Università degli Studi di Padova

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE


Università degli Studi di Padova

Department of Information Engineering

Master Degree in
Control System Engineering


# Nonlinear Model-based Predictive Control for anti-wheelie in high performance motorcycles


*Supervisor:*
Prof. Mattia Bruschetta

*Student:*
Irene Morando
2026724



Academic year 2021/2022

Graduation date 20/10/2022

*"Satisfaction of one's curiosity is one of the greatest sources of happiness in life."*
Linus Pauling

# Abstract

The goal of the thesis is the design of a Nonlinear Model Predictive Controller (NMPC) to avoid wheelie phenomena and maximize performance in high-performance motorcycles.

In order to find the optimal pitch angle to track, a preliminary analysis of maximum performance is carried out. A NMPC controller has been designed based on a simplified motorcycle model, which takes into account the human rider action.

In particular, the control reduces the rider demand to avoid wheelies. The control is studied for limiting the pitch angle via soft constraints and tracking the pitch angle of maximum acceleration determined in the preliminary analysis.

Moreover, in order to favour a more realistic implementation, an actuation delay compensation is performed.

The control law is tested at first in a simulative scenario with a multibody vehicle model, and then in an experimental context, on a real racing motorcycle.

# Abstract

L'obiettivo di questa tesi è lo sviluppo di un controllore predittivo non lineare (NMPC), implementato per evitare fenomeni di impennata e massimizzare le performance di una moto in un contesto racing.

Inizialmente, con lo scopo di trovare l'angolo ottimo per il tracking, è stata svolta un'analisi preliminare sulle massime performance.

Il controllore NMPC è stato realizzato basandosi su un modello semplificato di moto e prendendo in considerazione la richiesta pilota. In particolare, lo schema di controllo agisce riducendo la domanda del rider per evitare impennate.

La legge di controllo è stata studiata per limitare l'angolo di impennata tramite vincoli di tipo soft e facendo il tracking dell'angolo di massima accelerazione ricavato nell'analisi preliminare.

Inoltre, per un'implementazione più realistica del problema in analisi, è stata studiata una compensazione sul delay di attuazione. Infine, il controllore è stato testato inizialmente in un contesto simulativo, su un modello di veicolo multibody, e poi in un contesto sperimentale, su una vera moto da corsa.

# Contents

# List of Figures

# List of Tables

# Acronyms

**CU** Control Unit

**HPIPM** High-Performance Interior-Point-Method

**IMU** Inertial Measurement Unit

**KKT** Karush-Kuhn-Tucker

**LP** Linear Program

**MATMPC** MATLAB Model Predictive Control

**MEX** MATLAB Executable function

**MPC** Model Predictive Control

**NLP** Nonlinear programming

**NMPC** Nonlinear Model Predictive Control

**OCP** Optimal Control Problem

**QP** Quadratic Programming

**RH** Receding Horizon

**RHC** Receding Horizon Control

**RK** Runge-Kutta

**RK4** Runge-Kutta 4th order

**SQP** Sequential quadratic programming

# Introduction

Significant investments have been made in the automotive sector for years, in terms of autonomous driving, security and, more in general, in terms of active control systems. Nowadays, a lot of vehicles are equipped with Adaptive Cruise-Control, Anti-lock Braking System, Anti-Slip Regulation, Lane-Keep Assist or Traction-Control System.

The scope of this work is the design of an anti-wheelie controller for a high-performance motorcycle. In literature, only few controllers have been presented regarding this control field, except for ad hoc solutions. Among these, a fuzzy logic to improve motorcycle stability regulating the torque is described in [1], designed to track the longitudinal slip, the pitch and the cornering angle. Another example, illustrated in [2], is a wheelie detection algorithm, which exploits the estimate of the linear velocities to identify a front tire lifting.

The other active control systems available at the state of the art are patents, such as [3–6], however, these control schemes act only once the front wheel is lifting, causing a strong and late action to solve this phenomenon.

This thesis proposes a control that aims at reducing the rider demand and acts to restrict it, preventing a wheelie. This is possible thanks to a model-based technique. In detail, a Nonlinear Model Predictive Control (NMPC) is chosen, which has the capability of both predicting the system behaviour and handling constraints on the problem at hand. To implement the control law, a motorcycle model that describes the wheelie dynamics is adopted.

In this work, two models are used, a system that fully describes the motorcycle behaviour, and a simplified plant. The first one describes the wheelie dynamics including suspensions kinematics and chain forces, but the variables used to describe the dynamics are not entirely accessible (e.g. suspensions elongation). To cope with this, a simplified model is used for control.

Firstly, a preliminary analysis in simulation is performed in order to find the pitch angle that maximizes the motorcycle acceleration.

In this project, three controllers implementation is presented. They are based on the common idea of limiting the rider demand and bounding the pitch angle variable with soft constraints, while they differ about the cost function minimized by the NMPC.

The first implementation does not consider the pitch angle variable in the cost index,

but only as a soft constraint, leading to a control law that strongly depends on the upper bound value imposed on the wheelie angle. In the second implementation, the pitch angle error with respect to the optimal values is only weighted in the last stage of the cost function, in order to add a stabilizing term at the last point of the objective function. The last implementation is the most general, which minimizes the error on the wheelie angle in the entire prediction horizon, thus solving a pitch angle tracking problem.

On top of that, an input delay is considered into the dynamical model in order to take into account the actuation delay and to compensate it. The tool MATMPC [7] is exploited in the entire project to implement the control law and to test it in simulation. It is an open-source software based on MATLAB, which exploits functions compiled in C code in order to solve nonlinear MPC problems in efficiently way.

Simulations are performed in order to test the effectiveness of the proposed methods, and a sensitivity analysis is performed to evaluate the response to parameters variations.

The first control implementation is tested in experimental scenario on an instrumented motorcycle, where NMPC problem is solved via MATMPC on embedded hardware.

The thesis is organized as follows.

In Chapter 1 the theoretical background on Nonlinear Model Predictive Control is described, analyzing all the phases of the control process. Then, a brief description of MATMPC is presented, the software used to carry out all the simulations.

In Chapter 2 the Euler-Lagrangian method to derive the dynamic model of the motorcycle is reported. After that, the two used motorcycle prototypes are described: a multibody model and a single body plant, which simplifies the first one.

In Chapter 3 a preliminary control is designed, which aims to find the best performance tracking angle. Then, a suitable controller is designed for the problem in analysis. Three different control laws are designed and discussed, leading to the definitive one, which both maximizes performances and minimizes sensitivity to the model parameters. It is implemented to handle a rider request and permits a smooth action to avoid wheeling. On top of that, in this chapter an input delay is modelled in order to make simulations closer to the real system behaviour.

In the first part of the Chapter 4, the angle that guarantees maximum acceleration is found out. Then, the control schemes are tested in a simulative scenario and the related data are discussed. Moreover, to compare the designed controllers, some simulations are performed to understand how they can react in case of models mismatch.

The first designed control law, is validated in an experimental context, which results are reported and discussed in Chapter 5.

Finally, in Chapter 6 the outcome and possible future works are discussed.

# Chapter 1

# Model Predictive Control

In this chapter a brief introduction to MPC technique is carried out [8].

In particular, the MPC strategy is described, and then the Nonlinear Model Predictive Control process is explained, because of the nonlinearities of the used models.

## 1.1 MPC strategy

The Model Predictive Control is an effective way to exploit the system model to compute control laws that are optimal according to a given criterion.

MPC acts as a human who operates naturally when executes control tasks, such as playing chess. Therefore, at each step it works following three main phases:

1. on the basis of model prediction, computes the process output;
2. it evaluates a control sequence that minimises an objective function;
3. it applies only the first element of the control sequence, implementing the receding horizon strategy.

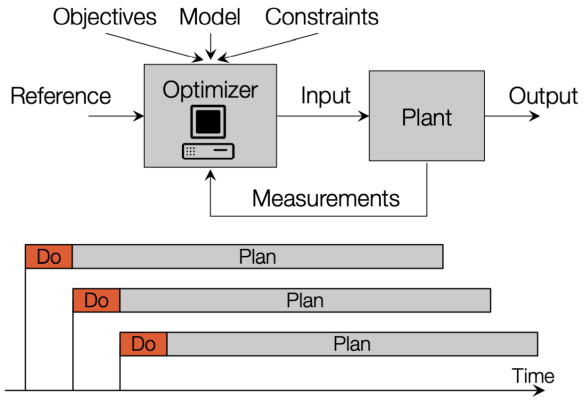


Figure 1.1: Receding Horizon strategy

The MPC advantage is that it can handle constraints on the input, state, and output variables.

The variable constraints are used in order to face real problems with, for example, a physical limitation associated with the application scenarios.

In order to get a control law for the system in analysis, first of all a model that describes its dynamic is needed. The Model Predictive Controller uses this model in order to predict the future behaviour of the system and it applies the best control input for the plant. For this reason a suitable definition of the mathematical model is the basis for the success of this control strategy. The description of the model dynamics plays an important role: the model has to take into account all the possible aspects of the system, in order to make consistent predictions as much as possible near to the real evolution of the process. On the other hand, if the model is too descriptive, the complexity is increased too, therefore the control law could have a high computational cost, that can be processed just by expensive hardware devices.

In Figure 1.1 is reported the block scheme of how a generic MPC control operates. From this picture it can be noticed how the Receding Horizon introduces feedback in the control scheme.

At each sample time, the MPC computes a sequence of control inputs for the actual instant and next sample times in open loop, but just the first control input of this sequence is applied to the plant. At the next sample time the procedure is repeated, but neglecting the other control input values obtained before. The input control sequence evaluated by MPC is the result of the control capability to predict the future system behaviour. The capability of the controller to "look forward" in the future is defined by a parameter called prediction horizon. It is a parameter set in the design control phase and it is chosen as compromise between the computational limits and the improvement of the optimal control input obtained from the controller. In Figure 1.2 the MPC working idea is reported. It can be noticed from the figure how the control strategy solves the optimisation problem using all the entire control sequence calculated at time $k$ to reach the reference signal; but then only the first value of the control sequence is applied to the plant.

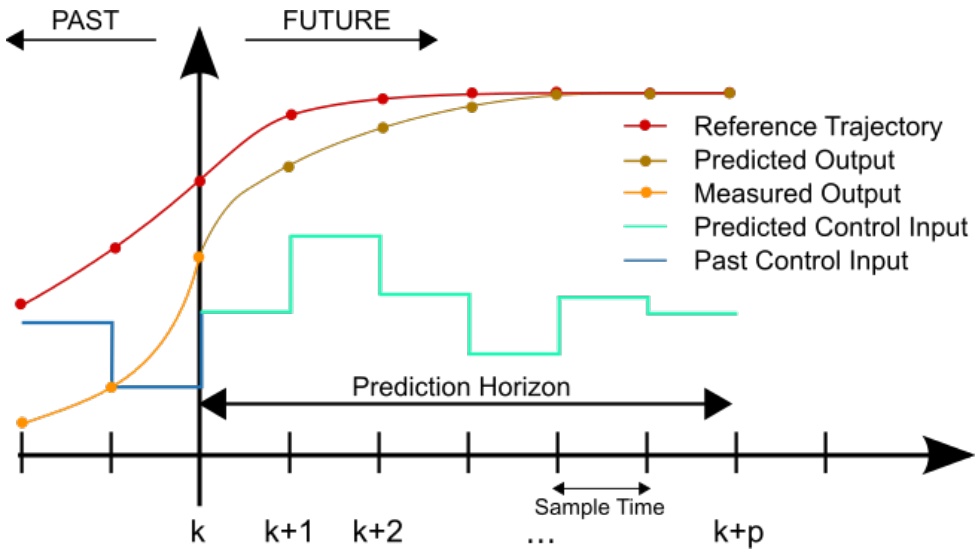


Figure 1.2: MPC working idea

MPC is an implementation of RHC, in which off-line evaluation of the RHC law is

replaced by online determination of its value.

Summarizing, MPC has three main ingredients:

1. an objective function that has to be minimized (e.g., distance of the state from the origin, sum of squared or absolute tracking errors, etc.);
2. an internal system model, that is used to predict the system behaviour;
3. a set of constraints to be satisfied (e.g, on control inputs, controlled outputs, state variables) that can be of different type (e.g., linear, quadratic, etc.).

## 1.2 Nonlinear MPC

When the considered system is a nonlinear one, MPC is called Nonlinear Model Predictive Control. This technique permits to examine the elements in the plant accurately, and hence an improved controller can be derived. On the other hand, a nonlinear description of the system can cause more complexity in terms of time demand.

In this research a Nonlinear Model Predictive Control is implemented, therefore a theoretical background on it is reported in this section.

Working with a real-time implementation, the computational time used to solve the optimal solution is a very sensitive aspect.

In the MPC approach, one of the main steps to be computed is to find the solution of the continuous control problem reported in Eq. 1.1.

$$\min_{\boldsymbol{u}(\cdot)} \boldsymbol{M}(\boldsymbol{x}(t_f)) + \int_{t_0}^{t_f} \boldsymbol{L}(\boldsymbol{x}(t), \boldsymbol{u}(t))dt \tag{1.1}$$

$$\text{such that} \quad \begin{cases} \boldsymbol{x}(t_0) = \hat{\boldsymbol{x}}_0 \\ \dot{\boldsymbol{x}}(t) = F((\boldsymbol{x}(t), \boldsymbol{u}(t)), t \in [t_0, t_f] \\ H((\boldsymbol{x}(t), \boldsymbol{u}(t)) \leq 0, t \in [t_0, t_f] \end{cases} \tag{1.2}$$

Where $\boldsymbol{L}$ represents the cost function, $t_0$ is the initial time (the actual time instant) and $t_f$ is the final time of the prediction horizon. The final stage has a separated treatment, indeed $\boldsymbol{M}$ represents the cost function for the final instant. In Eq. 1.2 the initial condition, the state evolution and the nonlinear limitations are reported.

To solve this problem there are four main approaches, which can find the global or local minimum for discrete or continuous time case. In this work the discrete equation is considered, in which the local minimum will be found, in order to solve the problem fast. Therefore, first of all the system will be discretized, and then the optimal control problem will be solved. The discrete form of this problem is called Nonlinear Programming. This approach has different advantages, such as:

- The flexibility, this means that it is easy to switch from a problem to another.

- The efficiency in terms of computational time requirement.
- The ease of formulation.
- The different types of available solvers.

In order to discretize the Continuous OCP, the input has to be sampled following the Eq. 1.3.

$$\boldsymbol{u}(t) = \boldsymbol{u}_k \quad \text{with} \quad t \in [kt_s, (k+1)t_s] \tag{1.3}$$

Working with nonlinear systems, the discretization phase will be called integration phase. An important point to be discussed is how the model has to be integrated in order to be accurate enough.

According to Eq. 1.4, once the $\dot{\boldsymbol{x}}(t)$ is obtained through approximation, a sample of the state can be evaluated.

$$f(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{x}(t_{k+1}) \quad \text{with} \quad \begin{cases} \dot{\boldsymbol{x}}(t) & = F(\boldsymbol{x}(t), \boldsymbol{u}(t)) \\ \boldsymbol{x}(t_k) & = \boldsymbol{x}_k \end{cases} \tag{1.4}$$

Also the nonlinear constraints have to be sampled, following the Eq. 1.5.

$$\boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \begin{bmatrix} H(\boldsymbol{x}(t_{k,0}, \boldsymbol{u}(t_{k,0})) \\ \vdots \\ H(\boldsymbol{x}(t_{k,n}, \boldsymbol{u}(t_{k,n})) \end{bmatrix} \quad \text{with} \quad t_{k,0}, \cdots t_{k,n} \in [t_k, t_{k+1}] \tag{1.5}$$

The intuition behind this idea is being fast enough to catch all the nonlinearities of the system.

At the end, the NLP problem is obtained, which is reported below.

$$\min_{\boldsymbol{w}} f(\boldsymbol{w}) \quad \text{such that} \quad g(\boldsymbol{w}) = 0, h(\boldsymbol{w}) \leq 0 \tag{1.6}$$

It has the aim of minimising the nonlinear cost function $f$, defined on $\boldsymbol{w}$, which represents the samples on both inputs and states. The relation between input and states is expressed in 1.6 with the equality constraints $g(\boldsymbol{w}) = 0$.

## 1.3 Outline of the entire process

The outline followed by the process in real-time framework is described in this section and a scheme of the different phases is reported in Figure 1.3. Only the steps used for this work are described in details.

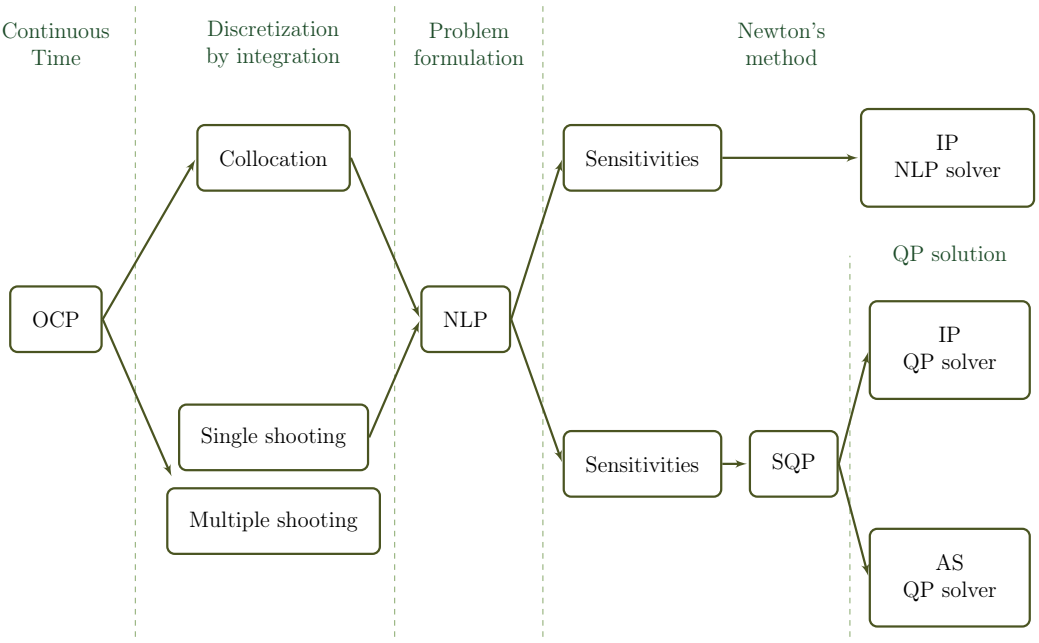


Figure 1.3: Outline of the entire process

### 1.3.1 OCP

The first step is the formulation of the continuous Optimal Control Problem (OCP), which is mentioned above in Eq. 1.1- 1.2. Once the continuous OCP is derived, the next phase is taken into account.

### 1.3.2 Discretization by integration

Looking at Figure 1.3, it can be noticed that the following step is the disctetization of the Optimal Control Problem just obtained.

There are a couple of different approaches: collocation and shooting (single or multiple shooting). Only the multiple shooting will be used, its advantages with respect to the single shooting are presented below.

The shooting strategy comes from the pure integration problem. With the single shooting strategy, an approximation of the state is obtained firstly, then its integration is evaluated. The integration can be obtained for example with the Euler method, which follows a very basic and simple idea, reported in Eq. 1.7:

$$\dot{\boldsymbol{y}} = f(t, \boldsymbol{y}(t)), \quad \boldsymbol{y}(t_0) = \boldsymbol{y}_0 \quad \boldsymbol{y}_{n+1} = \boldsymbol{y}_n + hf(t_n, \boldsymbol{y}_n) \tag{1.7}$$

where $h$ value is the size of every step. One step of the Euler method from $t_n$ to $t_{n+1} = t_n + h$ is expressed by $\boldsymbol{y}_{n+1}$.

The Euler method is not so precise, and at the same time it is too demanding in terms of computational time, hence it is not working for this purpose. Therefore, the multiple shooting will be considered. Looking at previous equation, it can be noticed that $\boldsymbol{u}(\cdot)$ is infinite dimensional, hence a finite-dimensional parametrization is selected, for example

with zero order hold or first order hold. But also the state $\boldsymbol{x}(\cdot)$ is infinite dimensional and an integrator is needed in order to discretize it.

Runge Kutta integrators are used to face this problem, they are a family of integrators used for their accuracy. Their equations are reported below:

$$k_j = F(t_i + c_j, s_i + h\sum_{l=1}^{q} a_{jl}k_l) \qquad\qquad s_{i+1} = s_i + h\sum_{l=1}^{q} b_l k_l \tag{1.8}$$

The aim of Runge Kutta integrators is to approximate the derivative. In case of RK of 4th order, to accomplish the goal, it splits the sample time into two samples, as reported in Figure 1.4.

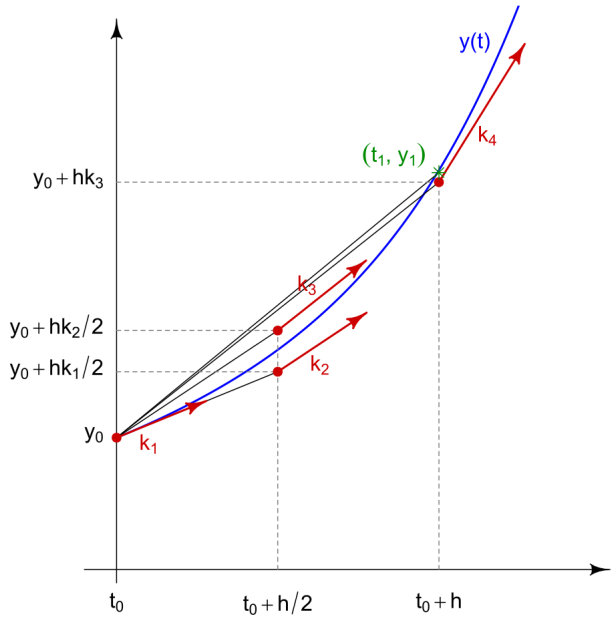


Figure 1.4: Slopes used by the classical Runge-Kutta method

The explicit Runge Kutta 4th order is considered, as Eq. 1.9a-1.9d state.

$$k_1 = F(t_i, s_i) \tag{1.9a}$$
$$k_2 = F(t_i + \frac{1}{2}h, s_i + \frac{1}{2}hk_1) \tag{1.9b}$$
$$k_3 = F(t_i + \frac{1}{2}h, s_i + \frac{1}{2}hk_2) \tag{1.9c}$$
$$k_4 = F(t_i + h, s_i + hk_3) \tag{1.9d}$$

Starting from $t_0$ instant, $k_1$ is easily obtained through approximation (1.9a). The RK4 goes ahead of half of a step and computes two others intermediate derivatives $k_2$ and $k_3$ for $t_0 + \frac{h}{2}$ using respectively $k_1$ and $k_2$, where $h$ is the sampling interval. Then, with $k_3$ RK4 goes ahead of the full step and $k_4$ is obtained.

Then the weighted sum is picked, as Eq. 1.10 suggests. It can be noticed how the two intermediate derivatives are weighted more, because of their accuracy with respect to the others.

$$s_{i+1} = s_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{1.10}$$

### 1.3.3 Problem formulation

The NLP problem is the core of the process, and an efficient way to solve it is needed.

Its formulation is already introduced in Eq. 1.6, in which $\boldsymbol{w}$ is the optimization variable.

First of all, the feasible set has to be defined. It is strongly related to the equality constraints (state update) and inequality constraints (actuator or output limitations).

The huge problem is that the NLP can pass from a feasible domain to an infeasible one in a step, and it can be caused by the actual state and the related constraint, for this reason sometimes the use of soft constraints is preferred.

The aim is to find the local minimum, that to be optimal has to respect two conditions:

1. First order necessary condition : $x^*$ is local optimum, then $\nabla f(x^*) = 0$

2. Second order sufficient condition : $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) > 0$ then $x^*$ is strict local minimum. If $\nabla^2 f(x^*)$ is indefinite, nothing can be deduced.

The first order of KKT is used to evaluate the optimality conditions. In case of equality constraints, the KKT conditions are reported in Eq. 1.11, where $\lambda$ is the typical Lagrangian multiplier.

$$\begin{cases} x^* \text{is regular} \\ g(x^*) = 0 \\ \nabla f(x^*) + \nabla g(x^*)\lambda^* = 0 \end{cases} \tag{1.11}$$

Defining the Lagrangian as $\mathcal{L}(x, \lambda) = f(x) + \lambda^T g(x)$, then the Eq. 1.11 can be seen as $\nabla \mathcal{L}(x^*, \lambda^*) = 0$. Therefore, the problem with equality constraints can be led to a problem without constraints.

The case of inequality constraints is more complicated, and the complementary slackness $\mu_i^* h_i(x^*) = 0$ is needed. The result is reported in Eq. 1.12.

$$\begin{cases} x^* \text{is regular} \\ g(x^*) = 0, \quad h(x^*) \leq 0 \\ \nabla f(x^*) + \nabla g(x^*)\lambda^* + \nabla h(x^*)\mu^* = 0 \\ \mu^* \geq 0, \quad \mu_i^* h_i(x^*) = 0 \quad i = 1, \ldots, n_h \end{cases} \tag{1.12}$$

In this way, the problem of solving NLP is switched to solve the KKT, which is summarized in 1.13. The Lagrangian and its derivatives are providing also a direction for the research of the local minimum. Solving the problem using the Lagrangian has the advantage of evaluate a search direction.

$$\begin{cases} \nabla_x \mathcal{L}(x, \lambda, \mu) = 0 \quad \mu \geq 0 \\ g(x) = 0 \quad h(x) \leq 0 \\ h_i(x)\mu_i = 0 \quad i = 1, \ldots, n_h \\ \text{with} \quad \mathcal{L}(x, \lambda, \mu) = f(x) + \lambda^T g(x) + \mu^T h(x) \end{cases} \tag{1.13}$$

To understand if the stationary point is a local minimum, the second order sufficient condition (Eq. 1.14) is exploited.

$$\begin{cases} x^* \text{is regular satisfying KKT conditions and} \\ d^T \nabla^2_{xx} \mathcal{L}(x, \lambda, \mu) d > 0 \quad \text{with} \quad \forall d \in \mathcal{F}(x^*) \end{cases} \tag{1.14}$$

### 1.3.4 Sensitivities

There are different ways to compute the sensitivities. This step is a simple but very demanding phase in terms of computational time. Using the analytical differentiation is quite expensive in terms of computational effort, but it is accurate. The numerical differentiation is inaccurate, but fast. The algorithmic differentiation has both the advantages: accurateness and fastness. It is performed by a specific toolbox CasADi. The idea is that, using the chain rule, the basic derivative are computed and then they are summed up together online.

### 1.3.5 Newton's method

Next step is to find a solution for NLP just derived in 1.13. Essentially, a Newton problem has to be solved. Therefore the gradient has to be computed, in order to find the minimum in an iterative way. The line search method is used, hence the step amplitude to reach the minimum can be chosen, namely the parameter $\alpha \in ]0, 1]$ in Eq. 1.15.

$$r(x+\Delta x) \approx r(x) + \nabla r(x)^T \Delta x = 0 \quad x^{(k+1)} = x^{(k)} + \alpha \Delta x \tag{1.15}$$

The Newton problem can be solved finding the Newton direction, expressed by $\Delta x$. In case of equality constraints, it can be obtained through Eq. 1.16.

$$\begin{bmatrix} B(x,\lambda) & \nabla g(x) \\ \nabla g(x)^T & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \lambda^+ \end{bmatrix} = - \begin{bmatrix} \nabla f(x) \\ g(x) \end{bmatrix} \tag{1.16}$$

In 1.16 $f(x)$ is the function that has to be minimised, $g(x)$ expresses the equality constraints and $B(x,\lambda)$ is a soft of approximation of the Lagrangian Hessian (due to the computational effort needed to compute the Lagrangian Hessian).

The Newton direction can also be found solving the Quadratic Programming problem at each step, as expressed in Eq. 1.17.

$$\begin{cases} \min_{\Delta x} \quad \frac{1}{2}\Delta x^T B(x,\lambda)\Delta x + \nabla f(x)^T \Delta x \quad \text{such that} \\ \nabla g(x)^T \Delta x + g(x) = 0 \quad \text{given} \quad \lambda^{QP} \quad \text{, where} \quad \lambda^+ = \lambda^{QP} \end{cases} \tag{1.17}$$

The SQP solve the QP problem to find the Newton direction in an iterative way, starting with an initial value for $x$, $\lambda$ and $\mu$. There are different solvers for the QP problem. SQP is the most efficient with respect to the computational time, in particular, the Active Set QP solver is described. AS-QP methods were originally developed as extensions of the simplex method for solving LP problems. The fundamental idea of all active-set methods is to fix a working set, a maximal linearly independent subset of the active constraints, and to solve the resulting equality constrained QP problem. The working set is then updated repeatedly until optimality is reached.

## 1.4 MATMPC

There are different open-source software to solve MPC problems, but not all of them are useful with nonlinear systems in terms of computational time and easy to use. In this project MATMPC is used, a software that solves NMPC problems, which is based on MATLAB and it is developed by Yutao Chen at the University of Padova [7].

Using MATLAB as the base software leads to some advantages. In particular all the algebraic or mathematical routine are implemented, with the relative documentation, and the computational efficiency of the MATLAB Executable functions (MEX), which permit to use C code directly on MATLAB.

In order to use MATMPC, a continuous time model is needed and it is written in CasADi. This model will be discretized through the multiple shooting strategy described before, exploiting the Runge-Kutta method.

At this point, the problem is prepared for a specific solver, in this project it is built

for a condensed QP problem, for the package qpOASES. Then an iterative procedure is executed, in order to obtain an optimal solution for the problem expressed in Eq. 1.18.

$$\begin{cases} \min_{\boldsymbol{x},\boldsymbol{u}} \sum_{k=0}^{N-1} \|\boldsymbol{x}_k - \bar{\boldsymbol{x}}\|^2_{\boldsymbol{Q}} + \|\boldsymbol{u}_k - \bar{\boldsymbol{u}}\|^2_{\boldsymbol{R}} + \|\boldsymbol{x}_N - \bar{\boldsymbol{x}}\|^2_{\boldsymbol{Q}_N} \quad \text{with} \\ g_m(\boldsymbol{u}, \boldsymbol{x}) \leq g(\boldsymbol{x}, \boldsymbol{u}) \leq g_M(\boldsymbol{u}, \boldsymbol{x}), \quad k \geq 0 \end{cases} \tag{1.18}$$

The second line of Eq. 1.18 represents the constraints imposed on the problem that has to be minimised.

A line search algorithm is applied to find a local minimum, then the first order KKT optimality conditions are evaluated in this point: the lower the KKT value is, the closer to the real one the found solution is.

# Chapter 2

# Motorcycle Models

The scope of this work is to design a wheel control for a motorcycle.

As mentioned before, a model-based controller is implemented, therefore a prototype of the real vehicle is needed. In this work, two dynamical models are used. The first (Section 2.2) is more accurate and takes into account all the forces acting on the real motorcycle. On the other hand, its dynamics is described also by non-accessible variables. For this reason a simplified version is introduced (Section 2.3). The second model is used in the control phase, while the first one is used to simulate the behaviour of the motorbike and to test the control action.

## 2.1 The Euler-Lagrangian method

This section is dedicated to the theoretical background useful to understand the derivation of the models.

In general, there are two main approaches to dynamic modelling: the Euler-Lagrangian method and the Newton-Euler method. The first method is based on energy, while the second one is based on the balance between forces and torques acting on the system.

In this case, the Lagrangian function derivation is described, which is the technique used with the aim of obtaining the dynamic model of the motorcycle.

The Lagrangian can be defined as

$$\mathcal{L} = \mathcal{T} - \mathcal{U}, \tag{2.1}$$

where $\mathcal{T}$ represents the kinetic energy, while $\mathcal{U}$ is the potential energy. The Lagrangian equations can be obtained starting from Eq. 2.1, they are of the type:

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{q}}_i} - \frac{\partial \mathcal{L}}{\partial \boldsymbol{q}_i} = \xi_i \quad \text{with} \quad i = 1, \ldots, n \tag{2.2}$$

In Eq. 2.2, $\boldsymbol{q}_i$ are the generalized coordinates that describe the $n$ degrees of freedom of the system in analysis, whereas $\xi_i$ symbolizes the generalized forces associated with the generalized coordinate $\boldsymbol{q}_i$. The contribution of the generalized forces are given by the non-conservative forces, namely the joint actuator torques, the joint friction torques, or the joint torques induced by the contact with the environment.

Considering a system with $n$ rigid bodies, the kinetic and potential energy can be evaluated, in order to be summed up to find the Lagrangian in Eq. 2.1.

### 2.1.1 Kinetic energy

The total kinetic energy is given by the sum of the contributions $\mathcal{T}_i$, related to the motion of each body $i$. It is expressed in Eq. 2.3.

$$\mathcal{T} = \sum_{i=1}^{n} \mathcal{T}_i \tag{2.3}$$

Each kinetic contribution $\mathcal{T}_i$ is split in a translational term and in a rotational term, as shown below. The used symbols $m_i$ and $\dot{\boldsymbol{p}}_i$ are respectively the mass of $i$ and the linear velocity of the interested body, while $\boldsymbol{\omega}_i, \boldsymbol{R}_i$ and $\boldsymbol{I}_i^i$ are the rotational velocity, the rotational matrix and the inertia tensor relative to the centre of mass of the rigid body $i$.

$$\mathcal{T}_i = \underbrace{\frac{1}{2} m_i \dot{\boldsymbol{p}}_i^T \dot{\boldsymbol{p}}_i}_{\text{translations}} + \underbrace{\frac{1}{2} \boldsymbol{\omega}_i^T \boldsymbol{R}_i \boldsymbol{I}_i^i \boldsymbol{R}_i^T \boldsymbol{\omega}_i}_{\text{rotations}} \tag{2.4}$$

To determine the velocities $\dot{\boldsymbol{p}}_i$ and $\boldsymbol{\omega}_i$, the geometric method for the Jacobian computation can be applied to $i$. Therefore, the two quantities can be expressed as

$$\dot{\boldsymbol{p}}_i = \boldsymbol{J}_P^{(i)} \dot{\boldsymbol{q}} \qquad\qquad \boldsymbol{\omega}_i = \boldsymbol{J}_O^{(i)} \dot{\boldsymbol{q}} \tag{2.5}$$

where $\boldsymbol{J}_P^i$ and $\boldsymbol{J}_O^i$ are equal to the geometric Jacobian matrix.

Performing a substitution of 2.5 in Eq. 2.4, a new formulation of kinetic energy of rigid body $i$ (2.4) is obtained:

$$\mathcal{T}_i = \frac{1}{2} m_i \dot{\boldsymbol{q}}^T \boldsymbol{J}_P^{(i)T} \boldsymbol{J}_P^{(i)} \dot{\boldsymbol{q}} + \frac{1}{2} \dot{\boldsymbol{q}}^T \boldsymbol{J}_O^{(i)T} \boldsymbol{R}_i \boldsymbol{I}_i^i \boldsymbol{R}_i^T \boldsymbol{J}_O^{(i)} \dot{\boldsymbol{q}}. \tag{2.6}$$

Therefore, the total kinetic energy can be written as Eq. 2.7 states.

$$\mathcal{T} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij}(\boldsymbol{q}) \dot{q}_i \dot{q}_j = \frac{1}{2} \dot{\boldsymbol{q}}^T \boldsymbol{B}(\boldsymbol{q}) \dot{\boldsymbol{q}} \tag{2.7}$$

Where matrix $\boldsymbol{B}(\boldsymbol{q})$ is the inertia matrix defined as:

$$\boldsymbol{B}(\boldsymbol{q}) = \sum_{i=1}^{n} (m_i \boldsymbol{J}_P^{(i)T} \boldsymbol{J}_P^{(i)} + \boldsymbol{J}_O^{(i)T} \boldsymbol{R}_i \boldsymbol{I}_i^i \boldsymbol{R}_i^T \boldsymbol{J}_O^{(i)}) \tag{2.8}$$

### 2.1.2 Potential energy

As in case of kinetic energy, the potential energy is given by the sum of the contributions relative to each rigid body:

$$\mathcal{U} = \sum_{i=1}^{n} \mathcal{U}_i \tag{2.9}$$

where the contribution of the body $i$ is expressed by Eq. 2.10, in which $\boldsymbol{g}_0$ represents the gravity acceleration vector in the base frame: $\boldsymbol{g}_0 = [0\ 0\ g]$.

$$\mathcal{U}_i = -m_i \boldsymbol{g}_0^T \boldsymbol{p}_i \tag{2.10}$$

### 2.1.3 Lagrangian equations

In this Subsection, the Lagrangian equation in scalar form is derived. Once the Lagrangian in Eq. 2.1 is obtained, its derivative is needed, and hence the following equations are evaluated in order to get the Lagrangian equations.

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) = \sum_{j=1}^{n} b_{ij}(\boldsymbol{q})\ddot{q}_j + \sum_{j=1}^{n} \frac{db_{ij}(\boldsymbol{q})}{dt}\dot{q}_j = \sum_{j=1}^{n} b_{ij}(\boldsymbol{q})\ddot{q}_j + \sum_{j=1}^{n}\sum_{k=1}^{n} \frac{\partial b_{ij}(\boldsymbol{q})}{\partial q_k}\dot{q}_k\dot{q}_j$$

$$\frac{\partial \mathcal{L}}{\partial q_i} = \frac{1}{2}\sum_{j=1}^{n}\sum_{k=1}^{n} \frac{\partial b_{ij}(q)}{\partial q_i}\dot{q}_k\dot{q}_j - \frac{\partial \mathcal{U}}{\partial q_i}$$

In this way, following the Eq. 2.2 and the formulas just described, the Lagrangian equation can be written as:

$$\underbrace{\sum_{j=1}^{n} b_{ij}(\boldsymbol{q})\ddot{q}_j}_{\text{Inertial terms}} + \underbrace{\sum_{j=1}^{n}\sum_{k=1}^{n} h_{ijk}(\boldsymbol{q})\dot{q}_k\dot{q}_j}_{\text{Centrifugal and Coriolis terms}} + \underbrace{g_i(\boldsymbol{q})}_{\text{Gravity terms}} = \xi_i \tag{2.11}$$

The variable $\xi_i$ represents the difference between active forces and dissipative forces. Therefore, the Eq. 2.12 is obtained, where $\boldsymbol{Q}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ represents the generalized force contribution, which can be computed as in Eq. 2.13.

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{Q}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{2.12}$$

In Eq. 2.13 $F_k$ and $T_k$ are the forces and the torques respectively, while $\boldsymbol{p}_k$ and $\boldsymbol{s}_k$ stay for the application points of the forces and the coordinates of application of the torques.

$$\boldsymbol{Q}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \sum_{k=1}^{m} \left(\frac{\partial \boldsymbol{p}_k}{\partial \boldsymbol{q}}\right)^T F_k + \left(\frac{\partial \boldsymbol{s}_k}{\partial \boldsymbol{q}}\right)^T T_k \tag{2.13}$$

In Eq. 2.12, the $n \times n$ matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is not unique, its elements satisfy the Eq. 2.14.

$$\sum_{j=1}^{n} c_{ij}\dot{q}_j = \sum_{j=1}^{n}\sum_{k=1}^{n} h_{ijk}\dot{q}_k\dot{q}_j = \sum_{j=1}^{n}\sum_{k=1}^{n} \frac{1}{2}\left(\frac{\partial b_{ij}}{\partial q_k} + \frac{\partial b_{ik}}{\partial q_j} - \frac{\partial b_{jk}}{\partial q_i}\right)\dot{q}_k\dot{q}_j \tag{2.14}$$

A generic element of $\boldsymbol{C}$ is $c_{ij} = \sum_{k=1}^{n} c_{ijk}\dot{q}_k$, and it is called Christoffel symbol of the first type.

The Christoffel matrix is characterized by two main properties:

- $\boldsymbol{N}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \dot{\boldsymbol{B}}(\boldsymbol{q}) - 2\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is skew symmetric, this means that $\boldsymbol{\omega}^T\boldsymbol{N}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{\omega} = 0 \quad \forall \boldsymbol{\omega}$
- $\dot{\boldsymbol{q}}^T\boldsymbol{N}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = 0$ for any choice of matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$.

## 2.2 Multibody model

The complex model is derived using the Lagrangian formulation described in the previous Section. In particular, the motorcycle model shown in Figure 2.1 is formed by seven rigid bodies with six joints. The mentioned rigid bodies are the rear wheel, the swingarm, the main frame, the front upper fork, the front lower fork, the front wheel and the rider. Each body is considered to have a centre of mass and a known inertia along y-axis. Rider, main frame and upper front fork are considered as in-built separated bodies, because they do not have any joint between them.

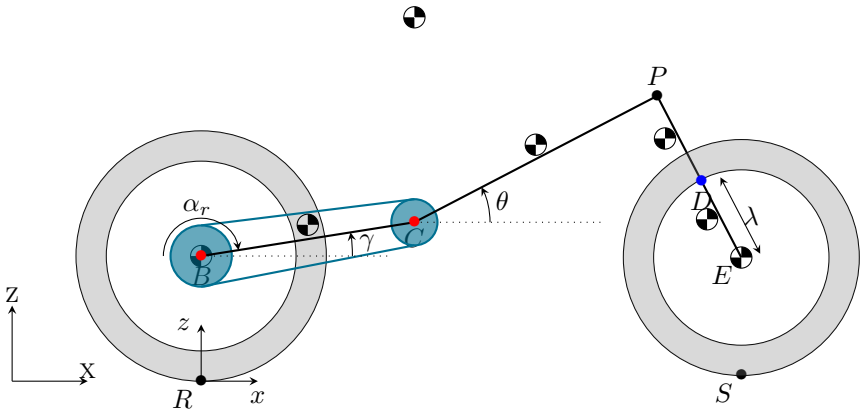


Figure 2.1: Multibody model

This model describes the motorcycle longitudinal and vertical motion (together with rotation about y-axis), which is characterized by six degrees of freedom, that are:

1. Swingarm angle $\gamma$: angle between the swingarm (segment $\overline{BC}$) and the global x-axis. This makes the joint B a revolute joint.
2. Main frame angle $\theta$: angle between main frame and global x-axis. The relative motion between $\gamma$ and $\theta$ is given thanks to rear suspension, which acts between these two rigid bodies. This generates a revolute joint on C.

3. Front suspension travel $\lambda$: front suspension elongation, direct acts on segment DE. This makes the joint D a prismatic joint.

4. Global rear contact horizontal position $x_r$, which is highlighted in the Figure 2.1 with $R$.

5. Global rear contact vertical position $z_r$.

6. Rear wheel rotation angle $\alpha_r$.

Therefore, the model is derived through generalized coordinates $\boldsymbol{q}$ and the generalized forces applied on them. The generalized coordinates are

$$\boldsymbol{q} = \left[\gamma, \theta, \lambda, z_r, x_r, \alpha_r, \dot{\gamma}, \dot{\theta}, \dot{\lambda}, \dot{z}_r, \dot{x}_r, \dot{\alpha}_r\right]^T$$

### 2.2.1 Kinematics

This motorcycle model is composed by seven bodies. To describe them, their centre of mass and the position of the joints are considered. For each one, the rotational matrices to switch from the world frame to the local coordinate system have to be taken into account. All the rotations are made around y-axis, and they are positive if motorcycle pitches down. Since generalized coordinates are positive in the other direction, the rotation matrices are all transposed.

In particular, the following rotational matrices are useful:

$$\boldsymbol{R}_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \quad \boldsymbol{R}_y(\theta_0) = \begin{bmatrix} \cos(\theta_0) & 0 & -\sin(\theta_0) \\ 0 & 1 & 0 \\ \sin(\theta_0) & 0 & \cos(\theta_0) \end{bmatrix} \tag{2.15}$$

$$\boldsymbol{R}_y(\gamma) = \begin{bmatrix} \cos(\gamma) & 0 & -\sin(\gamma) \\ 0 & 1 & 0 \\ \sin(\gamma) & 0 & \cos(\gamma) \end{bmatrix} \quad \boldsymbol{R}_y(\gamma_0) = \begin{bmatrix} \cos(\gamma_0) & 0 & -\sin(\gamma_0) \\ 0 & 1 & 0 \\ \sin(\gamma_0) & 0 & \cos(\gamma_0) \end{bmatrix} \tag{2.16}$$

$$\boldsymbol{R}_y(\nu) = \begin{bmatrix} \cos(\nu) & 0 & -\sin(\nu) \\ 0 & 1 & 0 \\ \sin(\nu) & 0 & \cos(\nu) \end{bmatrix} \tag{2.17}$$

The angles $\theta_0$ and $\gamma_0$ are the initial pitch angle and the swingarm initial angle respectively. The symbol $\nu$ is the angle of front suspension with respect to the vertical line, supposing orthogonal with respect to the frame. Hence, it is $\nu = \theta - \theta_0 + \epsilon$, with $\epsilon$ as the caster angle.

The joints positions are reported below, expressed by $\boldsymbol{O}_i$.

$$\boldsymbol{O}_W = [0\ 0\ 0]^T \tag{2.18}$$
$$\boldsymbol{O}_0 = \boldsymbol{O}_W + [x_r\ 0\ z_r]^T \tag{2.19}$$
$$\boldsymbol{O}_1 = \boldsymbol{O}_0 + [0\ 0\ rr]^T \tag{2.20}$$
$$\boldsymbol{O}_2 = \boldsymbol{O}_1 + \boldsymbol{R}_y(\gamma)\,[l_1\ 0\ 0]^T \tag{2.21}$$
$$\boldsymbol{O}_3 = \boldsymbol{O}_2 + \boldsymbol{R}_y(\theta)\,[l_2\ 0\ 0]^T \tag{2.22}$$
$$\boldsymbol{O}_4 = \boldsymbol{O}_3 + \boldsymbol{R}_y(\nu)\,[\sigma\ 0\ -l3]^T \tag{2.23}$$
$$\boldsymbol{O}_5 = \boldsymbol{O}_4 + \boldsymbol{R}_y(\nu)\,[\sigma\ 0\ -d]^T \tag{2.24}$$
$$\boldsymbol{O}_6 = \boldsymbol{O}_5 + [0\ 0\ -r_f]^T \tag{2.25}$$

The Eq. 2.18 symbolizes the position of the word frame, while $\boldsymbol{O}_0, \boldsymbol{O}_1, \boldsymbol{O}_2, \boldsymbol{O}_3, \boldsymbol{O}_4, \boldsymbol{O}_5$ stay for the pose of the rear contact point of the motorcycle, the joint between the rear wheel and the swingarm, the joint between the swingarm and the main frame, the joint between the main frame and the front upper fork and the joint between the front upper and lower fork. The frontal wheel contact point is formalized with $\boldsymbol{O}_6$.

The variable $\sigma$ represents the offset with respect to front suspension and wheel. The parameters $l_i$, with $i$ the number of the considered body, represent the length of the links. The values $\boldsymbol{p}_i$ represent the position of the centres of mass. The point application for the drag force is expressed by

$$\boldsymbol{O}_{drag} = \boldsymbol{O}_2 + \boldsymbol{R}_{\theta_0}\boldsymbol{R}_{\theta_0}\boldsymbol{p}_{drag}.$$

The kinematic analysis includes also the definition of the centre of mass of each body. They are represented with $\boldsymbol{O}_{m_i}$, with $i$ the number of the considered body. In details, they are

$$\boldsymbol{O}_{m_1} = \boldsymbol{O}_2 + \boldsymbol{R}_y(\gamma)\boldsymbol{p}_1 \tag{2.26}$$
$$\boldsymbol{O}_{m_2} = \boldsymbol{O}_2 + \boldsymbol{R}_y(\theta)^T\boldsymbol{R}_y(\theta)\boldsymbol{p}_2 \tag{2.27}$$
$$\boldsymbol{O}_{m_3} = \boldsymbol{O}_3 + \boldsymbol{R}_y(\nu)\boldsymbol{p}_3 \tag{2.28}$$
$$\boldsymbol{O}_{m_4} = \boldsymbol{O}_3 + \boldsymbol{R}_y(\nu)([0\ 0\ l_4]^T - [0\ 0\ d]^T + \boldsymbol{p}_4) \tag{2.29}$$
$$\boldsymbol{O}_{m_5} = \boldsymbol{O}_2 + \boldsymbol{R}_y(\theta_0)^T\boldsymbol{R}_y(\theta_0)\boldsymbol{p}_5 \tag{2.30}$$
$$\boldsymbol{O}_{m_6} = \boldsymbol{O}_5 \tag{2.31}$$
$$\boldsymbol{O}_{m_7} = \boldsymbol{O}_1 \tag{2.32}$$

The previous equations, in order, represent the centres of mass of the swingarm, the main frame, the front upper fork, the front lower fork, the rider, the front and rear wheel.

In order to evaluate the inertia matrix $\boldsymbol{B}(\boldsymbol{q})$, the derivatives of the positions $\boldsymbol{p}_i$ are computed following the Eq. 2.33.

$$\dot{\boldsymbol{p}}_i = \frac{\partial \boldsymbol{O}_{m_i}}{\partial \boldsymbol{q}} \tag{2.33}$$

Moreover, the angular velocities $\boldsymbol{\omega}_i$ are calculated as in Eq. 2.34-2.35 .

$$\boldsymbol{\omega}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \boldsymbol{\omega}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \boldsymbol{\omega}_7 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.34}$$

$$\boldsymbol{\omega}_3 = \boldsymbol{\omega}_4 = \boldsymbol{\omega}_5 = \boldsymbol{\omega}_6 = \boldsymbol{\omega}_2 \tag{2.35}$$

In this way the inertia matrix $\boldsymbol{B}(\boldsymbol{q})$ can be obtained following the Eq. 2.8. Therefore,

$$\boldsymbol{B}(\boldsymbol{q}) = \sum_{i=1}^{7} m_i \dot{\boldsymbol{p}}_i^T \dot{\boldsymbol{p}}_i + \boldsymbol{\omega}_i^T \begin{bmatrix} 0 & 0 & 0 \\ 0 & I_i & 0 \\ 0 & 0 & 0 \end{bmatrix} \boldsymbol{\omega}_i$$

The symbol $I_i$ is the inertia of the bodies, while $m_i$ represents the mass of the $i^{th}$ rigid body.

Following the Eq. 2.14, the Christoffel matrix $\boldsymbol{C}$ can be derived.

### 2.2.2 Dynamics

On the vehicle in analysis different forces are applied, some of them are internal forces, while others are external. In particular, the considered forces are:

- Rear suspension (internal): acts as a torque between swingarm and main frame, modelled as a spring-damper.
- Front suspension (internal): acts as a force between upper front fork and front wheel centre, modelled as a spring-damper.
- Chain force (internal): acts as a force plus torque between swingarm and main frame. Pinion torque obtained from chain force is reacting on main frame. Chain forces also make rear wheel rotate.
- Normal wheel forces (external): act as forces on contact point, modelled as a saturated spring-damper.
- Rear wheel longitudinal force (external): acts as a longitudinal force on rear contact point. It also decelerates the rear wheel.
- Drag forces (external): acts as longitudinal and vertical forces which are opposed to vehicle motion, therefore, also these forces decelerate the motorcycle.

### Rear suspension force

The rear suspension scheme is shown in Figure 2.2, its kinematics can be evaluated by geometrical relations, knowing swingarm and main frame angles, namely $\gamma$ and $\theta$. It is important to notice that the points D and H in figure are in-built to main frame, F is integral to swingarm, while the points E and G are moving with respect to both swingarm and main frame.

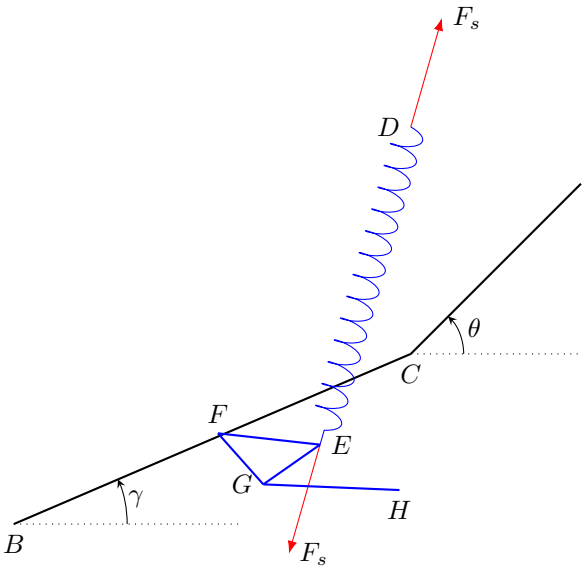


Figure 2.2: Rear Suspension

The segments $\overline{EF}, \overline{EG}, \overline{GF}, \overline{GH}$ are fixed, while $\overline{DE}$ is depending on spring compression, hence it is varying. The nomenclature $l_1(\gamma, \theta) = \overline{DE}$ is used to express the explicit dependence on generalized variables. The notation $(\cdot)^S$ or $(\cdot)^M$ is exploited to reference the mentioned vector to swingarm or main frame. Experimental values of suspension coordinates are given in a design condition, where the rear spring is in neutral position, namely with $\gamma_0$, $\theta_0$. Therefore, considering all the previous elements, some points can be defined, all expressed in a frame that is solidal to the vertical plane of the vehicle. They are reported in Table 2.1.

| Symbol | Definition | Meaning |
|---|---|---|
| $\boldsymbol{p}_1(\gamma)$ | $\boldsymbol{R}_y(\gamma)\boldsymbol{p}_{1o}$ | link pivot point |
| $\boldsymbol{p}_2(\theta)$ | $\boldsymbol{R}_y(\theta)\boldsymbol{R}_y^T(\theta_0)\boldsymbol{p}_{2o}$ | spring bottom point |
| $\boldsymbol{p}_3(\theta)$ | $\boldsymbol{R}_y(\theta)\boldsymbol{R}_y^T(\theta_0)\boldsymbol{p}_{3o}$ | rod bottom point |
| $\boldsymbol{p}_4(\gamma, \theta)$ | | rod top point |
| $\boldsymbol{p}_5(\gamma, \theta)$ | | spring top point |

Table 2.1: Rear suspension points

The distances between the just mentioned points are those reported in Table 2.2.

The position of points $\boldsymbol{p}_4(\gamma, \theta)$ and $\boldsymbol{p}_5(\gamma, \theta)$ are unknown, and they have to be determined. Knowing the swingarm and main frame angles, $\gamma$ and $\theta$, it is possible to find the

| Symbol | Definition | Meaning |
|---|---|---|
| $l_1(\gamma,\theta)$ | $\|\boldsymbol{p}_2(\gamma,\theta)-\boldsymbol{p}_5(\gamma,\theta)\|$ | spring length |
| $l_2$ | $\|\boldsymbol{p}_1(\gamma_0)-\boldsymbol{p}_{4o}\|$ | link length |
| $l_3$ | $\|\boldsymbol{p}_1(\gamma_0)-\boldsymbol{p}_{5o}\|$ | first moving link |
| $l_4$ | $\|\boldsymbol{p}_{5o}-\boldsymbol{p}_{4o}\|$ | second moving link |
| $l_5$ | $\|\boldsymbol{p}_{3o}-\boldsymbol{p}_{5o}\|$ | rod length |

Table 2.2: Rear suspension distances

position $\boldsymbol{p}_4(\gamma,\theta)$ as the intersection of two circumferences centred at $\boldsymbol{p}_1(\gamma)$ and $\boldsymbol{p}_3(\theta)$ with radius respectively of $l_3$ and $l_5$. The formula for circumference intersection is given by

$$l_s(\gamma,\theta) = \|\boldsymbol{p}_1(\gamma)-\boldsymbol{p}_3(\theta)\| \tag{2.36}$$

$$l_c(\gamma,\theta) = \frac{l_3^2-l_5^2+l_s(\gamma,\theta)^2}{2l_s(\gamma,\theta)} \tag{2.37}$$

$$l_h(\gamma,\theta) = \sqrt{l_5^2-l_s(\gamma,\theta)^2} \tag{2.38}$$

$$\boldsymbol{p}_4(\gamma,\theta) = \frac{l_s(\gamma,\theta)}{l_c(\gamma,\theta)}\left[\boldsymbol{p}_1(\gamma)-\boldsymbol{p}_3(\theta)\right] \pm \frac{l_h(\gamma,\theta)}{l_c(\gamma,\theta)}\left[\boldsymbol{p}_1(\gamma)-\boldsymbol{p}_3(\theta)\right]+\boldsymbol{p}_3(\theta) \tag{2.39}$$

Once $\boldsymbol{p}_4(\gamma,\theta)$ is evaluated, it is possible to get also $\boldsymbol{p}_5(\gamma,\theta)$, following the same reasoning.

$$l_s(\gamma,\theta) = \|\boldsymbol{p}_4(\gamma)-\boldsymbol{p}_3(\theta)\| \tag{2.40}$$

$$l_c(\gamma,\theta) = \frac{l_4^2-l_5^2+l_s(\gamma,\theta)^2}{2l_s(\gamma,\theta)} \tag{2.41}$$

$$l_h(\gamma,\theta) = \sqrt{l_5^2-l_s(\gamma,\theta)^2} \tag{2.42}$$

$$\boldsymbol{p}_5(\gamma,\theta) = \frac{l_s(\gamma,\theta)}{l_c(\gamma,\theta)}\left[\boldsymbol{p}_4(\gamma)-\boldsymbol{p}_3(\theta)\right] \pm \frac{l_h(\gamma,\theta)}{l_c(\gamma,\theta)}\left[\boldsymbol{p}_4(\gamma)-\boldsymbol{p}_3(\theta)\right]+\boldsymbol{p}_3(\theta) \tag{2.43}$$

At this point, it is possible to define the spring length as the quantity $l_1(\gamma,\theta) = \|\boldsymbol{p}_2(\gamma,\theta)-\boldsymbol{p}_5(\gamma,\theta)\|$, which is only function of $\gamma$ and $\theta$.

In order to compute the spring-damper force that describes the suspension behaviour, the spring velocity is needed, and it is found as

$$\dot{l}_1(\gamma,\theta,\dot{\gamma},\dot{\theta}) = \frac{\partial l_1}{\partial\gamma}(\gamma,\theta)\dot{\gamma}+\frac{\partial l_1}{\partial\theta}(\gamma,\theta)\dot{\theta} \tag{2.44}$$

Therefore, the force caused by spring-damper action is reported in 2.45.

$$F_s(\gamma,\theta,\dot{\gamma},\dot{\theta}) = k\left[l_1(\gamma,\theta)-l_{1o}\right]+c\dot{l}_1(\gamma,\theta,\dot{\gamma},\dot{\theta}) \tag{2.45}$$

The suspension force generates a set of applied forces to swingarm and main frame,

because it includes forces on links and rod. A simple way to derive the force contribution is to find the generalized momentum generated by the force to the swingarm and main frame angles, by using jacobians.

In this way, the generated torques are equal to

$$\boldsymbol{Q}_{M_\gamma}(\gamma,\theta,\dot{\gamma},\dot{\theta}) = \frac{\partial l_1}{\partial \gamma}(\gamma,\theta)F_s(\gamma,\theta,\dot{\gamma},\dot{\theta}) \tag{2.46}$$

$$\boldsymbol{Q}_{M_\theta}(\gamma,\theta,\dot{\gamma},\dot{\theta}) = \frac{\partial l_1}{\partial \theta}(\gamma,\theta)F_s(\gamma,\theta,\dot{\gamma},\dot{\theta}) \tag{2.47}$$

**Front suspension force**

Front suspension force acts as a force between upper front fork and front wheel centre. As all the suspension forces, it is modelled as a spring-damper system, which is reported in Eq. 2.48, depending on generalized variables $\lambda, \dot{\lambda}$. The symbols $k_f$ and $c_f$ represent respectively the stiffness coefficient of the spring and the damping coefficient of the front suspension. $l_4$ is the front lower fork length.

$$F_{fs}(\lambda,\dot{\lambda}) = k_f(\lambda - l_4) + c_f\dot{\lambda} \tag{2.48}$$

The interested points for this force are D and E. The point D is the intersection point between the front upper fork and the front lower fork, while E is the joint between the front lower fork and the front wheel. They can be defined as

$$D = P + \boldsymbol{R}_y(\nu)\begin{bmatrix}\sigma & 0 & -l_3\end{bmatrix}^T \tag{2.49}$$

$$E(\lambda) = D + \boldsymbol{R}_y(\nu)\begin{bmatrix}\sigma & 0 & -\lambda\end{bmatrix}^T \tag{2.50}$$

where $\nu$ and $\sigma$ are the variables defined before, while $l_3$ is the front upper fork length. The variable $P$ is the intersection point of the front upper fork and the main frame.

Also in this case, to evaluate the generalized forces arising from the front suspension force, the jacobians are used:

$$\boldsymbol{Q}_{M_{fs}} = \frac{\partial E}{\partial \boldsymbol{q}}\boldsymbol{R}_y^T(\nu)F_{fs} - \frac{\partial D}{\partial \boldsymbol{q}}\boldsymbol{R}_y^T(\nu)F_{fs} \tag{2.51}$$

**Chain force**

The chain force derivation is the most complex part of this model, it is based on the relative slope between the upper segment of the chain and the swingarm. Considering $\alpha$ as the relative inclination angle, it can be seen from Figure 2.3 that $\alpha = \alpha_1 + \alpha_2 - \pi/2$.

This relation is due to the orthogonality of the chain segment with respect to the point A in Figure 2.3. With the aim of evaluate angle $\alpha$, the computation of $\alpha_1$ and $\alpha_2$

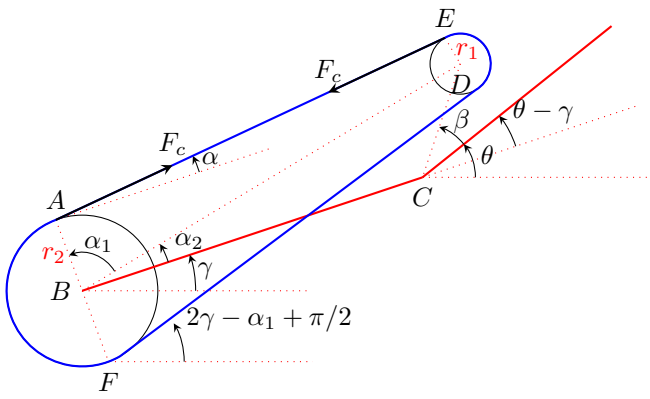


Figure 2.3: Chain description

is needed. To face this problem, it is important to consider where the points B, C and D are attached. In particular, B moves with the rear wheel, but is independent of the swingarm and main frame orientation; point C coincides with the swingarm pivot joint and depends on the swingarm angle, while D is attached to the main frame and then depends on both swingarm and main frame angle. The vehicle parameters and distances are given in an initial condition frame, therefore, relative position and orientation with respect to the initial frame has to be taken into account. In particular, some points can be written with explicit dependencies, such as $B = B(\gamma)$ and $D = D(\gamma, \theta, \theta_0)$. Since the whole chain mechanism is solidal to the main vehicle frame, all components have the same yaw and roll angles. Following this idea, the chain reference frame can be set in the swingarm pivot point, oriented as the world frame. Hence, points B, C and D are set as follows:

$$C = [0 \quad 0 \quad 0]^T \tag{2.52}$$

$$B = \boldsymbol{R}_y(\gamma)[-l_1 \quad 0 \quad 0]^T, \text{ with } l_1 \text{ the swingarm lenght} \tag{2.53}$$

$$D = C + \boldsymbol{R}_y(\theta)\boldsymbol{R}_y(\theta_0)^T D_0 \text{ with } D_0 \text{ the initial position of the pinion centre} \tag{2.54}$$

The angle $\alpha_1$ is computed considering the trapezoid ABDE, following Eq. 2.55.

$$\alpha_1 = \frac{r_1 - r_2}{\overline{BD}} \tag{2.55}$$

Whereas the angle $\alpha_2$ is evaluated considering the triangle BCD and the projection of D into the BC line, as expressed in Eq. 2.56.

$$\alpha_2 = atan\left(\frac{\sin(\beta + \theta - \gamma - \theta_0)}{\overline{BC} + \overline{CD}\cos(\beta + \theta - \gamma - \theta_0)}\right) \tag{2.56}$$

Once the relative orientation $\alpha$ is known, it is possible to obtain the absolute orientation of the force as $\eta_{up} = \alpha + \gamma$. The variable $\eta_{up}$ is the absolute angle of the upper chain segment. At this point, all the involved angles are known, hence it is necessary to model the force generation: the chain is considered as a spring-damper mechanism that acts only in extension.

The amount of extension can be computed by considering the nominal segment length and the wrapping around crown and pinion as

$$\delta l_{up} = \overline{AE} - l_{up_0} + r_1\alpha_r - r_2\alpha_p \tag{2.57}$$

Then, the upper chain force can be obtained as reported in Eq. 2.58.

$$F_{ch_{up}} = k_{ch}l_{up} + c_{ch}l_{up} \tag{2.58}$$

This force is applied on the pinion and the crown, but not directly on the swingarm and main frame. Adding two moments resulting from the force translation, the chain force can be considered as applied to the centre of pinion and crown. The moment acting on the crown is responsible for the rear wheel rotation, indeed it is the one that transfer the actual torque. The moment in pinion is the reaction torque that the main frame receive from the chain and tends to rotate the main frame counter-clockwise.

The generalized contribution of the chain force is computed as

$$\boldsymbol{Q}_{M_{ch_{up}}} = \frac{\partial B}{\partial q}\boldsymbol{R}_y^T(\eta_{up})F_{ch_{up}} - \frac{\partial D}{\partial q}\boldsymbol{R}_y^T(\eta_{up})F_{ch_{up}} \tag{2.59}$$

Then the just described generalized torques generated by the chain force are applied to the rear wheel as $M_{\alpha_r} = F_{ch}r_1$, while the control torque is applied to the pinion.

The force generated by the lower part of the chain is not considered. This is not a lack in the modelling because in this work the motorcycle acceleration is taken into account. The braking situation is neglecting, which corresponds to the situation in which the lower area of the chain generate force cased by the extension in this part.

### Normal wheel force

The front normal force affects the point G in Figure 2.1, namely the front wheel contact point. It can be defined as $G = E + [0 \quad 0 \quad -r_f]^T$, where $r_f$ is the front wheel radius and $E$ is the centre of the front wheel (Figure 2.1). The $z$ component of the point $G$ is considered as $z_G$, while its derivative is $\dot{z}_G$. Therefore, on the front wheel contact point acts a force $F_{nf}$, which is modelled as a saturated spring-damper force expressed in Eq. 2.60.

$$F_{fn} = \frac{1}{2}\left(-k_2 z_G - c_2\dot{z}_G + \sqrt{(-k_2 z_G - c_2\dot{z}_G)^2}\right) \tag{2.60}$$

It can be noticed how the force $F_{fn}$ contains the spring and the damper effects, and a third part. The symbols $k_2$ and $c_2$ are the stiffness and damping coefficients. The square root term is used in order to consider both the case in which the front wheel is on the

surface and when the wheel is lift. In particular, in the case in which the motorcycle is performing a wheelie, the front normal force is equal to 0 N.

Also the generalized forces made by the normal wheel force are evaluated thanks to the jacobians:

$$\boldsymbol{Q}_{M_{fn}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \frac{\partial G}{\partial \boldsymbol{q}} F_{fn} \tag{2.61}$$

The point $R$ in Figure 2.1 is the rear wheel contact point, namely:

$$R(x_r, z_r) = [x_r \quad 0 \quad z_r]^T$$

On this point acts a normal force reported in 2.62, as $F_{fn}$ just described.

$$F_{rn} = \frac{1}{2}\left(-k_1 z_R - c_1 \dot{z}_R + \sqrt{(-k_1 z_R - c_1 \dot{z}_R)^2}\right) \tag{2.62}$$

Its effects are reported in Eq. 2.63, they are taken into account also in the following section, with those caused by the rear wheel longitudinal force.

$$\boldsymbol{Q}_{M_{rn}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \frac{\partial R}{\partial \boldsymbol{q}} F_{rn} \tag{2.63}$$

**Rear wheel longitudinal force**

On $R$ point acts both rear suspension force $F_{nr}$ and thrust force $F_{x_r}$.

$F_{rn}$ force is described in the previous section. In order to define $F_{x_r}$ force, the evaluation of the longitudinal slip is needed, which is reported in Eq. 2.64.

$$S_{long} = \frac{\dot{x}_r - r_{eff}\dot{\alpha}_r}{\dot{x}_r} \tag{2.64}$$

where $\dot{x}_r$ is the longitudinal speed of the wheel centre, $\dot{\alpha}_r$ is the angular speed of the tire and $r_{eff}$ is the effective tire radius. The effective radius is defined to be the radius of the tire when rolling with no external torque applied about the spin axis.

Figure 2.4 shows the general shape of the force versus slip curve generated from the called "Magic Formula" tire model [9].

The general form of the Magic Formula, given by Pacejka, is:

$$F_{x_r} = Dx \sin\{Cx \arctan[Bx - Ex(Bx - \arctan(Bx))]\} \tag{2.65}$$

where B, C, D and E represent fitting constants and $F_{x_r}$ is a force resulting from a slip parameter $x$.

The generalized force produced by rear wheel longitudinal force is expressed as:

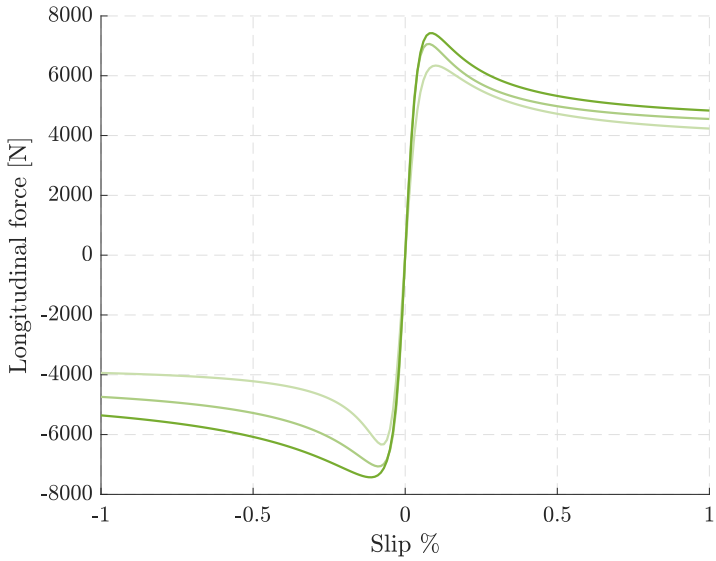


Figure 2.4: Magic formula curve

$$\boldsymbol{Q}_{M_{rl}}(x_r, z_r) = \frac{\partial R}{\partial \boldsymbol{q}} F_{x_r} \tag{2.66}$$

Summed to the generalized force arose thanks to the rear wheel longitudinal force there is the following torque: $\boldsymbol{Q}_{M_{rw}} = T_w - F_{xr} r_{eff}$, where $T_w$ is the torque applied on the wheel, while $r_{eff}$ is the effective position vector, defined as $r_{eff} = rr + z_r$, with $rr$ the radius of rear wheel.

### Drag forces

Looking at the drag forces, it can be noticed that they act on pitch angle and generate a momentum caused by their effect on both x and z coordinate of the motorcycle. These forces are reported in Eq. 2.67a-2.67c, in which $\rho$ represent the air density, A is the front area of the motorcycle subject to the forces. $c_{drag}$, $c_{lift}$ and $c_\theta$ are the drag coefficients.

$$F_{drag_x}(\dot{x}_r) = \frac{1}{2}\rho A c_{drag} \dot{x}_r^2 \tag{2.67a}$$

$$F_{drag_z}(\dot{x}_r) = \frac{1}{2}\rho A c_{lift} \dot{x}_r^2 \tag{2.67b}$$

$$F_{drag_\theta}(\dot{x}_r) = \frac{1}{2}\rho A c_\theta \dot{x}_r^2 \tag{2.67c}$$

The drag forces arise generalized forces, which are computed exploiting jacobians, as before. In particular, they are:

$$\boldsymbol{Q}_{M_{drag}} = \frac{\partial O_{drag}}{\partial \boldsymbol{q}} F_{drag_z} - \frac{\partial O_{drag}}{\partial \boldsymbol{q}} F_{drag_x} + F_{drag_\theta} \tag{2.68}$$

Of particular importance is the $\dot{x}_r^2$ dependence, which means that the fluid drag in-

creases with the square of flow velocity.

Summing up all the elements presented in this Section, the dynamics of the generalized coordinates can be derived.

The generalized forces include all the forces described above, hence:

$$\boldsymbol{Q}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{Q}_{M_\gamma} + \boldsymbol{Q}_{M_\theta} + \boldsymbol{Q}_{M_{fs}} + \boldsymbol{Q}_{M_{chup}} + \boldsymbol{Q}_{M_{\alpha_r}} + \boldsymbol{Q}_{M_{fn}} + \boldsymbol{Q}_{M_{rl}} + \boldsymbol{Q}_{M_{rw}} + \boldsymbol{Q}_{M_{drag}} \quad (2.69)$$

Once that the matrices are defined, following the Lagrangian equations, the equations of motion can be evaluated as

$$\ddot{\boldsymbol{q}} = \boldsymbol{B}(\boldsymbol{q})^{-1}(\boldsymbol{Q}(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q})); \quad (2.70)$$

In Eq. 2.70, $\boldsymbol{g}(\boldsymbol{q})$ represents the Jacobian of the potential energy $\mathcal{U}$

$$\mathcal{U} = \sum_{i=1}^{n} -m_i \boldsymbol{g}^T \boldsymbol{O}_{m_i},$$

where $m_i$ are the mass of the bodies, $\boldsymbol{O}_{m_i}$ the centres of mass and $\boldsymbol{g} = [0\ 0\ -9.81]$ is the gravitational acceleration. The matrix $\boldsymbol{B}$ represents the inertia matrix, $\boldsymbol{C}$ is the Christoffel matrix, and they are evaluated as written above.

## 2.3 Single body model

The motorcycle is equipped with sensors, but they can not measure all the generalized coordinate used in the previous section. For example, the electronics produces the pitch angle and the rotational velocity signals. On the other hand, the fork elongation $\lambda$ is not readable.

In order to face the problem, the model is simplified using the single body idea, which is reported in this section.

The states variables of the reduced system are:

1. $\theta$: the angle between the ground and the front contact point
2. $x_r$: the rear contact point
3. $\dot{\theta}$: the angular velocity
4. $\dot{x}_r$: the linear velocity

Therefore, the generalized coordinates are $\boldsymbol{q} = \begin{bmatrix}\theta\ x_r\ \dot{\theta}\ \dot{x}_r\end{bmatrix}^T$. A graphical representation of the system is shown in Figure 2.5.

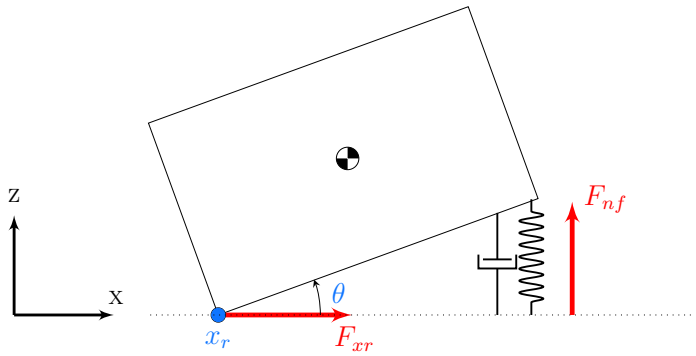


Figure 2.5: Single body model

### 2.3.1 Kinematics

Also for the simplified model the kinematics analysis is carried out, but only few definitions are made, due to the single body structure of the prototype.

In details, only the pitch angle rotation matrix is considered, namely

$$\boldsymbol{R}_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta). \end{bmatrix}$$

Then, only three joints position are needed: the position of the world frame and the position of the rear and front contact point. They are reported below, where $w_b$ represents the distance between the front and rear contact point.

$$\begin{aligned} \boldsymbol{O}_W &= [0\ 0\ 0]^T \\ \boldsymbol{O}_0 &= \boldsymbol{O}_W + [x_r\ 0\ 0]^T \\ \boldsymbol{O}_1 &= \boldsymbol{O}_0 + \boldsymbol{R}_y(\theta)\,[w_b\ 0\ 0]^T \end{aligned}$$

Being this model a single body system, only a centre of mass has to be considered. It is $\boldsymbol{O}_m = \boldsymbol{O}_0 + \boldsymbol{R}_y(\theta)\boldsymbol{p}_1$.

In order to construct the kinetic matrix $\boldsymbol{B}(\boldsymbol{q})$, the Jacobian derivation is performed. Therefore, $\dot{\boldsymbol{p}}_1$ and $\boldsymbol{\omega}_1$ are derived as

$$\dot{\boldsymbol{p}}_1 = \frac{\partial \boldsymbol{O}_m}{\partial \boldsymbol{q}} \qquad \boldsymbol{\omega}_1 = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \end{bmatrix} \tag{2.71}$$

The potential energy $\mathcal{U}$ is expressed by $\mathcal{U} = -m_1 \boldsymbol{g}^T \boldsymbol{O}_m$, where $m_1$ is the mass of the body and $\boldsymbol{g} = [0\ 0\ -9.81]$ is the gravitational acceleration.

### 2.3.2 Dynamics

As it can be seen in Figure 2.5, this model does not consider the internal forces such as the chain and rear suspension forces. The only forces acting on this system are the frontal normal force $F_{fn}$, the rear contact point force $F_{x_r}$ and the drag force $F_{drag}$.

The frontal normal force is modelled as a spring-damper system with an adding term $\mu = 1e^{-3}$, useful to make the Eq. 2.72 differentiable in all points.

$$F_{fn} = \frac{1}{2}\left(-kz_f - c\dot{z}_f + \sqrt{(-kz_f - c\dot{z}_f)^2 + \mu^2}\right) \tag{2.72}$$

The force $F_{x_r}$ is the longitudinal force applied on rear contact point.

In this case, the generalized forces are expressed by:

$$\boldsymbol{Q}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{Q}_{M_{fn}} + \boldsymbol{Q}_{M_{fx_r}} + \boldsymbol{Q}_{M_{drag}}$$

The drag force is obtained as in the case of the multibody model. Analogically to the complex model, the inertia matrix $\boldsymbol{B}$ and the Christoffel matrix $\boldsymbol{C}$ are obtained. All these elements can be substituted in Eq. 2.70 to obtain the equation of motion, as made in the case of complex model.

### 2.3.3 Implementation

Both the motorcycle prototypes are written in continuous time domain with CasADi. Then, they will be discretized by ERK4, following the procedure described in Chapter 1.

The single body model will be the model inserted in MATMPC software, hence it will be expoited to predict the system evolution. On the other hand, the multibody will be used to simulate the motorcycle behaviour and to test the designed Nonlinear Model Predictive Control.

# Chapter 3

# Controller Design

In this Chapter, a study to determine the pitch angle that guarantees maximum acceleration is carried out. Then, a controller that handles both the rider demand and the time delay is designed.

Therefore, to achieve the goals four NMPC controllers are designed.

- A tracking control law is implemented, but neither the rider request nor the time delay on the input are considered. This analysis is performed to find a promising reference signal for the variable $\theta$.
- A controller that handles rider demand, where the pitch angle is not inserted in the cost function, in order to let the variable $\theta$ as free as possible.
- A second control law is designed to cope the possible problems that could arise with the first implementation. Indeed, improvements can be obtained weighing the pitch angle in the last stage of the prediction horizon.
- The definitive control scheme, which aims to track a reference value for the pitch angle.

The last three implemented controllers follow a common design idea: the control input restricts the rider demand and acts only to avoid wheeling.

In the last part of the Chapter, the electro-mechanical delay is modelled in order compensate the time delay arising on the input.

## 3.1 Pitch angle of maximum performance

The controller designed in this section is a tracking NMPC controller and it aims to find the pitch angle value that guarantees maximum performances.

The $\theta_{ref}$ that will be obtained with this controller will be used as reference signal in the implementation phase of the controllers designed below.

In this preliminary analysis, the rider demand is not considered. Therefore, MPC evaluates the input force value to be applied to the system in order to reach the desired angle. The input force arrives at the point $x_r$, and the model inserted in MPC is described by the following state variables: $\boldsymbol{x} = \left[\theta, x_r, \dot{\theta}, \dot{x}_r\right]^T$. Indeed, the NMPC exploits the structure of the single body model and the NMPC problem minimizes the following objective function:

$$\min_{x,u} \frac{1}{2} \sum_{k=0}^{N-1} \|\theta_k - \theta_{ref_k}\|_Q^2 + \|u_k\|_R^2 + \|\theta_N - \theta_{ref_N}\|_{Q_N}^2 \tag{3.1}$$

In Eq. 3.1, $\theta_{ref}$ represents the reference for the $\theta$ variable.

The unique scope of this Section is finding the best pitch angle to reach. Therefore, the error on $\theta$ and the control input are the only variables weighted in the cost function.

To depict a situation near to the real physical actuation, the bounds reported in Eq. 3.2 are applied on the input variable $u$.

$$\underline{u} \leq u \leq \bar{u} \tag{3.2}$$

## 3.2 Controller design

The proposed control law is thought to be coupled with the rider request.

To implement the control schemes, the single body model is inserted in MPC in order to predict its evolution and to evaluate the control input to apply at the motorcycle prototype.

In particular, the motorcycle is driven by the sum of the rider action and an additional state of the system, called $\Delta_u$. The new state is used to limit the thrust on the motorcycle, avoiding wheelie, and its dynamics is described by $\dot{\Delta}_u = u$, where $u$ is the control input computed by MPC. In this way, the control action will be smoother because of the integral made to obtain $\Delta_u$ starting from the control $u$ provided by MPC.

The $\Delta_u$ values are bounded by hard constraints, to make it feasible and to give it only the possibility of reducing the input. Therefore, the new state will be $\underline{\Delta_u} \leq \Delta_u \leq 0$ [N], while the input will be bounded by $\underline{u} \leq u \leq \bar{u}$.

In MPC, the rider demand enters as a constant parameter in the entire prediction horizon, because of the uncertainties on what is the rider future action.

In order to simplify the model used in the control phase and leading to an easier problem in terms of computational time, the state variable $x_r$ is deleted. This is possible because $x_r$, the contact point of the rear wheel with the ground, is an ignorable coordinate.

Summarising, the input force $F_{input} = F_{rider} + \Delta_u$ arrives at the point $x_r$ and the model used to predict the system behaviour and to compute the control has the following

states:

$$\boldsymbol{x} = \left[\theta\ \dot{\theta}\ \dot{x}_r\ \Delta_u\right],\ \text{with}\ \dot{\Delta}_u = u.$$

To find out the best solution for the analysed control problem, three controllers are designed, and they are described in the following Sections, highlighting the positive and negative aspects.

### 3.2.1 Pitch angle not considered in the objective function

Starting from Eq.1.18, the NMPC problem of the first implementation is obtained in Eq.3.3.

$$\min_{x,u} \frac{1}{2} \sum_{k=0}^{N-1} \|\Delta_{u_k}\|_Q^2 + \|\dot{\Delta}_{u_k}\|_{\boldsymbol{R}}^2 + \|\epsilon_k\|_{\boldsymbol{R}}^2 + \|\Delta_{u_N}\|_{Q_N}^2 \tag{3.3}$$

The cost function just presented is subjected to

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, u, F_{rider}) \tag{3.4}$$
$$\boldsymbol{x}_0 = \boldsymbol{x}(t_0) \tag{3.5}$$
$$\theta_{min} \leq \theta + \epsilon \leq \theta_{max} \tag{3.6}$$
$$\epsilon \geq 0 \tag{3.7}$$
$$\Delta_u \leq 0 \tag{3.8}$$
$$u_{min} \leq u \leq u_{max} \tag{3.9}$$

The model dynamics is presented in Eq. 3.4, while Eq. 3.5 states the initial conditions.

Looking at the problem formulation, it can be noticed that the pitch angle state is limited via soft constraints. This choice has been made to avoid forcing the system variable, as far as possible. In general, the input constraints represent physical limits, while the output or state constraints are usually desirable. According to this idea, the input variables are limited by the hard constraints, while states and outputs by the soft ones.

Therefore, in this case, $\theta$ variable will be limited by $\theta_m \leq \theta \leq \theta_M$, with $\theta_M = \theta_{max} + \epsilon$. The lower bound relaxation $\theta_m$ is not very important, because the work is focused on the high values of $\theta$, the values that can cause a wheelie.

In this way, the slack variable $\epsilon$ is inserted in MPC as a control variable, hence it will be weighted in cost function by the matrix $\boldsymbol{R}$. Moreover, it has to be $\epsilon \geq 0$ (Eq.3.7).

Except for the cost index expressed in Eq.3.3, the other elements remain true for all the mentioned elements.

Looking at Eq. 3.3, it can be deduced that this control scheme does not need to track the pitch angle, because the $\theta$ angle is not weighted in the cost function minimized by

MPC.

This choice is due to the fact of maintaining the pitch angle as free as possible according to the rider demand, and limiting it only with the slack constraints.

However, this control law aims to only reduce the rider demand, maintaining the pitch angle lower than the constraints imposed on the wheelie angle variable. This means that the pitch angle stabilization could strongly depends on the soft constraints variables.

Therefore, a second implementation is derived in order to face the problems that could arise with the first one.

### 3.2.2 Pitch angle weighted in the last stage of cost function

The second designed control is depicted in this Section. The controller described before evaluates a control input which depends on the $\theta$ variable only via constraints, and this can lead to undesired behaviours.

For this reason, the objective function minimized by NMPC is changed and it is presented below, slightly modifing the role of the wheelie angle, with respect to Section 3.2.1.

In particular, the objective function is of the form presented in Eq.1.18, but the variables weighted in cost function are different with respect to the scenario described in Section 3.2.1. Indeed, also the pitch angle variable is inserted in the objective function, considering it in the last stage of the prediction horizon. Following this reasoning, the objective function minimized by MPC is

$$\min_{\boldsymbol{x},\boldsymbol{u}} \frac{1}{2} \sum_{k=0}^{N-1} \|\Delta_{u_k}\|_Q^2 + \|\dot{\Delta}_{u_k}\|_{\boldsymbol{R}}^2 + \|\epsilon_k\|_{\boldsymbol{R}}^2 + \|\Delta_{u_N}\|_{\boldsymbol{Q}_N}^2 + \|\theta_N - \theta_{ref_N}\|_{\boldsymbol{Q}_N}^2 \tag{3.10}$$

which is subjected to the same conditions reported in Eq. 3.4- 3.9.

This control structure aims to maintain the pitch angle variable near to the reference value at the prediction instant $N$. In this way, the soft constraints are no more the only parameter from which $\theta$ depends.

### 3.2.3 Tracking on pitch angle

Having considered the pitch error minimization at the last sample of the prediction horizon in Section 3.2.2, in this part the last designed control scheme is presented, which could lead to an optimization problem easier to calculate.

This consideration can be made looking at the cost function presented in Section 3.2.2, because the problem to be solved is irregular, being the last stage very different to the other ones.

To address the possible compuattional problem, a new controller is implemented and the pitch angle variable is inserted in the objective function for all the stages. This leads

to define the objective function reported in Eq. 3.11.

$$\min_{\boldsymbol{x},\boldsymbol{u}} \frac{1}{2} \sum_{k=0}^{N-1} \|\Delta_{u_k}\|^2_{\boldsymbol{Q}} + \|\theta_k - \theta_{ref_k}\|^2_{\boldsymbol{Q}} + \|\dot{\Delta}_{u_k}\|^2_{\boldsymbol{R}} + \|\epsilon_k\|^2_{\boldsymbol{R}} + \|\Delta_{u_N}\|^2_{\boldsymbol{Q}_N} + \|\theta_N - \theta_{ref_N}\|^2_{\boldsymbol{Q}_N} \quad (3.11)$$

In this phase, the pitch angle is desired to reach a reference value that guarantees maximum performances.

All the other elements remain unchanged, namely the hard constraints on $\Delta_u$ and $u$, the slack variable on $\theta$ and the states references.

This control scheme lead to solution that tracks wheelie angle that guarantees the motorcycle to reach the maximum accelerations, handling the rider demand maintaining the computational time low.

## 3.3 Time delay model

The input is applied to the real motorcycle with a time delay due to the electro-mechanical structure of the motorbike.

With the aim of predict the closest situation to the real one, the delay is modelled and then added to the last control law designed in Section 3.2.3, in order to compensate the possible input time delay and to improve its robustness.

The delay has to be written in continuous time domain as the entire motorcycle model, and to achieve the goal two techniques are used.

### Padè approximation

The first method is based on the Padè approximation, which is one of the most known method used to implement delay in continuous time. It is used to approximate a dead time delay $e^{-sT}$, where $T$ is the time delay. This formulation derives from the Laplace transform of a delayed step-signal $h(t) = 1(t - T)$. Therefore, Padè approximation is exploited to round a time delay with a transfer function in continuous time domain.

In general, the basic formulation is employed, namely with equal numerator and denominator degree. However, as highlighted in [10], this leads to a jump at time $t = 0$.

To face the problem, only Padè transfer function with relative degree $r = (n - m) = 1$ are considered, where $n$ and $m$ are the denominator and numerator degree respectively.

Among many, the Padè approximation with $n = 2$ and $m = 1$ is chosen. This choice is a consequence of the transient analysis. Consequently, the delay is modelled with the following transfer function $H_P$:

$$H_P = \frac{6 - 2sT}{6 + 4sT + (sT)^2},$$

where the symbol $T$ represents the time delay.

## Butterworth filter

The other technique is exploiting the delay caused by a Butterworth filter [11].

The Butterworth filters can be used to remove signals with high frequency. They allow passing only the data with a frequency lower than a specified cut-off frequency $f$. The other effect, less known, of the Butterworth filter is the delay introduced in the output signal starting from the filter input. A causal low-pass filter with a fixed order and cut-off frequency delays sinusoids of different frequencies by different amounts.

In general this phenomenon is undesired, but in this case the low-pass filter is chosen exactly for this reason.

Therefore, a Butterworth filter of the $6^{th}$ order is selected.

## The combination of Padè approximation and Butterworth filter

As the definitive time delay model a combination of Padè approximation and a Butterworth filter is chosen, exploiting the two components: the pure continuous time delay and the delay introduced by a low pass filter, which is typical of the actuation mechanism.

Consequently, the Padè approximation with the numerator of first order and a denominator of the second combined with the Butterworth filter of the fourth order is selected as the tool to model the time delay.

The model phase consists of writing the delay transfer function $H_d$ in continuous time, which will be integrated by MATMPC through ERK4. To accomplish the aim, the transfer function is constructed as $H_d = H_P \cdot H_B$, where $H_P$ and $H_B$ stay for Padè approximation and Butterworth filter respectively. Both $H_P$ and $H_B$ are obviously noted in continuous time domain.

The final transfer function is of the form:

$$H_d = \frac{b_1 s + b_0}{s^6 + a_5 s^5 + a_4 s^4 + a_3 s^3 + a_2 s^2 + a_1 s + a_0}.$$

The idea behind the time delay implementation is schematized in Figure 3.1.

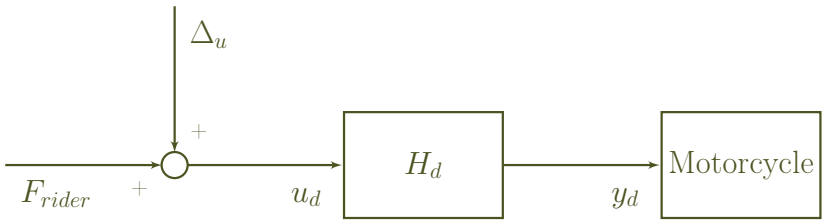


Figure 3.1: Time delay implementation scheme

Looking at Figure 3.1, it can be noticed how the motorcycle input considered so far ($u_d$) is the input of the delay filter. The output $y_d$ will be the delayed system input, hence the new control for the plant.

To formalize this reasoning, the following equation is used.

$$y_d = \boldsymbol{C}_d\boldsymbol{\gamma} + \boldsymbol{D}_d u_d \tag{3.12}$$

In Eq.3.12, the variable $\boldsymbol{\gamma}$ represents a set of additional states, included in the single body model, which is used in the prediction phase of MPC. In order to construct a consistent model, the dynamics of the $\boldsymbol{\gamma}$ states is useful. Therefore, the Eq. 3.13 is implemented.

$$\dot{\boldsymbol{\gamma}} = \boldsymbol{A}_d\boldsymbol{\gamma} + \boldsymbol{B}_d u_d \tag{3.13}$$

To put in practice Eq. 3.12 and 3.13, the realisation theory is exploited obtaining the state space matrices $\boldsymbol{A}_d, \boldsymbol{B}_d, \boldsymbol{C}_d$ and $\boldsymbol{D}_d$.

# Chapter 4

# Simulation Results

In this Chapter, the simulations results related to the previous Chapter are depicted.

At first the preliminary analysis results to find out the best performance tracking angle are presented. Then, the simulation data related to the three implemented controllers are depicted, discussing the strengths and weaknesses of the control schemes.

This study leads to determine the definitive control scheme in terms of tracking control and sensitivity to the model uncertainties.

Then, the electro-mechanical delay is tested to understand the improvements that the time delay compensation can introduce. In particular, the worse case is simulated, hence the case in which the controller does not predict the input delay, then the complete delay compensation case and robustness tests are carried out.

## 4.1 Pitch angle of maximum performance

The aim of this section is finding out the pitch angle value that guarantees maximum acceleration for the motorcycle. Once the angle is found, its value will be used as reference signal in the rest of the Chapter, namely the value that $\theta$ wants to reach.

The implemented NMPC problem is described in the previous Chapter, which minimizes the objective function reported in Eq. 3.1.

The cost function expressed in Eq. 3.1 is minimized by MPC with the following weights matrices:

$$Q = Q_\theta = 1e5 \quad R = R_u = 1e-6 \tag{4.1}$$

The last stage has the same weights of the others stages, therefore $Q = Q_N$. The pitch angle $\theta$ is not limited, while hard constraints are applied on the control evaluated by MPC, to better portray the real vehicle, they are reported in Eq. 4.2.

$$-1000 \leq u \leq 2700 \tag{4.2}$$

To achieve the goal of this section, simulations with different $\theta_{ref}$ are carried out, they last 8s.

Then, the obtained results are compared and discussed. In particular, the travelled space and the stabilization force values are analysed.

As already said, the pitch angle reference is set to a different $\theta_{ref}$ for each simulation, while the other states are desired to be zero. The $\theta_{ref}$ is set to a range of angles from 0° to 45°.

To collect the data, the tests are conducted as follows. The single body model (Section 2.3) is used by MPC to predict the plant behaviour and to compute the control sequence, while the multibody system (Section 2.2) is exploited to simulate the motorcycle behaviour. Therefore, the MPC input is applied to the multibody plant. This implementation aspect is used across the entire Chapter 4.

The two models are sampled at different shooting times: the single body is sampled with $T_s = 10ms$, whereas the complex one has $T_s = 1ms$. This means that for every iteration of single body plant, the multibody one makes 10 iterations.

The initial conditions for the system are obtained from the multibody model: before MPC starts minimizing the objective function, the motorcycle is run free for 1s to obtain a stable initial condition, in terms of pitch angle. Then, the initial value of linear velocity is set.

In this way, the state variables at the initial time are

$$\boldsymbol{x} = \begin{bmatrix} \theta \; x_r \; \dot{\theta} \; \dot{x}_r \end{bmatrix}^T = [-1.35 \; 0 \; 0 \; 20]^T .$$

The pitch angle initial value is equal to $\theta(t_0) = -1.35°$, which is the result of the motorcycle stabilization on the ground. The negative value corresponds to the rest condition, and the motorcycle is maintained at this point by the front suspension action.

Analysing the obtained results, it can be deduced how a greater angle permits a smaller value of input force, leading to a slower performance. For this reason, a study on the small angles $\theta$ is carried out. The collected data are depicted below.

In Figure 4.1 the relation between the pitch angle and the input force at the final instant is reported. In particular, the image highlights the six angles of greater interest, namely
$\theta_{ref} = 0.4°, 0.5°, 0.6°, 0.7°, 0.8°, 0.9°$. This is due to the fact that smaller angles correspond to a motorcycle pressed on the ground, while greater value of $\theta$ leads to a slower velocity.

In Figure 4.2, the tracking of the $\theta$ angle is reported for the studied $\theta_{ref}$ values.

As it can be observed in Figure 4.1, the maximum thrust condition is achieved with angle $\theta = 0.6°$. This reference angle corresponds to the condition in which the motorcycle

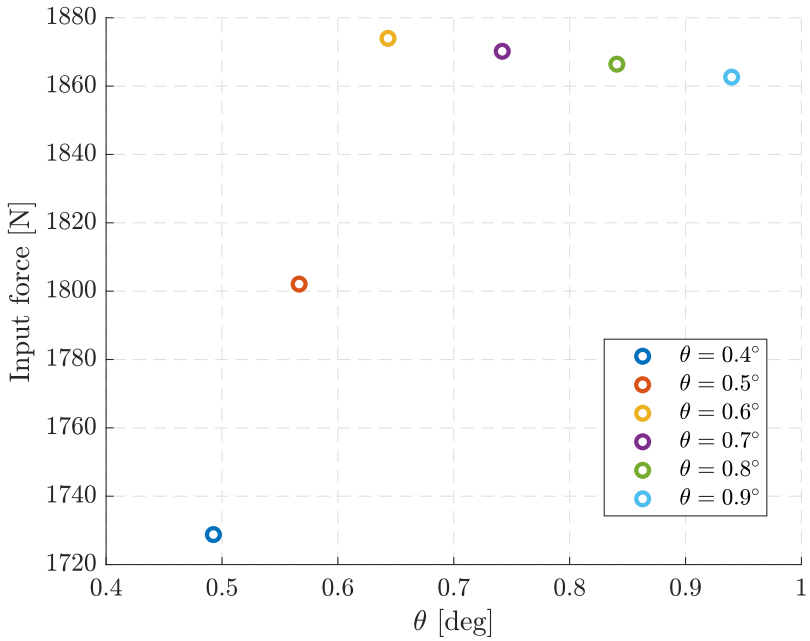


Figure 4.1: Input force-pitch angle relation

is slightly lifting from the ground, as the Figure 4.3 suggests.

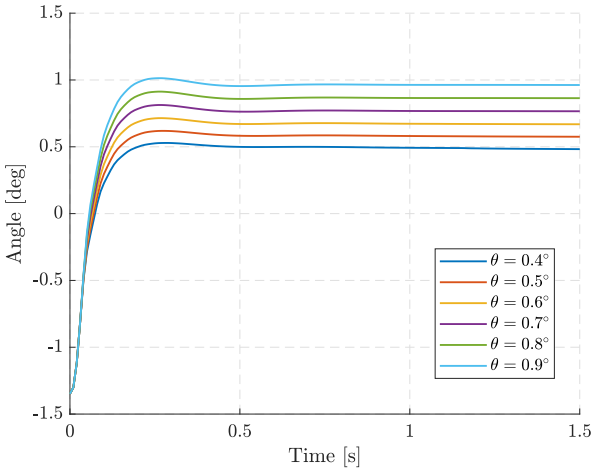


Figure 4.2: Zoom on pitch angle

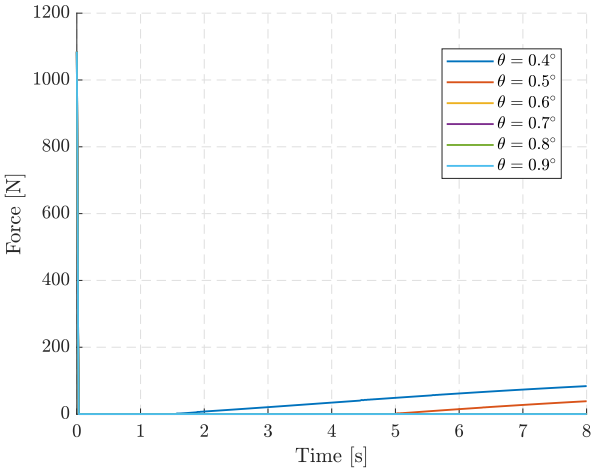


Figure 4.3: Frontal normal force

This means that the motorcycle succeeded in case of zeroed normal force, but maintaining a small pitch angle $\theta$.

Following this reasoning, the $\theta_{ref} = 0.6°$ will be used in this Chapter as the pitch angle reference.

## 4.2 Pitch angle not considered in the objective function

In this section the simulation results related to the controller designed in Section 3.2.1 are discussed. To recap, it minimizes the cost index stated by Eq.3.3, without considering the tracking error on $\theta$ variable.

In this case, the weights matrices are

$$Q = Q_{\Delta_u} = 1e4 \quad \boldsymbol{R} = \begin{bmatrix} R_u \\ R_\epsilon \end{bmatrix} = \begin{bmatrix} 1 \\ 1e15 \end{bmatrix} \tag{4.3}$$

The last stage has the same cost as the others, therefore, $Q = Q_N$.

In these simulations the control $u$ and the new state $\Delta_u$ are bounded as

$$-3000 \leq \Delta_u \leq 0$$
$$-50000 \leq u \leq 50000$$

Also the $\epsilon$ variable needs to be limited, hence it is set to $0 \leq \epsilon \leq 500$.

The curve used in the tests of this work to simulate the rider request corresponds to a progressive acceleration manoeuvre, and it comes from experimental data.

Except for the cost function expressed in Eq.3.3 and its related weights, all the elements just mentioned will remain true also for the Sections 4.3 and 4.4. Therefore, the constraints on $\Delta_u$, $u$ and $\epsilon$ will be set to these values for all the simulations of this project.

As already said, the tracking of the pitch angle is not the goal, however its reference is set to $\theta_{ref} = 0.6°$ (the best performance angle), while the other states are desired to be zero. The prediction horizon is set to $N = 20$.

In order to test the designed control action, different simulations are carried out, with the procedure described in the section above.

The initial conditions for the system are obtained from the multibody model, as before. Therefore, the state variables at the initial time are

$$\boldsymbol{x} = \begin{bmatrix} \theta & \dot{\theta} & \dot{x}_r & \Delta_u \end{bmatrix}^T = [-1.35 \; 0 \; 20 \; 0]^T$$

In Figures below the simulations results are shown. They are related to the conditions illustrated before and setting $\theta_M$ to different values. The parameter $\theta_m$ is set equal for all the tests, $\theta_m = -10°$.

The $\theta$ signal does not reach the reference value of $\theta_{ref} = 0.6°$, as expected. However, it is important to notice how the $\theta$ variable is stabilized at different values with respect to $\theta_M$. In particular, in Figure 4.4, the soft constraints and the related pitch angles are highlighted with the same colour, with dotted and continuous line respectively.

Looking at Figure 4.4, it is possible to observe how the bounds on $\theta$ are respected. This leads also to undesired behaviour: with large bounds the motorcycle is free to wheelie, regardless of the angle reference.

The phenomenon depicted above is reported also in Figure 4.5, where the input force tries to follow the rider request and $\Delta_u$ acts to limit the pilot. The $\Delta_u$ intervention occurs later as $\theta_M$ increases. This concept is notable also in Figure 4.6, where the new state evolution is reported.

The delay in $\Delta_u$ action is due to both the soft constraint value and the exclusion of $\theta$ from the cost function. This leads to a stronger intervention on the motorcycle input, as

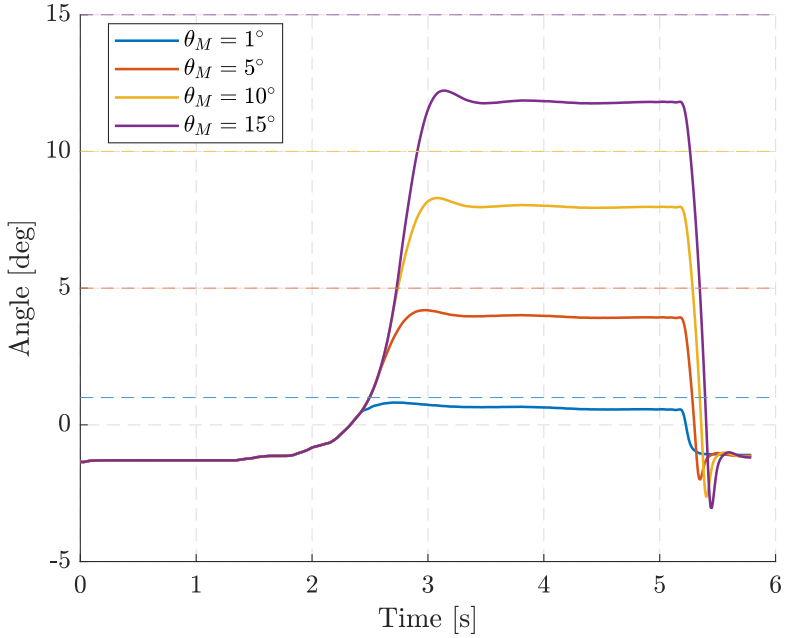


Figure 4.4: Pitch angle with different pitch upper bound $\theta_M$ and N=20

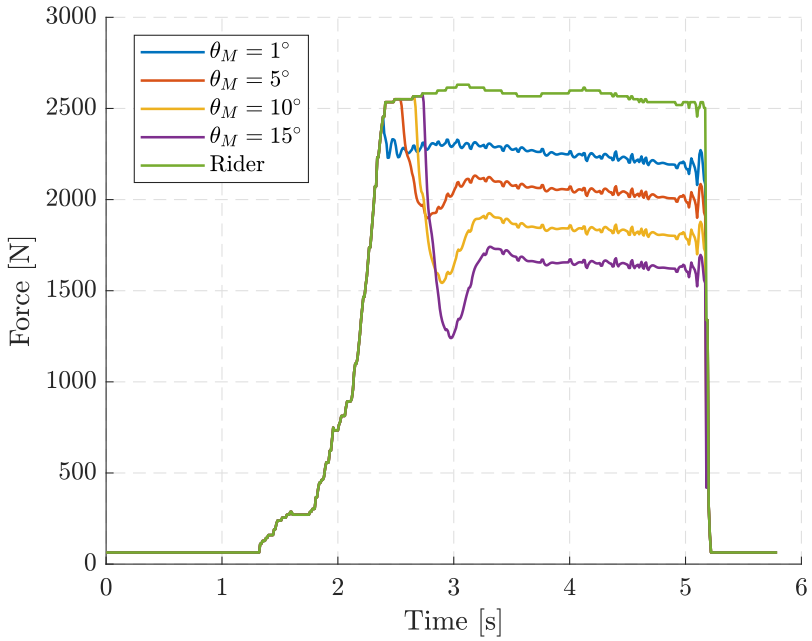


Figure 4.5: Input force with different pitch upper bound $\theta_M$ and N=20

it can be observed in Figure 4.6.

The wheelie phenomenon can be appreciated also in Figure 4.7, where the frontal normal force can be observed. Indeed, the frontal normal forces are zeroed when the motorcycle is lifting from the ground, returning different from zero when the motorcycle is hold on the road, with a value that increases proportionally to the value reached by the pitch angle.

In this situation, the control saves the pitch angles just before it goes outside the boundaries. This is appreciable in Figure 4.8, where the twenty predicted values of $\theta$ are reported, referred to the $4^{th}$ second of the simulation. In Figure 4.8, it can be seen as the pitch angle is far from the $\theta_{ref}$, as expected, and the control input acts only reducing the

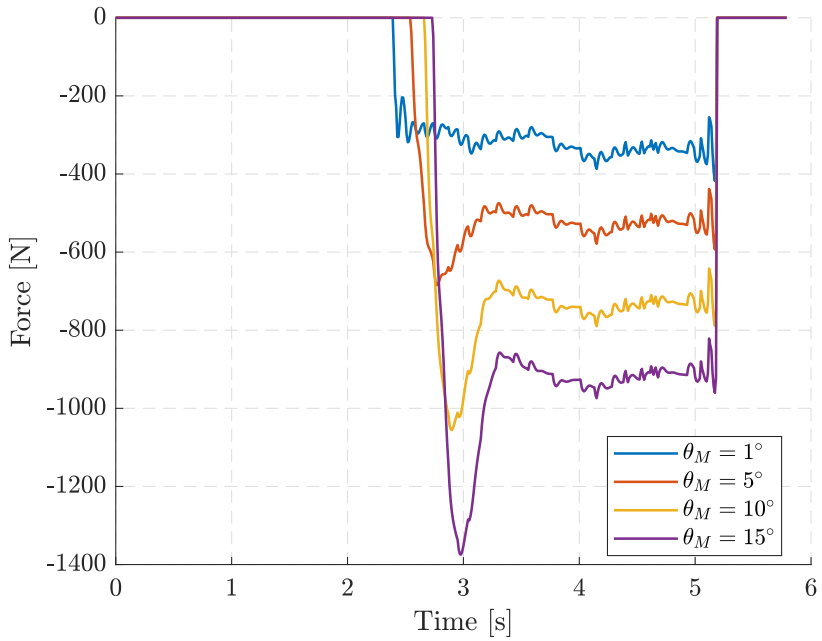


Figure 4.6: $\Delta_u$ state with different pitch upper bound $\theta_M$ and N=20

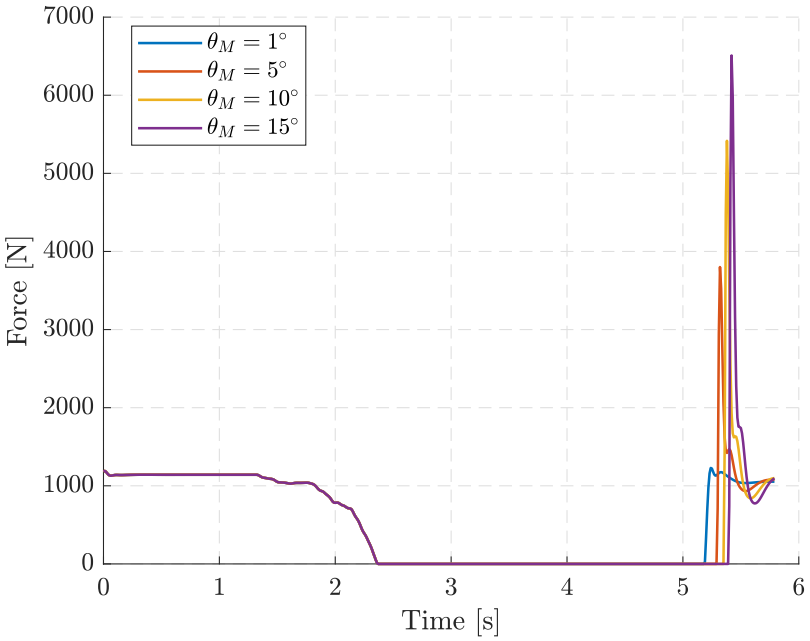


Figure 4.7: Frontal normal force with different pitch upper bound $\theta_M$ and N=20

rider input, otherwise the $\theta$ variable would overcome the limits, as predicted at sample N=20.

This control action is permitted because of the prediction phase carried out by MPC, indeed these results are dependent on the prediction horizon $N$.

For this reason, other simulations are performed, with a larger prediction horizon. All the other parameters are the same as before.

In Figures 4.9-4.12 the collected data are reported. Looking at Figure 4.10, it can be noticed that a larger prediction horizon leads to a preventive action on the motorcycle.

This is observable also in the Figure 4.11, where the limiting action on the rider demand is shown. This aspect can be highlighted also comparing 4.6 and 4.11, where the

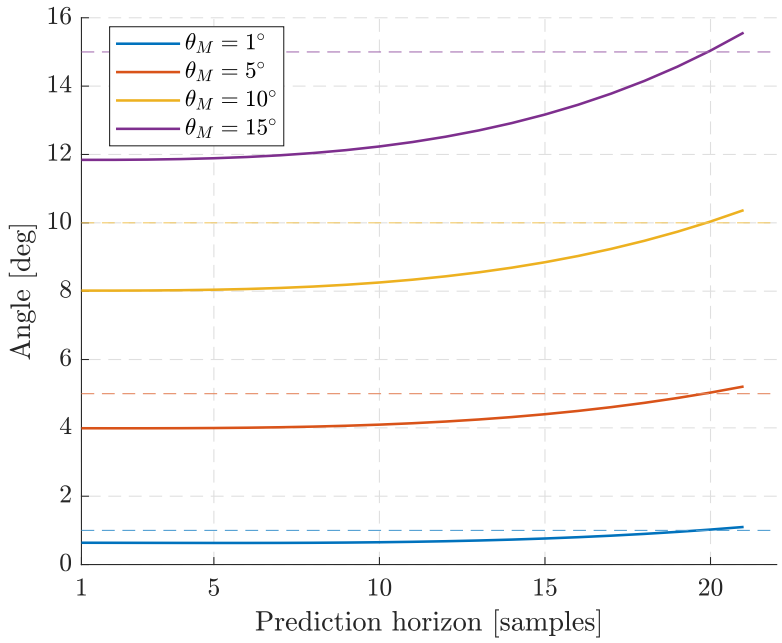


Figure 4.8: $\theta$ variable predicted at t=4s, when N=20

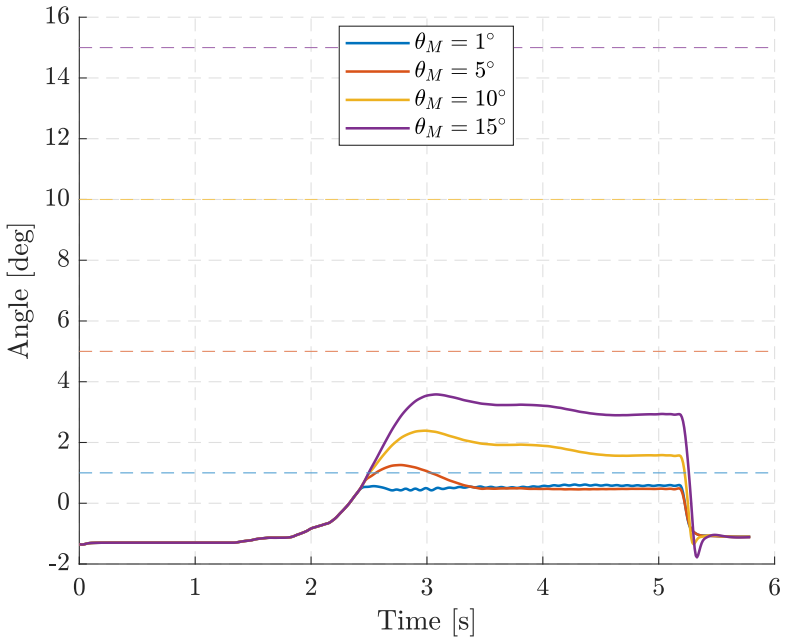


Figure 4.9: Pitch angle with different pitch upper bound $\theta_M$ and N= 50

$\Delta_u$ role is smoother with a larger $N$ value.

Operating before, and with smaller intervention of $\Delta_u$, it is allowed to stabilize the force input at a greater value, also maintaining the $\theta$ values quite small (Figure 4.9).

In the Figures just presented, it can be observed how the system reacts, imposing $-10° \leq \theta \leq 1°$. In particular, in Figure 4.12, the value of the frontal normal force is not always equal to zero in the time interval $[2.5 - 4.5]\, s$. The oscillatory behaviour depicted in this Figure is reflected also in the other Figures of these simulations. It is due to the fact that the constraints set on the pitch angle, even if they are soft, are too stringent for the simulated scenario, causing the motorcycle to bounce.

Except for the undesired oscillatory behaviour obtained with the smallest value of $\theta_M$,

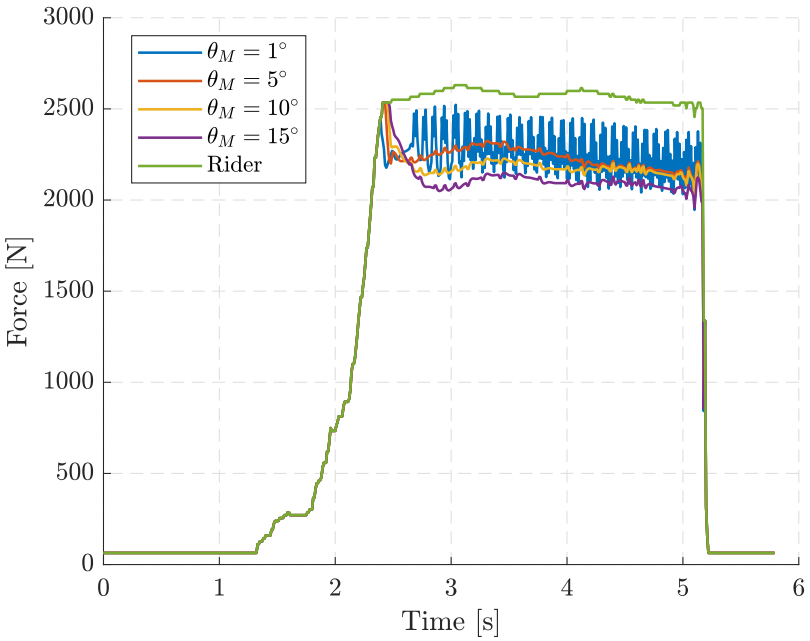


Figure 4.10: Input force with different pitch upper bound $\theta_M$ and N= 50

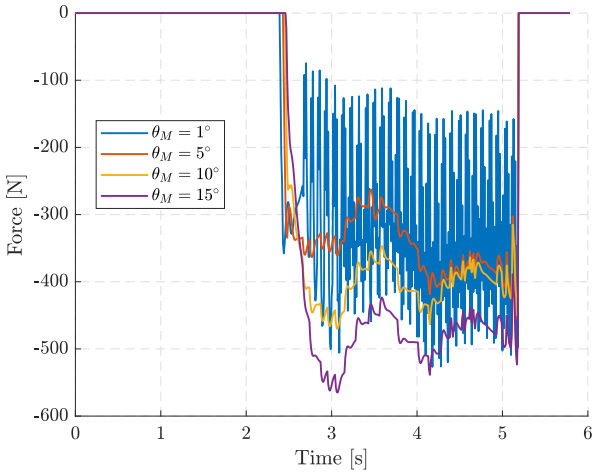


Figure 4.11: $\Delta_u$ state with different pitch upper bound $\theta_M$ and N=50

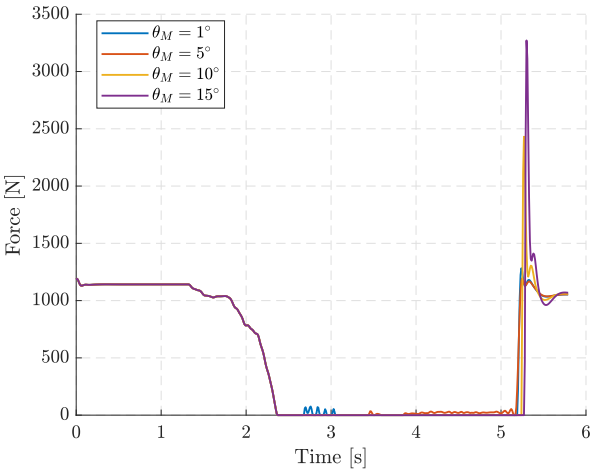


Figure 4.12: Frontal normal force with different pitch upper bound $\theta_M$ and N=50

the larger prediction horizon seems to be preferred. Unfortunately, working with a tool of real-time implementation, this is not sufficient. The computational-time demand has to be considered.

In Tables 4.1 and 4.2 the MATMPC effort is reported in terms of CPT and QP times. CPT stays for the total computational time used to handle the posed problem per sampling instant, while QP denotes the time needed to solve the Quadratic Programming problem per sampling instant.

It is important to remark that all the simulations reported in this work are carried out in *Real-time* setup, therefore only an iteration to solve the SQP is performed.

In the Tables reported below, the average (avg) and the maximum (max) values of the mentioned quantities are reported, in milliseconds.

Looking at the computational-time values, $N = 20$ is certainly the best choice in this

| $\theta_M$ | CPT avg | CPT max | QP avg | QP max |
|---|---|---|---|---|
| 1° | 0.83 | 7.47 | 0.12 | 0.47 |
| 5° | 0.84 | 6.13 | 0.14 | 0.60 |
| 10° | 0.82 | 7.45 | 0.13 | 0.35 |
| 15° | 0.84 | 8.44 | 0.13 | 0.33 |

Table 4.1: Computational time with N=20

| $\theta_M$ | CPT avg | CPT max | QP avg | QP max |
|---|---|---|---|---|
| 1° | 4.21 | 9.20 | 1.80 | 5.94 |
| 5° | 3.85 | 21.67 | 1.41 | 19.26 |
| 10° | 4.69 | 21.25 | 1.72 | 18.5 |
| 15° | 4.80 | 10.10 | 1.67 | 7.25 |

Table 4.2: Computational time with N=50

sense, indeed in real-time implementations also few milliseconds are crucial.

Therefore, changing the amplitude of the prediction horizon is not the best solution that it could be found.

The same conclusion can be obtained looking at Figure 4.13, where it is possible to see how the control acts following the same mechanism as in case of $N = 20$.

In conclusion, the controller presented in this scenario is lacking, because, even if it maintains the pitch angle into the set bounds, the simulations performances are extremely dependent on the value of $\theta_M$.

This happens because the controller acts only if the predicted pitch angle at the final sample $N$ overcomes the bounds, regardless to the $\theta_{ref}$ value.

Studying the behaviour described in Figure 4.8 and 4.13, it leads to take into account another control law. It will consider the value of $\theta$ in the last stage of the prediction horizon, in order to maintain a small pitch angle at the sample N=20.

## 4.3 Pitch angle weighted in the last stage of cost function

After discussing the results in previous section, the second controller is tested.

In this case, the role of the pitch angle is slightly changed with respect to Section 4.2.

In particular, the objective function is presented in Eq.3.10, where the pitch angle error is minimized in the last stage of the prediction horizon.

The hard and soft constraints applied on the input and states variables remain the same as above. Adding the pitch angle variable in the last stage cost function, the weights matrices become:

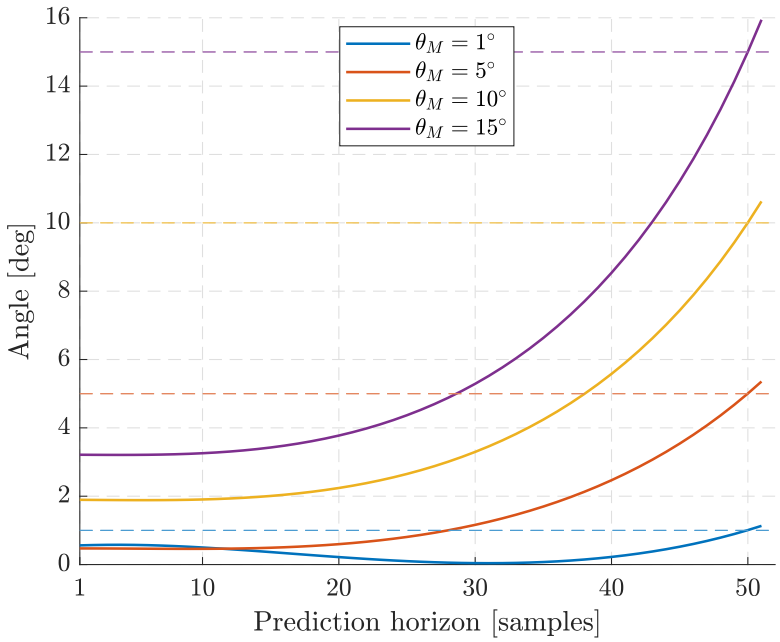


Figure 4.13: $\theta$ variable predicted at t=4s, when N=50

$$\boldsymbol{Q} = \begin{bmatrix} Q_\theta \\ Q_{\Delta_u} \end{bmatrix} = \begin{bmatrix} 0 \\ 1e4 \end{bmatrix} \quad \boldsymbol{Q}_N = \begin{bmatrix} Q_{N_\theta} \\ Q_{N_{\Delta_u}} \end{bmatrix} = \begin{bmatrix} 1e15 \\ 1e4 \end{bmatrix}$$

The matrix $\boldsymbol{R}$ remains the same as above, it is reported in Eq. 4.4.

$$\boldsymbol{R} = \begin{bmatrix} R_u \\ R_\epsilon \end{bmatrix} = \begin{bmatrix} 1 \\ 1e15 \end{bmatrix} \tag{4.4}$$

The simulations are carried out as before, hence the initial variable conditions, the variable references and the constraints remain unchanged. The prediction horizon is set to $N = 20$.

With this structure, the pitch angle stabilization is no longer dependent on the soft constraint values. It can be observed looking at Figure 4.14, where the pitch angle is the same also with different $\theta_M$ values. Indeed, setting $Q_{N_\theta} \neq 0$, the $\theta_N$ variable has to remain near to the reference value.

This behaviour can be appreciated also in the Figures 4.16-4.18. The pitch angle aims to track the reference value only in the last stage of the prediction horizon. However, the addition of the $Q_{N_\theta}$ value acts maintaining the $\theta$ value near to the desired $\theta_{ref}$ for all the simulation, guaranteeing satisfying performance. In Figure 4.15, the predicted $\theta$ values at $4^{th}$ second of the simulation are reported. Compared with Figure 4.8, the improvements introduced with this Section are observable. Not only $\theta$ is kept inside the bounds at the predicted stage N, but the pitch angle is maintained near the desired value in the entire prediction horizon. This evolution is caused by the optimization made on the control input.

The soft constraints are no more parameters that can change the results in terms of the $\theta$ steady state value, hence the pitch angle is stabilized around the reference signal.

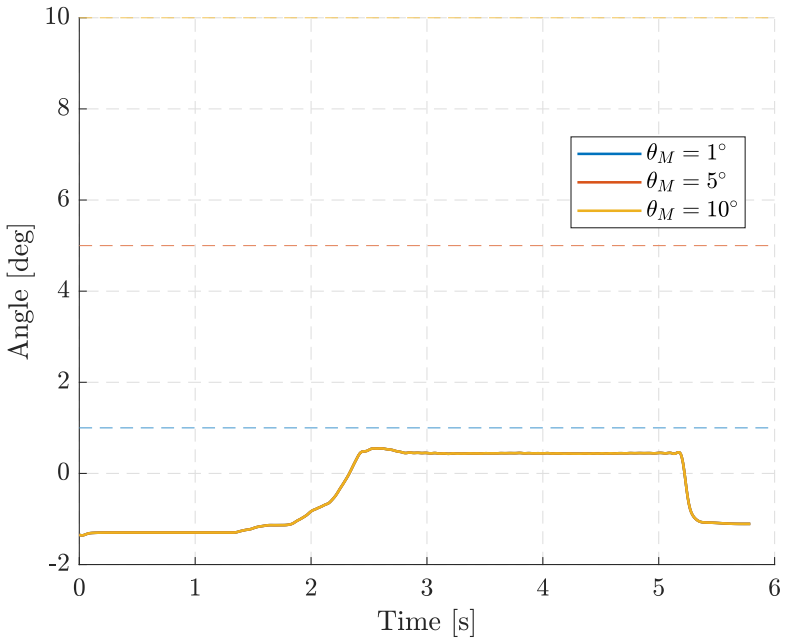


Figure 4.14: Pitch angle with different pitch upper bound $\theta_M$ and $Q_{N_\theta} \neq 0$

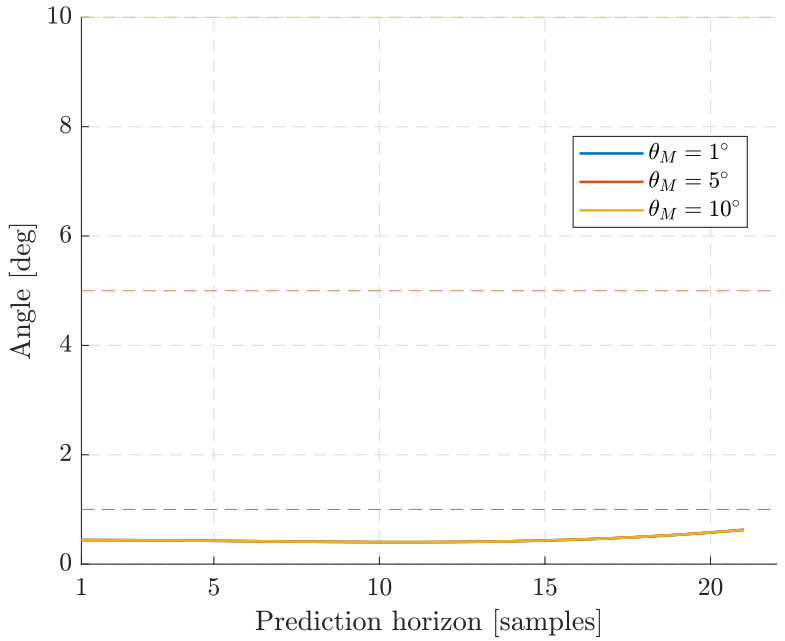


Figure 4.15: $\theta$ variable predicted at t=4s, when $Q_{N_\theta} \neq 0$

All these considerations have effects also in the Figures of the state $\Delta_u$ and of the frontal normal forces, namely in Figure 4.17 and 4.18.

It is important to notice that in Figure 4.18, the frontal normal forces are not completely zeroed, even if the constraints are not so binding. This means that the motorcycle cannot perform its maximal acceleration in this condition, because the frontal suspension force are working to maintain the frontal wheel compressed on the road.

The time demand needed to solve the problem is reported in milliseconds in Table 4.3, in terms of CPT and QP time. Obviously, the computational effort is greater than the case with same prediction horizon $N$ but without weighting $\theta$ in the objective function. This is caused by the request on $\theta$ error minimization.

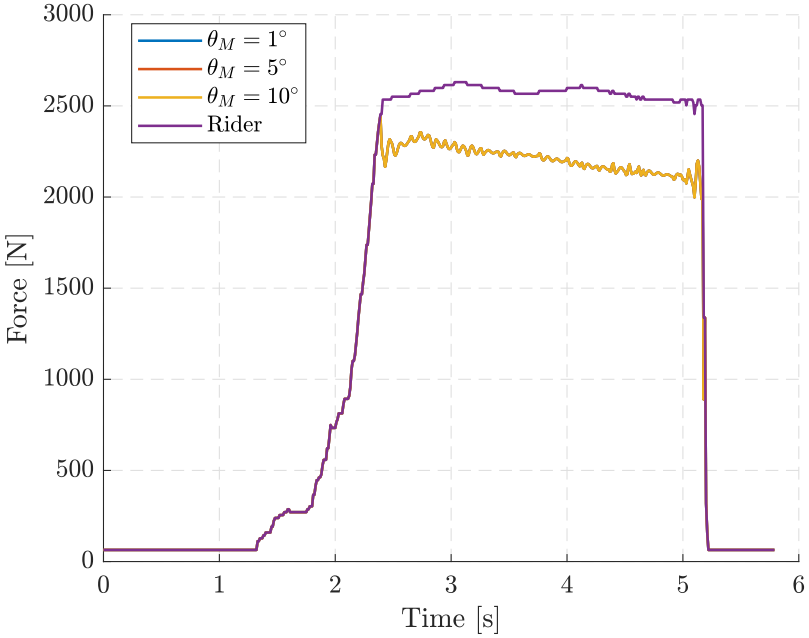


Figure 4.16: Input force with different pitch upper bound $\theta_M$ and $Q_{N_\theta} \neq 0$

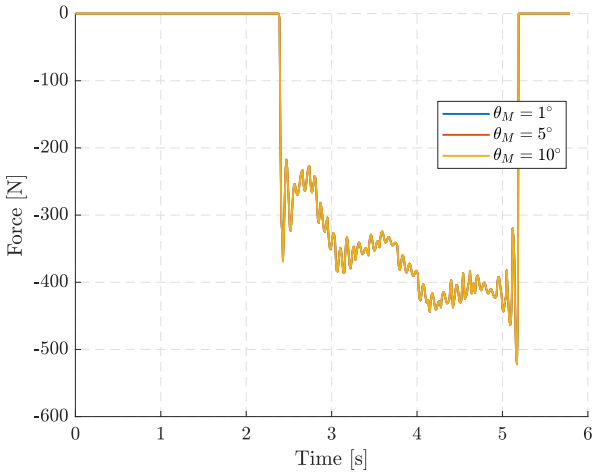


Figure 4.17: $\Delta_u$ state with different pitch upper bound $\theta_M$ and $Q_{N_\theta} \neq 0$

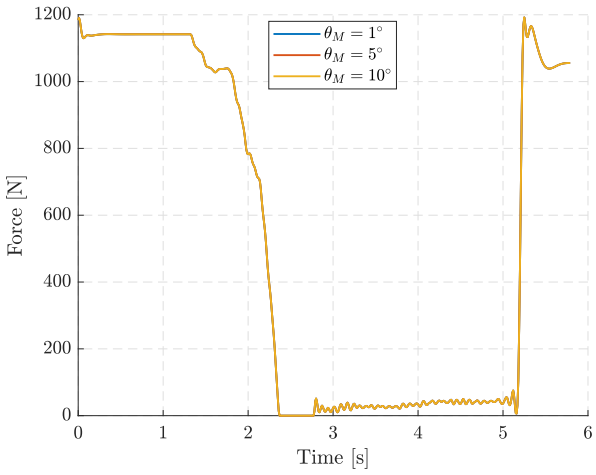


Figure 4.18: Frontal normal force with different pitch upper bound $\theta_M$ and $Q_{N_\theta} \neq 0$

| $\theta_M$ | CPT avg | CPT max | QP avg | QP max |
|---|---|---|---|---|
| 1° | 2.51 | 15.2 | 0.40 | 3.98 |
| 5° | 2.18 | 11.8 | 0.37 | 1.46 |
| 10° | 2.70 | 13.24 | 0.44 | 2.32 |

Table 4.3: Computational time with N=20 and $Q_{N_\theta} \neq 0$

## 4.4 Tracking on pitch angle

As it comprehended in the previous Sections, setting a weight value $Q_{N_\theta} \neq 0$ is a good choice to obtain appreciable results for the problem in analysis.

In this Section, to demonstrate the effectiveness of the controller designed in Section 3.2.3, some simulations are carried out.

In these tests, the pitch angle is desired to reach a reference value, therefore the minimized objective function is that reported in Eq. 3.11.

In this way, also the weight $Q_\theta$ will be different from 0. In order to perform the simulations, the cost function is constructed with the following weights matrices:

$$\boldsymbol{Q} = \begin{bmatrix} Q_\theta \\ Q_{\Delta_u} \end{bmatrix} = \begin{bmatrix} 1e13 \\ 1e4 \end{bmatrix} \quad \boldsymbol{Q}_N = \begin{bmatrix} Q_{N_\theta} \\ Q_{N_{\Delta_u}} \end{bmatrix} = \begin{bmatrix} 1e13 \\ 1e4 \end{bmatrix} \quad \boldsymbol{R} = \begin{bmatrix} R_u \\ R_\epsilon \end{bmatrix} = \begin{bmatrix} 1 \\ 1e15 \end{bmatrix}.$$

All the other elements remain unchanged, namely the hard constraints on $\Delta_u$ and $u$, the slack variable on $\theta$ and the states references. Also the initial conditions are equal as above.

In particular, for these tests, the prediction horizon is $N = 20$, while the soft constraint is varying: $\theta_M = 1°,\ 3°$.

The pitch angle reference is obtained in Section 4.1, where the analysis on the maximum performance is carried out. Therefore, the maximum performance angle is set as the $\theta_{ref}$ value.

In the following Figures the results are shown. In detail, in Figure 4.19 it is possible to see how the pitch angle is near to the desired value, with lower tracking error in case of $\theta_M = 3°$. This happens because a greater value of $\theta_M$ permits to relax the constraints on the pitch angle and leads to a better tracking performance.

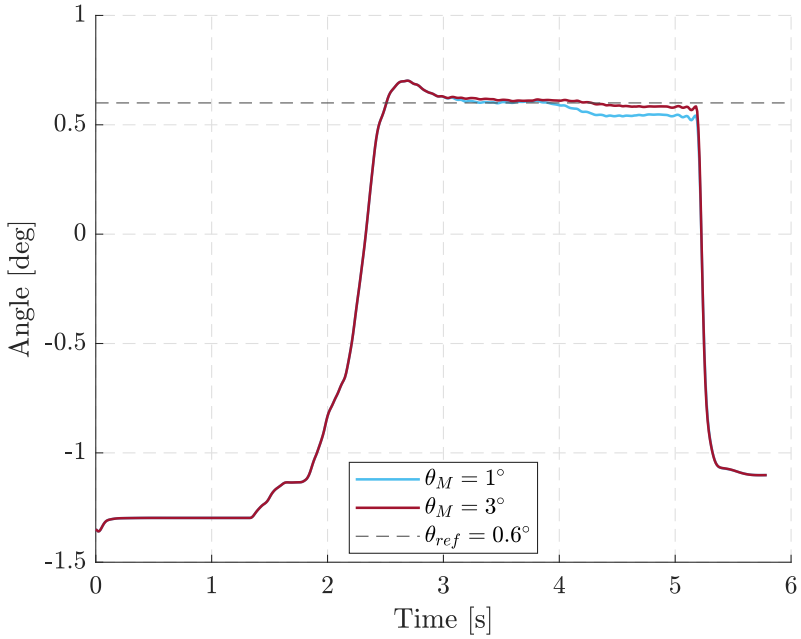


Figure 4.19: Pitch angle tracking with different pitch upper bound $\theta_M$

Looking at Figures 4.20-4.22 it can be seen how these two simulations results are not so different from each other. Indeed, in Figures 4.21 and 4.22 the curves related to $\theta_M = 1°$ are quite overlapped to the case of $\theta_M = 3°$.

This aspect is also revealed in terms of computational-time demand, whose data are reported in Table 4.4.

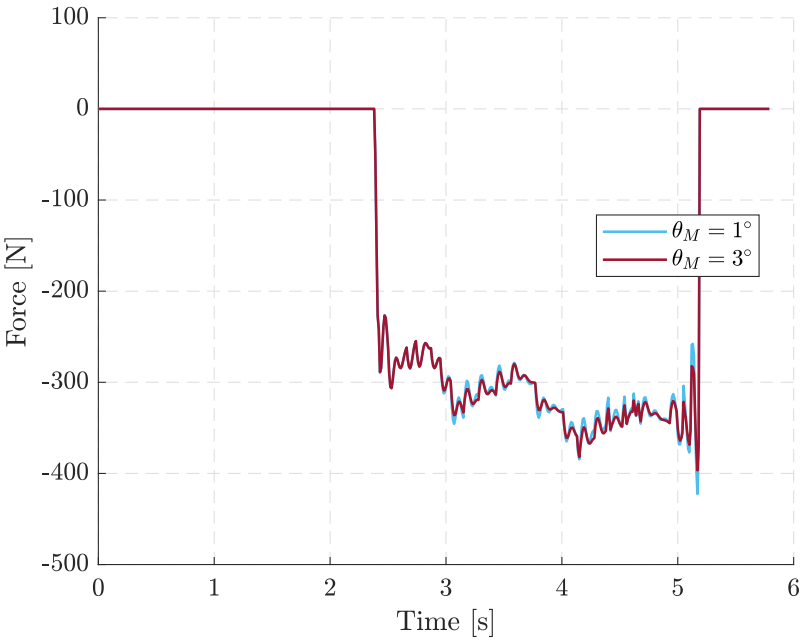


Figure 4.20: $\Delta_u$ state in case of $\theta$ tracking with different pitch upper bound $\theta_M$

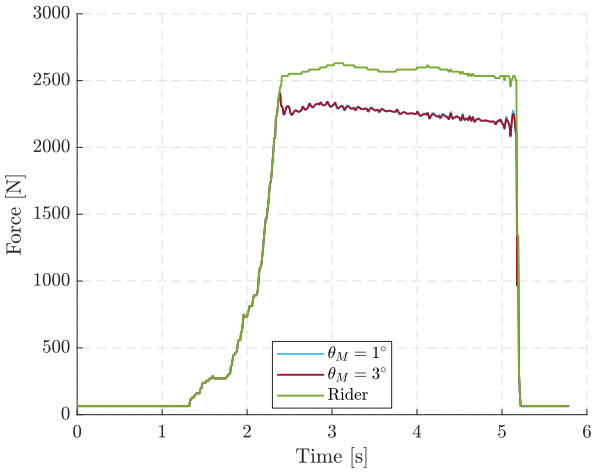


Figure 4.21: Input force in case of $\theta$ tracking with different pitch upper bound $\theta_M$

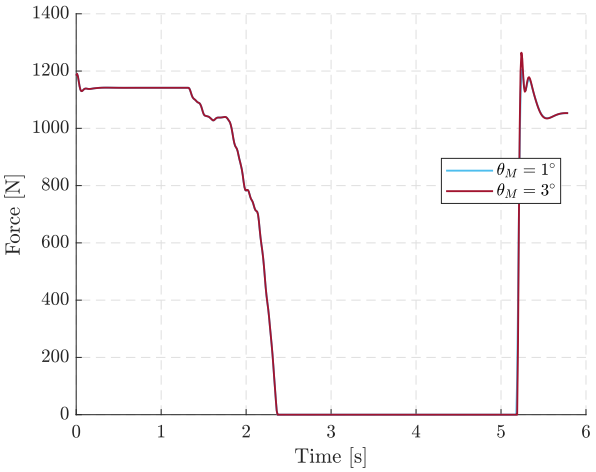


Figure 4.22: Frontal normal force in case of $\theta$ tracking with different pitch upper bound $\theta_M$

## 4.5 Controllers comparison

To see the differences between the control law depicted in Section 3.2.2 and that implemented in Section 3.2.3, the figures below are reported. They show the performed simulations with $\theta_M = 3°$. This choice is made in order to let the pitch angle free enough to avoid both bouncing and wheeling phenomenon. All the other elements are equal as above.

Comparing the simulations data, the third scenario (Section 4.4) seems the best. Indeed, observing Figure 4.23, it can be seen how the desired value of $\theta_{ref}$ is reached only in this case, while in the case in which only $Q_N \neq 0$ (Section 4.3) the pitch angle remains lower.

| $\theta_M$ | CPT avg | CPT max | QP avg | QP max |
|---|---|---|---|---|
| 1° | 1.12 | 7.47 | 0.25 | 1.65 |
| 3° | 1.07 | 6.48 | 0.18 | 0.9 |

Table 4.4: Computational time in case of $\theta$ tracking

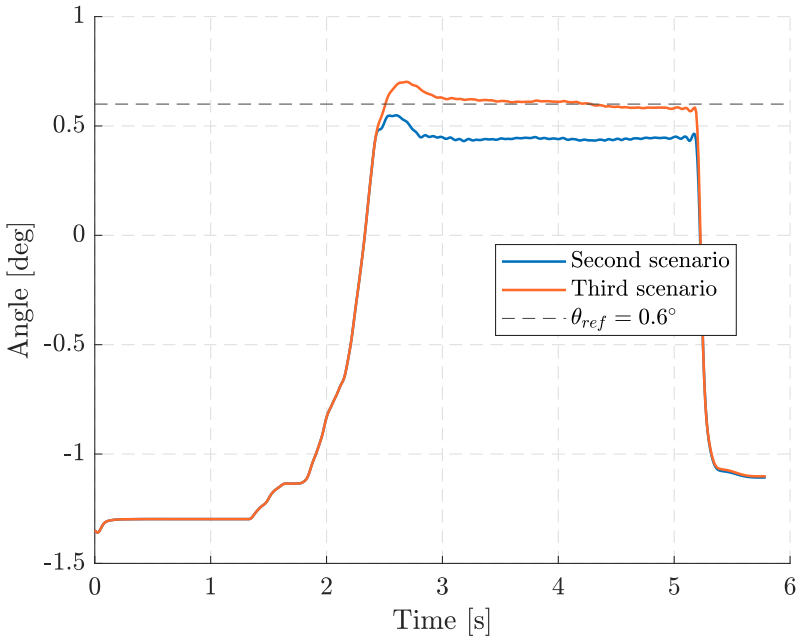


Figure 4.23: Controller comparison in nominal case: pitch angle

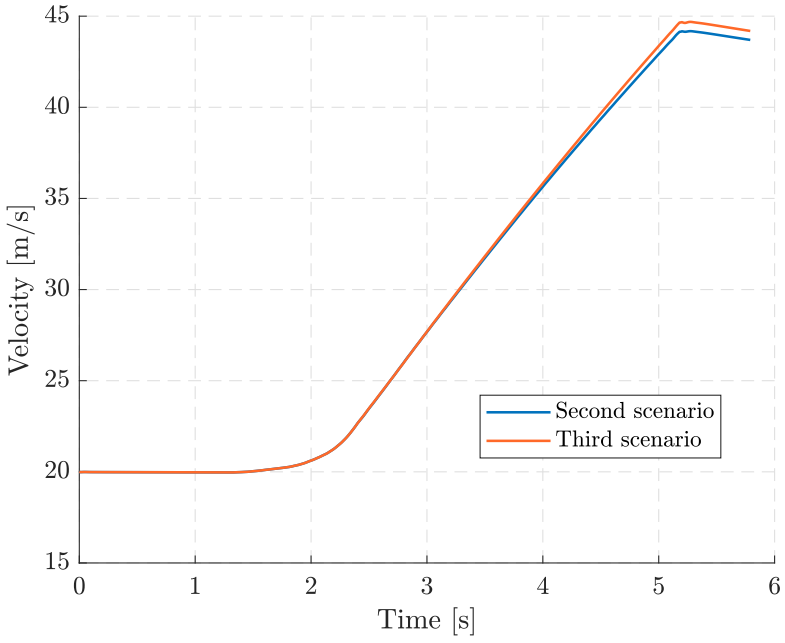


Figure 4.24: Controller comparison in nominal case: linear velocity

With the configuration of the third scenario, the motorcycle reaches a greater velocity (Figure 4.24) and hence in same time it can travel more space. This is also notable in Figure 4.25, in terms of input force.

Looking at Figure 4.26, it can be noticed how the motorcycle is still on the ground in

the second case, this leads to a non maximal thrust, which could be guaranteed with an angle of $\theta = 0.6°$, as seen in Section 4.1

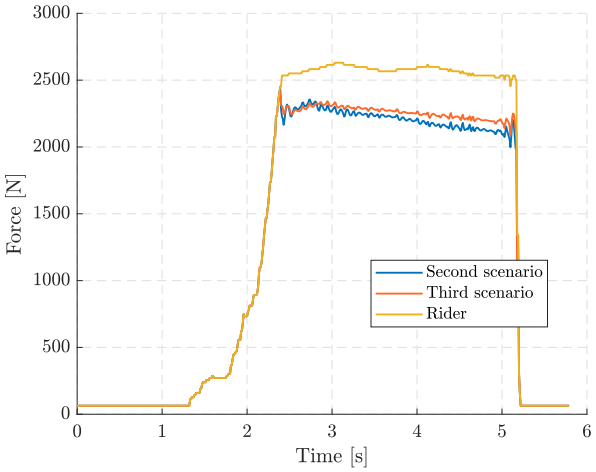


Figure 4.25: Controller comparison in nominal case: input force

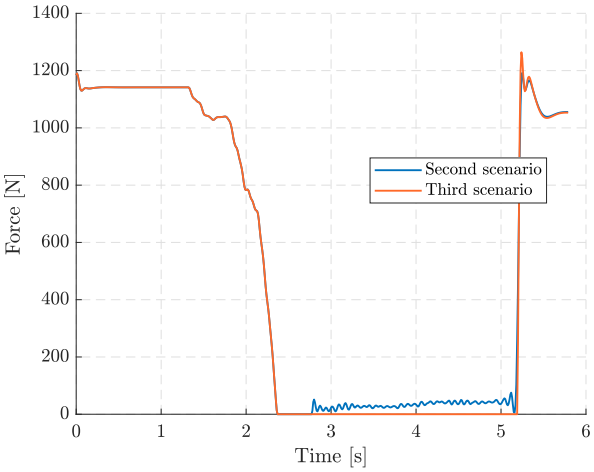


Figure 4.26: Controller comparison in nominal case: frontal normal force

The arrangement with both $Q_\theta$ and $Q_{N_\theta}$ different from zero is promising also looking at the computational time. This is due to the fact that the MPC maintains the pitch angle near the $\theta_{ref}$ signal for all the prediction horizon. On the other hand, the case in which only $Q_N \neq 0$ allows to let the angle more free in the prediction horizon, but with the last sample near to the desired value $\theta_{ref}$. Therefore, in this sense, the controller of Section 4.3 is aimed to solve an irregular NMPC problem. The effect is a more expensive behaviour in terms of computational time with respect to the pitch angle tracking case, as annotated in Table 4.5.

| Scenario | CPT avg | CPT max | QP avg | QP max |
|---|---|---|---|---|
| Second scenario | 1.94 | 13.26 | 0.33 | 1.42 |
| Third scenario | 1.07 | 6.48 | 0.18 | 0.84 |

Table 4.5: Controller comparison in nominal case: computational time with $\theta_M = 3°$

### 4.5.1 Models uncertainties

As mentioned in Chapter 2, the two used models describe the same plant, but there are some critical aspects to consider. Indeed, working with a model-based control technique, all the parameters used to model the system are important. This happens because these plant uncertainties can cause dramatic effects. In particular, in this work the rider mass and the offset on the pitch angle are taken into account. In this section the sensitivity to the model uncertainties is studied, for both the controller with $Q_N \neq 0$ and the $\theta$ tracking control law.

### Pitch angle offset

The first gap between the single body model and the multibody one is the pitch angle offset. This is due to the fact that the multibody plant used in simulation starts wheeling at $\theta_0 = 25.5°$, whereas the front wheel of the simplified model inserted in MPC starts lifting with $\theta_0 = 0°$.

In simulation, and consequentially in an experimental context, if the value of $\theta_0$ is badly evaluated it can lead to a tracking error different from 0. In order to understand the effects caused by this type of uncertainty, a set of simulations are carried out. In detail, the results are obtained with the procedure depicted before, testing both the second and third scenario. All the parameters are unchanged, hence the initial conditions, the weight matrices, the variables reference and the constraints are the same. The simulations are performed with different value of the offset $\theta_0$. In the images below, the cases of $\theta_0 = 23.5°, 24.5°, 25.5°, 26.5°, 27.5°$ are reported.

The collected data referred to the the controller presented in Section 4.4 can be appreciated in Figure 4.27, where the tracking of the $\theta$ variable can be observed.

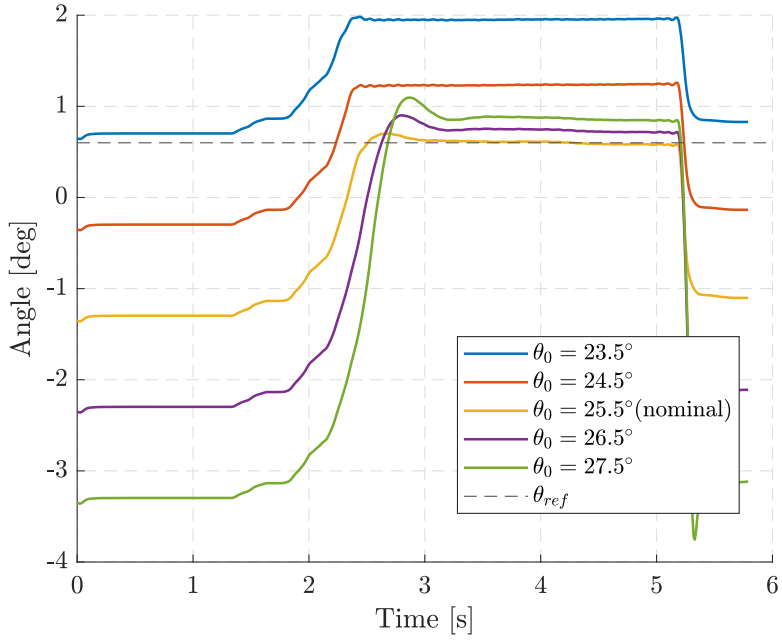


Figure 4.27: $Q_\theta \neq 0$ and $Q_{\theta_N} \neq 0$: pitch angle with different pitch angle offset $\theta_0$

It can be noticed how by increasing the offset $\theta_0$ the magnitude of the error increases. On the other hand, it can be observed that in case of $\theta_0 = 23.5°$ and $\theta_0 = 24.5°$ the error is greater in absolute value, this is due to the fact that the motorcycle is like breaking down. This phenomenon is depicted in Figure 4.28, where the frontal normal forces remain different from zero in these two cases.

Being $\theta_{ref}$ the angle that permits a maximum thrust, an error $e_\theta \neq 0$ corresponds to a slower performance for the motorcycle. It can be observed in Figure 4.29, in terms of velocity, and in Figure 4.30, as input force applied to the motorcycle.

The results related to the control scheme with only $Q_{\theta_N} \neq 0$ are reported in Figures

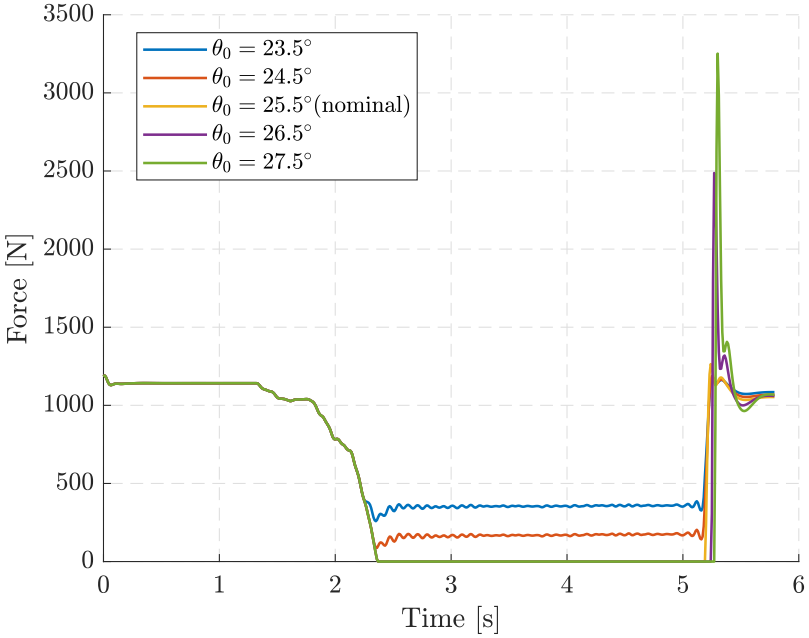


Figure 4.28: $Q_\theta \neq 0$ and $Q_{\theta_N} \neq 0$: frontal normal force with different pitch angle offset $\theta_0$

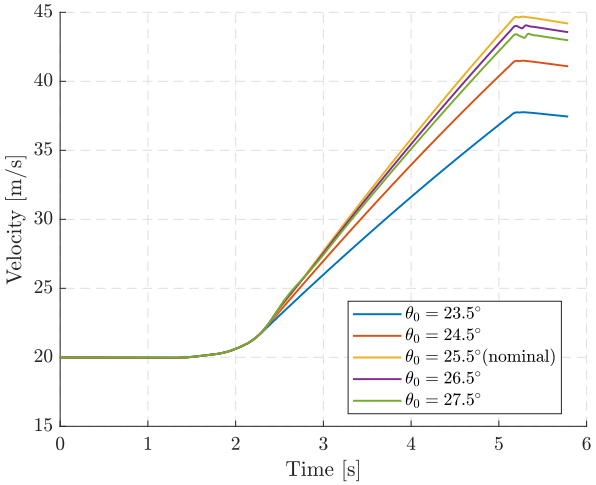


Figure 4.29: $Q_\theta \neq 0$ and $Q_{\theta_N} \neq 0$: velocity with different pitch angle offset $\theta_0$

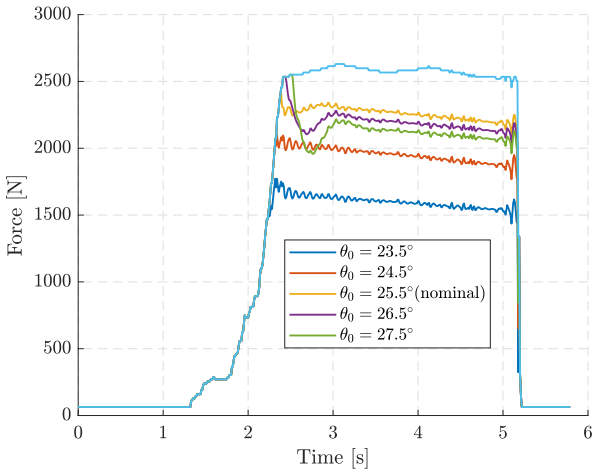


Figure 4.30: $Q_\theta \neq 0$ and $Q_{\theta_N} \neq 0$: input force with different pitch angle offset $\theta_0$

4.31-4.34.

As shown at the beginning of this Section, the second scenario is less efficient than the third one. It can be concluded also looking at the Figure 4.31, where the pitch angle is reported. In this case, the controller is not performing the tracking, as highlighted in Section 4.3. Moreover, adding uncertainty on the pitch angle offset $\theta_0$, it causes a greater error in absolute value, showing off the fragility of the controller.

The control brittleness is obviously reported also in Figure 4.32-4.34, where the velocities, input forces and frontal normal forces are shown.

### Rider mass

Another significant element to consider is the mass of the rider. Indeed, it is important to understand how the rider body could condition the control performance, being the

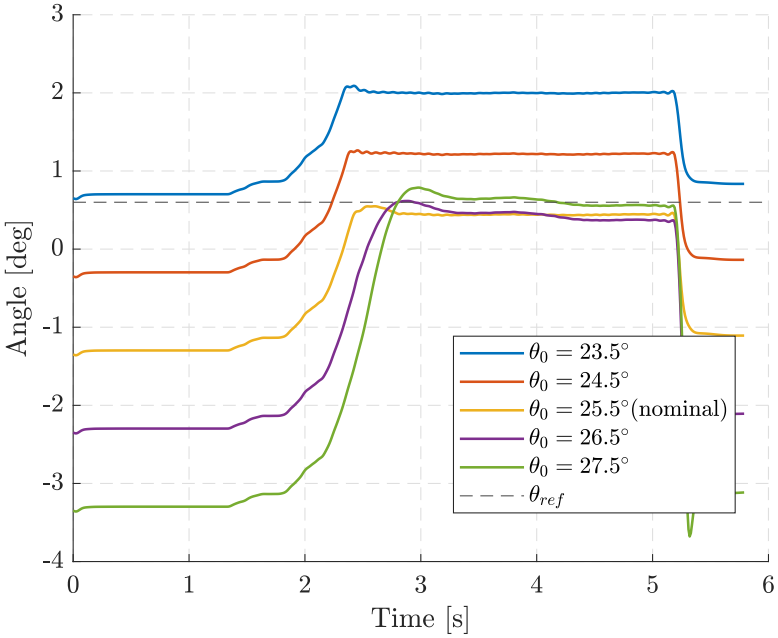


Figure 4.31: $Q_{\theta_N} \neq 0$: pitch angle with different pitch angle offset $\theta_0$

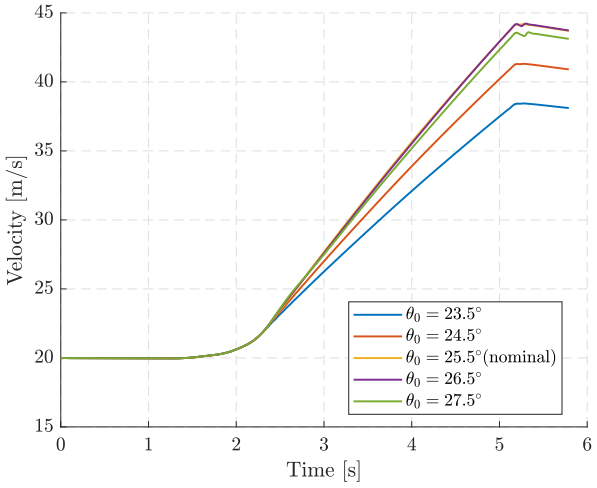


Figure 4.32: $Q_{\theta_N} \neq 0$: velocity with different pitch angle offset $\theta_0$

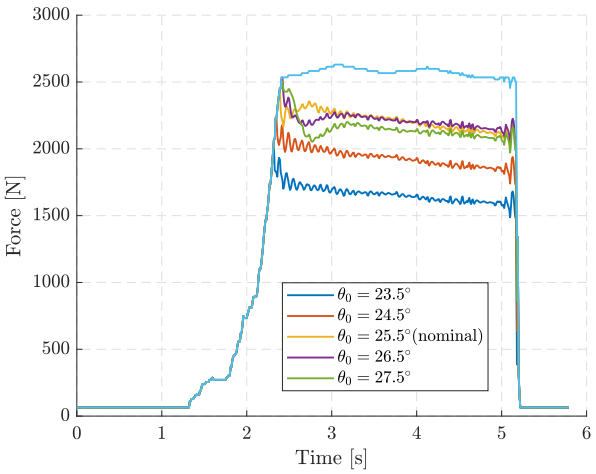


Figure 4.33: $Q_{\theta_N} \neq 0$: input force with different pitch angle offset $\theta_0$

implemented controller a model based control and the mass a peculiar characteristic of each pilot.

To carry out the analysis, different simulations are made. Specifically, the rider mass of the model used to evaluate the control input is changed in order to see the effects. The perfect models match is obtained with $m_{rider} = 70kg$, the other tests are made with $m_{rider} = 80kg, 90kg$ and $100kg$.

The tests are carried out with the same conditions of the previous section and the collected data are shown below.

Even if the rider mass is an important element to consider, this model uncertainty has fewer consequences than the $\theta_0$ parameter. Indeed, in case of $\theta$ tracking control, the simulation performances are similar one to each other, as depicted in Figure 4.36. Considering the control law designed in Section 4.4, it can be noticed how the tracking of

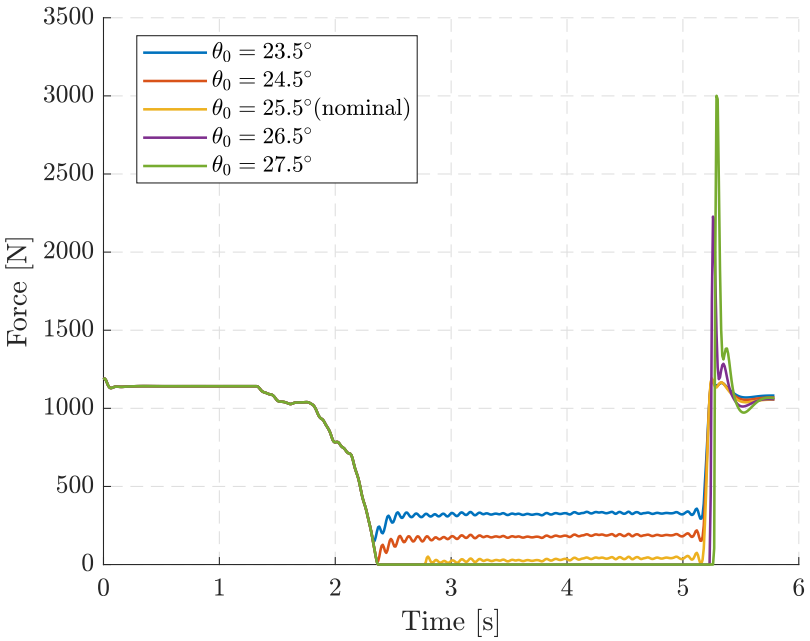


Figure 4.34: $Q_{\theta_N} \neq 0$: frontal normal force with different pitch angle offset $\theta_0$

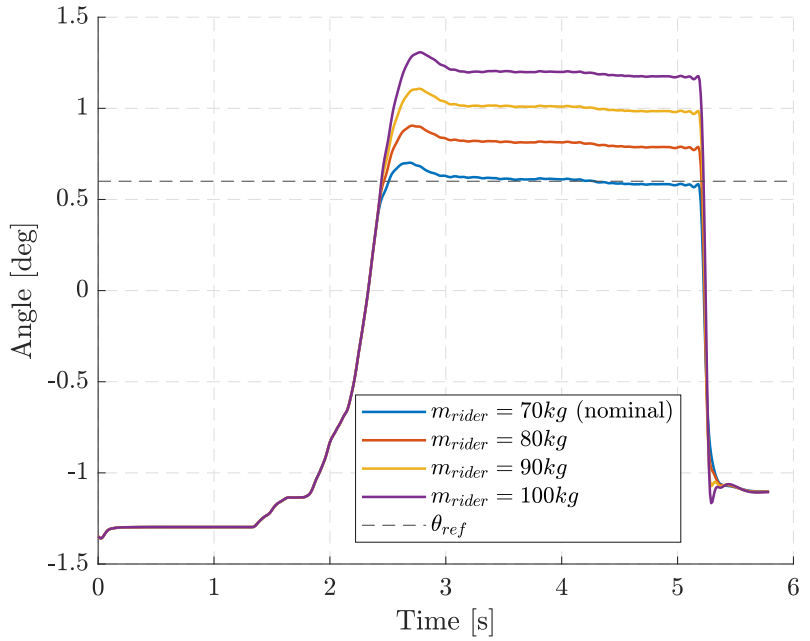


Figure 4.35: $Q_\theta \neq 0$ and $Q_{\theta_N} \neq 0$: tracking on $\theta$ with different rider mass

the $\theta$ signal slightly varies as pilot body does (Figure 4.35).

In this sense the models match is robust, looking at the mass rider parameter, this leads to satisfying performances for the motorcycle. For example, looking at Figure 4.36 and 4.37, it can be seen how the simulations results are similar also with different values of $m_{rider}$.

The Figures 4.38-4.39 depict what happens in the case of the controller with only the last stage weighted in cost function.

Also in this situation, the rider mass variation causes few effects in the motorcycle performances, with respect to the nominal case. Nevertheless, the results are worse than the case of the controller presented in Section 4.4, because the undesired phenomenon

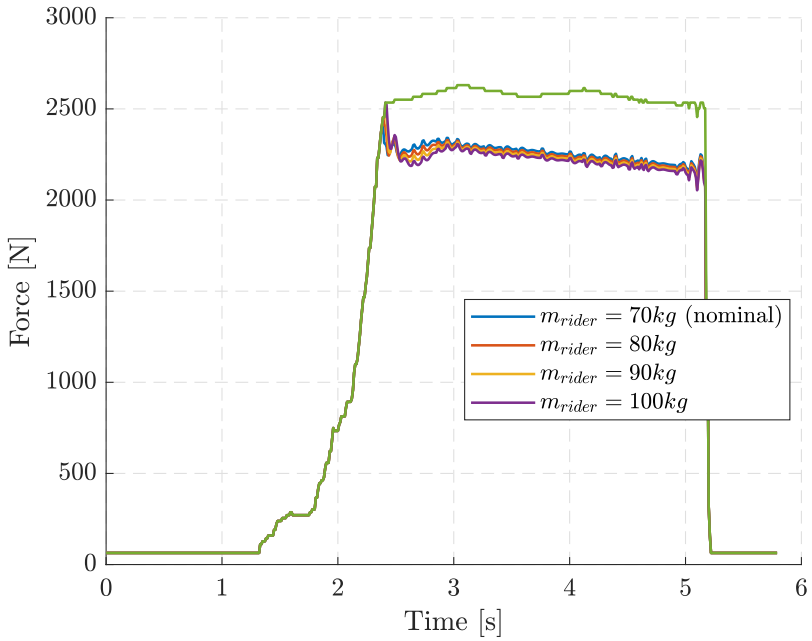


Figure 4.36: $Q_\theta \neq 0$ and $Q_{\theta_N} \neq 0$: input force with different rider mass

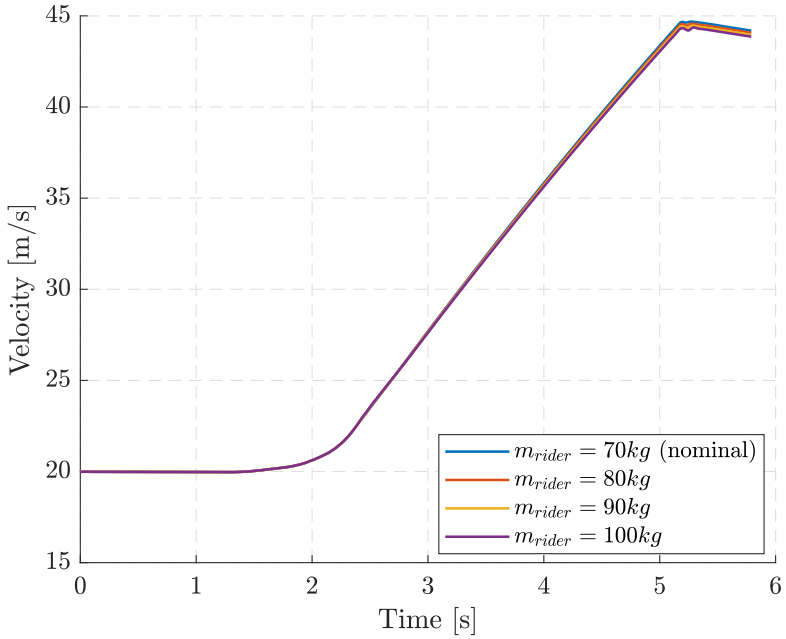


Figure 4.37: $Q_\theta \neq 0$ and $Q_{\theta_N} \neq 0$: velocity with different rider mass

depicted in the nominal case are revealed also in this situation. In particular, the pitch angle is not stabilized at a constant value, as in case of third scenario, and the tracking error is grater in absolute module.

Taking into account all the considerations made before, based on both the nominal and parameters variation cases, the controller developed in Section 4.4 is chosen. Therefore the control law that perform the tracking of the pitch angle will be used in the following Section to model the input time delay.

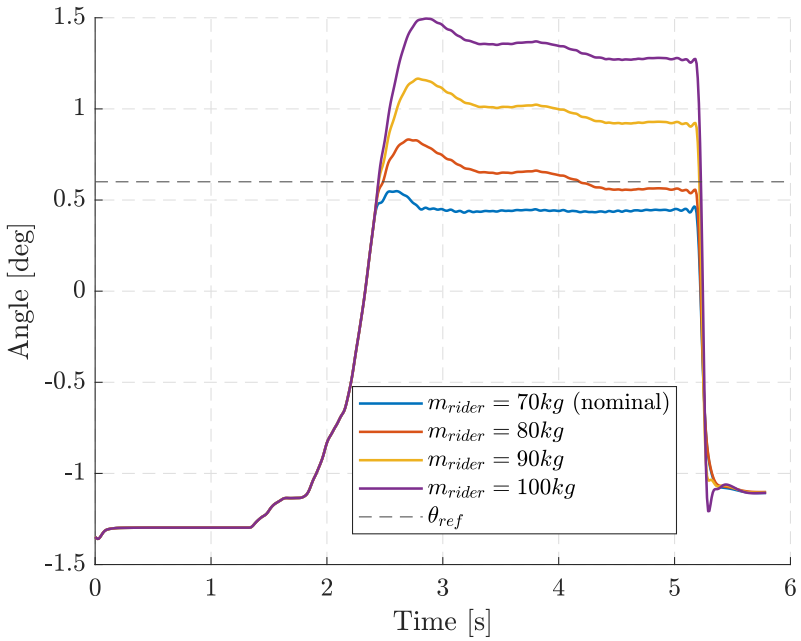


Figure 4.38: $Q_{\theta_N} \neq 0$: tracking on $\theta$ with different rider mass

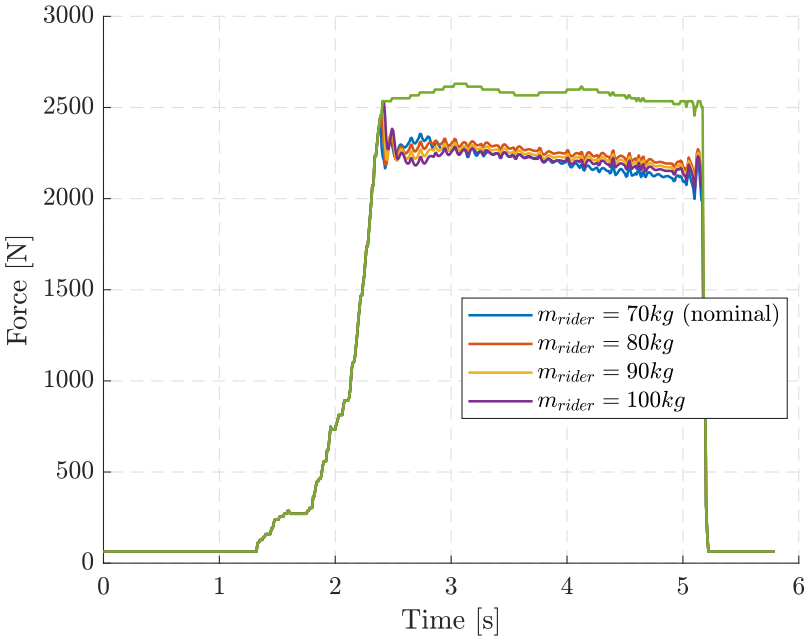


Figure 4.39: $Q_{\theta_N} \neq 0$: input force with different rider mass

## 4.6 Time delay

The delay caused by the electro-mechanical structure is modelled in Section 3.3, and it is of $t_{delay} \approx 100ms$.

To understand the effect of the delay derived in Section 3.3, a step response of the transfer functions related to Padè approximation with $T = 100ms$ and Butterwoth filter with a cut-off frequency set to $f = 7Hz$ is evaluated, and the results are shown in Figure 4.40.

Looking at this picture, it can be seen how the transients are not very smooth, in both

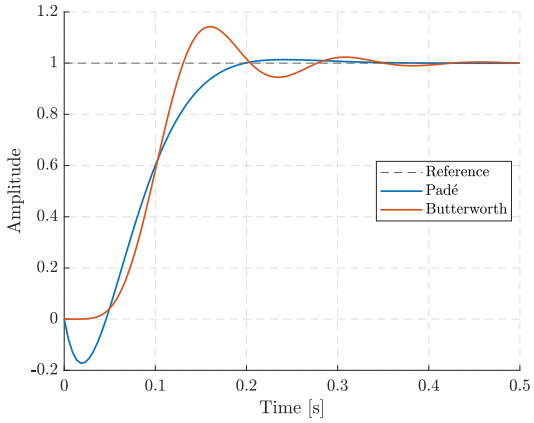


Figure 4.40: Step response: Padè $1^{st}$, $2^{nd}$ order and Butterworth $6^{th}$ order

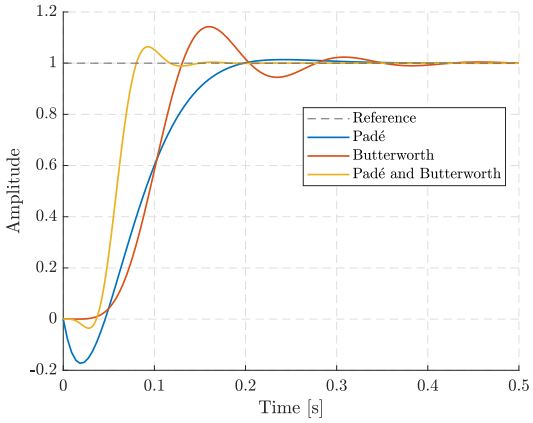


Figure 4.41: Step response: combination of time delay methods

cases. The Padè approximation makes an undershoot before reaching the desired value, while the low pass filter performs an overshoot, and then it sets the signal at the reference value. This is caused by the order of the transfer functions.

With the aim of becoming the step response smoother in the transients, a compromise in terms of overshoot and undershoot is considered. Therefore, the delay is built as a union of the two methods, as mentioned in the previous Chapter.

The two intermediate transfer functions that lead to the final time delay are constructed as before, with different values for $T$, the time delay in Padè approximation, and $f$, the cut-off frequency of the low-pass filter. Then, the mixed time delay is the product between:

- Padè approximation with denominator of $2^{nd}$ order and numerator of the $1^{st}$ order, with $T = 30ms$, and
- Butterworth filter of $4^{th}$ order, with $f = 15Hz$.

The step response of this delay formulation is shown in Figure 4.41, where it is compared with the previous ones. It can be observed how both the overshoot and the undershoot are limited. In this way, the modelled time delay is smaller than the previous cases, but is not a crucial aspect. This is due to the fact that the time delay is not measured in experimental context precisely.

With the aim of analysing the improvements obtained introducing the time delay compensation, different simulations are carried out.

At first attempt, the worse case is presented, it corresponds to the case in which the delay arises in the simulated part, without a compensation in the control phase.

Then, a simulation in which the controller predicts the delay on the input is considered.

Finally, the results related to the robustness tests are discussed.

### 4.6.1 Time delay only in simulation

In order to understand the effects of non consider the time delay in the control scheme designed before, a simulation is carried out. This test is made to observe the worse case performance results. In particular, the time delay is added only in the simulation part, but not in the model used to predict the motorcycle behaviour, hence it could be the actual scenario in case of unmodelled delay.

The simulation is performed as before. In detail, the implemented control law is the same depicted in Section 4.4. All the parameters are unchanged, in particular the weights matrices, the initial conditions and the cost function that has to be minimized. The prediction horizon and the soft constraint on $\theta$ are $N = 20$ and $\theta_M = 3°$ respectively. The difference is that the control scheme is applied on a motorcycle with a time delay introduced by the combination of Padè and Butterworth. This is possible adding the dynamics written in Eq. 3.12-3.13 only in the simulation code. This means that in the simulation scenario, six states $\gamma_{s_i}$ (where the letter $s$ stays for *simulation*) are added to the twelve known variables presented in Chapter 2.

In Figure 4.42 the time delay effect on the input force is shown. In this situation, where the delay is not inserted in the MATMPC model, the control leads to oscillatory movements.

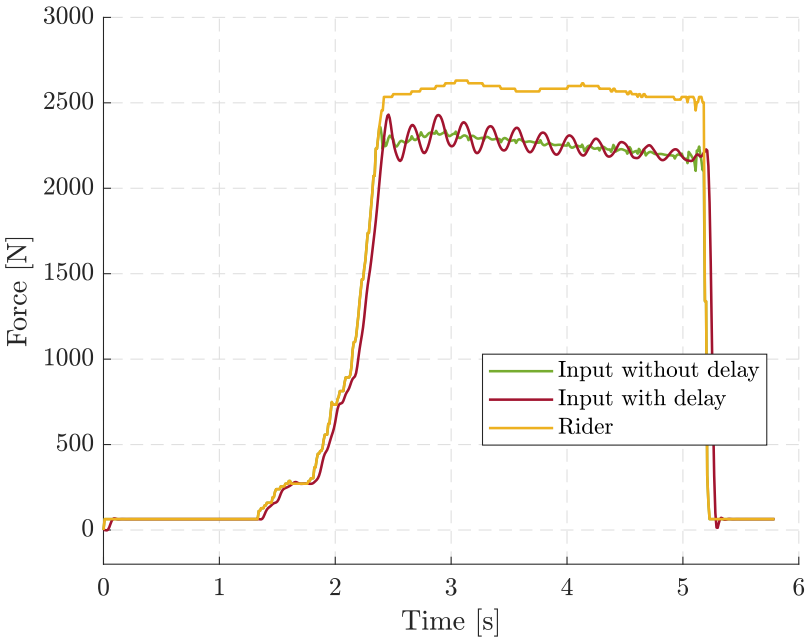


Figure 4.42: Input force with delay in simulation

This behaviour is caused by the fact that the controller acts predicting to arrive at the motorcycle at a determined instant, while the actual control input arrives at the plant with a delay. This obviously origins an expected evolution also in terms of pitch angle, as it can be noticed in Figure 4.43. It is maintained in the bounds and near the desired value $\theta_{ref}$, but a stabilization of the signal is not guaranteed.

The delayed action can be noticed also in Figures 4.44 and 4.45, where the rate of

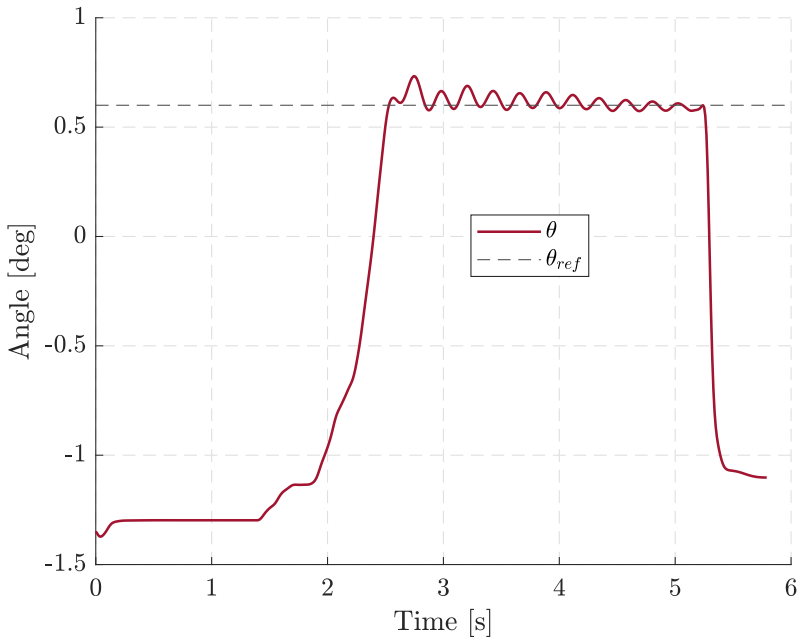


Figure 4.43: Pitch angle with delay in simulation

the pitch angle and the force state $\Delta_u$ are shown. Looking at these two figures, it can be appreciated how the signals are packed with spikes.

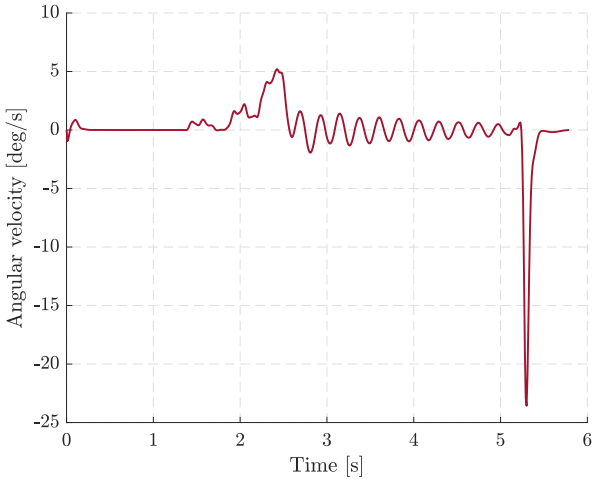


Figure 4.44: Angular velocity with delay in simulation

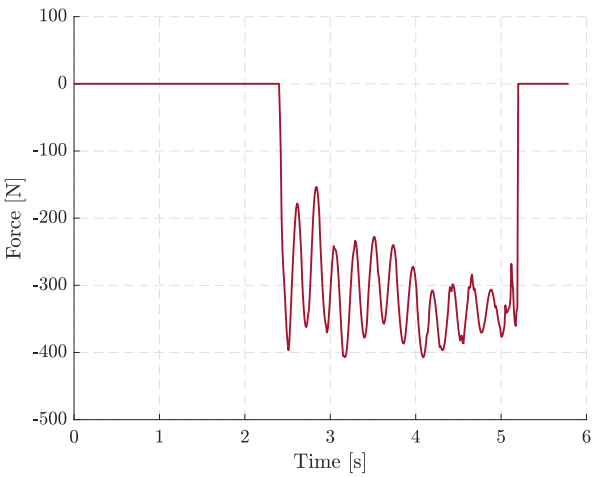


Figure 4.45: $\Delta_u$ state with delay in simulation

### 4.6.2 Time delay in the model

In order to avoid the phenomenon just described, the time delay is added in the single body model, which is used to predict the motorcycle behaviour and to evaluate the control action. In this way, the time delay will be compensated by the controller.

Predicting the time shift with which the control arrives to the vehicle, the pitch angle evolution will be stabilized at a constant value.

To put in practice this idea, the number of state variables becomes $n_{\boldsymbol{x}} = 10$. Indeed, six new states $\boldsymbol{\gamma}_i$, with $i = 1, \cdots, 6$, are stacked up to the single body states with the aim

of realize the transfer function $H_d$ through the state space matrices $\boldsymbol{A}_d, \boldsymbol{B}_b, \boldsymbol{C}_d$ and $\boldsymbol{D}_d$. Therefore, the system states are

$$\boldsymbol{x} = \begin{bmatrix} \theta \ \dot{\theta} \ \dot{x}_r \ \Delta_u \ \gamma_1 \ \gamma_2 \ \gamma_3 \ \gamma_4 \ \gamma_5 \ \gamma_6 \end{bmatrix}.$$

Exploiting the Eq. 3.12, the delayed control is evaluated, and it is considered as the control of the system, as suggested by Figure 3.1.

Also in this case, the simulations are performed with the multibody model, which does not have the $\boldsymbol{\gamma}_i$ dynamics, hence these states are evaluated not as output of the system, but integrating the variables from the MATMPC predicted states.

The tests are carried out following the procedure used before. The objective function minimized by MPC is the one reported in Eq.3.11. The new states that realize the delay transfer function are not considered in the cost function, because they have only the objective of model the time delay. Also the constraints and the desired signals values are the same. The first simulation is performed with $N = 20$ and the variable $\theta_M = 3°$. The weights matrices of the cost function are:

$$\boldsymbol{Q} = \begin{bmatrix} Q_\theta \\ Q_{\Delta_u} \end{bmatrix} = \begin{bmatrix} 1e13 \\ 1e-3 \end{bmatrix} \quad \boldsymbol{Q}_N = \begin{bmatrix} Q_{N_\theta} \\ Q_{N_{\Delta_u}} \end{bmatrix} = \begin{bmatrix} 1e13 \\ 1e-3 \end{bmatrix} \quad \boldsymbol{R} = \begin{bmatrix} R_u \\ R_\epsilon \end{bmatrix} = \begin{bmatrix} 1e2 \\ 1e15 \end{bmatrix}.$$

The results are shown in the Figures below. In Figure 4.46, it can be seen how the controller reacts in case of time delay compensation. The red-violet line represents the input force that arrives at the motorcycle, which is the postponed control input obtained starting from the blue one. It can be noticed how the time delay modelling is useful to avoid oscillatory behaviour.

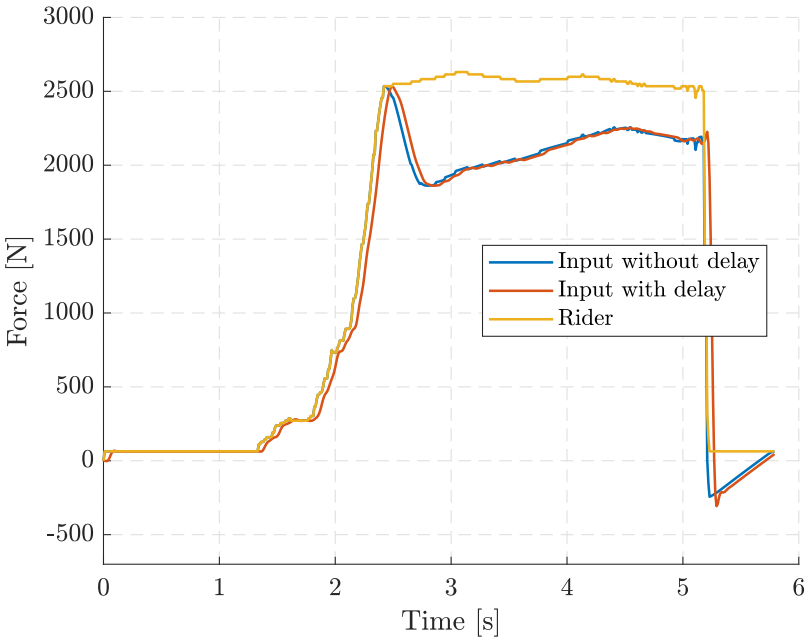


Figure 4.46: Input force with time delay and N=20

Looking at the transients of Figure 4.46, the smoothness of the delayed input can be appreciated. It is a consequence of the small overshoot and undershoot shown in Figure 4.41.

This is also observable in Figure 4.47. However, the tracking of the $\theta$ angle is no more performed. This especially happens in the transient, where the pitch angle remains in the set bounds but with higher values with respect to the case of Section 4.4.

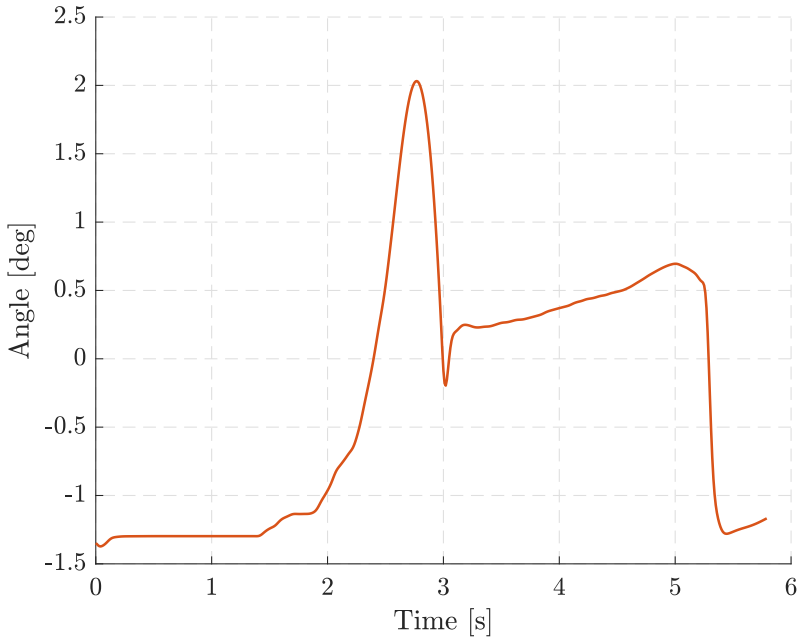


Figure 4.47: Pitch angle with time delay and N=20

This behaviour is also dependent on the prediction horizon, indeed a greater value of $N$ could help in this sense.

At this aim the value $N$ is increased to $N = 50$. With $N$ value, other changes are made in terms of the objective function weights matrices, which are become:

$$\boldsymbol{Q} = \begin{bmatrix} Q_\theta \\ Q_{\Delta_u} \end{bmatrix} = \begin{bmatrix} 1e10 \\ 1e-7 \end{bmatrix} \quad \boldsymbol{Q}_N = \begin{bmatrix} Q_{N_\theta} \\ Q_{N_{\Delta_u}} \end{bmatrix} = \begin{bmatrix} 1e10 \\ 1e-7 \end{bmatrix} \quad \boldsymbol{R} = \begin{bmatrix} R_u \\ R_\epsilon \end{bmatrix} = \begin{bmatrix} 1 \\ 1e15 \end{bmatrix}.$$

The related simulation data are reported below. In particular, in Figure 4.48 the input force is reported and the improvements in terms of the input thrust can be highlighted.

Unlike the case of $N = 20$, in this simulation the controller is capable of handle the delay maintaining a great value of input force.

This obviously leads to a smaller spike in the transient of the pitch angle evolution, which is reported in Figure 4.49.

The prediction horizon $N = 50$, in addition to guaranteeing a satisfying input force, allows $\Delta_u$ to intervene in a softer way. It can be observed looking at Figure 4.50, where the $\Delta_u$ action of the two cases is compared.

Even if a smaller prediction horizon guarantees an imperfect performance, it is preferred because of the time demanded to solve the problem in MATMPC.

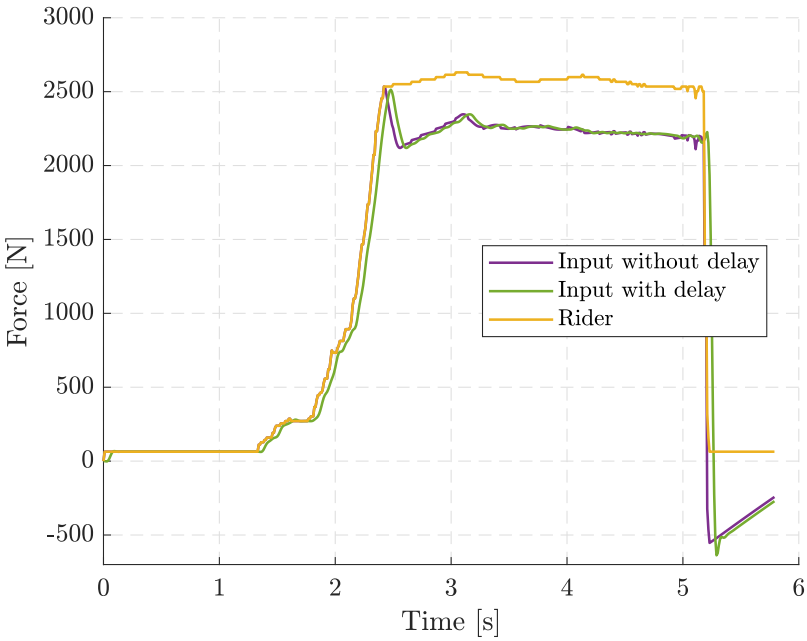


Figure 4.48: Input force with time delay and N=50

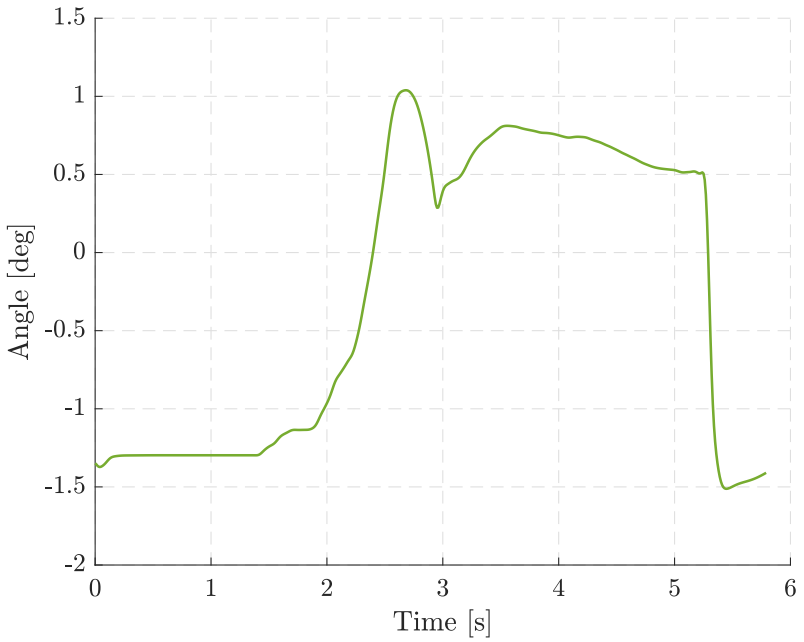


Figure 4.49: Pitch angle with time delay and N=50

Looking at the computational time quantities noted in Tab. 4.6, it can be noticed how the cases of $N = 50$ is more expensive in terms of time request.

| N | CPT avg | CPT max | QP avg | QP max |
|---|---|---|---|---|
| 20 | 1.88 | 15.03 | 0.24 | 0.73 |
| 50 | 9.09 | 29.21 | 2.48 | 23.67 |

Table 4.6: Time effort comparison between $N = 20$ and $N = 50$

Summarizing, the insertion of the time delay in the model can guarantee an appreciable performance in case of input delay, because it can predict the time shift introduced by

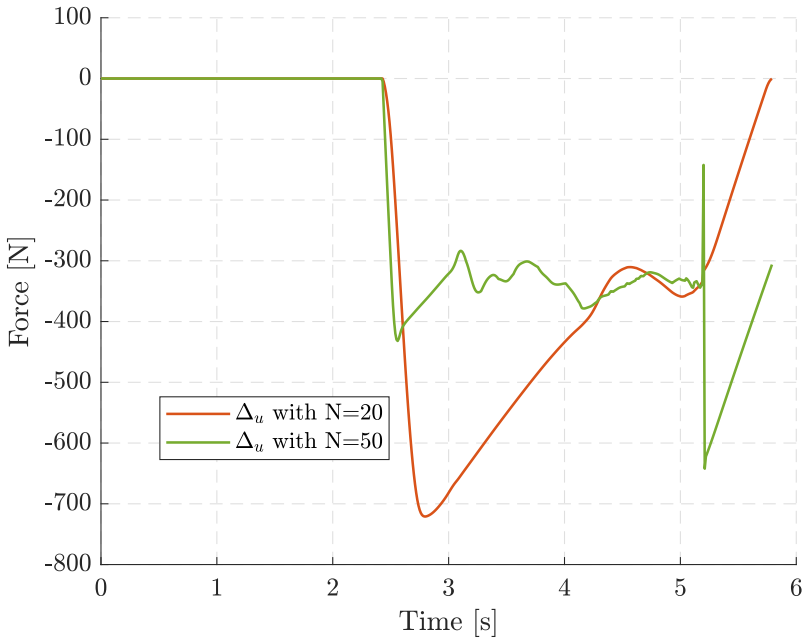


Figure 4.50: $\Delta_u$ state with $N = 20$ and $N = 50$

the electro-mechanical structure of the motorcycle. Working with a real-time control technique, the scenario with a smaller prediction horizon is chosen.

### 4.6.3 Time delay sensitivity

In this section, the sensitivity of the time delay model is tested. The time delay with which the input arrives at the motorcycle is not measured precisely in an experimental context. Therefore, it is important to analyse how the model can react in case of a non optimum estimate of the delay quantity.

Following this reasoning, the model depicted in the previous section is tested with a different delay in simulation.

In particular, the Figure 4.51 shows the used delays. With the blue line the delay inserted in the model is noted, while the tester delays are drawn with orange and yellow lines. The yellow signal (called Padè and Butterworth 2) is a delay that anticipates the time delay predicted by the model, whereas the orange line (named Padè and Butterworth 1) is delayed further with respect to the modelled delay.

The postponed signals used to prove the control robustness shown in Figure 4.51 are obtained as a combination of Padè approximation and Butterworth filter. They are computed setting the $T$ (time delay) and $f$ (cut-off frequency) at different values with respect to the value selected before.

In detail, the two delays are:

- Padè and Butterworth 1: Padè approximation with denominator of $2^{nd}$ order and numerator of the $1^{st}$ order, with $T = 40ms$, and Butterworth filter of $4^{th}$ order, with $f = 12Hz$;

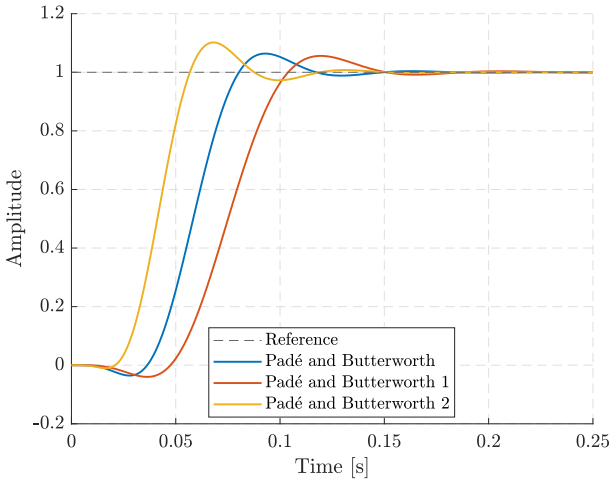


Figure 4.51: Step response: delay methods to test the delay sensitivity

- Padè and Butterworth 2: Padè approximation with denominator of $2^{nd}$ order and numerator of the $1^{st}$ order, with $T = 15ms$, and Butterworth filter of $4^{th}$ order, with $f = 17Hz$.

To build different delay in model and in simulation, two different transfer functions $H_d$ are constructed. The delay inserted in the model is realized thanks to $\boldsymbol{\gamma}$ states, while the tester delays are built applying Eq. 3.12-3.13 to the states $\boldsymbol{\gamma}_s$. The variables $\gamma_{s_i}$ and $\gamma_i$ are maintained independent of one another, to carry out the desired tests and avoiding a mixture of different dynamics.

### A greater simulated delay with respect to the modelled delay compensation

The first test is made with

- Padè and Butterworth in model: Padè approximation with denominator of $2^{nd}$ order and numerator of the $1^{st}$ order, with $T = 30ms$, and Butterworth filter of $4^{th}$ order, with $f = 15Hz$.

- Padè and Butterworth 1 in simulation: Padè approximation with denominator of $2^{nd}$ order and numerator of the $1^{st}$ order, with $T = 40ms$, and Butterworth filter of $4^{th}$ order, with $f = 12Hz$.

The simulation is carried out with the parameters described in the previous section. In particular the prediction horizon is $N = 20$ and the weights matrices of the objective function are unchanged.

The collected data are reported below. In Figure 4.52 the drawn input force is coherent with which in Figure 4.46. Indeed, the curve has the same behaviour, but the delay between the two line is slightly greater.

Also the pitch angle evolution (Figure 4.53) is very similar to that in Figure 4.47.

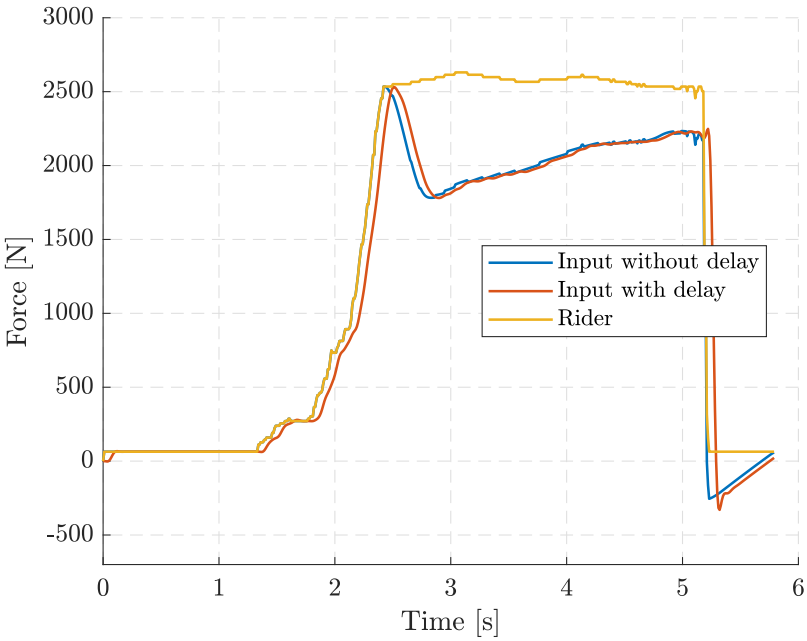


Figure 4.52: Time delay sensitivity test 1: input force

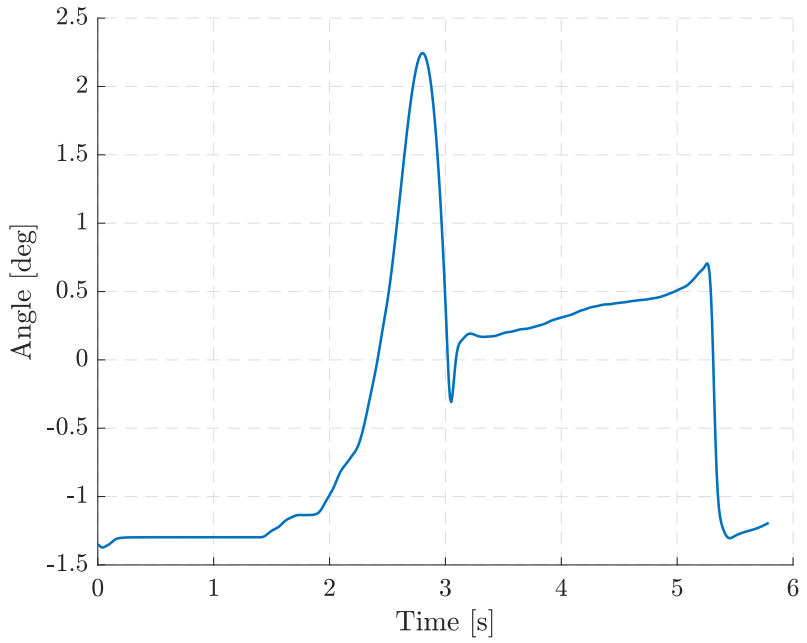


Figure 4.53: Time delay sensitivity test 1: pitch angle

Unfortunately, the gap between the predicted and the real delay leads to a bigger spike in the transient of the pitch angle. The control acts after then what it has supposed, therefore $\Delta_u$ state intervenes with a strong action to compensate. Being the constraints $\theta_M = 3°$, the MPC is successful in these terms because the angle $\theta$ is maintained far from this bound.

**A smaller simulated delay with respect to the modelled delay compensation**

The second validation is carried out with

- Padè and Butterworth in model: Padè approximation with denominator of $2^{nd}$ order

and numerator of the $1^{st}$ order, with $T = 30ms$, and Butterworth filter of $4^{th}$ order, with $f = 15Hz$.

- Padè and Butterworth 2 in simulation: Padè approximation with denominator of $2^{nd}$ order and numerator of the $1^{st}$ order, with $T = 15ms$, and Butterworth filter of $4^{th}$ order, with $f = 17Hz$.

In this case, the unique change is the time delay in simulation. The results related to the input force are presented in Figure 4.54.

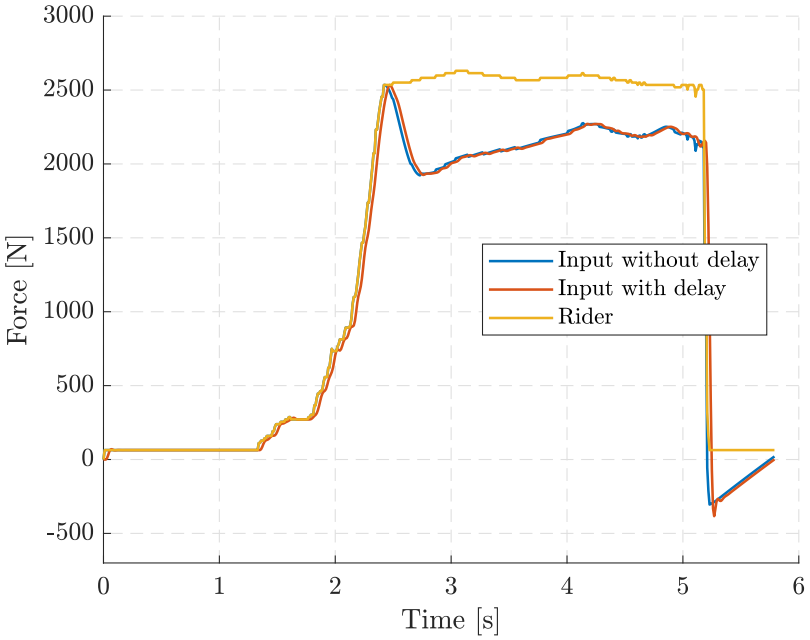


Figure 4.54: Time delay sensitivity test 2: input force

It can be seen how the delay is smaller in this situation. The other collected data are shown compared with the case of the previous section and with that at in which there is a perfect match in delays model-simulation.

In Figures 4.55 and 4.56, it can be seen how the delay implemented in the model is robust at the parameters variation. In particular, the behaviour of the motorcycle is very similar to that of the perfect models match. It can be observed how a faster input guarantees a softer action of $\Delta_u$. Then it obviously leads to a better evolution of the pitch angle $\theta$.

The crucial parts of the signals evolution are those in the transients. In these instants the effects of the model mismatches are highlighted. For example, it can be remarked in Figure 4.56, when the $\Delta_u$ state is returning to be zero.

In Table 4.7 the computational time efforts are noted. As expected, the three cases demand the same quantity of time to solve the related control problems.

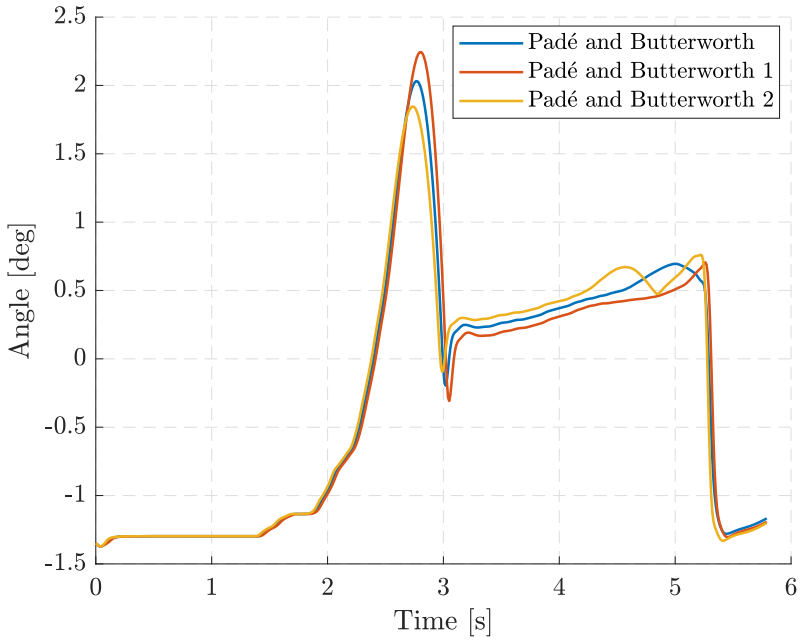


Figure 4.55: Time delay sensitivity tests: pitch angle

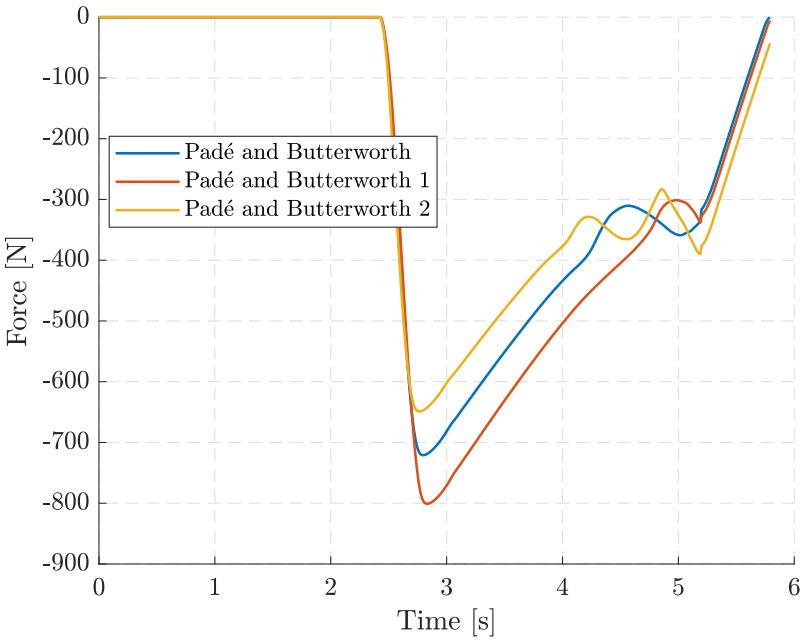


Figure 4.56: Time delay sensitivity tests: $\Delta_u$ state

| Delay in simulation | CPT avg | CPT max | QP avg | QP max |
|---|---|---|---|---|
| Padè and Butterworth | 1.88 | 15.03 | 0.24 | 0.73 |
| Padè and Butterworth 1 | 1.95 | 12.25 | 0.25 | 0.72 |
| Padè and Butterworth 2 | 1.93 | 12.36 | 0.25 | 1.02 |

Table 4.7: Time delay sensitivity tests: time effort comparison

# Chapter 5

# Experimental Tests

In this Chapter, the experimental tests are made to validate the control law described in Section 4.2. In particular, the experiment is made on RSV4 Aprilia motorcycle. First, a brief introduction on the experimental background is made. Then, the collected data are shown and discussed.

## 5.1 Experimental context

With the aim of collecting data similar to race conditions, the control law is tested on a racetrack. The first attempt is carried out on a flat ground, with large distances of straight road. In this way, no difficulties on the environment interaction would arise, and the rider is free to accelerate.

As depicted above, the control is designed on MATMPC, hence exploiting MATLAB and MEX functions. It is based on the single body model presented before and tested on the Aprilia motorcycle. To bring the controller on the real motorcycle, the electronics field is exploited. In particular, the code is totally written in C code, which is the programming language of the motorcycle control unit (CU).

The control unit minimizes the objective function exploiting the HPIPM solver [12]. HPIPM is a High-Performance Interior-Point solver, it provides a solution to small and medium scale problems arising in MPC and in embedded optimization. Therefore, it is chosen because of its computational time and its compatibility with the CU on board the motorcycle.

Moreover, the motorbike is equipped with a set of sensors. In particular, an IMU is used to collect the telemetries data. The Inertial Measurement Unit (IMU) is an electronic device that measures and store the angular rate, the accelerations and orientations. This is possible thanks to a combination of accelerometers and gyroscopes. In particular, on board the vehicle used in the experimental part, it is possible to read the pitch angle value

and its derivative, the motorcycle velocity and the frontal suspension elongation.

## 5.2 Experimental data

The experimental evaluation is made with the control law depicted in Section 4.2. Even if it is not the definitive designed control scheme, it is tested on the motorcycle in order to see how it works.

Therefore, the problem minimized by MPC is of the form reported in Eq. 3.3, which is subject to :

$$-10000 \leq u \leq 10000 \tag{5.1}$$

$$-3000 \leq \Delta_u \leq 0 \tag{5.2}$$

$$\epsilon \geq 0 \tag{5.3}$$

The pitch angle is characterized by relaxed bounds, hence it is constrained as reported in Eq. 5.4.

$$\theta \leq \theta_M \quad \text{with} \quad \theta_M = 3^\circ \tag{5.4}$$

The objective function is weighted with the following matrices:

$$Q = Q_N = 5 \quad \boldsymbol{R} = \begin{bmatrix} R_u \\ R_\epsilon \end{bmatrix} = \begin{bmatrix} 5 \\ 5e11 \end{bmatrix}.$$

Looking at the weights matrices, it can be noticed how the slack variable $\epsilon$ is accommodating, with respect to the simulations reported in the previous Chapter.

The prediction horizon is set to $N = 10$, and the problem is solved in *real-time* setup. In the following subsections, some collected data are presented. A remark has to be written with respect to the data illustrated below. Indeed, with the aim of guaranteeing the privacy on the Aprilia research, all the data are normalized in the interval [0:1].

Moreover, in the following sections the collected data are reported. In particular, three manoeuvres will be discussed, which order of presentation in not relevant.

#### Progressive manoeuvre scenario

The results correspondent to a progressive rider manoeuvre are reported below. For simplicity, this scenario will be named first test. Looking at Figure 5.1, it can be noticed how the rider demand reaches the maximum torque, with a progressive manoeuvre. The control acts with a soft action, to maintain the motorcycle low.

Around the instant 0.6 and 0.8, highlighted by the dashed line, a gearshift takes place. This factor is reported in all the collected data (Figures 5.2-5.4), which report spikes in this time intervals.

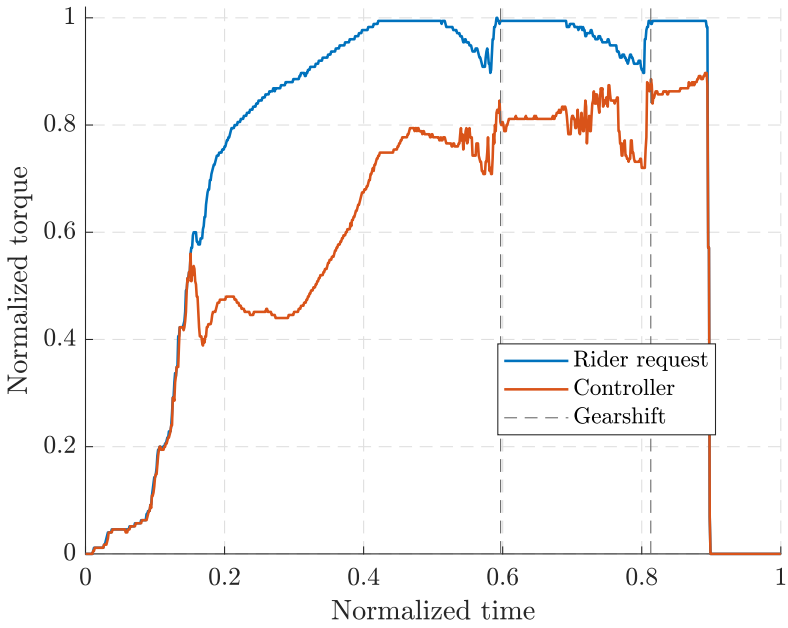


Figure 5.1: Experimental test 1: Normalized input torque

In Figure 5.2 the wheels velocities can be observed. In general, the rear wheel has a greater velocity with respect to the front one. This is due to the fact that the rear wheel accelerates more, generating slip and causing motorcycle movement. This leads to a non-overlap of the two curves in Figure 5.2.

The comparison between the front and rear wheels velocity is a method to understand if the motorcycle is wheeling. In particular, when a wheelie occurs the front wheel is lifting, causing its deceleration. On the contrary, the rear wheel acceleration does not have variations.

This phenomenon can be noticed in the time interval $[0.45 - 0.55]$ of Figure 5.2.

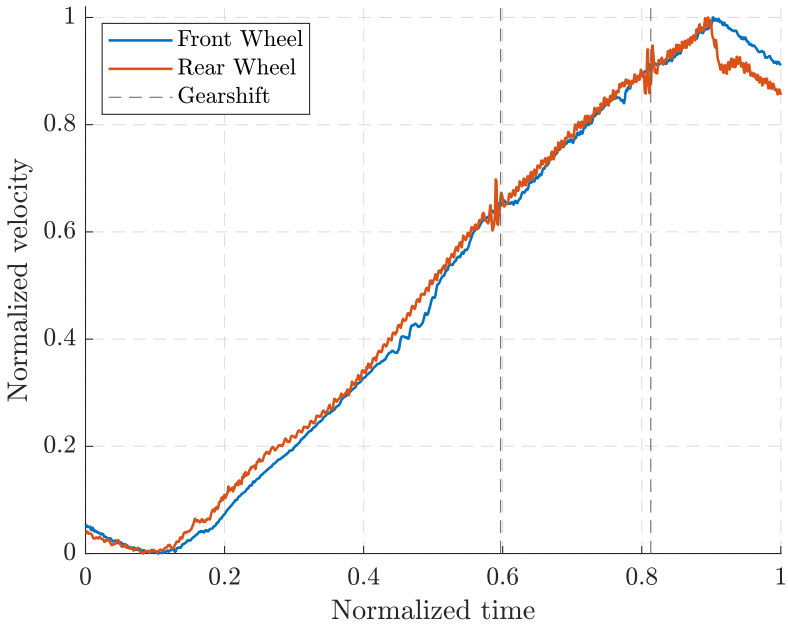


Figure 5.2: Experimental test 1: Normalized wheels velocities

Also in Figure 5.3 it is possible to observe the wheelie. In the simulations shown in

Chapter 4, the frontal normal forces are used to detect wheelies, but in the experimental context is not possible to read the normal force values. This lack can be filled looking at the compression of the frontal suspension.

In particular, great values in Figure 5.3 are caused by a compression on the ground, while small values correspond to a wheelie condition.

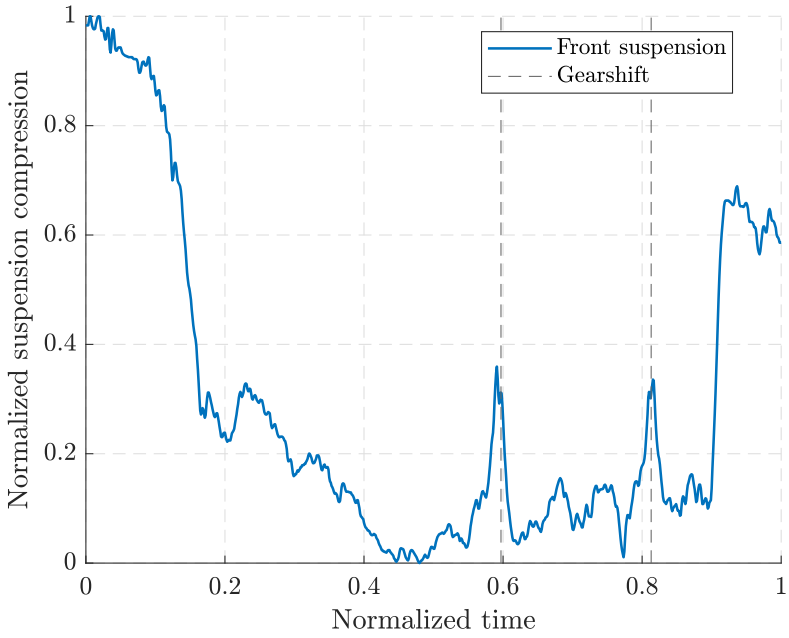


Figure 5.3: Experimental test 1: Normalized front suspension

The pitch angle is reported in Figure 5.4. Being the data normalized, is not easy to understand when the pitch angle becomes greater enough to lift the front wheel from the ground. Furthermore, the angle values detected by the IMU includes also the incline of the ground, hence if the track in not perfectly flat the pitch angle data could be not so useful.

### Preservative control input scenario

In this Section the second test data are discussed. This name is used for conciseness, and it describes a scenario with a preservative control input.

In the Figures related to the second validation, the control acts in a softer way with respect to the first experimental validation. This allows to maintain a greater input torque during the entire manoeuvre (Figure 5.5).

Also in this case, the rider makes a gearshift, as it can be appreciated in Figures 5.5-5.8, in which the instant of gearshift is noted with the dashed line.

Looking at Figure 5.6, a wheelie can be detected around the normalized instant 0.8, because in these interval the front suspension is elongated.

The motorcycle wheeling is noticed also in Figure 5.7. As mentioned before, this phenomenon is characterized by a gap between the two wheels velocities. In this case it happens around the instant referred above.

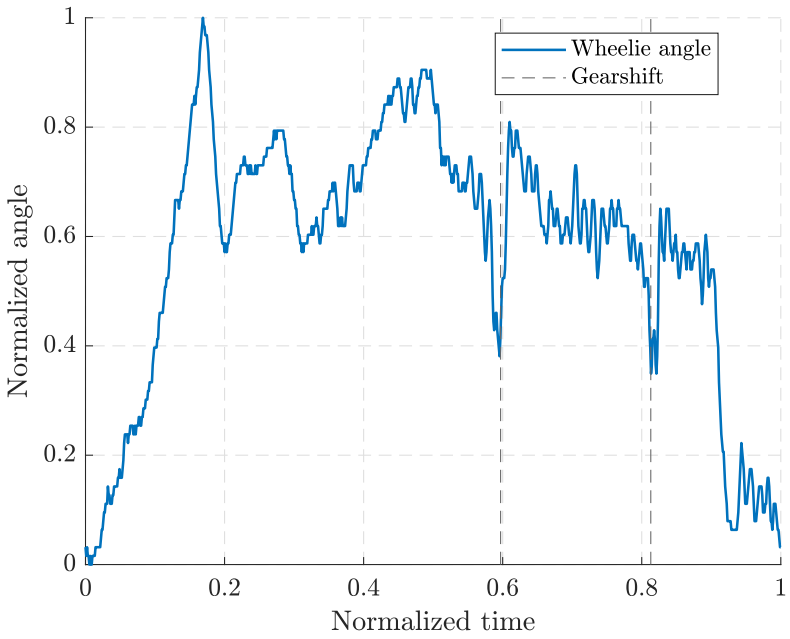


Figure 5.4: Experimental test 1: Normalized pitch angle

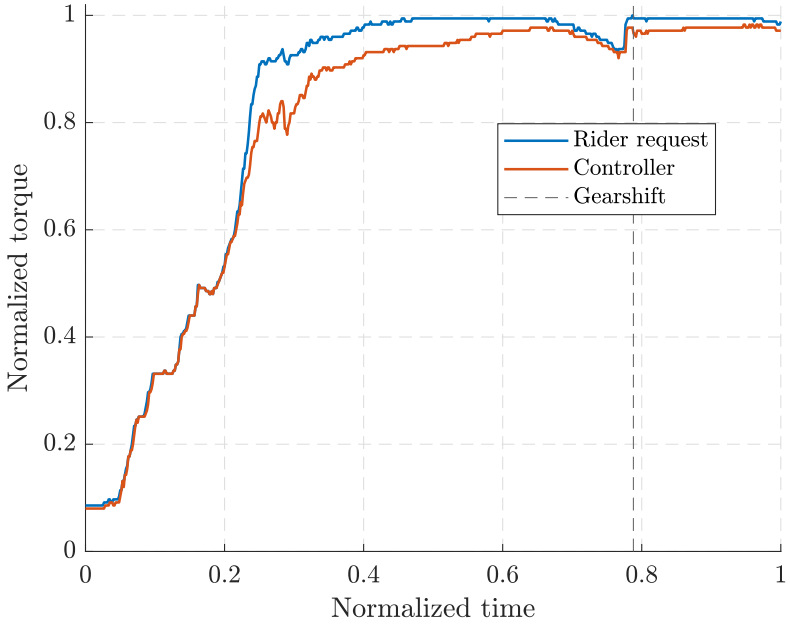


Figure 5.5: Experimental test 2: Normalized input torque

The Figure 5.8 depicts the wheelie angle. Even if the signal is normalized, it can be noticed how it is varying fast, this could be a consequence of the racetrack layout.

### Full throttle manoeuvre scenario

In this scenario, named third test for simplicity, the maximum torque value is obtained with a full throttle manoeuvre.

The third test represents a longer manoeuvre, with three gearshifts, highlighted by the dashed lines.

In this situations, the torque reaches its maximum value and the rider changes gear, maintaining the motorcycle at high velocities for the entire manoeuvre.

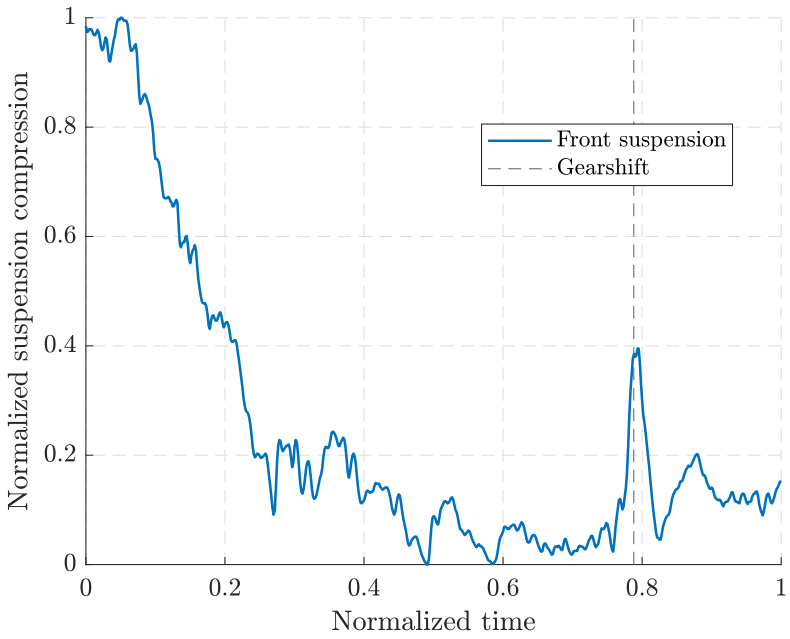


Figure 5.6: Experimental test 2: Normalized front suspension

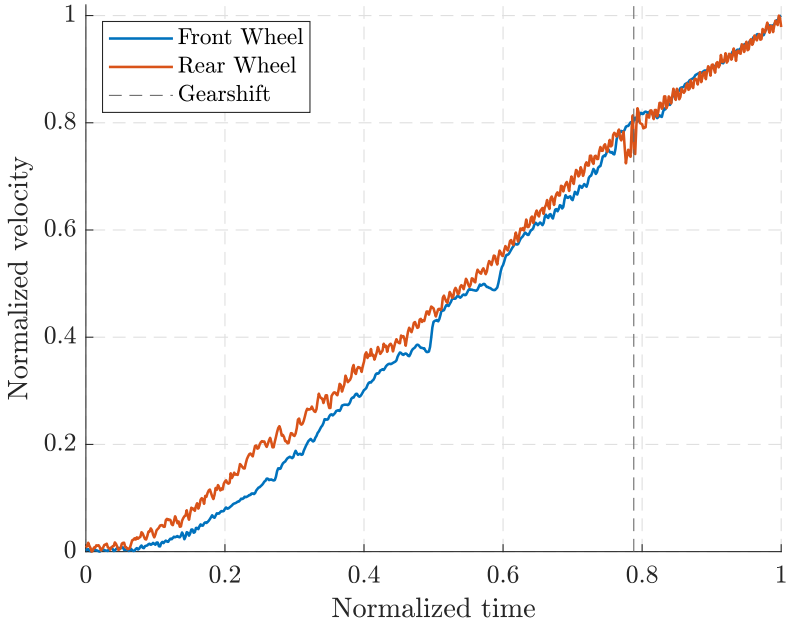


Figure 5.7: Experimental test 2: Normalized wheels velocities

Looking at Figure 5.9, it can be observed how the controller tends to preserve the rider demand, acting only in the first part of the test.

The performed gearshifts are clearly reported in Figure 5.10, where the signal of the front suspension shows off great spikes. In general, in this test, the motorcycle is not braking. It can be deduced both looking at Figure 5.9 and 5.10, where the compression values of the suspension are very small for the main part of the test.

In this case, a wheelie can be detected around the normalized instant 0.3, where the front wheel velocity decreases (Figure 5.11).

This evaluation is confirmed by the normalized compression of the suspension (Figure 5.10) and by the input torque (Figure 5.9). Indeed, the controller intervenes to limit the

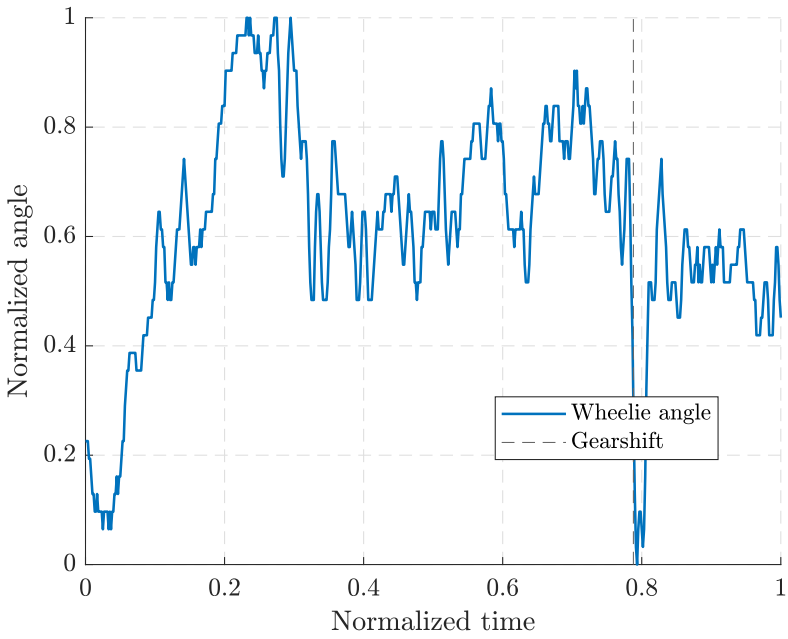


Figure 5.8: Experimental test 2: Normalized pitch angle

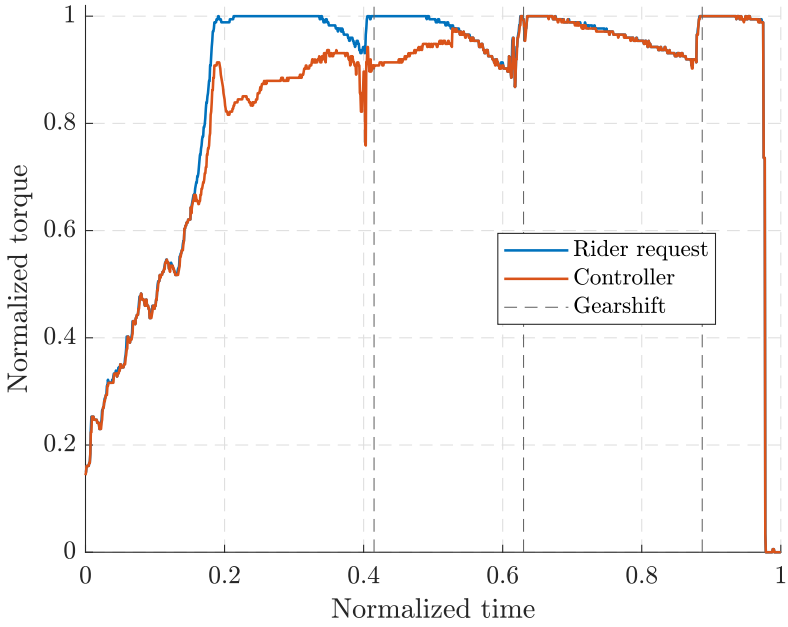


Figure 5.9: Experimental test 3: Normalized input torque

rider request only in the initial part. This happens because the slack variable of the soft constraint set on the pitch angle is few weighted in cost function, letting the wheel angle more free than the simulations cases depicted in the previous chapter.

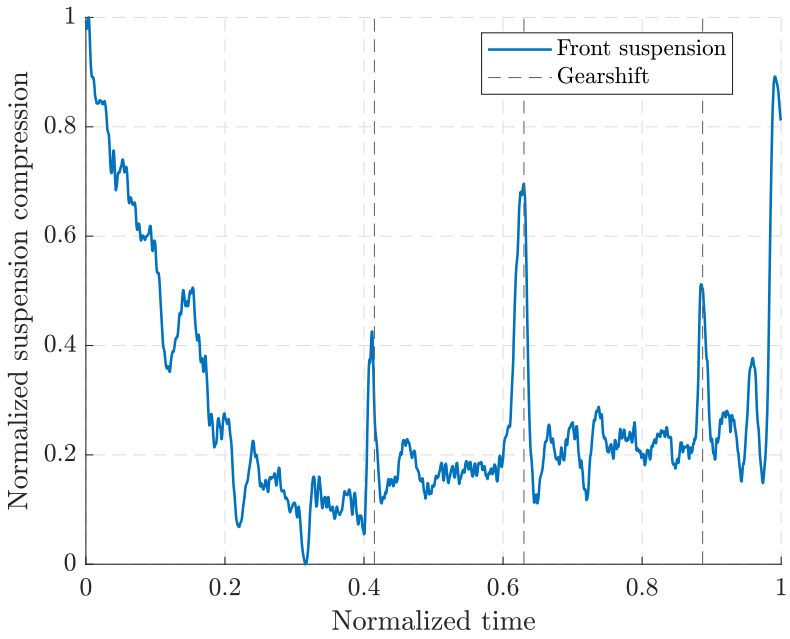


Figure 5.10: Experimental test 3: Normalized front suspension

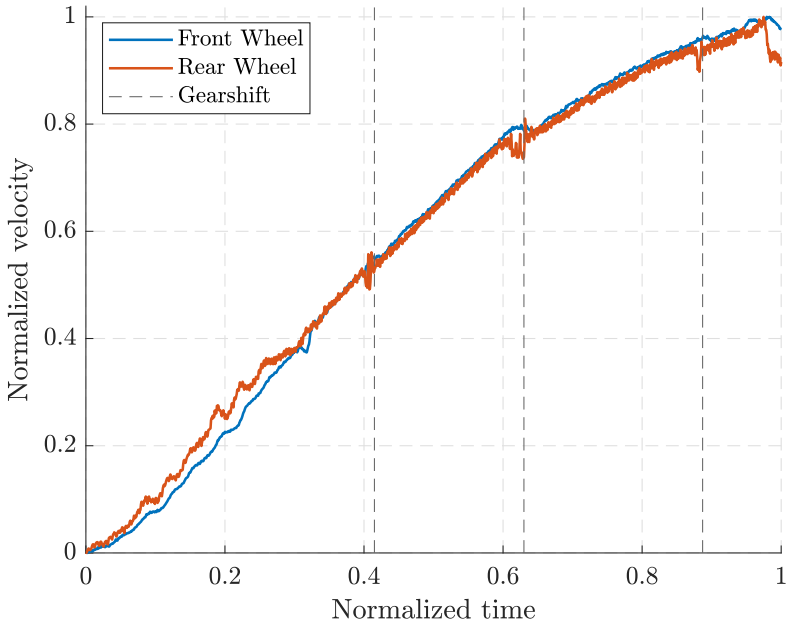


Figure 5.11: Experimental test 3: Normalized wheels velocities

# Chapter 6

# Conclusions

This thesis was born to describe how a model-based predictive control can avoid wheelies in high performance motorcycle.

Unlike the autonomous driving, the controller studied in this project interacts with the pilot demand. Moreover, the anti-wheelie techniques already developed act once the front wheel is lift from the ground, while the designed control wants to prevent this phenomenon. For this reason, a model-based control is needed.

Working in a real-time high-performance scenario and aiming to describe the real process conditions, an MPC is selected. This design choice is made to face the physical limits and the desired performance imposed by the context.

Therefore, as first step, a suitable prototype of a motorcycle is required.

The utilised models are illustrated in the Chapter 2, after introducing the NMPC theory in Chapter 1. The complete model is described also by non accessible variables, hence it is only used to test the controller, which is built exploiting the simplified motorcycle model.

The Chapter 3 shows how the procedure of designing control is developed. The first version of the control law regulate the pitch angle only via soft constraints, and it lead to an evolution sensitive to the constraints imposed on the pitch angle. The idea based on letting as free as possible the pitch angle does not correspond to the aimed robust solution.

Consequently, other versions of this control scheme are tested, finding the best form. In particular, the definitive control scheme is able to track a pitch angle reference that guarantees maximal acceleration, also handling the rider request.

Finally, with the scope of studying the situation closest to the actual one, a time delay is taken into account. Normally, the input arises to the motorcycle with a time delay, caused by the motorcycle structure.

It can be conclude that control law is capable to compensate a postponed input,

avoiding undesired motorcycle behaviour.

The most of the project is carried out in a simulated scenario, as shown in Chapter 4, where the MATMPC software is used to implement and validate the controllers.

The last Chapter of the thesis is focused on the experimental validation carried out on the RSV4 Aprilia motorcycle. The control framework depicted in Section 4.2 is evaluated on a real test vehicle. As discussed in Chapter 5, where the experimental collected data are presented, the control acts to avoid wheeling, but it turned out to be very preservative with respect to the rider command.

## 6.1 Future works

Even if the experimental test are performed with a first version of the designed controller, it is a great starting point for anti-wheelie control. Indeed, it is innovative, capable to handle the rider request and to predict the motorcycle behaviour. It would be interesting to test the definitive control scheme evaluated in Section 4.4 on the real motorcycle, in order to see what advantages can be introduced.

Since the controller is based on the motorcycle prototype, analysis on the parameters variation is carried out. In particular, it is seen how the controller reacts changing the rider mass or the offset on the pitch angle. The performed tests highlighted how a non zeroed tracking error arises in these cases. For this reason, an important improvement to introduce in the control law is the integral action. It could solve all the problems caused by the models mismatches, which are crucial, since the designed controller is model-based.

A notable remark is that the IMU set on board the motorcycle provides the pitch angle measure, namely the angle between the main frame and the road. In the actual case, this means that the measured signal includes also the slope of the hill. Therefore, an estimator for the slope of the ground would be an important contribution for this control scheme, it could guarantee great accelerations in terms of input force even if the motorcycle was uphill.

# Bibliography


[1] A Surana, J Jeffs, and TQ Dinh. "Accident prevention in motorcycles with 3 dimensional fuzzy logic traction control system". In: *2020 8th International Conference on Control, Mechatronics and Automation (ICCMA)*. IEEE, Nov. 2020. DOI: 10.1109/iccma51325.2020.9301500.

[2] Matteo Corno, Giulio Panzani, and Sergio M. Savaresi. "Traction-Control-Oriented State Estimation for Motorcycles". In: *IEEE Transactions on Control Systems Technology* 21.6 (Nov. 2013), pp. 2400–2407. DOI: 10.1109/tcst.2013.2238539.

[3] Kawasaki Jukogyo Kabushiki Kaisha. "Wheelie suppressing control unit". Mar. 2021. URL: https://patents.google.com/patent/US20210061099A1/en.

[4] Robert Bosch GmbH. "Wheelie controller and control method thereof". Jan. 2022. URL: https://patents.google.com/patent/US11230272B2/en.

[5] Masatoshi Nakatsu. "Wheel control device, vehicle, wheel control method". Oct. 2016. URL: https://patents.google.com/patent/US9475404B2/en.

[6] Yoshimoto Matsuda. "Control system in vehicle, wheelie determining method, and driving power suppressing method". Oct. 2013. URL: https://patents.google.com/patent/US8560199B2/en.

[7] Yutao Chen et al. "MATMPC - A MATLAB Based Toolbox for Real-time Nonlinear Model Predictive Control". In: *2019 18th European Control Conference (ECC)*. IEEE, June 2019. DOI: 10.23919/ecc.2019.8795788.

[8] M. M. Diehl. J.B. Rowlings D. Q. Mayne. *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, LLC, 2020. URL: https://sites.engineering.ucsb.edu/~jbraw/mpc/.

[9] Roberto Lot. "A Motorcycle Tire Model for Dynamic Simulations: Theoretical and Experimental Aspects". In: *Meccanica* 39 (June 2004), pp. 207–220. DOI: 10.1023/B:MECC.0000022842.12077.5c.

[10] Miklos Vajta. "Some remarks on Padé-approximations". In: *Proceedings of 3rd TEMPUS-INTCOM Symposium* (Oct. 2000). URL: https://ris.utwente.nl/ws/portalfiles/portal/134422804/Some_remarks_on_Pade-approximations.pdf.

[11] Kurt Manal and William Rose. “A general solution for the time delay introduced by a low-pass Butterworth digital filter: An application to musculoskeletal modeling”. In: *Journal of Biomechanics* 40.3 (Jan. 2007), pp. 678–681. DOI: 10.1016/j.jbiomech.2006.02.001.

[12] Gianluca Frison and Moritz Diehl. “HPIPM: a high-performance quadratic programming framework for model predictive control”. In: (2020). DOI: 10.48550/ARXIV.2003.02547.

# Acknowledgments

I would like to express my acknowledgments to significant people who have contributed to the completion of this course of study and this final project.

First of all I want to thank my supervisor Professor Mattia Bruschetta, who gave me the possibility to join this research project, and finally reach this important goal.

Then, I wish to acknowledge my co-supervisor Luca Caiaffa, who gave me all the support and supervision that I needed during the course of the project. Thank you for your time.

Thanks to my parents Adriano and Monica. They have always believed in me, even more than I do, giving me constant support during my studies and in particular during this final project.

Thanks to my sisters Ilaria, Silvia and Annachiara, I will never be able to thank them enough for all they did and do for me.

Thanks to my boyfriend Marco, my reference point. Thanks for listening to me talk about motorcycles and NMPC for the past months and to feed my curiosity every day.

Thanks to Giulia, to be my friend, study partner and flatmate. With her, thanks to Ginevra and Elisa.

Thanks to Elisabetta, Francesca, Sofia, Margherita and Amina, friends for some time who came with me on this journey.