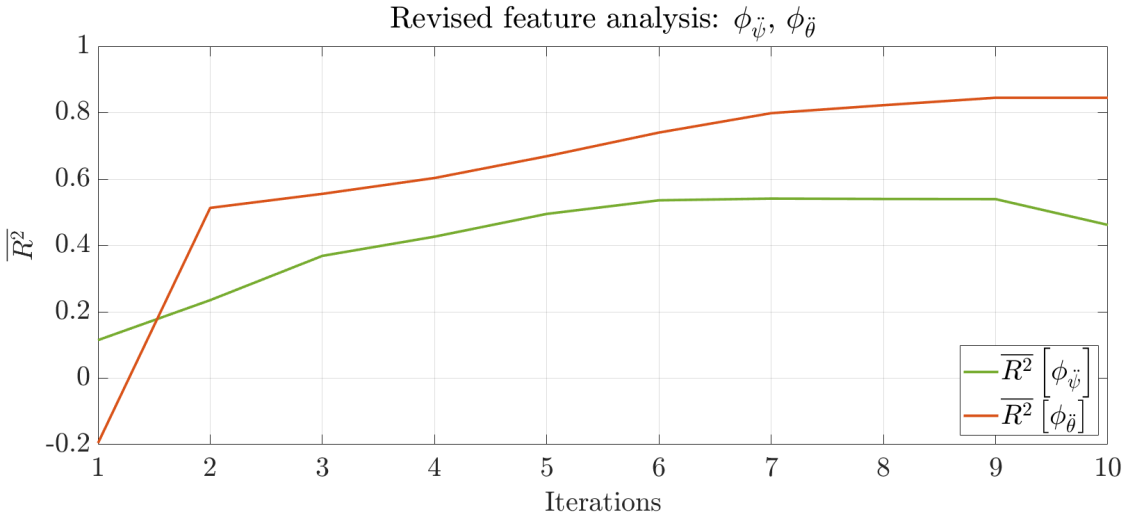


Figure 5.4: The $\overline{R^2}$ index obtained from the full **grey-box** GP models trained according to the revised feature analysis on the extended dataset.

Table 5.2: Feature combinations resulting from the revised input analysis for the **grey-box** target $\phi_{\ddot{\psi}}$

ITER	1	2	3	4	5
	[17]	[17,7]	[17,7,12]	[17,7,12,8]	[17,7,12,8,9]
ITER	6	7	8	9	10
	[17,7,12,8,9,2]	[7,12,8,9,2,1]	[7,12,8,9,2,1,16]	[7,12,8,9,2,1,16,17]	[7,12,8,9,2,1,16,15]

Table 5.3: Feature combinations resulting from the revised input analysis for the **grey-box** target $\phi_{\ddot{\theta}}$

ITER	1	2	3	4	5
	[17]	[17,8]	[17,8,5]	[17,8,5,12]	[17,8,5,12,4]
ITER	6	7	8	9	10
	[17,8,5,12,4,2]	[17,8,5,12,4,2,15]	[17,8,5,12,4,2,15,14]	[17,8,5,12,4,2,15,14,9]	[17,8,5,12,4,2,15,14,9]

6

DEFINITIVE CLOSED-LOOP RESULTS

6.1 TESTING OF THE REVISED LEARNING-BASED MODELS

The current section is devoted to the presentation of the results that were derived by testing the best grey-box and black-box models in the Virtual Rider scenario. In particular, learning-based augmentation entailed improvements to the $\ddot{\psi}$ and $\ddot{\theta}$ dynamics to be applied in the closed-loop scenario. Among the tested configurations, the ones that provided the most promising results are the following:

- Black-box modeling of $\ddot{\psi}$ (using features [5($\dot{\theta}$), 16(c_θ), 2(v_x), 6(δ), 3(v_y), 7(γ_t), 13($\dot{y}_r$), 4($\dot{\psi}$)]), according to the revised feature analysis and of $\ddot{\theta}$ (using features [1(θ), 2(v_x), 4($\dot{\psi}$), 9(y_r)]), according to the original feature analysis.
- Grey-box modeling of $\ddot{\psi}$ (using features [17(s_θ), 7(γ_t), 12($\dot{\gamma}_b$), 8(γ_b), 9(y_r), 2(v_x)]) and $\ddot{\theta}$ (using features [17(s_θ), 8(γ_t), 5($\dot{\theta}$), 12($\dot{\gamma}_b$), 4($\dot{\psi}$), 2(v_x)]), both in accord with the revised feature analysis.
- Grey-box modeling of $\ddot{\psi}$ and $\ddot{\theta}$ using the same features as the black-box models; this additional solution was tested in order to investigate the differences between the two strategies (black/grey-box) when using the same exact information.

In particular, the devised models were tested according to a common framework. First of all, reduced models employing 50 inducing points for each learning target were used for all models. After that, the augmented models were tested on *VI-Track* on a preliminary trial. Before evaluating the final closed-loop performance of the models, a further learning improvement is made. Specifically, data from the first run that appears as useful is employed to refine the models, according to an approach that may be linked to the general field of *incremental learning* [28]. The complications of adapting learning models through streaming data are avoided for the scope of this thesis, meaning that adaptation of the models is performed inbetween runs through a new training procedure.
The incremental learning procedure may be summarized as follows:

1. A trial run is performed in the closed-loop scenario employing the base models trained on the extended dataset and reduced to 50 inducing points.
2. Data that appears as "interesting" in order to improve the learning models is collected. In particular two main criteria are used to determine the usefulness of new data:

- A *variance criterion*: new datapoints are considered as "interesting" if the predictive variance of the model on the location is sufficiently high, which means that the point belongs to a scarcely known region. More formally, new datapoints $z_* = (x_*, y_*)$ are collected if

$$\mathcal{V}[f_*] \geq 2 \cdot \overline{V[\mathbf{f}]} \tag{6.1}$$

  where $\mathcal{V}[f_*]$ is the predictive variance on the new datapoint and $\overline{V[\mathbf{f}]}$ is the average variance over the datapoints of the base model.

- A *mean criterion*: new datapoints are selected if they are interpreted as "non-disruptive", meaning that the regression target at the new location is sufficiently in line with the base model in terms of predictive mean. This is done as to discard outliers. More formally, selection is performed when

$$\mu_* - 3\sigma \leq y_* \leq \mu_* + 3\sigma \tag{6.2}$$

  with y_* being the acceleration target at the current location, μ_* the predictive mean at the current location and σ the corresponding standard deviation.

The idea behind these criteria is thus to select datapoints that store new information and that are guaranteed not be disruptive to the base models. It was largely inspired by [29].

3. The newly collected points are used for a retraining of the sparse GP models to be used for the definitive run: 50-point reduced models are created using the established features and the additional datapoints in order to be finally tested in the closed-loop scenario on the usual *VI-Track*.

The results that will be presented in the next sections are derived from the models trained according to the pipeline presented above, which led to the best performances all-around. Since incremental learning leads to models that are based on partly different data, it may be interesting to also perform comparisons on the base models which depend on the same dataset, especially to study their generalizability properties. Such analyses are deferred to Appendix D.

6.1.1 *Prediction quality analysis*

The models that underwent the procedure of Section 6.1 were finally tested in the general Virtual Rider scenario. As a first step, the quality of the models was verified in an offline test aimed at determining the prediction capability of each model. Such test was conducted exploiting the data obtained by the physics-based Virtual Rider on *VI-Track*. In particular, given the real state-input trajectory along the track, it is useful to verify how close each model is to recreating the real system behavior: this may be done studying the predictions provided by the model starting from each state and comparing them with the actual evolution of the system trajectories. Comparisons were provided in terms of velocities: starting from a given state, the model predicts the future velocity

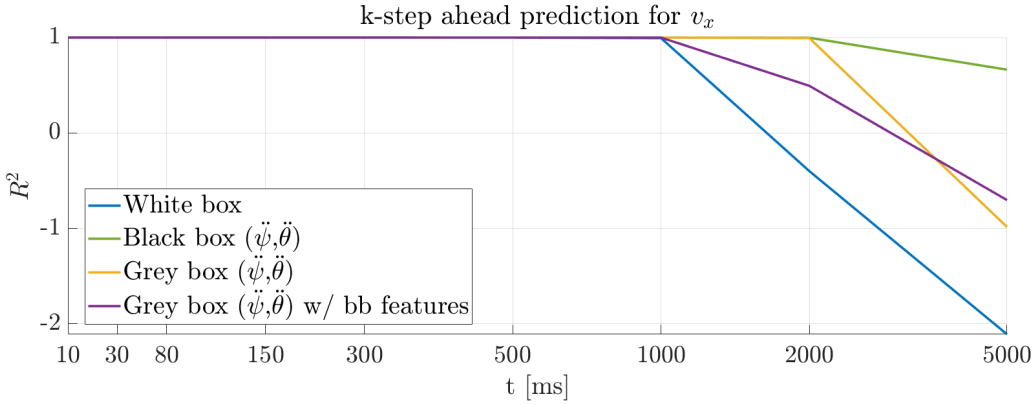
evolution according to the input trajectory. The predicted velocities are then compared to the actual ones. [1] The specifications of the test are provided in the following.
In the devised test, the obtained dynamics models underwent k-step integration in order to verify the adherence of the k-step ahead velocity predictions w.r.t. the real physical quantities. In particular, subsequent prediction steps are separated by $T_s = 10$ms and the different models introduced in Section 6.1 (plus the physics-based model) are evaluated at $k = [1, 3, 8, 15, 30, 50, 100, 200, 500]$ steps (i.e. at $[10, 30, 80, 150, 300, 500, 1000, 2000, 5000]$ms). To do so, the real inputs along the trajectory have been used. The prediction results are then compared with the real physical evolution in terms of R^2 index, in order to evaluate how closely the models match the real motorcycle system. The results obtained for the velocity quantities, namely v_x, v_y, $\dot{\psi}$, $\dot{\theta}$, have been reported in Figure 6.1, in which comparisons between the 3 different learning-based schemes and the physics-based model are shown. Despite having provided learning improvements only to $\dot{\psi}$ and $\dot{\theta}$, it makes sense to also evaluate v_x and v_y predictions, as dynamics are highly coupled. As a consequence, it is generally expected that improvements to a given acceleration model will have an impact on the other accelerations/velocities as well.
The obtained results may lead to the following observations:

- All learning-based models provide better predictions than the white-box (physics-based) one. This is true for all 4 velocities, both in the short and long term range. It can be noted that improvements are present for both the quantities that are directly related to learning (i.e. $\dot{\psi}$ and $\dot{\theta}$) and those that are not (i.e. v_x and v_y), thus indicating that there is also an indirect (positive) impact of learning-based dynamics models.
  Additionally, it can be seen that the prediction quality of the white-box model degrades faster (especially for v_y and $\dot{\theta}$), while the learning-based models provide good results up to 1s (v_y, $\dot{\psi}$ and $\dot{\theta}$) and 2s (v_x).

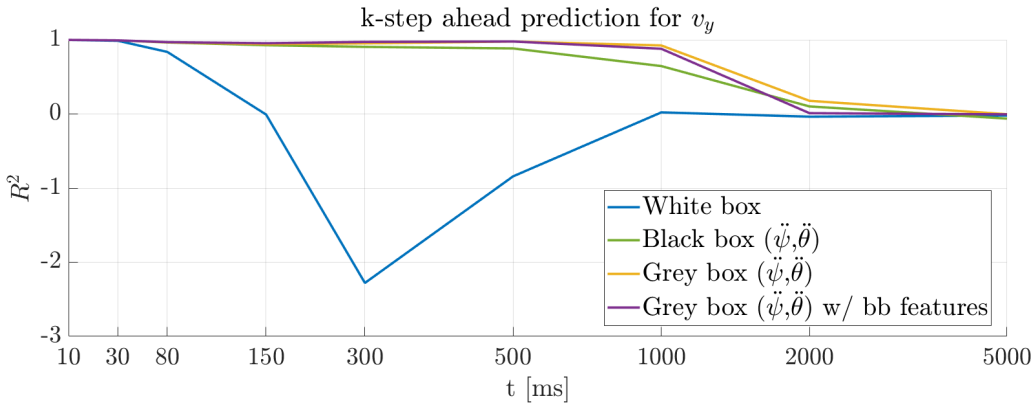
- When comparing the learning-based models between each other, it must be noted that the black-box one generally appears as the superior one, providing better predictions than the others for v_x, $\dot{\psi}$ and $\dot{\theta}$ up to 1s, after which predictions are less meaningful.
  Among the grey-box models, a clear distinction in the prediction capabilities is not apparent, even though the grey-box with custom features seems to perform slightly better than the grey-box model employing the same features as the black-box one.

The models, which have now been tested off-line in terms of prediction capabilities, will now be tested in closed-loop scenario, in order to gauge whether they provide improvements to the general riding performance.

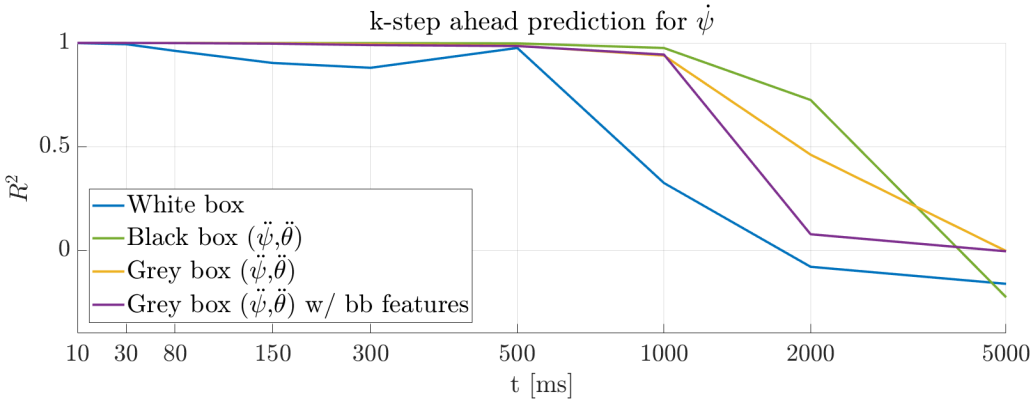
1 Predictions are computed for all velocities, both those that are directly related to the learning-based accelerations ($\dot{\psi}$, $\dot{\theta}$) and those that aren't (v_x, v_y). This is due to the fact that modifications to a given dynamics portion normally have a generalized impact, due to the significant coupling effects.



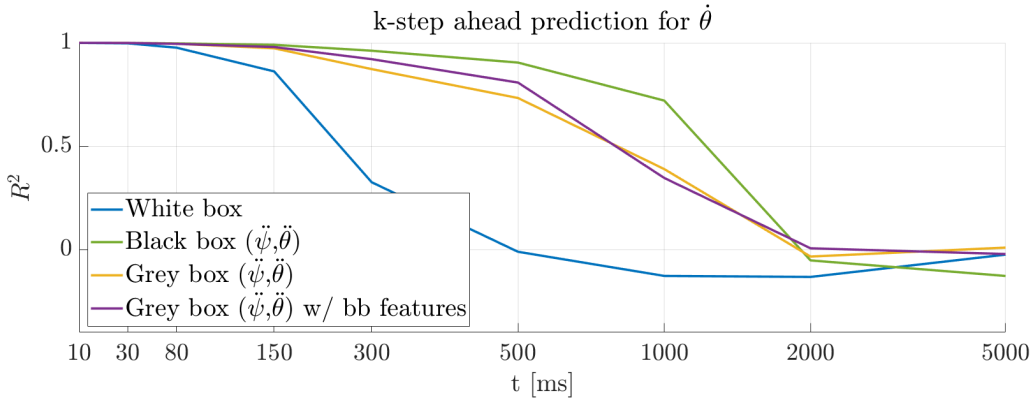
(a) R^2 index evolution for the v_x predictions



(b) R^2 index evolution for the v_y predictions



(c) R^2 index evolution for the $\dot{\psi}$ predictions



(d) R^2 index evolution for the $\dot{\theta}$ predictions

Figure 6.1: Prediction analysis of the learning-based models. The R^2 index evolution of the k-step prediction for v_x (a), v_y (b), $\dot{\psi}$ (c) and $\dot{\theta}$ (d) is reported for each model (the white-box one in blue, the black-box one in green, the grey-box one w/ custom features in yellow, the grey-box one w/ bb features in purple). Note that the x-axis is freely-scaled for better visibility.

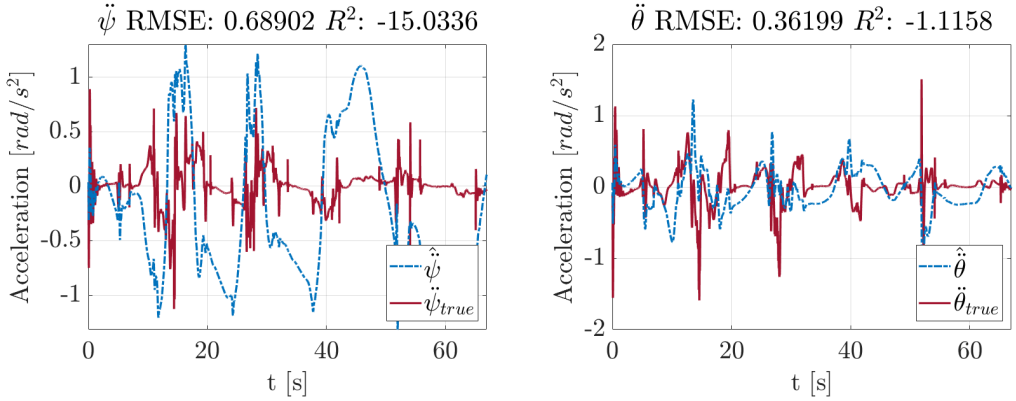
6.1.2 *Closed-loop analysis*

After performing the off-line prediction analysis presented in Section 6.1.1, the 3 learning-based models were finally tested in the closed-loop scenario, employing a common NMPC configuration (the one already presented in (2.57)). As previously mentioned, the riding task used for testing is given by the completion of *VI-Track* according to a reference tracking strategy. The results that were obtained for the 3 models are reported and analysed in this section. In particular, the 3 learning models are compared between each other and w.r.t. the physics-based models.

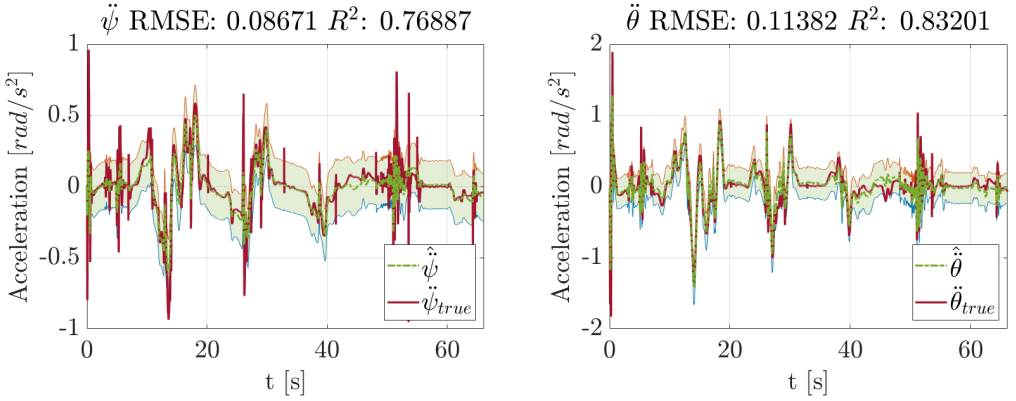
6.1.2.1 *Closed-loop prediction analysis*

First of all, the accuracy of the models will be evaluated and compared with the one shown by the white-box model. In such sense, it is natural to check how the new models compare in terms of acceleration estimation, since learning is employed for continuous dynamics modeling. Since the models are augmented w.r.t. the $\ddot{\psi}$ and $\ddot{\theta}$ accelerations, it is natural to check how each of the models fares in terms of acceleration estimation w.r.t. the actual evolution of those two targets. The results of the learning-based models are shown in Figure 6.2(b-d), while the corresponding white-box estimations are reported in Figure 6.2(a). Specifically, the fit provided along the trajectory by the black-box model is represented in green (with the dashed line representing the predictive mean and the light green area being the 2σ region). The fit attained by the grey-box model with custom features is shown in yellow, while the estimations obtained using the grey-box model with the black-box model's features are shown in purple.
Such results lend themselves to the following observations:

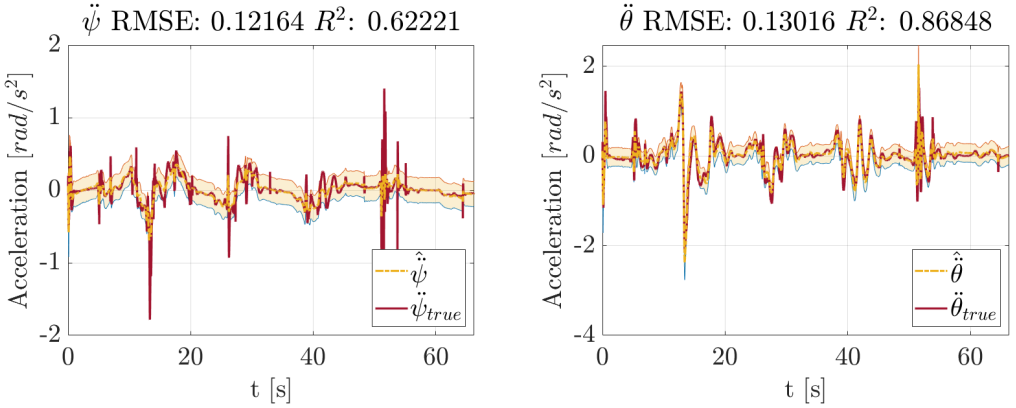
- In general, all learning solutions lead to closed-loop acceleration estimations that are significantly improved w.r.t. the physics-based ones; in particular, the fits provided by the learning-based models lead to a R^2 index that ranges from 0.48 to 0.77 for the $\ddot{\psi}$ acceleration and from 0.62 o 0.87 for the $\ddot{\theta}$ acceleration.
- When compared to each other, it can be seen that the black-box solution provides the best fit for $\ddot{\psi}$, while the grey-box (w/ custom features) leads to the highest $\ddot{\theta}$ accuracy (with black-box being a very close second). The grey-box model using the black-box features leads to the lowest acceleration estimation accuracy.
- As could have been expected, the grey-box model employing custom features fares better than the other grey-box solution. This is intuitive, as features were deliberately selected as to maximise the prediction capabilities. Still, other performance criteria will show that a better fit capability does not always lead to the best closed-loop riding behavior.



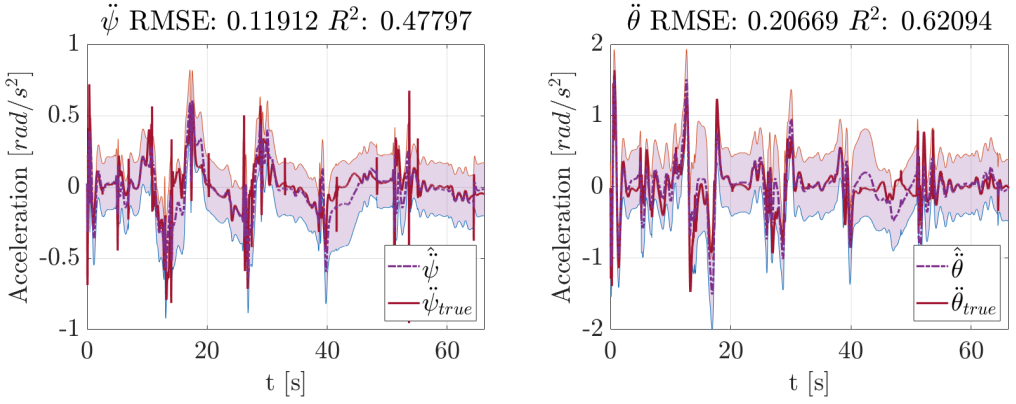
(a) Closed-loop fit for the white-box model.



(b) Closed-loop fit for the black-box model.



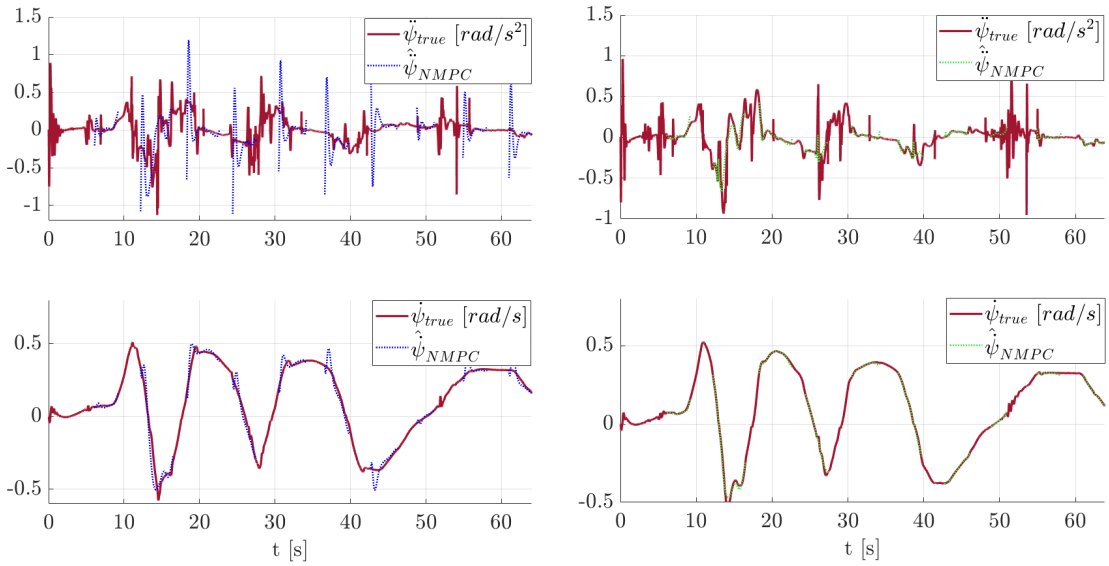
(c) Closed-loop fit for the grey-box model w/ custom features.



(d) Closed-loop fit for the grey-box model w/ bb features

Figure 6.2: The fit provided by learning-based models in their respective closed-loop trials. Results for the $\ddot{\psi}$ and $\ddot{\theta}$ targets are shown according to the following color scheme: green for the black-box model (b), yellow for the grey-box model w/ custom features (c), purple for the grey-box model w/ bb features (d). The 2σ region is shown as a colored shaded area.

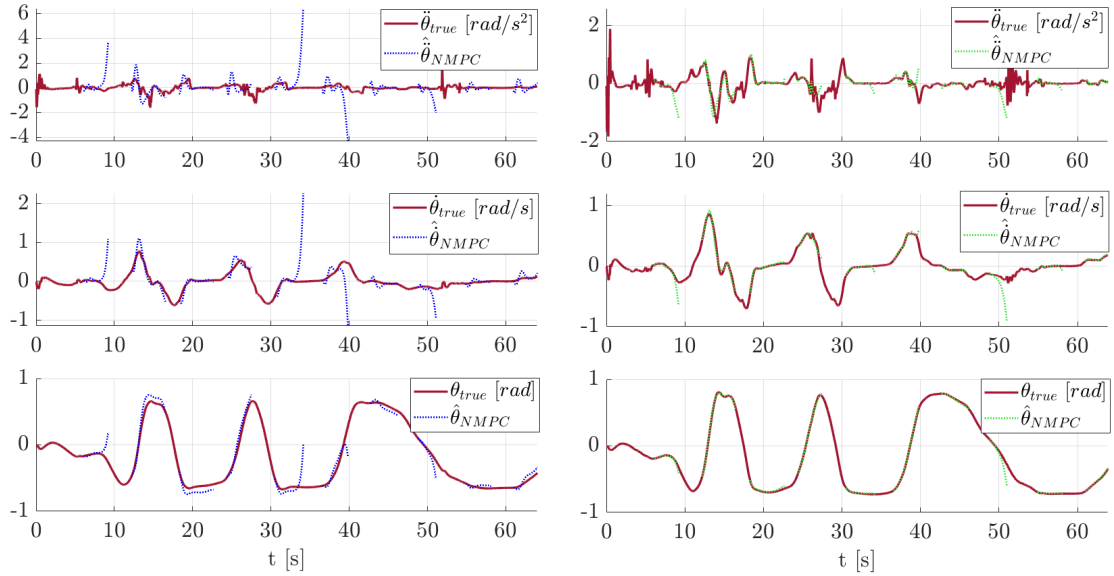
An additional test on the closed-loop predictive capabilities was also conducted. In particular, the reliability of the internal model used by the Virtual Rider was compared when using different strategies. In such sense, the internal model reliability may be evaluated by testing whether the NMPC predicted trajectories match the future behavior of the real system. In such sense, Figure 6.3 and Figure 6.4 provide a graphical comparison between the physics-based model (left) and the best internal model (the black-box one, on the right), showing some sample NMPC trajectories (in dotted lines) and how they compare w.r.t. the real trajectories (in red). It can be seen how the NMPC predictions are much more representative of the future system trajectories when using the black-box model, both for the yaw-related quantities ($\ddot{\psi}$, $\dot{\psi}$, in Figure 6.3) and the roll-related ones ($\ddot{\theta}$, $\dot{\theta}$, θ, in Figure 6.4). For completeness' sake, the corresponding results obtained for the two grey-box models have been reported in the appendix (Figure D.1). In particular, they show that NMPC predictions based on the grey-box model w/ optimized features are not satisfactory (a numerical comparison will follow), while the grey-box model employing the black-box features leads to higher quality NMPC predictions, even if not at the same level of the black-box counterpart. This is in contrast with the punctual acceleration estimations from Figure 6.2, which showed better results when using the grey-box solution w/ custom features. This reversal will also appear in the forthcoming comparisons and is a testament to the fragility of the feature selection procedure.



(a) Sample NMPC predicted trajectories for yaw-related quantities using the physics-based internal model.

(b) Sample NMPC predicted trajectories for yaw-related quantities using the black-box learning-based internal model.

Figure 6.3: Graphical comparison showing the reliability of the physics-based model for the yaw-related quantities (a) w.r.t. learning-based black-box internal model (b). Sampled NMPC predicted trajectories are shown with dotted lines (blue for the physics-based model, green for the learning-based one).



(a) Sample NMPC predicted trajectories for roll-related quantities using the physics-based internal model.

(b) Sample NMPC predicted trajectories for roll-related quantities using the black-box learning-based internal model.

Figure 6.4: Graphical comparison showing the reliability of the physics-based model for the roll-related quantities (a) w.r.t. the learning-based black-box internal model (b). Sampled NMPC predicted trajectories are shown with dotted lines (blue for the physics-based model, green for the learning-based one).

Beside the graphical evidence behind the higher reliability of the learning-based internal models, it is possible to produce a numerical comparison, evaluating how close the predictions at k spatial steps are to the actual trajectory. A quick comparison, showing the reliability of NMPC predictions at $k = 20$m for some selected quantities is reported in Table 6.1, where predictions are evaluated in terms of R^2 index w.r.t. the real trajectories. They highlight that NMPC velocity predictions at $k = 20$m are improved all-around when employing the black-box model, confirming the previous prediction analyses, which also showed this learning solution coming on top. The other solutions do not always provide improvements, with the grey-box "1" solution being better only for v_x and $\dot{\psi}$ and the grey-box "2" solution improving on v_x, v_y and $\dot{\psi}$.

Table 6.1: Reliability of NMPC predictions: accelerations and velocity predictions are compared to the real trajectories in terms of R^2 index. Note that grey-box 1 indicates the grey-box model w/ custom features and grey-box 2 indicates the grey-box model w/ bb features.

$R^2(k = 20m)$	NOMINAL	BLACK-BOX	GREY-BOX 1	GREY-BOX 2
v_x	0.901	0.908	0.906	0.906
v_y	0.835	0.863	0.824	0.863
$\dot{\psi}$	0.786	0.830	0.839	0.909
$\dot{\theta}$	0.328	0.400	0.185	0.273

6.1.2.2 *General closed-loop performance*

After analysing the predictive/accuracy properties of the learning-based models, it is interesting to focus on other evaluation criteria, that are more indicative of the actual riding performance. In such sense, Table 6.2 provides a synthesis that includes some track averages of interest for the white-box model (*nominal*), the black-box one and the two grey-box ones (the one w/ custom features is denoted as "1", while the one w/ bb features is denoted as "2"). In particular, the following quantities have been highlighted:

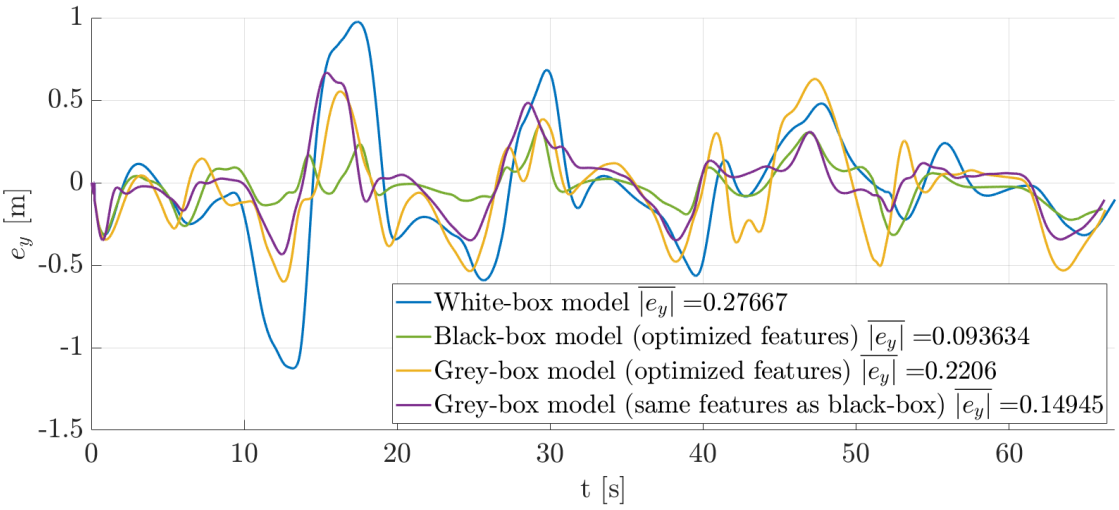
- In terms of tracking performance, the average absolute errors along the track were evaluated. It is clear that learning-based models provide a much better tracking performance, with the black-box model reducing the lateral error (the more significant one from a path following point of view) by 66% and the velocity tracking error by more than 80%, while the grey-box model w/ the same features manages to reduce the average yaw tracking error by 65%
- The average input commands along the track were also evaluated. They generally highlight a less "wasteful" approach to riding when using the learning-based strategies, w/ a reduced steering action, less braking and a lowered reliance on the rider lateral movement.
- Riding aggressiveness was also evaluated in terms of average absolute side-slip angles. They are not in general an indicator of "good" or "bad" riding performance, but lowered side-slip angles indicate a less risky riding behavior, which is the case for the learning-based strategies (in particular for the black-box model)
- Lap times were also compared: they were up to 0.8s lower for the learning-based solutions; this may be attributed to the generally improved riding behavior.
- Finally, average computational times required for the NMPC solution were analysed. They emphasize the added computational load of learning-based solutions. Still, it must be said that the current solutions were aimed at determining the maximal improvements that could be provided through learning, without taking computational matters into account (as will be done in Section 6.2).

It is interesting to note that the grey-box model using the same features as the black-box one has an all-around better performance than the grey-box model with custom features. This seems to testify a general observation of this thesis work: models that provide the best fit and prediction capabilities in a priori trials are not necessarily the best performing in the closed-loop scenario, as will be further discussed in Chapter 7.

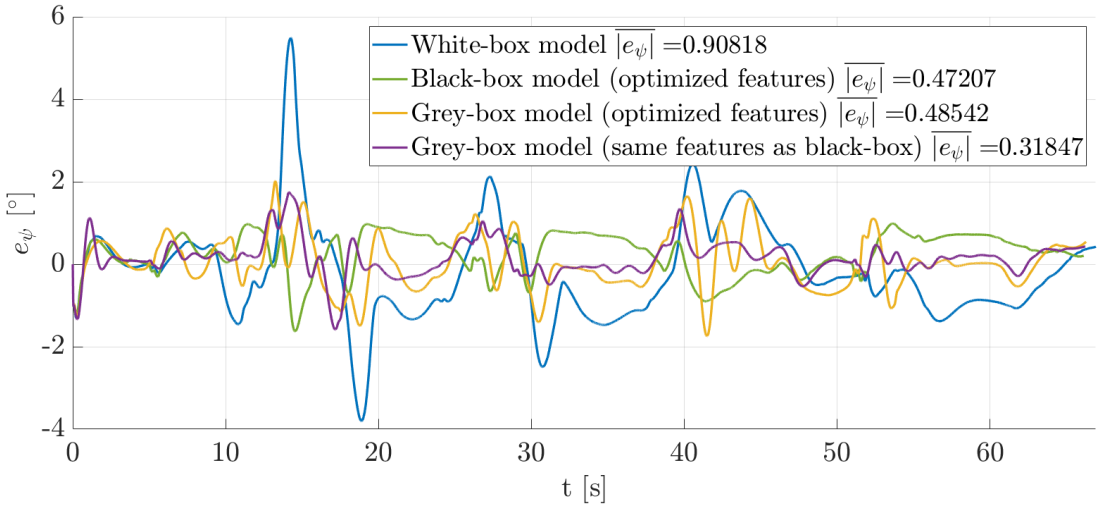
Table 6.2: Summary table of the closed-loop results in terms of performance.

	NOMINAL	BLACK-BOX	GREY-BOX 1	GREY-BOX 2
Tracking performance				
$\overline{\lvert e_y \rvert}$ [m]	0.277	0.0936 (-66.2%)	0.221 (-20.3%)	0.149 (-46.0%)
$\overline{\lvert e_\psi \rvert}$ [°]	0.908	0.472 (-48.0%)	0.485 (-46.6%)	0.318 (-64.9%)
$\overline{\lvert e_v \rvert}$ [m/s]	0.268	0.0451 (-83.2%)	0.0861 (-67.9%)	0.0580 (-78.3%)
Input commands				
$\overline{\lvert \delta \rvert}$ [°]	1.194	1.026 (-14.0%)	1.132 (-5.2%)	1.091 (-8.6%)
$\overline{\gamma_t}$ [0-100]	48.84	48.90 (+0.14%)	48.48 (-0.73%)	48.83 (-0.02%)
$\overline{\gamma_b}$ [0-100]	4.24	3.67 (-13.4%)	3.45 (-18.7%)	3.72 (-12.37%)
$\overline{\lvert y_r \rvert}$ [m]	0.130	0.068 (-47.7%)	0.104 (-19.7%)	0.105 (-19.23%)
Riding aggressiveness				
$\overline{\lvert \alpha_f \rvert}$ [°]	1.055	0.837 (-20.6%)	0.993 (-5.89%)	0.918 (-13.0%)
$\overline{\lvert \alpha_r \rvert}$ [°]	0.564	0.474 (-15.9%)	0.545 (-3.3%)	0.511 (-9.5%)
Lap Time				
T_{lap} [s]	66.94	66.14 (-1.20%)	66.30 (-0.96%)	66.25 (-1.03%)
Computation Time				
$\overline{T_{solver}}$ [ms]	6.57	22.84 (+248%)	24.08 (+267%)	19.12 (+191%)

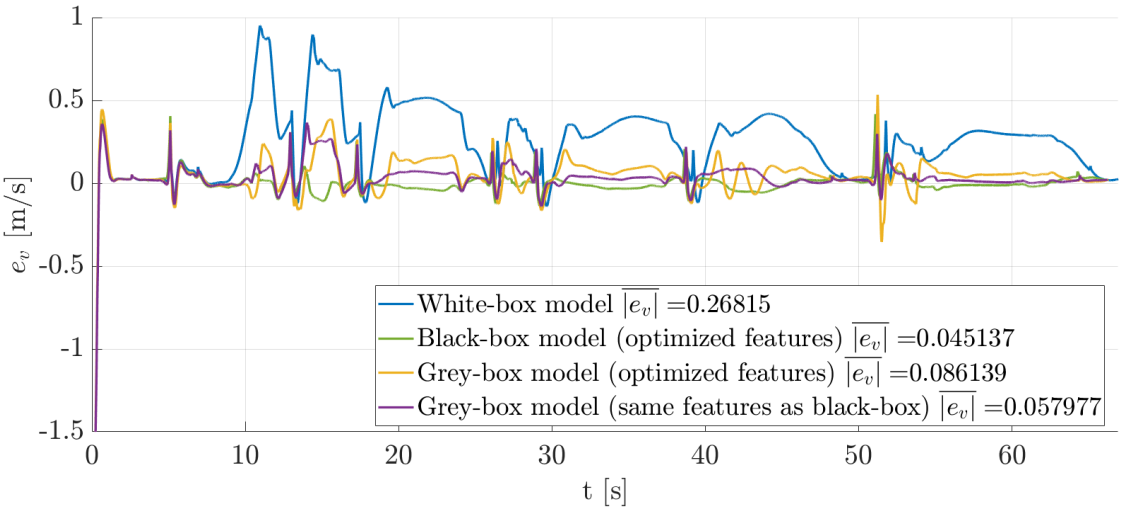
The detailed error trajectories obtained using the different strategies are reported in Figure 6.5 and they highlight the improvements provided by the learning-based strategies, as errors are consistently below the physics-based counterpart and the path following task is attained with a higher degree of success. This seems to be a general statement to the improved riding behavior of the learning-based solutions.



(a) Evolution of the lateral trajectory error e_y.



(b) Evolution of the yaw trajectory error e_ψ.



(c) Evolution of the velocity trajectory error e_v

Figure 6.5: Tracking errors' evolution attained using the physics-based model (blue) and the learning-based ones (the black-box one in green, the grey-box one w/ custom features in yellow, the grey-box one w/ bb features in purple). From top to bottom the lateral error e_y (a), the yaw error e_ψ (b), the velocity error (c).

A further error analysis is reported in Figure 6.6, where a specific focus was given to the tracking performance in the chicane maneuver of *VI-Track*. Again, it can be seen that learning-based strategies lead to the strongest adherence to the reference path.

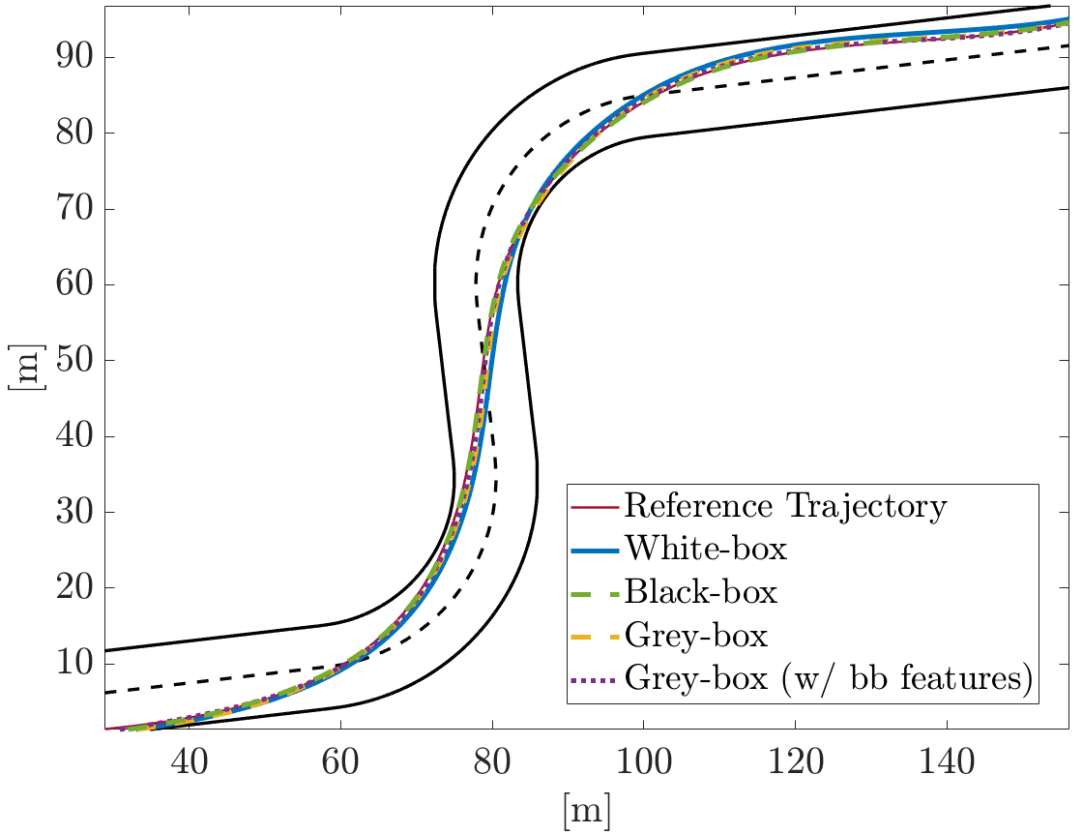


Figure 6.6: Greater detail of the tracking performance in the chicane maneuver of *VI-Track*.

From the analyses that were conducted up to now, the beneficial impact of learning-based NMPC strategies is evident. In particular, the black-box strategy appears as the gold standard, as it possesses the best predictive properties and leads to the best closed-loop performance. The grey-box learning strategies, while still providing improvements, are not up to par. It is also interesting to note that the grey-box model using the custom features found through the associated input analysis is better than the grey-box solution employing the black-box features in terms of prediction quality, but the converse is true in terms of performance. This highlights a general limitation of the feature selection procedure, as it does not necessarily lead to the best performing models in the closed-loop scenario.

Since a meaningful objective is to also reduce the computational complexity of the devised solutions, the next section is devoted to finding more computationally-sound strategies, taking the black-box model as a comparative benchmark, as it led to the best results so far.

6.2 COMPUTATIONALLY-SOUND LEARNING-BASED SOLUTIONS

The analysis undertaken in Section 6.1 was meant to compare high-fidelity models in order to establish a higher ceiling performance that could be attained through learning-based solutions. The computational load of the models was not a priority in such analysis.
This section is thus devoted to filling this gap, as new models were investigated in order to reduce the computational load of the resulting optimization problem. Before delving into the strategies that produced the most interesting results in such sense, some key observations are in order:

- The reduction produced by *VFE*, which leads to models depending on few inducing points, is not generally sufficient to guarantee a significant computational load reduction. Other online reductions should be considered, as already mentioned in Section 3.4. There, two main methods were presented:
  1. The *nearest neighbor* approach, which employs the closest GP points to the current state for prediction at each NMPC step;
  2. The *transductive learning* approach, which employs the previous-step NMPC trajectory in order to provide *transductive* predictions for the current state using the *FITC* formulas.
- Trials employing the two online approximation approaches highlighted two main facts
  1. The *transduction* approach leads to better prediction properties and to better closed-loop performances than the *nearest neighbor* counterpart.
  2. Online approximation methods work well only with grey-box modelling approaches. This might be related to the fact that the physics component grants additional robustness, which in turn can be exploited in order to further approximate the learning-based components.
- The computational load of GP models is also largely influenced by the number of features used for the learning models.

The considerations above led to exploring grey-box models based on few features that could be employed with online approximations, in order to reduce the computational load while maintaining part of the benefits shown by learning-based models. This twofold objective resulted in the following strategies:

- Strategy I: Grey-box modeling of $\ddot{\psi}$ and $\ddot{\theta}$ through a 100-point reduced GP model for each. Both targets are to be predicted with a reduced number of features:
  - 3 features for $\ddot{\psi}$ ($[1(\theta), 5(\dot{\theta}), 2(v_x)]$), according to the original analysis from Chapter 4
  - 3 features for $\ddot{\theta}$ ($[1(\theta), 8(\gamma_b), 2(v_x)]$), according to the original analysis from Chapter 4.

Transduction is used for both models employing a ratio of $h = 4$ (meaning that given a NMPC grid of 80 points, 20 of them are used for transduction).

- Strategy II: Grey-box modeling of $\ddot{\psi}$ through a 100-point reduced GP and black-box modeling of $\ddot{\theta}$ through a 50-point reduced GP. Modeling entails a reduced number of features:
  - 3 features for $\ddot{\psi}$ ($[1(\theta), 5(\dot{\theta}), 2(v_x)]$), according to the original analysis from Chapter 4
  - 4 features for $\ddot{\theta}$ ($[1(\theta), 2(v_x), 4(\dot{\psi}), 9(y_r)]$), according to the original analysis from Chapter 4.

  Transduction is used only for the yaw model, employing a ratio of $h = 4$. This "mixed" strategy (in terms of grey/black-box modeling), which may seem unusual, will soon be justified, as it retains most of the improvements provided by the high-quality GP models of Section 6.1, while leading to a reduced computational complexity.

- Strategy III: Black-box modeling of $\ddot{\psi}$ and $\ddot{\theta}$ through a 50-point reduced GP; it corresponds to the black-box model analysed in in Section 6.1. It is here used as benchmark for the other two more computationally-sound solutions.

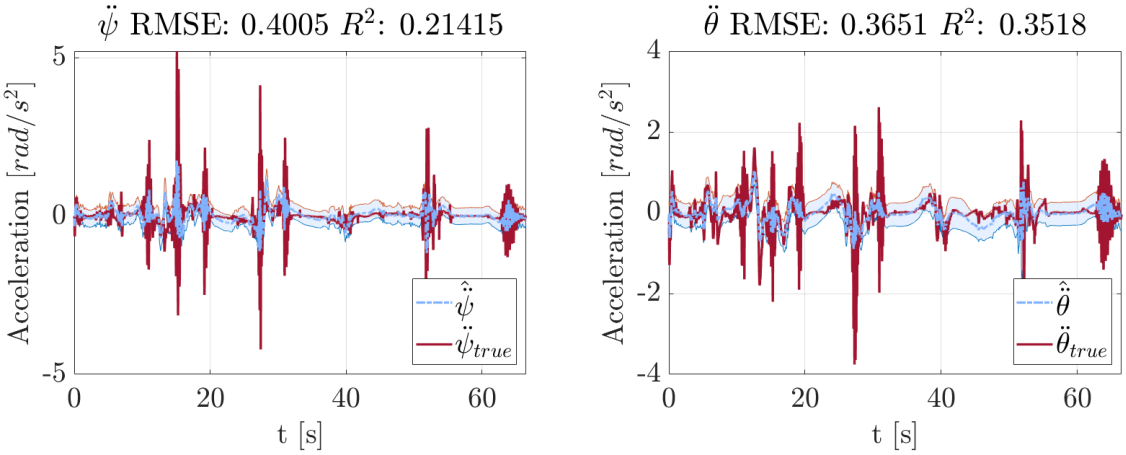
Strategies I/II will now be analysed in terms of prediction capabilities, performance and computational cost in order to verify how they compare with the gold-standard solution (strategy III). The testing approach is analogous to the one undertaken in Section 6.1.

6.2.1 *Prediction quality analysis*

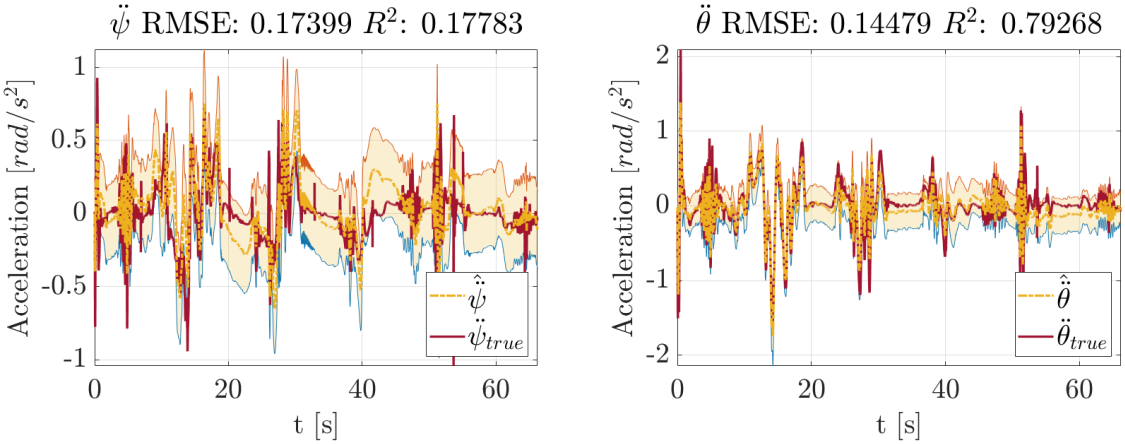
A brief prediction analysis is conducted by simply comparing the fit that models from strategies I/II attain on the targets $\ddot{\psi}$ and $\ddot{\theta}$ in the closed-loop trials w.r.t. the gold-standard solution of strategy III. The results of such analysis are reported in Figure 6.7, with strategy I in plot (a), strategy II in plot (b) and strategy III in plot (c). They highlight the following:

- Strategies I/II understandably provide worse fits than strategy III, especially in terms of $\ddot{\psi}$, as they employ additional approximations. The fit attained by strategy II on $\ddot{\theta}$ is higher quality as it entails no online approximation. In any case, even when using online reductions, the general tendencies of the accelerations are assessed correctly.

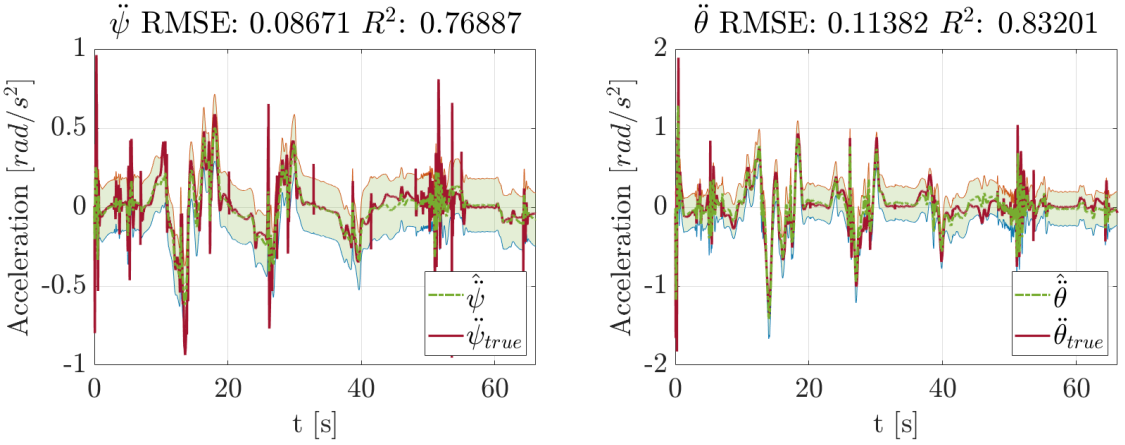
- The acceleration evolution shows some additional oscillatory behavior when employing strategy I. This is not true for strategy II, which is closer to the benchmark provided by strategy III in terms of acceleration evolution.



(a) Closed-loop fit using strategy I.



(b) Closed-loop fit using strategy II.



(c) Closed-loop fit using strategy III.

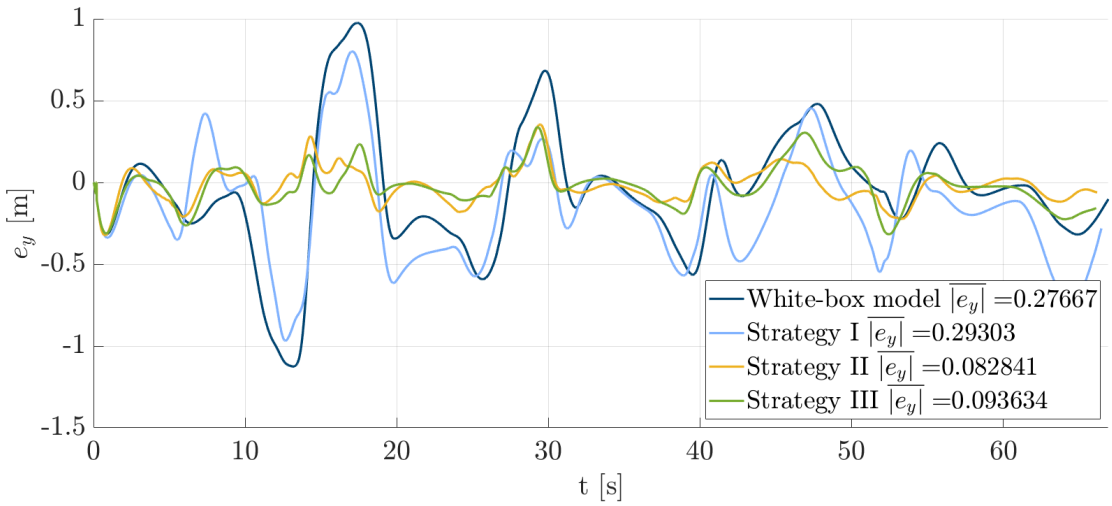
Figure 6.7: The fit provided by computationally-sound learning-based models in their respective closed-loop trials. Results for the $\ddot{\psi}$ and $\ddot{\theta}$ targets are shown according to the following color scheme: blue for strategy I, yellow for strategy II, purple for strategy III (the performance benchmark). The 2σ region is shown as a colored shaded area.

6.2.2 *Closed-loop performance analysis*

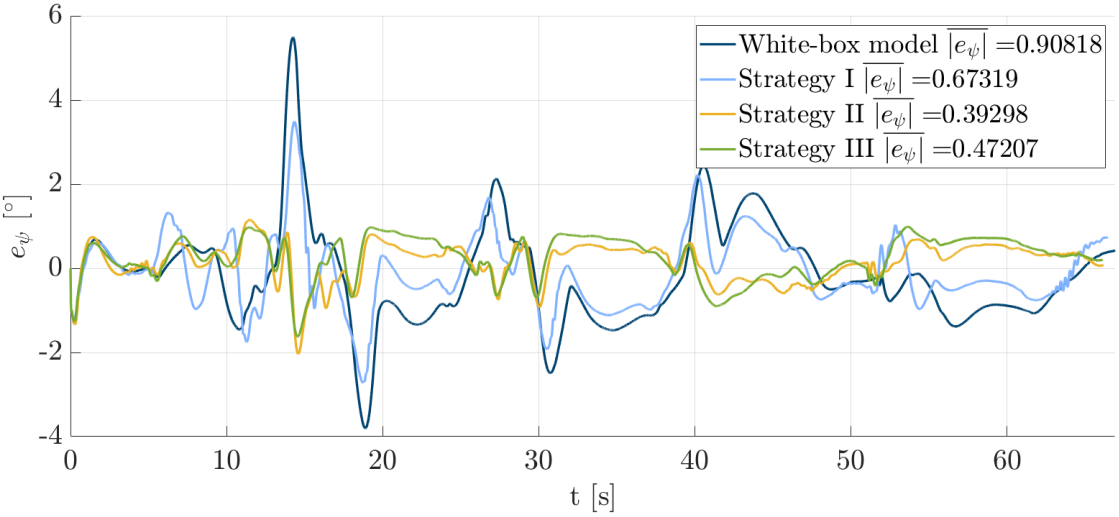
The closed-loop performance is evaluated according to criteria that are analogous to those used in Section 6.1. In particular, average quantities of interest have been computed and reported in Table 6.3. First of all, it can be noted that both Strategy I and II lead to a reduced computational time w.r.t. the benchmark of Strategy III. This is especially true for I, having $\overline{T_{solver}} = 11.5$ms; this is easily explained by the fact that both $\ddot{\psi}$ and $\ddot{\theta}$ models employ the *transduction* approach. The mixed Strategy II leads to smaller computational improvements ($\overline{T_{solver}} = 17.8$ms). At the same time, it retains most of the performance enhancements, with the tracking performance being close and in some regards better than the benchmark (Strategy III). It shows a similar performance also in terms of input requirements and riding aggressiveness. The same cannot be said for Strategy I, which has more muted performance improvements w.r.t. the physics-based model. As a consequence, it appears that Strategy II (the "mixed" model) constitutes a valid compromise between computational savings and performance requirements. A more detailed view of the trajectory error evolution is reported in Figure 6.8, which shows the improvements brought by strategy I and in particular Strategy II, as it behaves quite similarly to Strategy III in terms of tracking properties.

Table 6.3: Summary table of the closed-loop performance results for the computationally-sound learning-based strategies.

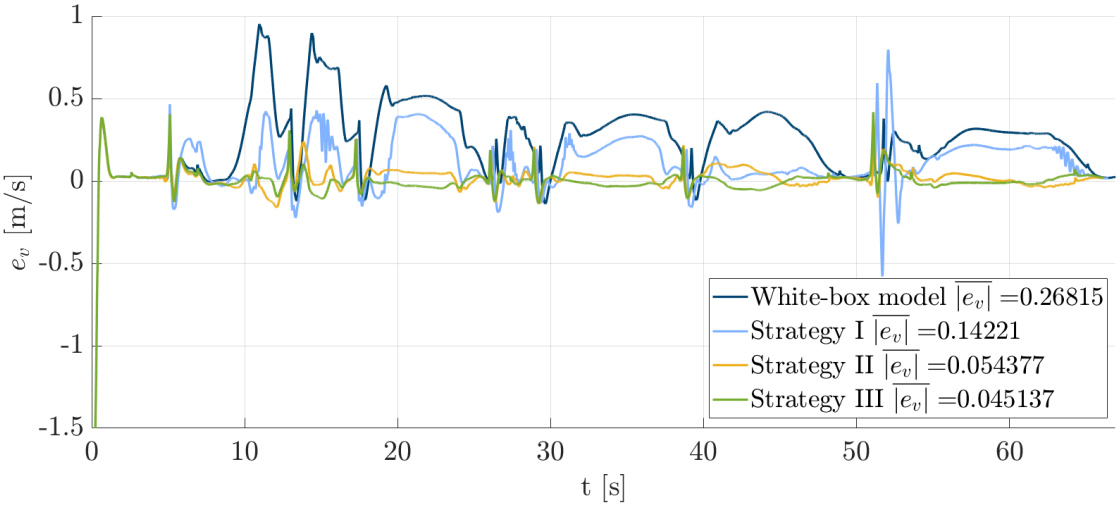
	NOMINAL	STRATEGY I	STRATEGY II	STRATEGY III
Tracking performance				
$\overline{\lvert e_y \rvert}$ [m]	0.277	0.293 (+5.92%)	0.083 (-70.0%)	0.094 (-66.2%)
$\overline{\lvert e_\psi \rvert}$ [°]	0.908	0.673 (-25.9%)	0.393 (-56.7%)	0.472 (-48.0%)
$\overline{\lvert e_v \rvert}$ [m/s]	0.268	0.142 (-47.0%)	0.054 (-79.7%)	0.045 (-83.2%)
Input commands				
$\overline{\lvert \delta \rvert}$ [°]	1.194	1.149 (-3.74%)	1.050 (-12.0%)	1.026 (-15.9%)
$\overline{\gamma_t}$ [0-100]	48.84	48.35 (-0.99%)	48.99 (+0.31%)	48.90 (+0.14%)
$\overline{\gamma_b}$ [0-100]	4.24	3.47 (-18.1%)	3.71 (-12.5%)	3.67 (-13.4%)
$\overline{\lvert y_r \rvert}$ [m]	0.130	0.107 (-17.4%)	0.056 (-57.0%)	0.068 (-47.7%)
Riding aggressiveness				
$\overline{\lvert \alpha_f \rvert}$ [°]	1.055	1.008 (-4.45%)	0.873 (-17.2%)	0.837 (-20.6%)
$\overline{\lvert \alpha_r \rvert}$ [°]	0.564	0.525 (-3.33%)	0.489 (-13.4%)	0.474 (-15.9%)
Lap Time				
T_{lap} [s]	66.94	66.48 (-0.69%)	66.20 (-1.11%)	66.14 (-1.20%)
Computation Time				
$\overline{T_{solver}}$ [ms]	6.57	11.5 (+75.3%)	17.79 (+171%)	22.84 (+248%)



(a) Evolution of the lateral trajectory error e_y.



(b) Evolution of the yaw trajectory error e_ψ.



(c) Evolution of the velocity trajectory error e_v

Figure 6.8: Tracking errors' evolution attained using Strategy I (light blue), Strategy II (yellow) and Strategy III (green), compared with physics-based model (blue). From top to bottom: the lateral error e_y (a), the yaw error e_ψ (b), the velocity error (c).

The presented comparisons prove that approaches for computational load reduction exist. In the case of the Virtual Rider implementation, reducing the number of features and using online approximations proved useful. Further analyses should be conducted in order to determine whether *VFE*-reduced models are effective even with less inducing points (i.e. $m \leq 50$) and if *transduction* may be applied with higher ratios h, in order to further reduce the computational complexity of the NMPC problem.

6.3 LAP TIME PERFORMANCE

One final analysis was undertaken in order to determine which is the best lap-time performance attainable through learning-based solutions and to verify how they compare with the physics-based counterpart. Such test was performed on the usual *VI-Track*.
In such sense, the already cited genetic tuning procedure from [13] was employed in order to find the best-performing NMPC configuration (in terms of T_{lap}) for the physics-based model and the gold standard learning-based model (the one employing black-box modeling of $\ddot{\psi}$ and $\ddot{\theta}$). As already noted, the genetic algorithm finds ideal configurations exploring a *performance* $\leftrightarrow$ *robustness* spectrum.
In particular, the procedure entailed the optimization of the NMPC weights attributed to e_ψ, e_y, e_v and α, while a constant velocity reference was maintained. Performing a 25-generation tuning on a population of 50 individuals, the obtained results are the ones shown in Figure 6.9. The configurations are evaluated and compared in terms of lap time and sensitivity index (a measure of robustness). The sensitivity index obtained for the tested strategies is in general very low, as the velocity reference is not particularly demanding (note that sensitivity is measured on a 0-100 scale). What is interesting to note is how much the different strategies may be pushed in order to reduce the lap time, with the best-performing configurations leading to:

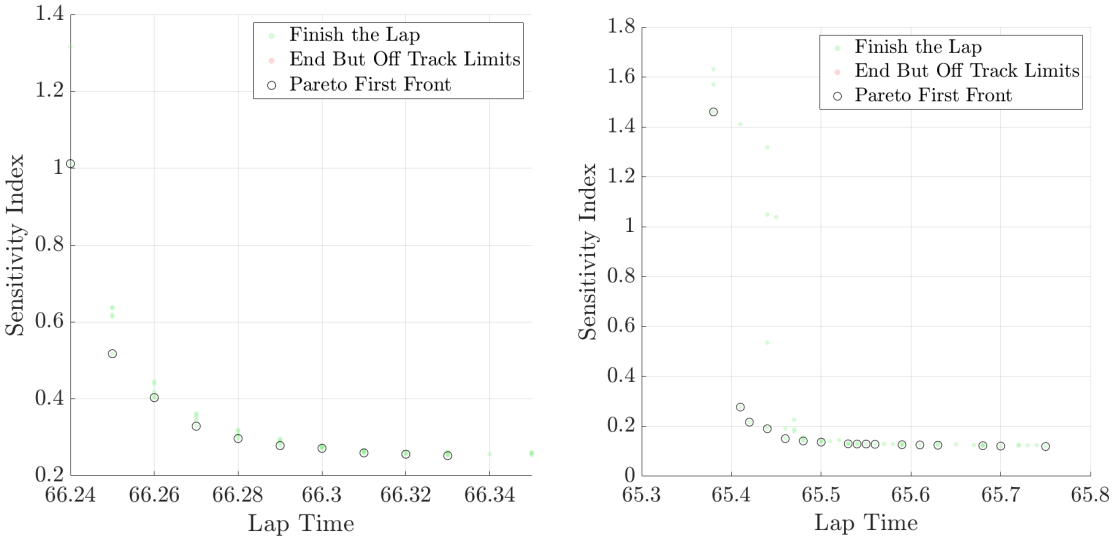
- $T^{min}_{lap,p} = 66.24$s for the physics-based model
- $T^{min}_{lap,l} = 65.38$s for the learning-based model

where p refers to the physics-based model and l to the learning-based one. Such results show that a performance-oriented tuning again leads to a lower lap time when employing the learning-based strategy w.r.t. the physics-based one. This confirms that the improvements provided by learning are not limited to the model accuracy.
Additionally, the tuned weight configurations corresponding to the fastest trials are:

$$h_{fast,p}\left(\begin{bmatrix} e_\psi + \alpha \\ e_y \\ e_v \\ \alpha \end{bmatrix}\right) = \begin{bmatrix} 0.683 \\ 0.047 \\ 0.970 \\ 7.30 \end{bmatrix} \qquad h_{fast,l}\left(\begin{bmatrix} e_\psi + \alpha \\ e_y \\ e_v \\ \alpha \end{bmatrix}\right) = \begin{bmatrix} 6.23 \\ 0.01 \\ 0.001 \\ 389 \end{bmatrix} \tag{6.3}$$

The configurations from (6.3) highlight some key differences between the strategies. Specifically, the fast configuration for the learning-based solution relies on considerably

low error weights, which may be attributed to the higher accuracy of the model, while more importance is given to the side-slip angle.
All in all, it is clear that the learning-based strategies provide improvements also in terms of lap-time, highlighting the all-around importance of moving towards higher-accuracy models.



(a) Genetic tuning results for the physics-based model.

(b) Genetic tuning results for the learning-based model.

Figure 6.9: Results obtained from the genetic tuning of the physics-based model (left) and the learning-based one (right).

Part III

CONCLUSIONS AND FUTURE DEVELOPMENTS

The objective of this conclusive section is twofold. First of all, a recap including the methodologies that were employed and the main experimental findings will be presented, along with an analysis of what worked according to expectations and what did not. Secondarily, a brief overview of the prospective research directions that may be undertaken in the world of LbMPC according to the gained insight will be presented.

7

CONCLUSIONS AND FUTURE DEVELOPMENTS

The main objective of this thesis was to explore the cutting edge field of learning-based MPC (LbMPC), which aims at improving the performance of NMPC control frameworks through the implementation of learning techniques. In particular, the sub-field of *learning dynamics* was investigated, as Gaussian Process-based dynamics modeling was the main focus of methodological research. In order to provide experimental significance to the devised learning strategies, the highly complex scenario constituted by the NMPC-based Virtual Rider was approached as a central case study to check the implementability and the usefulness of the devised learning-based models. The main keypoints and contributions of this thesis will now be summarized, before delving into prospective directions for future research.

7.1 CONCLUSIVE REMARKS

The Virtual Rider has been a case study of interest at the University of Padua during the past years. It falls under the general umbrella of virtual prototyping for the automotive industry. In particular, successful solutions for simulative motorcycle riding have already been attained in the past, employing a physics-based internal model to be used for the NMPC control problem. Such solutions were evaluated using a high-fidelity co-simulation environment, which includes a commercial simulation software (*VI-BRT*) that reproduces the motorcycle-rider behavior in a highly realistic way. The physics-based solutions proved successful in the assigned riding tasks, completing them in a timely fashion and reproducing a realistic riding behavior. This was done in conjuction with the real-time requirements of the control action. Still, open questions remained: it was clear that the underlying model was not particularly accurate, as the estimated dynamics (in terms of accelerations) were often off the mark w.r.t. the real ones. Such observation begged the question as to whether improving the accuracy of the NMPC internal model could foster significant improvements in the performance of the Virtual Rider in terms tracking, system input management and lead to lowered lap-times.
This thesis thus had as its main purpose the exploration of learning-based techniques to be used for the improvement of the Virtual Rider's internal model. The biggest focus was placed on Gaussian Process Regression modeling, which had already shown a lot of promise when applied into LbMPC schemes, proving successful also in practical implementations, as proven by the literature on the matter. Its achievements can be

traced to the high flexibility it allows, as it is a non-parametric probabilistic learning approach which can be adapted to most nonlinear modeling problems. GPR cannot be implemented without precautions in a time-sensitive field like NMPC: models should be carefully selected to be as computationally-sound as possible. For that reason, a lot of attention was devoted to researching sparse GP approximations, namely procedures that allow to distill the information of entire datasets into few inducing variables. Two main approximation methods exist in literature: the *VFE* one, which includes a variational objective function to be used for both hyperparameters and inducing input optimization, and the *FITC* one, which is based on an approximated prior which simplifies the GP predictive formulas from a computational point of view. The former is superior in terms of inducing variable selection; as such, it was used extensively to reduce the available datasets. The latter provides useful simplified formulas that may be employed online in order to provide additional computational savings (e.g. according to *transductive* learning schemes).

Another fundamental aspect to be investigated was feature selection: the acceleration dynamics clearly have different properties and should be predicted using only the most pertinent state-input information. This should be done in accord with the parsimony principle: the unnecessary feature information should be discarded to avoid misleading correlations and to simplify the model structure. To that end, feature selection methods were explored: two main criteria were devised, one based on a mutual information measure and the other based on a quality-of-fit measure. They led to competitive reduced models that could provide a satisfactory offline accuracy on the dataset used for learning. Nevertheless, it soon became clear that a satisfactory offline performance does not guarantee an effective closed-loop performance. This was particularly highlighted by some of the black-box models (the $\ddot{\psi}$ one especially), as they failed to gauge all of the acceleration dynamics properly. As a consequence, a new feature selection procedure was undertaken: it was now guided by an extended dataset, used to highlight all of the relevant dynamics. As a result, the most useful features to be used for each acceleration target were finally derived, according to both a grey-box and a black-box modeling strategy. Exploiting the feature analyses, 3 different modeling structures were derived: a grey-box model and a black-box one for the $\ddot{\psi}$ and $\ddot{\theta}$ accelerations using the optimized features, in addition to a grey-box model of the same accelerations employing the features derived for the black-box model. This last solution was added in order to verify how grey-box and black-box strategies fare when using the same information. $\dot{v}_x$ and $\dot{v}_x$ model improvements were instead avoided as they appeared unnecessary, if not harmful.

The devised learning-based strategies were then tested extensively. First of all, their predictive capabilities were evaluated in an off-line scenario, comparing their k-step accuracy. Secondly, they were tested in the closed-loop scenario: they were compared in terms of acceleration estimation accuracy, but also in terms of tracking performance, input management and lap-times. The black-box strategy proved to be the gold standard in view of its all-around quality results. The grey-box models were generally a little worse, but still provided improvements w.r.t. the physics-based solution. In particular,

the grey-box model employing the optimized features fared better in terms of predictions but worse in terms of closed-loop performance w.r.t. the grey-box solution using the same features as the black-box model. This highlighted a key finding of this thesis: models with carefully picked features have better accuracy but do not necessarily provide the best closed-loop performance. This, in turn, represents a critical limitation of the undertaken feature selection procedure and of the *learning design* framework in general, as the accuracy of the models seems to be a necessary but not sufficient condition for a good closed-loop performance.
Further tests were conducted in order to reduce the computational load of the employed models. This entailed the exploration of *transductive* methods, which allow to significantly reduce the complexity of the optimization problem to be solved by the NMPC. They proved effective only with grey-box learning formulations, presumably because the physics-informed foundation grants additional robustness. Trials showed that most of the learning-related improvements shown by the black-box model (the "gold standard") could be retained when using a simpler model, in which $\ddot{\psi}$ was modeled through a grey-box strategy (allowing *transductive* approaches) while $\ddot{\theta}$ was modeled through a black-box strategy.

7.2 FUTURE DIRECTIONS

Making the most of the insight which was gained through this thesis, a discussion on which branches of research may be investigated next is conducted.

7.2.1 *Providing a link between learning and performance through learning design approaches*

As mentioned multiple times, a critical limitation of the *learning dynamics* approach that was undertaken in this thesis is that it generally provides no a priori guarantees: it has become clear that models that seem to correctly represent the system dynamics in offline tests do not always fare as well in the closed-loop scenario. Such problem, which was in part tackled through an enlarged training set and a revised feature analysis, was never properly solved. While it is certainly possible that more robust feature selection approaches and more finely tuned models may be the key, it could be interesting to also explore learning approaches that explicitly take the closed-loop performance into account. That is the case when adopting a *learning design* point view. Works like [30] show that it is in general possible to link learning and performance objectives. In that case, Bayesian MPC is employed in order to provide a control policy that trades-off information extraction and knowledge exploitation when applied to the system. In such sense, it takes a step further w.r.t. the passive *dynamics learning* approach which has been followed in this thesis. The latter, while mostly beneficial, has proven that a general link between model accuracy and MPC efficacy cannot be provided in a straightforward way.

7.2.2 *Further Improvements to the Learning Dynamics approach*

While extending the point of view beyond *dynamics learning* is certainly appealing, there is still much to explore in this field.
It is clear that complex dynamical systems like the motorcycle one require a lot of care when implementing learning-based internal models. In particular, the success of a given model stems not only from its fit capabilities, but also from its properties; these, in turn, mostly depend on two factors when employing GP-based models: the features that are used for prediction and the kernel structure. In such sense, it is evident that renewed attention should be devoted to the feature selection procedure, which proved particularly fragile, as it provided no general guarantee for the closed-loop performance. Additional selection criteria and a refinement of the strategies devised as part of this thesis should be considered. Additionally, it would be advisable to embed the desired dynamic models' properties directly into the kernel structures; this could go a long way in providing additional guarantees for the closed-loop performance.
Possible GP model improvements do not end with kernel structure exploitation. Ditching the independence assumption made for the system accelerations, favoring instead a multi-output GP solution [31], could be beneficial for applications like the Virtual Rider, which entails highly coupled dynamics. Detecting such couplings may in fact lead to more robust learning-based dynamics models.
As a final note, active learning strategies [32] should be mentioned. This thesis mostly dealt with *passive learning* strategies, save for the incremental learning aspects: data is retrieved passively through simple trials and then exploited for model training. Conversely, *active learning* strategies aim to embed *exploration* $\leftrightarrow$ *exploitation* schemes directly into the MPC controller, so that safe exploration of the state-input space may be fostered. The additional system knowledge resulting from *exploration* can then be exploited to attain a better control performance.

Part IV

APPENDIX

The Appendix includes supplemental material to the thesis, including:

- Appendix A, where some complementary math to Chapter 3 is reported.
- Appendix B, where a comprehensive specification of the physics-based model presented in Chapter 2 is reported, along with the exact parameter configuration of the motorcycle model.
- Appendix C, where the feature analysis' results for $\dot{v}_x$ and $\dot{v}_y$ are reported; they are complementary to the results shown in Chapter 4.
- Appendix D, where additional results of the closed-loop trials employing the learning-based models are shown. It is supplementary w.r.t. the material shown in Chapter 5.

A

MATHEMATICAL APPENDIX

A.1 GAUSSIAN IDENTITIES

The following Gaussian properties have been used for the derivation of the marginal likelihood for the full Gaussian Process model.
The general notation is of the following type:

$$p(\boldsymbol{x}|\boldsymbol{m},\Sigma) = (2\pi)^{-D/2}|\Sigma|^{-1/2}\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{m})\right) \tag{A.1}$$

where

1. $\boldsymbol{m}$ is the *mean* vector (of length D)
2. Σ is the symmetric and positive definite *covariance* matrix (of size $(D \times D)$)

In particular, the product of two Gaussians gives another (un-normalized) Gaussian

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{a},A)\mathcal{N}(\boldsymbol{x}|\boldsymbol{b},B) = Z^{-1}\mathcal{N}(\boldsymbol{x}|\boldsymbol{c},C) \tag{A.2}$$

where

$$\boldsymbol{c} = C(A^{-1}\boldsymbol{a} + B^{-1}\boldsymbol{b}) \tag{A.3a}$$

$$C = (A^{-1} + B^{-1})^{-1} \tag{A.3b}$$

The normalizing constant has the form

$$Z^{-1} = (2\pi)^{-D/2}|A+B|^{-1/2}\exp\left(-\frac{1}{2}(\boldsymbol{a}-\boldsymbol{b})^T(A+B)^{-1}(\boldsymbol{a}-\boldsymbol{b})\right) \tag{A.4}$$

A.2 FITC POSTERIOR DERIVATION

The complete procedure for the derivation of the *FITC* posterior distribution is here reported.
The starting point is the *FITC* prior already introduced in (3.26), which is:

$$p(\mathbf{f},\mathbf{f}_*,\mathbf{f}_\mathbf{u}) \sim \mathcal{N}\left(\begin{bmatrix} m(x_1) \\ \vdots \\ m(x_n) \\ m(X_*) \\ m(X_u) \end{bmatrix}, \begin{bmatrix} K_{f_1f_1} & \dots & Q_{f_1f_n} & Q_{f_1*} & K_{f_1u} \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ Q_{f_nf_1} & \dots & K_{f_nf_n} & Q_{f_n*} & K_{f_nu} \\ Q_{*f_1} & \dots & Q_{*f_n} & K_{**} & k_{*u} \\ K_{uf_1} & \dots & K_{uf_n} & K_{u*} & K_{uu} \end{bmatrix}\right) \tag{A.5}$$

Clearly $\tilde{K}_{ff} = Q_{ff} + diag(K_{ff} - Q_{ff}) = Q_{ff} + \Lambda_{ff}$. Starting from this prior, it is easy to find the posterior distribution for the targets $\mathbf{f}_*$, which will be of the type $p(\mathbf{f}_*|\mathbf{f}) \sim \mathcal{N}(\tilde{\boldsymbol{\mu}}_*, \tilde{\Sigma}_*)$, where

$$\tilde{\boldsymbol{\mu}}_* = m(X_*) + Q_{*f}\tilde{K}_{ff}^{-1}(\mathbf{f} - m(X)) \tag{A.6a}$$

$$\tilde{\Sigma}_* = K_{**} - Q_{*f}\tilde{K}_{ff}^{-1}Q_{f*} \tag{A.6b}$$

Similarly, the posterior distribution for the inducing inputs will be $p(\mathbf{f}_u|\mathbf{f}) \sim \mathcal{N}(\tilde{\boldsymbol{\mu}}_u, \tilde{\Sigma}_u)$, where

$$\tilde{\boldsymbol{\mu}}_u = m(X_u) + K_{uf}\tilde{K}_{ff}^{-1}(\mathbf{f} - m(X)) \tag{A.7a}$$

$$\tilde{\Sigma}_u = K_{uu} - K_{uf}\tilde{K}_{ff}^{-1}K_{fu} \tag{A.7b}$$

The above equations are clearly not satisfactory as they require the inversion of a $n \times n$ matrix ($\tilde{K}_{ff}$). In order to find expressions that are computationally advantageous, the matrix inversion lemma (Woodbury matrix identity) must be applied repeatedly to the expressions (A.7a-A.7b) to obtain:

$$\tilde{\boldsymbol{\mu}}_u = m(X_u) + \tilde{\Sigma}_u K_{uu}^{-1}K_{uf}\Lambda_{ff}^{-1}(\mathbf{f} - m(X)) \tag{A.8a}$$

$$\tilde{\Sigma}_u = K_{uu}(K_{uu} + K_{uf}\Lambda_{ff}^{-1}K_{fu})^{-1}K_{uu} \tag{A.8b}$$

which leads to the final posterior formulas for the targets:

$$\begin{aligned}
\tilde{\boldsymbol{\mu}}_* &= m(X_*) + K_{*u}K_{uu}^{-1}(\tilde{\boldsymbol{\mu}}_u - m(X_u)) \\
&= m(X_*) + k_{*u}K_{uu}^{-1}\tilde{\Sigma}_u K_{uu}^{-1}K_{uf}\Lambda_{ff}^{-1}(\mathbf{f} - m(X)) \\
&= m(X_*) + k_{*u}\cancel{K_{uu}^{-1}}\,\cancel{K_{uu}}(K_{uu} + K_{uf}\Lambda_{ff}^{-1}K_{fu})^{-1}\cancel{K_{uu}}\,\cancel{K_{uu}^{-1}}K_{uf}\Lambda_{ff}^{-1}(\mathbf{f} - m(X)) \\
&= m(X_*) + k_{*u}(K_{uu} + K_{uf}\Lambda_{ff}^{-1}K_{fu})^{-1}K_{uf}\Lambda_{ff}^{-1}(\mathbf{f} - m(X)) \\
&= m(X_*) + k_{*u}\Sigma K_{uf}\Lambda_{ff}^{-1}(\mathbf{f} - m(X))
\end{aligned} \tag{A.9}$$

$$\begin{aligned}
\tilde{\Sigma}_* &= K_{**} - K_{*u}K_{uu}^{-1}(K_{uu} - \tilde{\Sigma}_u)K_{uu}^{-1}K_{u*} \\
&= K_{**} - K_{*u}K_{uu}^{-1}(K_{uu} - K_{uu}(K_{uu} + K_{uf}\Lambda_{ff}^{-1}K_{fu})^{-1}K_{uu})K_{uu}^{-1}K_{u*} \\
&= K_{**} - K_{*u}\cancel{K_{uu}^{-1}}\,\cancel{K_{uu}}K_{uu}^{-1}K_{u*} + K_{*u}\cancel{K_{uu}^{-1}}\,\cancel{K_{uu}}(K_{uu} + K_{uf}\Lambda_{ff}^{-1}K_{fu})^{-1}\cancel{K_{uu}}\,\cancel{K_{uu}^{-1}}K_{u*} \\
&= K_{**} - K_{*u}K_{uu}^{-1}K_{u*} + K_{*u}(K_{uu} + K_{uf}\Lambda_{ff}^{-1}K_{fu})^{-1}K_{u*} \\
&= K_{**} - Q_{**} + K_{*u}\Sigma K_{u*}
\end{aligned} \tag{A.10}$$

where the notation $\Sigma = (K_{uu} + K_{uf}\Lambda_{ff}^{-1}K_{fu})^{-1}$ has been used.

B

MODEL PARAMETERS AND SPECIFICATIONS

B.1 PHYSICS-BASED MODEL DYNAMICS SPECIFICATIONS

The physics-based model dynamics are characterized by the following matrices. The inertia matrix is

$$\tilde{M} = \begin{bmatrix} \tilde{M}_{11} & \tilde{M}_{12} & \tilde{M}_{13} & \tilde{M}_{14} \\ \tilde{M}_{21} & \tilde{M}_{22} & \tilde{M}_{23} & \tilde{M}_{24} \\ \tilde{M}_{31} & \tilde{M}_{32} & \tilde{M}_{33} & \tilde{M}_{34} \\ \tilde{M}_{41} & \tilde{M}_{42} & \tilde{M}_{43} & \tilde{M}_{44} \end{bmatrix} \tag{B.1}$$

with entries

- $\tilde{M}_{11} = m_b + m_r$
- $\tilde{M}_{12} = \tilde{M}_{21} = 0$
- $\tilde{M}_{13} = \tilde{M}_{31} = h_b m_b s_\theta + h_r m_r s_\theta + m_r y_r c_\theta$
- $\tilde{M}_{14} = \tilde{M}_{41} = 0$
- $\tilde{M}_{22} = m_b + m_r$
- $\tilde{M}_{23} = \tilde{M}_{32} = b m_b + b_r m_r$
- $\tilde{M}_{24} = \tilde{M}_{42} = m_r y_r s_\theta - h_r m_r c_\theta - h_b m_b c_\theta$
- $\tilde{M}_{33} = I_{yy} + b^2 m_b + b_r^2 m_r + h_b^2 m_b + h_r^2 m_r - I_{yy} c_\theta^2 + I_{zz} c_\theta^2 - h_b^2 m_b c_\theta^2 - h_r^2 m_r c_\theta^2 + m_r y_r^2 c_\theta^2 + 2 h_r m_r y_r c_\theta s_\theta$
- $\tilde{M}_{34} = \tilde{M}_{43} = b_r m_r y_r s_\theta - b_r h_r m_r c_\theta - b h_b m_b c_\theta$
- $\tilde{M}_{44} = m_b h_b^2 + m_r h_r^2 + m_r y_r^2 + I_{xx}$

The gravitational effects matrix is of the form

$$\tilde{G} = \begin{bmatrix} \tilde{G}_{11} \\ \tilde{G}_{21} \\ \tilde{G}_{31} \\ \tilde{G}_{41} \end{bmatrix} \tag{B.2}$$

with entries:

- $\tilde{G}_{11} = \tilde{G}_{21} = \tilde{G}_{31} = 0$
- $\tilde{G}_{41} = -gm_r(h_r s_\theta + y_r c_\theta) - gh_b m_b s_\theta$

The input matrix has the structure:

$$\tilde{B} = \begin{bmatrix} \tilde{B}_{11} & \tilde{B}_{12} & \tilde{B}_{13} & \tilde{B}_{14} & \tilde{B}_{15} \\ \tilde{B}_{21} & \tilde{B}_{22} & \tilde{B}_{23} & \tilde{B}_{24} & \tilde{B}_{25} \\ \tilde{B}_{31} & \tilde{B}_{32} & \tilde{B}_{33} & \tilde{B}_{34} & \tilde{B}_{35} \\ \tilde{B}_{41} & \tilde{B}_{42} & \tilde{B}_{43} & \tilde{B}_{44} & \tilde{B}_{45} \end{bmatrix} \tag{B.3}$$

and entries

- $\tilde{B}_{11} = c_{\delta_G}$
- $\tilde{B}_{12} = -s_{\delta_G}$
- $\tilde{B}_{13} = 1$
- $\tilde{B}_{14} = 0$
- $\tilde{B}_{15} = -1$
- $\tilde{B}_{21} = s_{\delta_G}$
- $\tilde{B}_{22} = c_{\delta_G}$
- $\tilde{B}_{23} = 0$
- $\tilde{B}_{24} = 1$
- $\tilde{B}_{25} = 0$
- $\tilde{B}_{31} = ps_{\delta_G}$
- $\tilde{B}_{32} = pc_{\delta_G}$
- $\tilde{B}_{33} = \tilde{B}_{34} = 0$
- $\tilde{B}_{35} = -h_d s_\theta$
- $\tilde{B}_{41} = \tilde{B}_{42} = \tilde{B}_{43} = \tilde{B}_{44} = \tilde{B}_{45} = 0$

Finally, the Coriolis and centrifugal effects matrix has the form

$$\bar{C} = \begin{bmatrix} \bar{C}_{11} \\ \bar{C}_{21} \\ \bar{C}_{31} \\ \bar{C}_{41} \end{bmatrix} \tag{B.4}$$

and entries

- $\bar{C}_{11} = \dot{\psi}(m_b v_y + m_r v_y + b\dot{\psi} m_b + b_r \dot{\psi} m_r - 2\dot{\theta} h_b m_b c_\theta - 2\dot{\theta} h_r m_r c_\theta + 2\dot{\theta} m_r y_r s_\theta)$
- $\bar{C}_{21} = \dot{\psi} m_b v_x + \dot{\psi} m_r v_x + \dot{\theta}^2 h_b m_b s_\theta + \dot{\theta}^2 h_r m_r s_\theta + \dot{\psi}^2 h_b m_b s_\theta + \dot{\psi}^2 h_r m_r s_\theta + \dot{\theta}^2 m_r y_r c_\theta + \dot{\psi}^2 m_r y_r c_\theta$
- $\bar{C}_{31} = I_{wf}\dot{\theta}\omega_{wf} + I_{wr}\dot{\theta}\omega_{wr} + b\dot{\psi} m_b v_x + b_r \dot{\psi} m_r v_x + I_{yy}\dot{\theta}\dot{\psi} s_{2\theta} - I_{zz}\dot{\theta}\dot{\psi} s_{2\theta} + b\dot{\theta}^2 h_b m_b s_\theta + b_r \dot{\theta}^2 h_r m_R s_\theta + b_r \dot{\theta}^2 m_r y_r c_\theta + \dot{\theta}\dot{\psi} h_b^2 m_b s_{2\theta} + \dot{\theta}\dot{\psi} h_r^2 m_r s_{2\theta} - \dot{\theta}\dot{\psi} m_r y_r^2 s_{2\theta} - \dot{\psi} h_b m_b v_y s_\theta - \dot{\psi} h_r m_r v_y s_\theta - \dot{\psi} m_r v_y y_r c_\theta + 2\dot{\theta}\dot{\psi} h_r m_r y_r c_{2\theta}$
- $\bar{C}_{41} = \frac{1}{2} I_{zz}\dot{\psi}^2 s_{2\theta} - \frac{1}{2} I_{yy}\dot{\psi}^2 s_{2\theta} - I_{wf}\dot{\psi}\omega_{wf} c_\theta - I_{wf}\dot{\psi}\omega_{wr} c_\theta - \frac{1}{2}\dot{\psi}^2 h_b^2 m_b s_{2\theta} - \frac{1}{2}\dot{\psi}^2 h_r^2 m_r s_{2\theta} + \frac{1}{2}\dot{\psi}^2 m_r y_r^2 s_{2\theta} + \dot{\psi} m_r v_x y_r s_\theta - \dot{\psi}^2 h_r m_r y_r c_{2\theta} - \dot{\psi} h_b m_b v_x c_\theta - \dot{\psi} h_r m_r v_x c_\theta$

B.2 MOTORCYCLE AND RIDER PARAMETERS

In the following table the main parameters of the motorcycle and the rider are reported

Table B.1: Motorcycle and rider parameters

PARAMETER	SYMBOL	VALUE	UNIT OF MEASURE
Bike mass	m_b	144.010	kg
Distance between wheel contact patches	p_b	1.32	m
Bike's CM x-coordinate	b_b	0.64515	m
Bike's CM z-coordinate (height)	h_b	0.60226	m
Bike's moment of Inertia along z-axis	I_{zz}	93.9329	$\mathrm{kg\,m^2}$
Bike's moment of Inertia along y-axis	I_{yy}	150.3812	$\mathrm{kg\,m^2}$
Bike's moment of Inertia along x-axis	I_{xx}	61.8557	$\mathrm{kg\,m^2}$
Caster angle	ϵ	$\frac{25}{180}\pi$	rad
Front wheel mass	m_{wf}	10.004	kg
Rear wheel mass	m_{wr}	10.004	kg
Front wheel radius	r_{wf}	0.306	m
Rear wheel radius	r_{wr}	0.322	m
Front wheel spin inertia	I_{wf}	0.4347	$\mathrm{kg\,m^2}$
Rear wheel spin inertia	I_{wr}	0.8032	$\mathrm{kg\,m^2}$
Initial Max engine torque	$\tau^M_{t,start}$	28.50	N m
Initial Min engine torque	$\tau^m_{t,start}$	0	N m
1^{st} gear ratio	gr_1	0.3333	–
2^{nd} gear ratio	gr_2	0.4255	–
3^{rd} gear ratio	gr_3	0.5263	–
4^{th} gear ratio	gr_4	0.5882	–
5^{th} gear ratio	gr_5	0.6369	–
6^{th} gear ratio	gr_6	0.6667	–
Primary gear ratio	$gr_{primary}$	0.5714	–

Final gear ratio	gr_{final}	0.4000	–
Maximum frontal braking torque	τ_b^f	648	N m
Maximum rear braking torque	τ_b^r	67.5	N m
Front-rear brake bias	λ	0.75	–
Air density	ρ	1.225	kg/m^3
Drag coefficient	C_d	0.41	–
Front section area	A	0.6	m^2
Air density	ρ	1.225	kg/m^3
Rider mass	m_r	70	kg
Rider's CM x-coordinate w.r.t. gyro reference frame	$x_{cm,r}$	0.5519	m
Rider's CM z-coordinate (height) w.r.t. gyro reference frame	$z_{cm,r}$	0.9140	m

C

FEATURE ANALYSES INVOLVING $\dot{v}_x$ AND $\dot{v}_y$

This chapter is devoted to reporting the feature analysis results obtained for $\dot{v}_x$ and $\dot{v}_y$, which were omitted from Chapter 4, as they were not part of the final closed-loop implementation.
Table C.1 and Table C.2 show the results for the mutual information feature analysis. Table C.3 and Table C.4 show the results derived from the fit-based feature analysis.

Table C.1: Feature combinations with the highest information content for the **black-box** targets. For each black-box target ($\dot{v}_x$, $\dot{v}_y$), the 3 best combinations of 1, 2, 3, 4 and 5 features are shown, along with the corresponding mutual information index.

TARGETS	$\dot{v}_x$			$\dot{v}_y$		
	1st	2nd	3rd	1st	2nd	3rd
1 FEAT	[7]	[2]	[6]	[6]	[5]	[1]
	2.6355	2.2729	2.0352	1.3692	1.3159	1.2655
2 FEAT	[2, 7]	[6, 7]	[4, 7]	[5, 6]	[3, 5]	[4, 5]
	3.9853	3.6959	3.6153	2.5371	2.5148	2.4977
3 FEAT	[2, 7, 8]	[2, 7, 12]	[2, 7, 9]	[2, 5, 15]	[1, 2, 5]	[2, 3, 5]
	3.921	3.7867	3.5814	2.4082	2.404	2.3973
4 FEAT	[2, 7, 8, 12]	[2, 7, 8, 9]	[6, 7, 8, 15]	[3, 5, 8, 11]	[3, 5, 6, 8]	[4, 5, 8, 15]
	3.6405	3.406	3.3524	2.1941	2.1884	2.1857
5 FEAT	[2, 7, 8, 9, 12]	[2, 7, 8, 12, 13]	[6, 7, 8, 12, 15]	[3, 5, 8, 11, 15]	[1, 4, 5, 8, 11]	[3, 5, 8, 11, 12]
	3.1357	3.1047	3.0917	2.0392	2.0375	2.031

Table C.2: Feature combinations with the highest information content for the **grey-box** targets. For each grey-box target ($\phi_{\dot{v}_x}$, $\phi_{\dot{v}_y}$), the 3 best combinations of 1, 2, 3, 4 and 5 features are shown, along with the corresponding mutual information index.

TARGETS	$\phi_{\dot{v}_x}$			$\phi_{\dot{v}_y}$		
	1st	2nd	3rd	1st	2nd	3rd
1 FEAT	[4]	[3]	[6]	[1]	[3]	[15]
	1.5006	1.4959	1.458	2.0644	2.0088	1.9068
2 FEAT	[2, 14]	[2, 3]	[4, 7]	[2, 3]	[2, 14]	[1, 2]
	2.7163	2.7043	2.6991	3.2181	3.1962	3.1789
3 FEAT	[1, 2, 7]	[2, 4, 7]	[2, 7, 14]	[1, 6, 14]	[2, 3, 6]	[1, 3, 7]
	2.74	2.7285	2.7215	3.1438	3.1314	3.1283
4 FEAT	[2, 4, 7, 12]	[1, 2, 7, 12]	[2, 7, 12, 16]	[1, 6, 8, 15]	[1, 6, 14, 15]	[1, 3, 4, 15]
	2.5375	2.5278	2.527	2.9597	2.9586	2.9539
5 FEAT	[1, 4, 7, 8, 12]	[1, 6, 7, 8, 12]	[4, 7, 8, 12, 16]	[1, 3, 6, 8, 15]	[1, 3, 4, 8, 15]	[1, 6, 8, 14, 15]
	2.3817	2.3714	2.371	2.8135	2.8026	2.802

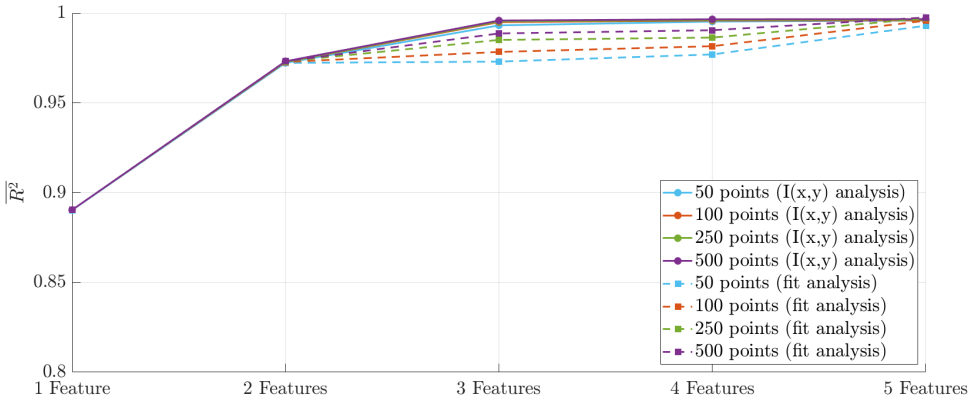
Table C.3: Feature combinations providing the best fit for the **black-box** targets on the whole dataset after training on a subset of 2000 points. The fit measure is given by $\overline{R^2}$, the average R^2 index over the 3 tracks' data, which is reported for each feature combination. For each black-box target ($\dot{v}_x$, $\dot{v}_y$), combinations of up to 7 features are shown.

TARGETS	$\dot{v}_x$	$\dot{v}_y$
1 Feat	[7]	[5]
	0.8861	0.3665
2 Feat	[7,2]	[5,10]
	0.9718	0.5619
3 Feat	[7,2,1]	[5,10,9]
	0.9828	0.6042
4 Feat	[7,2,1,9]	[5,10,9,14]
	0.9850	0.7305
5 Feat	[7,2,1,9,8]	[5,10,9,14,2]
	0.9882	0.7463
6 Feat	[7,2,1,9,8,5]	[5,10,9,14,2,15]
	0.9901	0.8393
7 Feat	[7,2,1,9,8,5,14]	[5,10,9,14,2,15,16]
	0.9905	0.8499

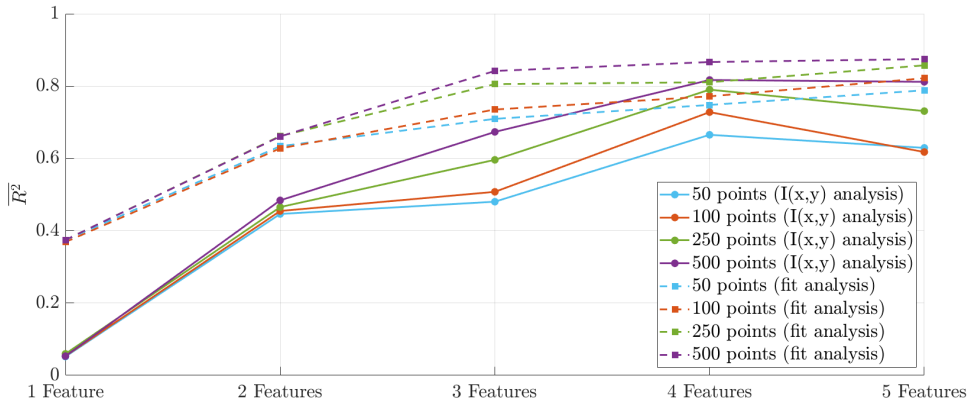
Table C.4: Feature combinations providing the best fit for the **grey-box** targets on the whole dataset after training on a subset of 2000 points. The fit measure is given by $\overline{R^2}$, the average R^2 index over the 3 tracks' data, which is reported for each feature combination. For each grey-box target ($\phi_{\dot{v}_x}$, $\phi_{\dot{v}_y}$), combinations of up to 7 features are shown.

TARGETS	$\phi_{\dot{v}_x}$	$\phi_{\dot{v}_y}$
1 Feat	[8]	[3]
	0.9849	-0.4384
2 Feat	[8,12]	[3,5]
	0.9909	0.2394
3 Feat	[8,12,15]	[3,5,2]
	0.9947	0.5369
4 Feat	[8,12,15,2]	[3,5,2,4],
	0.9955	0.7015
5 Feat	[8,12,15,2,5]	[3,5,2,4,15]
	0.9961	0.7727
6 Feat	[8,12,15,2,5,3]	[3,5,2,4,15,12]
	0.9962	0.8175,
7 Feat	[8,12,15,2,5,3,14]	[3,5,2,4,15,12,14]
	0.9962	0.8627

The chapter is completed by a comparative analysis of the best performing models (according to the feature analyses) as the feature numerosity and the number of inducing points varies (Figure C.1 for the black-box models and Figure C.2 for the grey-box models); finally, the fits provided by the 50-point models leading to the best $\overline{R^2}$ index on the *VI-Track* data are shown in Figure C.3 (black-box) and Figure C.4 (grey-box).

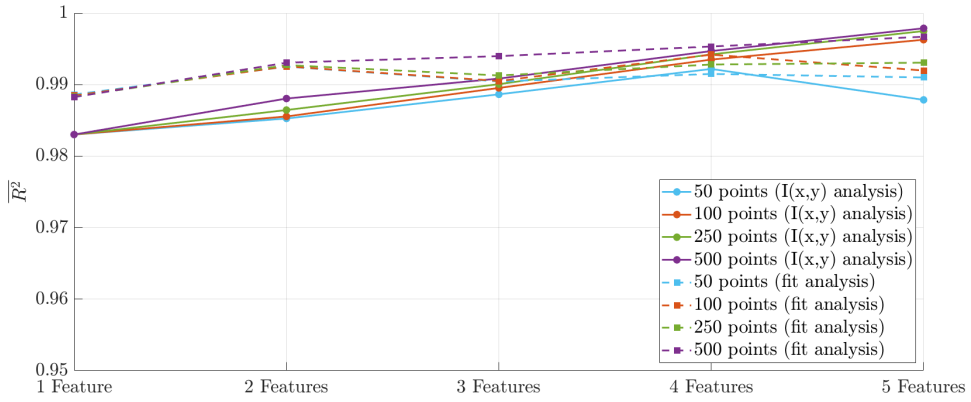


(a) $\overline{R^2}$ index for $\dot{v}_x$

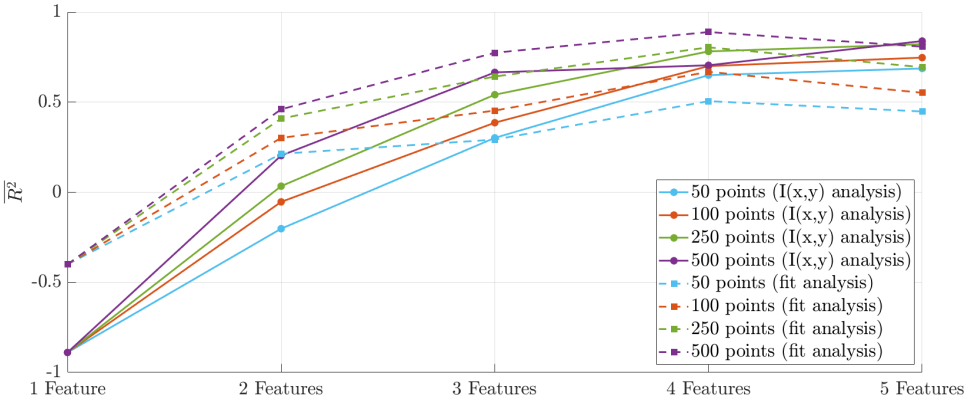


(b) $\overline{R^2}$ index for $\dot{v}_y$

Figure C.1: Results of the comparative analysis of the VSGP models for the **black-box** targets $\dot{v}_x$, $\dot{v}_y$.

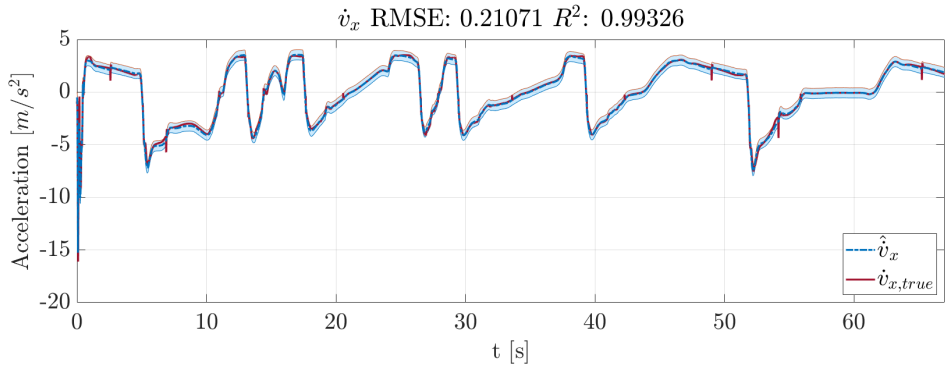


(a) $\overline{R^2}$ index for $\phi_{\dot{v}_x}$

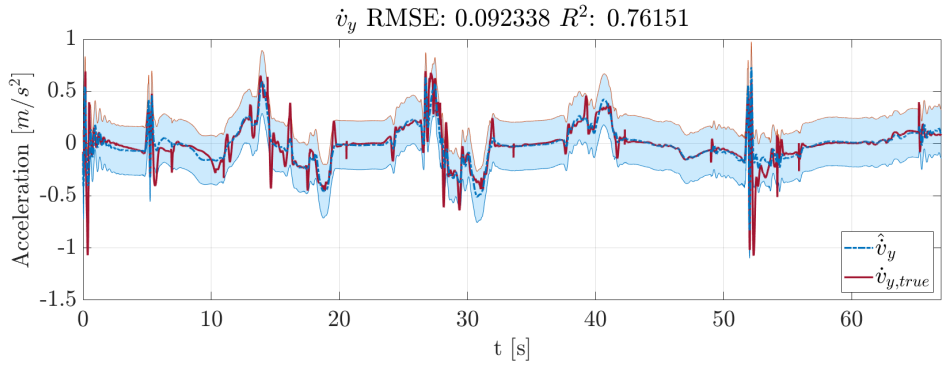


(b) $\overline{R^2}$ index for $\phi_{\dot{v}_y}$

Figure C.2: Results of the comparative analysis of the VSGP models for the **grey-box** targets $\phi_{\dot{v}_x}$, $\phi_{\dot{v}_y}$.

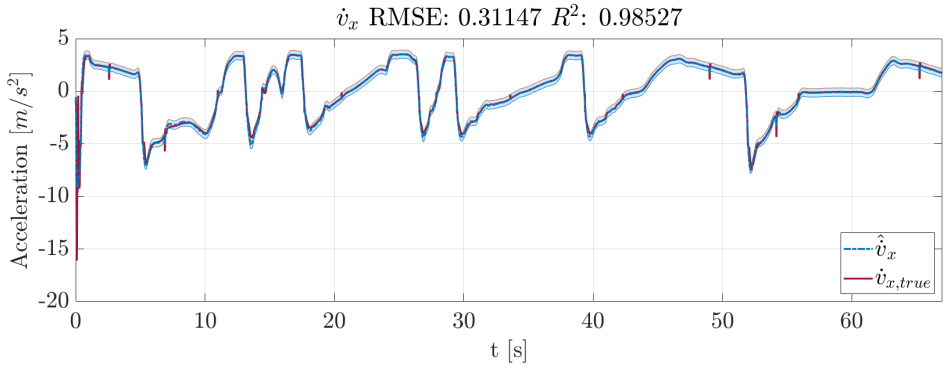


(a) Fit of the best 50-point $\dot{v}_x$ model

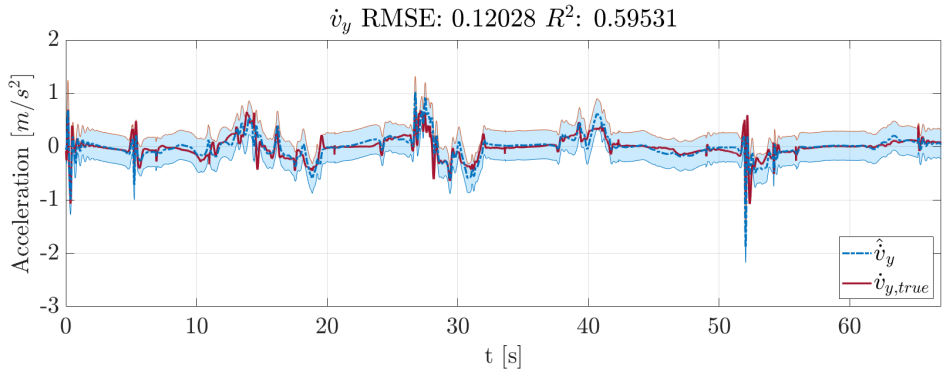


(b) Fit of the best 50-point $\dot{v}_y$ model

Figure C.3: The fit of the best 50-points **black-box** VSGP model configurations for $\dot{v}_x$, $\dot{v}_y$ (with μ_* shown with a dashed blue line and the light blue area representing $[\mu_* - 2\sigma, \mu_* + 2\sigma]$), compared with the target acceleration (shown in red).



(a) Fit using the best 50-point $\phi_{\dot{v}_x}$ model



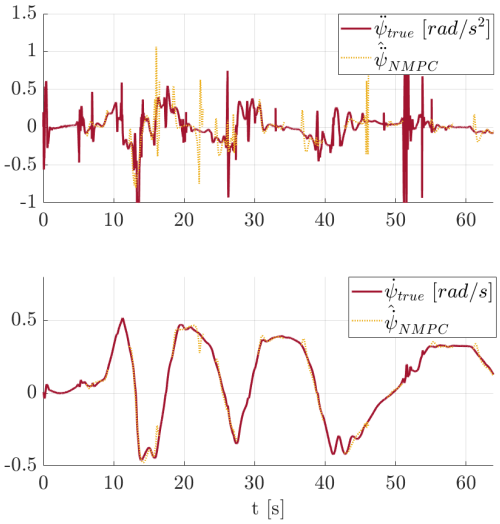
(b) Fit using the best 50-point $\phi_{\dot{v}_y}$ model

Figure C.4: The fit of the best 50-points **grey-box** VSGP model configurations for $\phi_{\dot{v}_x}$, $\phi_{\dot{v}_y}$ (with $\ddot{\tilde{q}} + \mu_*$ shown with a dashed blue line and the light blue area representing $[\ddot{\tilde{q}} + \mu_* - 2\sigma, \ddot{\tilde{q}} + \mu_* + 2\sigma]$), compared with the target acceleration (shown in red).

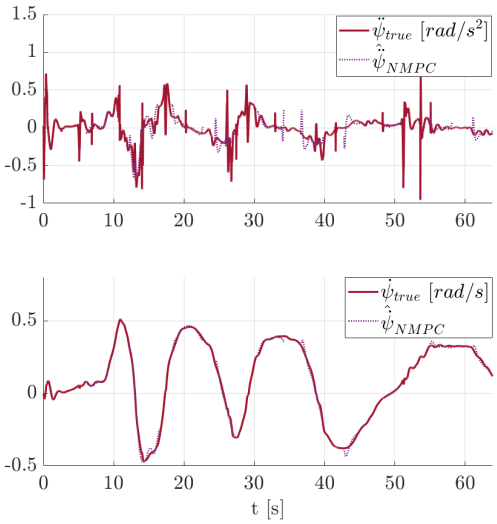
D

ADDITIONAL CLOSED-LOOP TRIALS

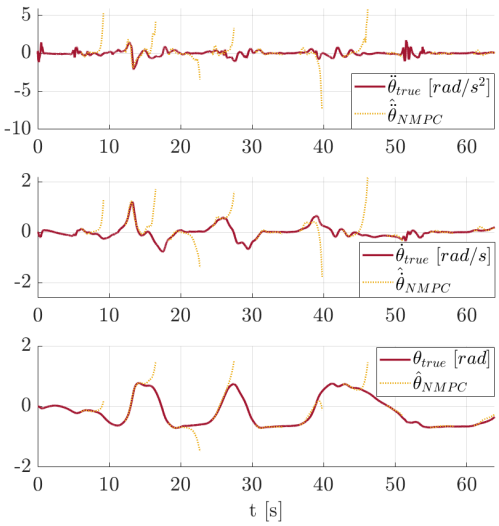
This chapter is devoted to showing some additional results that are supplementary to the analyses from Chapter 5.



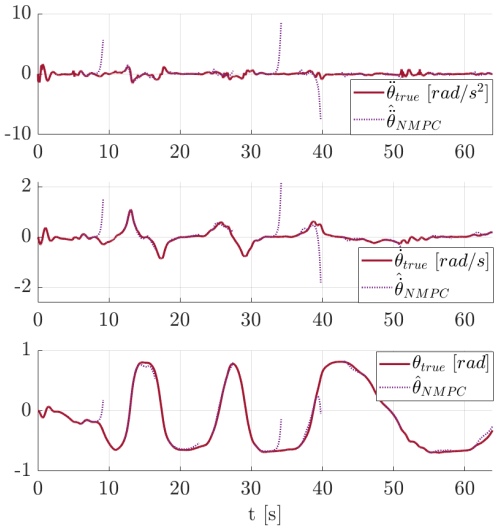
(a) Sample ψ-related NMPC predictions.



(b) Sample ψ-related NMPC predictions.



(c) Sample θ-related NMPC predictions.



(d) Sample θ-related NMPC predictions.

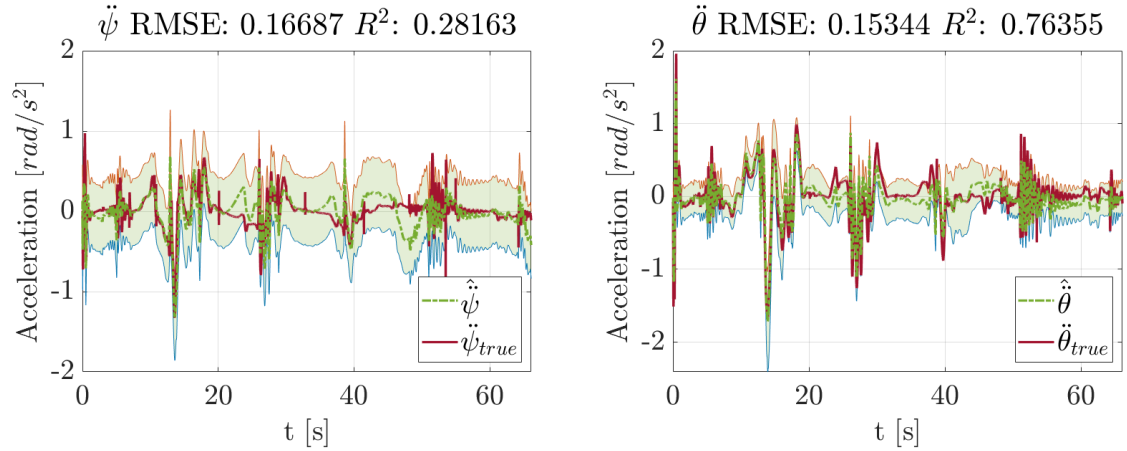
Figure D.1: Sample NMPC trajectories for the yaw-related and roll-related quantities provided by the grey-box model w/ custom features (left, in yellow) and the grey-box model w/ bb features (right, in purple).

D.1 GENERALIZABILITY ANALYSIS OF THE MODELS FROM SECTION 6.1

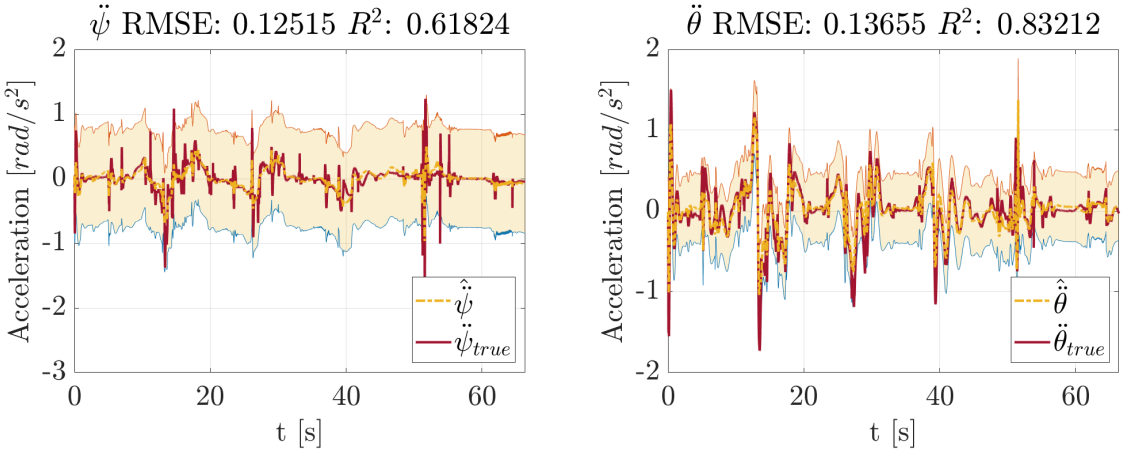
As already mentioned in Section 6.1, the models employed in the general analysis were subjected to a specific procedure, which included an incremental learning phase to improve their accuracy. It is still interesting to evaluate how the same models would fare when employing the same base dataset, in order to better gauge their generalizability properties.

In such sense, Figure D.2 shows the accuracy of the learning-based models trained just on the extended dataset (including 51180 points) and reduced to 50 inducing points through *VFE* inference. A quick analysis highlights how models employing the custom features obtained through the dedicated analysis (i.e. the black-box model shown in green and the grey-box one shown in yellow) perform well even without the incremental learning phase, showing satisfactory generalizability properties. Still, some reductions to the fit quality are present w.r.t. Figure 6.2, especially for the black-box modeled yaw acceleration. Nevertheless, the worst performance is related to the grey-box model which employs the same features as the black-box one (in purple), which seems to confirm the importance of feature selection in order to devise models that are sufficiently generalizable.

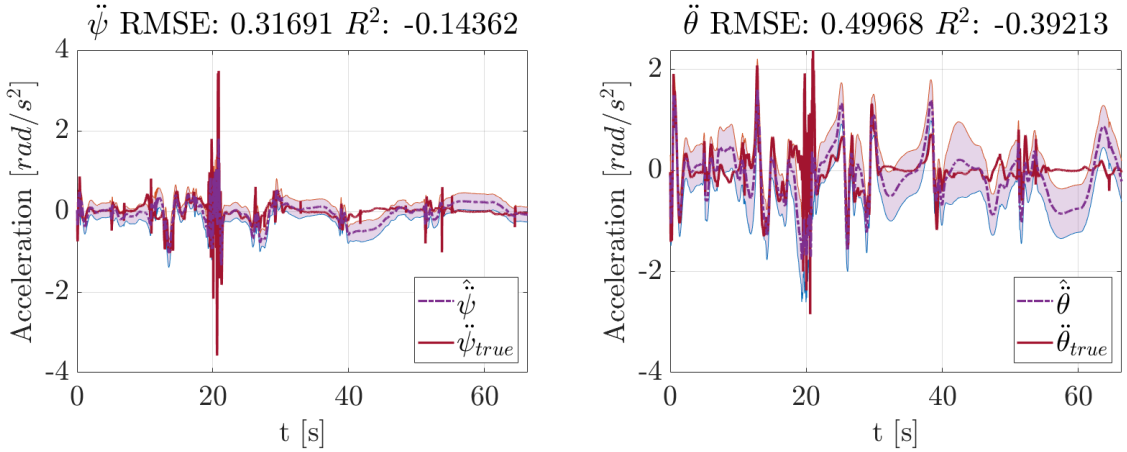
Further analysis is provided by Table D.1, showing some key average quantities resulting from the closed-loop trials. Again, they testify that the black-box and grey-box solutions employing dedicated features hold up well even without incremental learning (see Table 6.2 for comparison). The same cannot be said for the grey-box model employing the black-box features, whose performance is quite degraded w.r.t. the counterpart enriched with incremental learning (especially in terms of e_y, i.e. path following). Again, this seems to highlight the importance of dedicated features in order to guarantee an optimal closed-loop performance under varying conditions.



(a) Closed-loop fit for the black-box model.



(b) Closed-loop fit for the grey-box model w/ custom features.



(c) Closed-loop fit for the grey-box model w/ bb features

Figure D.2: The fit provided by learning-based models in their respective closed-loop trials. The models are based on the common extended dataset, without going through an incremental learning phase. Results for the $\ddot{\psi}$ and $\ddot{\theta}$ targets are shown according to the following color scheme: green for the black-box model, yellow for the grey-box model w/ custom features, purple for the grey-box model w/ bb features. The 2σ region is shown as a colored shaded area.

Table D.1: Summary table of the closed-loop results in terms of performance.

	NOMINAL	BLACK-BOX	GREY-BOX 1	GREY-BOX 2
Tracking performance				
$\overline{\lvert e_y \rvert}$ [m]	0.277	0.100 (-63.8%)	0.256 (-7.4%)	0.323 (+16.7%)
$\overline{\lvert e_\psi \rvert}$ [°]	0.908	0.452 (-50.3%)	0.595 (-34.4%)	0.883 (-2.77%)
$\overline{\lvert e_v \rvert}$ [m/s]	0.268	0.050 (-81.3%)	0.088 (-67.1%)	0.110 (-59.1%)
Input commands				
$\overline{\lvert \delta \rvert}$ [°]	1.194	1.050 (-12.1%)	1.122 (-6.1%)	1.038 (-13.1%)
$\overline{\gamma_t}$ [0-100]	48.84	48.93 (+0.19%)	49.11 (+0.57%)	49.28 (+0.92%)
$\overline{\gamma_b}$ [0-100]	4.24	3.71 (-12.6%)	3.03 (-28.3%)	4.09 (-3.6%)
$\overline{\lvert y_r \rvert}$ [m]	0.130	0.049 (-62.4%)	0.102 (-21.0%)	0.061 (-52.8%)
Riding aggressiveness				
$\overline{\lvert \alpha_f \rvert}$ [°]	1.055	0.881 (-16.5%)	0.973 (-7.8%)	0.826 (-21.7%)
$\overline{\lvert \alpha_r \rvert}$ [°]	0.564	0.496 (-12.0%)	0.535 (-5.2%)	0.471 (-16.6%)
Lap Time				
T_{lap} [s]	66.94	66.11 (-1.24%)	66.37 (-0.85%)	66.37 (-0.85%)
Computation Time				
$\overline{T_{solver}}$ [ms]	6.57	21.43 (+227%)	21.78 (+232%)	22.29 (+240%)

BIBLIOGRAPHY

[1] R. Frezza, A. Beghi, and A. Saccon. "Model predictive for path following with motorcycles: application to the development of the pilot model for virtual prototyping." In: *2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601)* 1 (2004), 767–772 Vol.1.

[2] M. Bruschetta, E. Picotti, A. De Simoi, Y. Chen, A. Beghi, M. Nishimura, Y. Tezuka, and F. Ambrogi. "Real-Time Nonlinear Model Predictive Control of a Virtual Motorcycle." In: *IEEE Transactions on Control Systems Technology* 29.5 (2021), pp. 2214–2222. DOI: 10.1109/TCST.2020.3022462.

[3] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger. "Learning-Based Model Predictive Control: Toward Safe Learning in Control." In: *Annual Review of Control, Robotics, and Autonomous Systems* 3.1 (2020), pp. 269–296. DOI: 10.1146/annurev-control-090419-075625. URL: https://doi.org/10.1146/annurev-control-090419-075625.

[4] L. Hewing, J. Kabzan, and M. N. Zeilinger. "Cautious Model Predictive Control Using Gaussian Process Regression." In: *IEEE Transactions on Control Systems Technology* 28.6 (2020), pp. 2736–2743. DOI: 10.1109/TCST.2019.2949757.

[5] A. Carron, E. Arcari, M. Wermelinger, L. Hewing, M. Hutter, and M. N. Zeilinger. "Data-Driven Model Predictive Control for Trajectory Tracking With a Robotic Arm." In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 3758–3765. DOI: 10.1109/LRA.2019.2929987.

[6] H. Liu, Y.-S. Ong, X. Shen, and J. Cai. "When Gaussian process meets big data: a review of scalable GPs." English. In: *IEEE Transactions on Neural Networks and Learning Systems* 31.11 (Jan. 2020), pp. 4405–4423. ISSN: 2162-237X. DOI: 10.1109/TNNLS.2019.2957109.

[7] R. Lot, M. Massaro, and R. Sartori. "Advanced motorcycle virtual rider." In: *Vehicle System Dynamics* 46.sup1 (2008), pp. 215–224. DOI: 10.1080/00423110801935822. eprint: https://doi.org/10.1080/00423110801935822. URL: https://doi.org/10.1080/00423110801935822.

[8] E. Picotti, M. Bruschetta, Y. Chen, A. Beghi, M. Nishimura, Y. Tezuka, and F. Ambrogi. "Design and implementation of a high-performance, nonlinear MPC-based virtual motorcycle rider." In: *2020 European Control Conference (ECC)*. 2020, pp. 706–711. DOI: 10.23919/ECC51009.2020.9143729.

[9] H. Pacejka. *Tire and Vehicle Dynamics*. Jan. 2012, p. 580. DOI: 10.1016/C2010-0-68548-8.

[10] J. T. Betts and J. M. Gablonsky. *A Comparison of Interior Point and SQP Methods on Optimal Control Problems*. Phantom Works, Mathematics & Computing Technology, A Division of The Boeing Company, Mar. 2002.

[11] Y. Chen, M. Bruschetta, E. Picotti, and A. Beghi. "MATMPC - A MATLAB Based Toolbox for Real-time Nonlinear Model Predictive Control." In: *2019 18th European Control Conference (ECC)*. 2019, pp. 3365–3370. DOI: 10.23919/ECC.2019.8795788.

[12] VI-Grade Eng. Softw. Services. *VI-BikeRealTime 20.0 Documentation*. VI-grade GmbH, Darmstadt, Germany, 2020.

[13] E. Picotti, L. Facin, A. Beghi, M. Nishimura, Y. Tezuka, F. Ambrogi, and M. Bruschetta. "Data-driven Tuning of a NMPC Controller for a Virtual Motorcycle through Genetic Algorithm." In: *2022 Conference on Control Technology and Applications (CCTA)*. 2022.

[14] M. Buisson-Fenet, F. Solowjow, and S. Trimpe. "Actively Learning Gaussian Process Dynamics." In: *Proceedings of the 2nd Conference on Learning for Dynamics and Control*. Ed. by Alexandre M. Bayen, Ali Jadbabaie, George Pappas, Pablo A. Parrilo, Benjamin Recht, Claire Tomlin, and Melanie Zeilinger. Vol. 120. Proceedings of Machine Learning Research. PMLR, 2020, pp. 5–15. URL: https://proceedings.mlr.press/v120/buisson-fenet20a.html.

[15] E. Picotti, A. Dalla Libera, R. Carli, and M. Bruschetta. "LbMATMPC: an open-source toolbox for Gaussian Process modeling within Learning-based Nonlinear Model Predictive Control." In: *2022 European Control Conference (ECC)*. 2022, pp. 736–742. DOI: 10.23919/ECC55457.2022.9838016.

[16] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, Nov. 2005. ISBN: 9780262256834. DOI: 10.7551/mitpress/3206.001.0001. URL: https://doi.org/10.7551/mitpress/3206.001.0001.

[17] G. Kimeldorf and G. Wahba. "Some results on Tchebycheffian spline functions." In: *Journal of mathematical analysis and applications* 33.1 (1971), pp. 82–95.

[18] E. Snelson and Z. Ghahramani. "Sparse Gaussian Processes using Pseudo-inputs." In: *Advances in Neural Information Processing Systems*. Ed. by Y. Weiss, B. Schölkopf, and J. Platt. Vol. 18. MIT Press, 2005. URL: https://proceedings.neurips.cc/paper/2005/file/4491777b1aa8b5b32c2e8666dbe1a495-Paper.pdf.

[19] H. Bijl, J.-W. van Wingerden, T. B. Schön, and M. Verhaegen. "Online sparse Gaussian process regression using FITC and PITC approximations." In: 48.28 (2015). 17th IFAC Symposium on System Identification SYSID 2015, pp. 703–708. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2015.12.212. URL: https://www.sciencedirect.com/science/article/pii/S2405896315028360.

[20] M. Titsias. "Variational Learning of Inducing Variables in Sparse Gaussian Processes." In: *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*. Ed. by David van Dyk and Max Welling. Vol. 5. Proceedings of Machine Learning Research. Hilton Clearwater Beach Resort, Clearwater Beach,

Florida USA: PMLR, 2009, pp. 567–574. URL: https://proceedings.mlr.press/v5/titsias09a.html.

[21] L. Csató and M. Opper. "Sparse On-Line Gaussian Processes." In: *Neural Computation* 14.3 (Mar. 2002), pp. 641–668. ISSN: 0899-7667. DOI: 10.1162/089976602317250933. eprint: https://direct.mit.edu/neco/article-pdf/14/3/641/815172/089976602317250933.pdf. URL: https://doi.org/10.1162/089976602317250933.

[22] M. Bauer, M. van der Wilk, and C. E. Rasmussen. "Understanding Probabilistic Sparse Gaussian Process Approximations." In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS'16. Barcelona, Spain: Curran Associates Inc., 2016, 1533–1541. ISBN: 9781510838819.

[23] E. Picotti, A. Dalla Libera, R. Carli, and M. Bruschetta. "Continuous-Time Acceleration Modeling through Gaussian Processes for Learning-Based Nonlinear Model Predictive Control." In: *2022 Conference on Control Technology and Applications (CCTA)*. 2022.

[24] G.-L. Tran, D. Milios, P. Michiardi, and M. Filippone. "Sparse within Sparse Gaussian Processes using Neighbor Information." In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 10369–10378. URL: https://proceedings.mlr.press/v139/tran21a.html.

[25] F. Rossi, A. Lendasse, D. François, V. Wertz, and M. Verleysen. "Mutual information for the selection of relevant variables in spectrometric nonlinear modelling." In: *Chemometrics and Intelligent Laboratory Systems* 80.2 (2006). CHIMIOMÉTRIE 2004, pp. 215–226. ISSN: 0169-7439. DOI: https://doi.org/10.1016/j.chemolab.2005.06.010. URL: https://www.sciencedirect.com/science/article/pii/S0169743905000985.

[26] C. E. Shannon. "A mathematical theory of communication." In: *The Bell System Technical Journal* 27.3 (1948), pp. 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[27] A. Kraskov, H. Stögbauer, and P. Grassberger. "Estimating mutual information." In: *Phys. Rev. E* 69 (6 2004), p. 066138. DOI: 10.1103/PhysRevE.69.066138. URL: https://link.aps.org/doi/10.1103/PhysRevE.69.066138.

[28] A. Gepperth and B. Hammer. "Incremental learning algorithms and applications." In: *European Symposium on Artificial Neural Networks (ESANN)*. Bruges, Belgium, 2016. URL: https://hal.archives-ouvertes.fr/hal-01418129.

[29] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger. "Learning-Based Model Predictive Control for Autonomous Racing." In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 3363–3370. DOI: 10.1109/LRA.2019.2926677.

[30] K. P. Wabersich and M. Zeilinger. "Bayesian model predictive control: Efficient model exploration and regret bounds using posterior sampling." In: *Learning for Dynamics and Control*. PMLR. 2020, pp. 455–464.

[31] H. Liu, J. Cai, and Y.-S. Ong. "Remarks on multi-output Gaussian process regression." In: *Knowledge-Based Systems* 144 (2018), pp. 102–121. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2017.12.034. URL: https://www.sciencedirect.com/science/article/pii/S0950705117306123.

[32] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause. "Learning-Based Model Predictive Control for Safe Exploration." In: *2018 IEEE Conference on Decision and Control (CDC)*. 2018, pp. 6059–6066. DOI: 10.1109/CDC.2018.8619572.