Control Lab Project

Modelling and Control of a Self-Balancing Motorcycle

Group:
Silvia Leo
Vito Daniele Perfetta
Giulia Gelsomina Romano

Contents

1 Introduction **2**

2 The Arduino self-balancing motorcycle **3**
2.1 Arduino boards . . . 3
2.1.1 Arduino MKR1000 . . . 3
2.1.2 Arduino MKR Motor Carrier . . . 4
2.1.3 Arduino MKR IMU Shield . . . 4
2.2 DC and Servo Motors . . . 5
2.3 Battery . . . 7

3 Modelling the motorcycle **9**
3.1 Mathematical Model . . . 9
3.2 Model in the Loop . . . 10

4 PID in place Control **12**
4.1 Simulated Motorcycle . . . 12
4.2 Real Motorcycle . . . 13

5 Feedback Linearization in place Control **18**

6 Sliding Mode in place Control **21**

7 PID straight motion Control **24**

8 Steering motion Control **28**
8.1 Vertical position PID Control . . . 28
8.2 Optimal lean angle Sliding Mode Control . . . 31

Chapter 1

Introduction

The main purpose of this project is controlling the self-balancing motorcycle provided by the Arduino Engineering Kit. A first step consists in balancing the motorcycle in place. Once the vertical position has been stabilized, the control goal has been extended in case of balancing in straight motion and while steering.
In order to do it, different types of controllers will be designed. A first approach with classical PID controller is implemented. Then, two more complex solutions respectively based on Feedback Linearization and Sliding Mode Control have been computed.

Figure 1.1: Arduino Self-balancing Motorcycle

Chapter 2

The Arduino self-balancing motorcycle

The following Arduino motorcycle project consists of a two-wheeled robot that can move and balance itself using an inertia wheel. To properly control the motorcycle it has been used an Arduino MKR1000 board, an Arduino MKR Motor Carrier and a MKR IMU Shield. Moreover, it's necessary a DC motor with tachometer to control the inertia wheel, a DC motor with an encoder to move the rear wheel, and a standard servo motor to make the motorcycle's steering.

2.1 Arduino boards

2.1.1 Arduino MKR1000

Over the years, Arduino has developed a variety of boards, each one with different capabilities and functionalities. The Arduino MKR1000 is a versatile board that includes a LiPo charging circuit that allows it to run on batteries, has a WiFi module with a crypto-chip to ensure secure communications, and can be configured to constrain its power consumption to keep it running for long periods.

Figure 2.1: Board MKR1000

The MKR1000 board has 28 pins, some of which have fixed functionalities and some of

which have configurable functionalities. Fixed functionality pins include GND and 5V, which provide access to the board's ground and 5-volt reference voltage lines respectively. The pins labeled A0 to A6 are analog pins. These pins can transmit or receive voltage values between 0 and 3.3 volts, relative to GND. The pins labeled 0 to 14 are digital pins. When these pins transmit data, they can only transmit voltage values of 0 or 3.3 volts, relative to GND. When they receive data, the voltage is interpreted as HIGH or LOW, relative to some threshold between 0 and 3.3 volts. The pins on the Arduino boards can be used to control and communicate with external components, such as LEDs, motors, or sensors. Pins can be configured either as INPUT or OUTPUT. Output pins in the MKR1000 can source up to 7 mA of current per I/O pin to other devices and circuits. So, it's not possible to drive a motor directly from a pin because the current will not suffice.

2.1.2 Arduino MKR Motor Carrier

Shields/Carriers are boards that can be plugged on top or underneath an Arduino board to extend its capabilities. They are generally used to simplify connecting sensors and actuators.
The MKR Motor Carrier is an MKR add-on board designed to control servo, DC, and stepper motors. The Carrier can also be used to connect other actuators and sensors via a series of 3-pin male headers. Moreover, it allows to connect LiPo battery and select it using an ON-OFF switch with Power ON LED. It can read the status of the battery too.

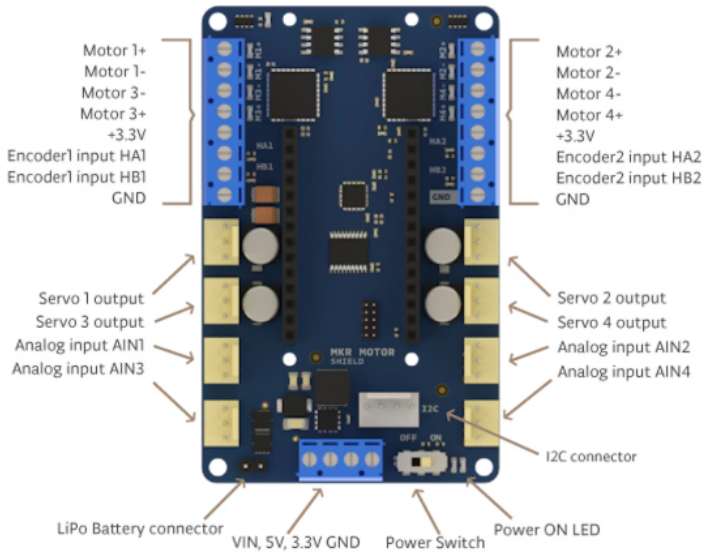

Figure 2.2: Board MKR Motor Carrier

2.1.3 Arduino MKR IMU Shield

The MKR IMU Shield is used to integrate inertial measurements: it can read three-dimensional acceleration, yaw rate and magnetical field by simply mounting it on top of a MKR family board.

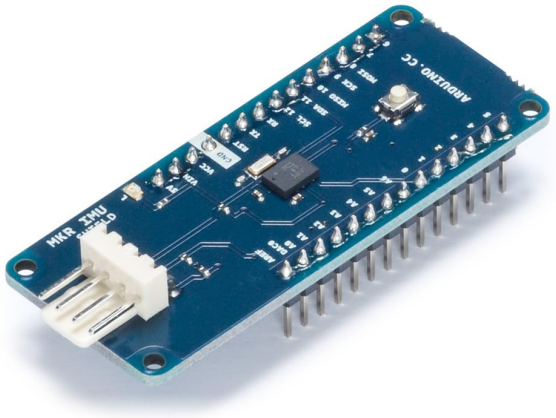

Figure 2.3: Board MKR IMU Shield

IMU(inertial measurement unit) uses a combination of accelerometers, gyroscopes and magnetometers in order to sense the gravitational vector and the magnetic field, from which it can determine rotational and translational motion. In particular, the motorcycle of this project uses the IMU BNO055, a 9-axis absolute orientation sensor, to measure the lean angle θ and its time derivative $\dot{\theta}$ in order to detect when the motorcycle is falling.

2.2 DC and Servo Motors

The self-balancing motorcycle carries one servo motor and two DC motors.
The former is a standard DC servo motor that can rotate between 0 and 180 degrees. As a result, it can not be used to drive a wheel, however, it can be used to move things back and forth at certain angles with high accuracy. In the self-balancing motorcycle, the servo motor is responsible for steering the front wheel.

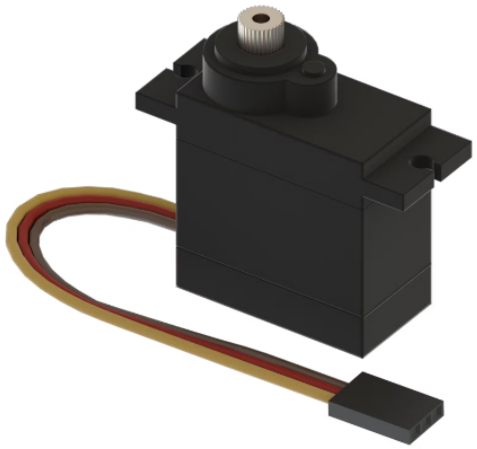

Figure 2.4: Servo Motor

The self-balancing motorcycle features one micro geared (100:1) DC motor with encoder and one bigger DC motor. The geared motor drives the motorcycle backwards and forwards by rotating the rear wheel, while the other DC motor powers the inertia wheel connected to the motorcycle. The inertia wheel DC motor can spin extremely fast when

full torque is applied for a sufficient duration. However, spinning the inertia wheel too fast for too long can permanently damage many components of the motorcycle, including the DC motor (overheating and melting), the motor carrier (overcurrent damage and overheating), and the structural members (vibrational stress and heat distortion). The idea is to spin the inertia wheel fast enough to keep the motorcycle balanced but without exceeding. To mitigate this risk, the motorcycle includes a tachometer (SL353 series micropower omnipolar digital Hall-effect sensor) inside the plastic casing directly in front of the inertia wheel.

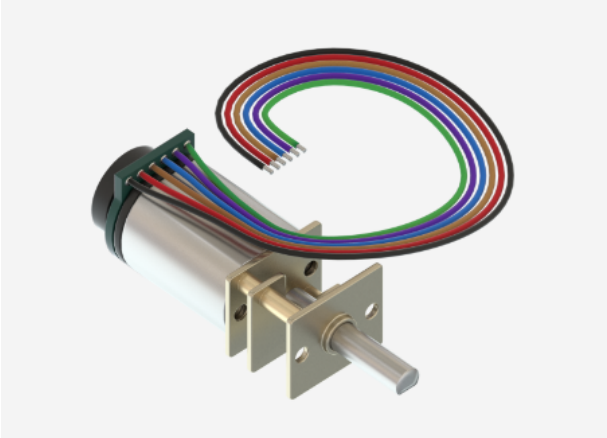

Figure 2.5: DC motor with encoder

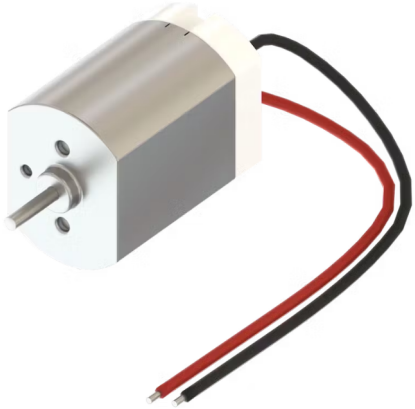

Figure 2.6: DC motor

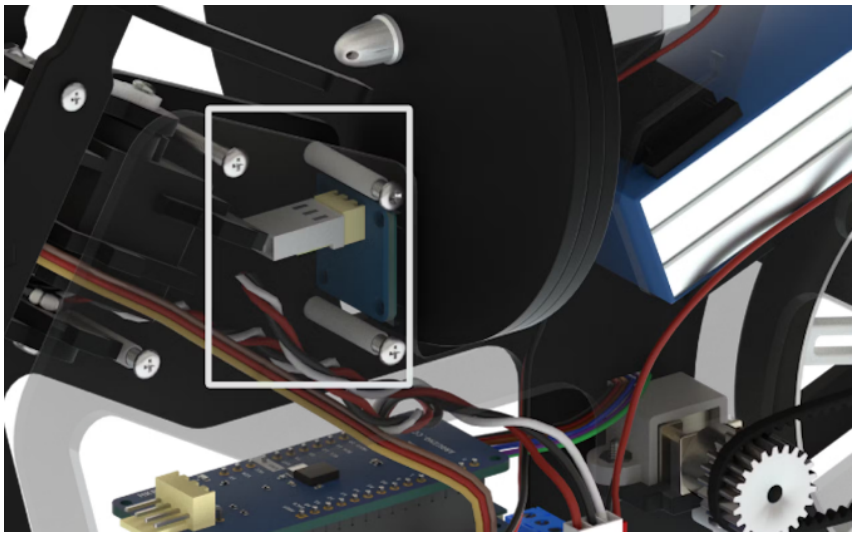

Figure 2.7: Tachometer

2.3 Battery

When working with motors, it's needed an external source to feed the motor drivers and power up the motors. This can be done by connecting a LiPo Battery to the battery connector or using an external power source.
The former are famous in applications where weight and shape are considered critical factors, but they must be handled with care. Discharging the battery under the minimum safe charge can cause irreparable damage to the battery. In addition, overcharge it can cause the battery to ignite. For this reason, Li-ion batteries should always be charged with a specialized charger and should never get charged unattended.

Figure 2.8: LiPo Battery

To avoid these problems, a common power supply connected to the mains can be used. To realize it, a female dc power jack connector joints a 12V AC adapter to the female DuPont jumper wires, which can be directly connected to the MKR1000 board's supply pins.

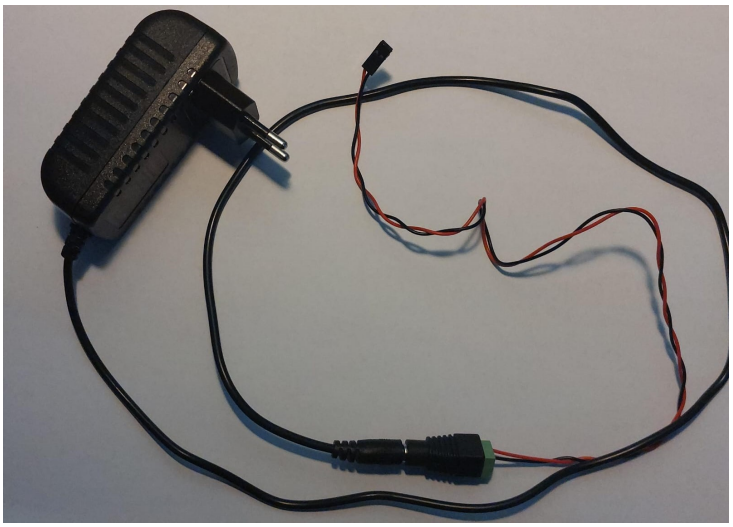

Figure 2.9: AC adapter

Chapter 3

Modelling the motorcycle

3.1 Mathematical Model

In order to control the motorcycle, a model base design approach can be used. Hence, the equations of motion for the system needs to be retrieved. To simplify the model definition, the following assumptions have been considered:

- the motorcycle is free to move only about the wheel-ground axis
- there is no rotational friction between the motorcycle and the ground or between the motorcycle and the inertia wheel
- the motorcycle wheels' thickness is negligible
- air resistance is negligible.

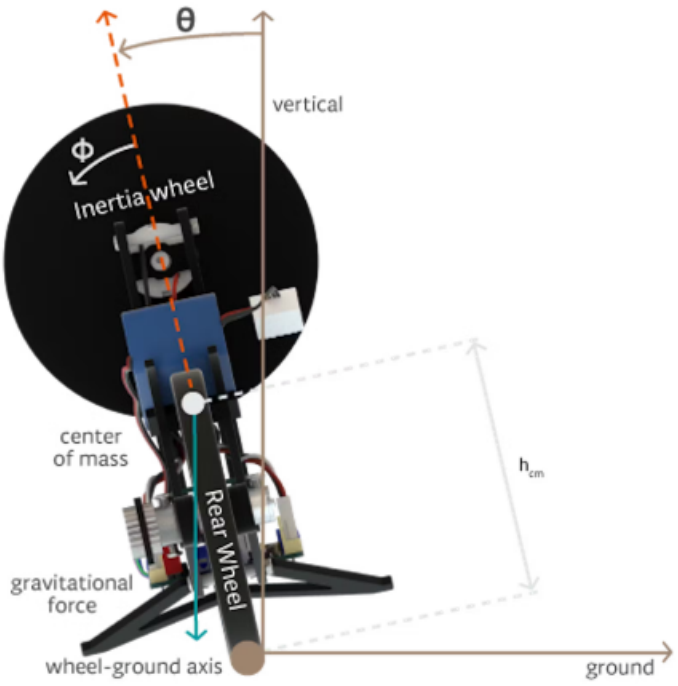

Figure 3.1: Motorcycle's free body diagram

The torque acting on the motorcycle about the wheel-ground axis has 3 major components: gravitational torque τ_g, inertia wheel torque τ_{IW}, external torque τ_{ext}.
The latter encompasses all other torque sources acting on the system like the USB cable pushing up against the motorcycle, air resistance, wind, and various other disturbances.
The net torque on the motorcycle about the wheel-ground axis is

$$\tau_{net} = \tau_g + \tau_{IW} + \tau_{ext} \tag{3.1}$$

Looking at the rotational axis that goes through the inertia wheel motor shaft, the torque that the motor shaft exerts on the inertia wheel is equal and opposite to the torque that the inertia wheel exerts on the motor shaft and thus, the rest of the motorcycle.

$$\tau_{IW} = -\tau_{motor} \tag{3.2}$$

The gravitational torque is equal to

$$\tau_g = M_M \cdot g \cdot h_{cm} \cdot \sin\theta \tag{3.3}$$

Assuming the external torque negligible, the lean angle dynamics is expressed as follows

$$I_M \cdot \ddot{\theta} \approx M_M \cdot g \cdot h_{cm} \cdot \sin\theta - \tau_{motor} \tag{3.4}$$

Where:

- θ is the lean angle
- ϕ is the inertia wheel angular displacement
- I_M is the inertia of the motorcycle
- M_M is the mass of the motorcycle
- h_{cm} is the center of mass height

It can be noticed that, for small angles of θ, the model can be linearized as follows:

$$\sin\theta \approx \theta$$

$$I_M \cdot \ddot{\theta} \approx M_M \cdot g \cdot h_{cm} \cdot \theta - \tau_{motor} \tag{3.5}$$

3.2 Model in the Loop

A control technique can be designed and verified exploiting a simulated model of the real motorcycle. In this case, a Simscape model is already provided inside the Arduino_-Engineering_Kit_Project_Files Matlab toolbox.

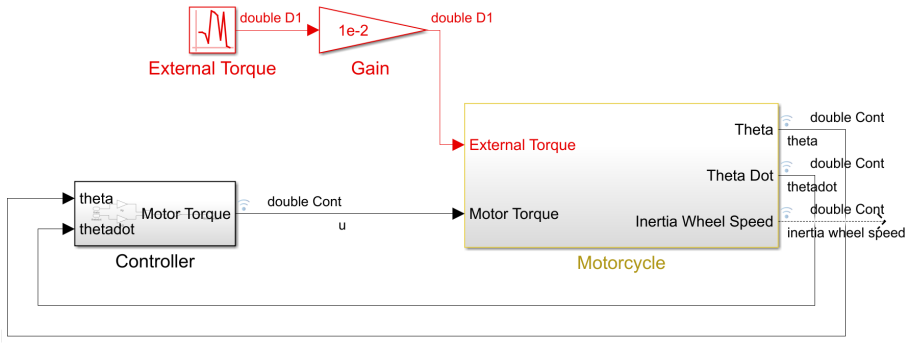

Figure 3.2: Model In the Loop scheme

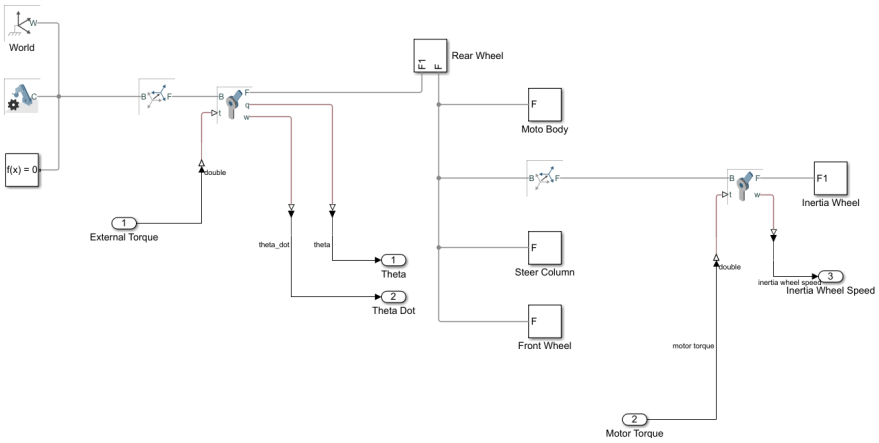

Figure 3.3: Zoom Motorcycle block

All the following control algorithms can be firstly tested with this simulated model, whose results are shown by the Simscape Multibody animation inside the Mechanics Explorer window.

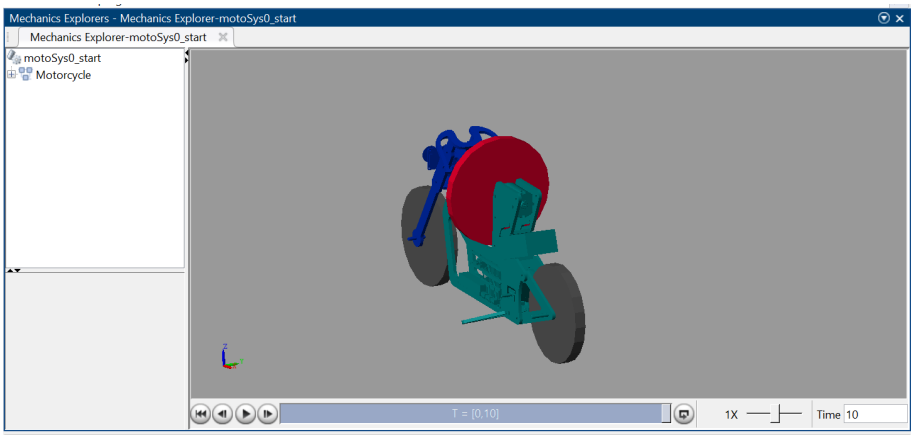

Figure 3.4: Simscape Multibody animation

Once a satisfactory result is obtained, the controller will be applied on the real motorcycle.

Chapter 4

PID in place Control

A simply control technique is based on PID. Retrieved the lean angular displacement and the corresponding velocity either from the simulated model or the IMU sensor on the real system, it's possible to apply a proportional and a derivative action in order to bring the lean angle to 0. To ensure a null steady state error, instead of introducing an integral action that brings an accumulation of errors, it's preferable to increase the proportional gain.

4.1 Simulated Motorcycle

Using as reference the control scheme shown in Figure 3.2, the controller block is implemented as follows

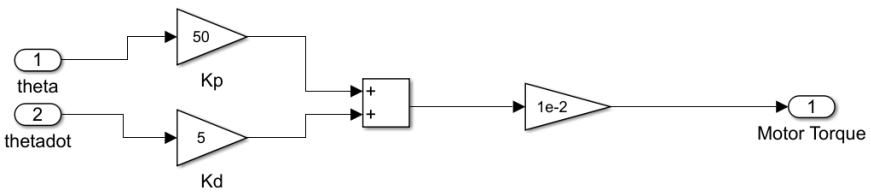

Figure 4.1: Simulated PID control scheme

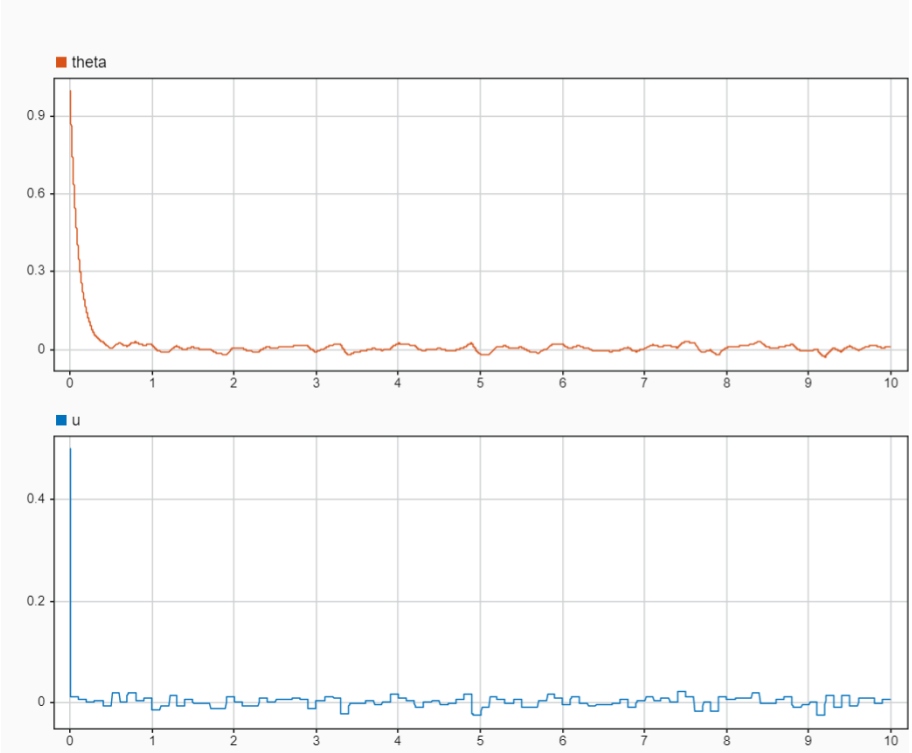

Figure 4.2: Simulated PID Data

4.2 Real Motorcycle

On the real system, the controller needs to take also into account some preventative measure in order to not damage the hardware components.
The resulting PID control scheme is depicted below

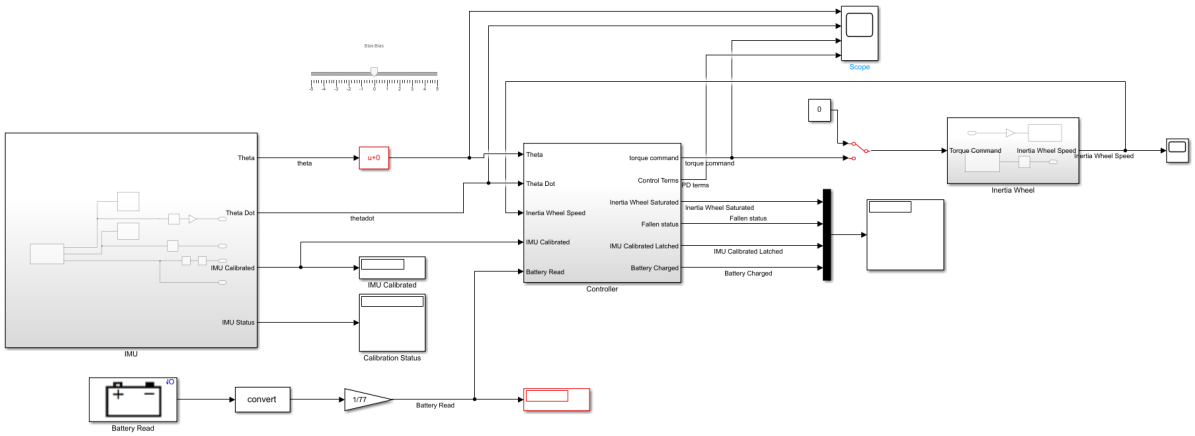

Figure 4.3: PID control scheme

It is possible to analyze each block of the control scheme separately.

In the IMU block, to ensure accurate outputs from the IMU, each sensor needs to be initialized to locate the gravity vector and magnetic field and offset the coordinates accordingly. The elements of the calibration status vector respectively represent: IMU system, Gyroscope, Accelerometer, Magnetometer. Each calibration status vector element has a value

of 0 to 3, indicating the degree to which the sensor is calibrated. For the motorcycle, the gyroscope and the magnetometer need to be fully calibrated (3). The calibration process is performed every time the IMU is powered on. The operator achieves it by rotating the motorcycle at least 90 degrees along each of the three spatial axes. When this procedure is completed, the IMU Calibrated status is true.

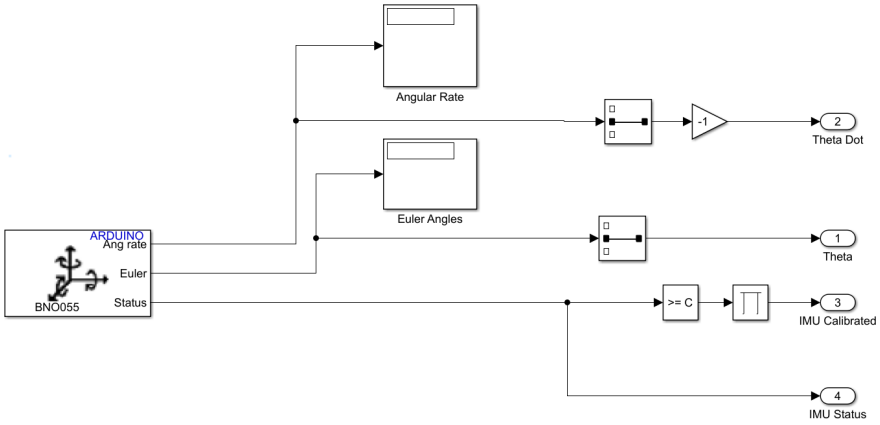

Figure 4.4: Zoom Imu Block

In the Inertia Wheel block, a tachometer is exploited to retrieve the Inertia Wheel Speed, meanwhile a DC motor is used to apply a torque to the inertia wheel, which then causes the motorcycle to exert an equal and opposite counter-torque. In the case analyzed, the inertia wheel is connected to the motor carrier using the Motor M3. The gain 255 is configured to send a torque command into the correct format to the DC motor. The positive sign of the gain is chosen such to have the inertia wheel accelerates counterclockwise (viewed from behind) when the torque command is positive.
Finally, an external manual switch of this block input is added to avoid the automatically spinning of the Inertia Wheel DC motor when the simulation is run.

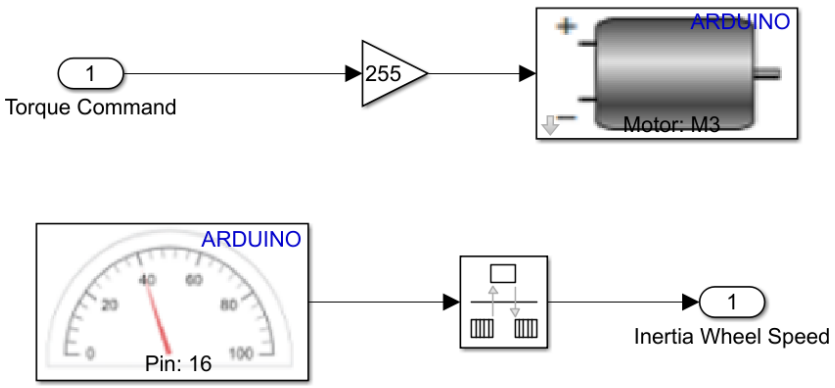

Figure 4.5: Zoom Inertia Wheel

In the PID controller block, a PID algorithm is implemented.

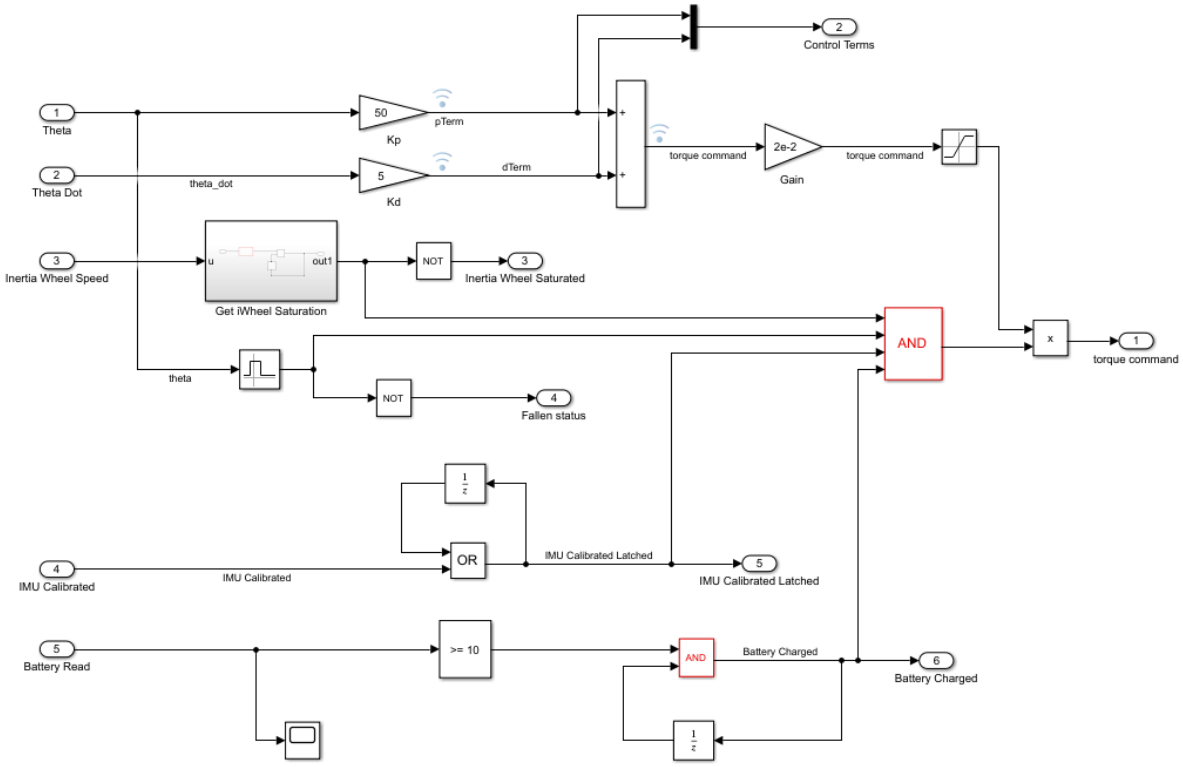

Figure 4.6: Zoom PID controller

Following a trial and error approach, the gain values and the sample time have been respectively chosen as $K_p = 50, K_d = 5, T_s = 0.05s$.

To enhance the robustness and autonomy of the implemented controller, the following safety features will automatically deactivate the inertia wheel under the following conditions:

- The battery level is insufficient
- The IMU is not calibrated
- The motorcycle is fallen
- The Inertia Wheel Speed is excessive

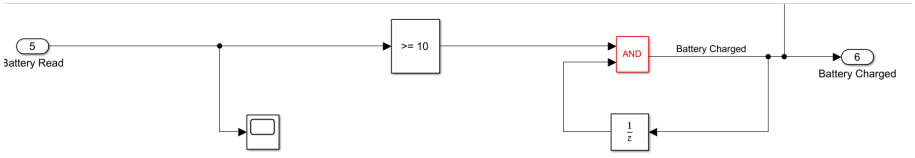

Figure 4.7: Battery Charged condition

Operating the inertia wheel with insufficient battery charge can damage the motor hardware since it will not be able to supply enough torque to correct the lean angle in the

way it was designed. In order to avoid this, a threshold of 10V has been defined. Once the provided voltage is below the threshold, the AND block ensures the Battery Charged condition is false.

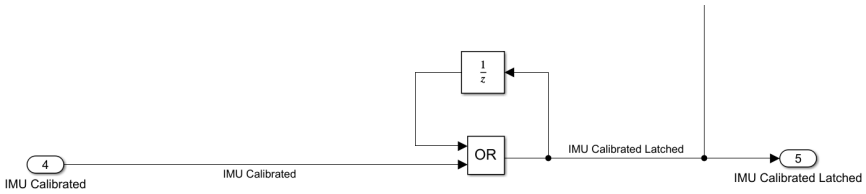

Figure 4.8: IMU Calibrated Latched condition

The IMU sometimes loses calibration to some degree for one or more sensors after it has initially been calibrated. Since it will recover the calibrated condition, an OR block with its previous value can be used to neglect this situation.

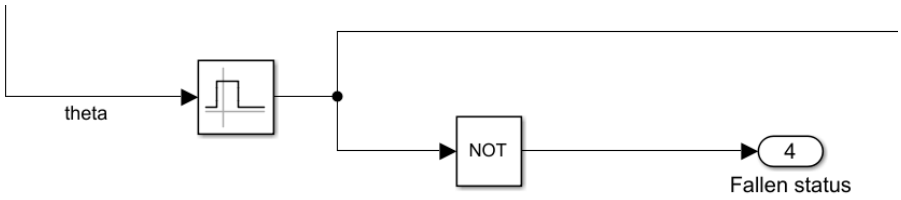

Figure 4.9: Fallen Status condition

If the lean angle magnitude exceeds 6 degrees, the motorcycle is not in a stable balancing mode and cannot reverse direction with just the inertia wheel. In this case, the motorcycle will be considered “fallen” and therefore the inertia wheel will be turned off.

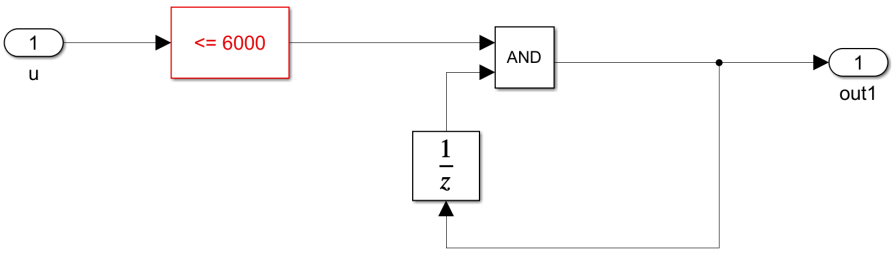

Figure 4.10: Inertia Wheel Saturated condition

As the inertia wheel speed increases, the frictional torque in the motor also increases and lots of heat is generated by the motor. This could lead to irreversible damage. To avoid this problem the maximum angular velocity is set at 6000.

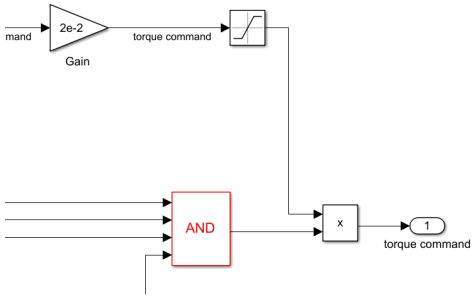

Figure 4.11: Torque command

The control input firstly takes into account the stall torque of the motor, approximately equals to $0.02Nm$. Then, it is saturated between $[-0.7, 0.7]$ to prevent that the Inertia Wheel reaches an excessive velocity. Finally the control input needs to fulfill all the discussed safety conditions.

With this procedure, the motorcycle has been balanced for almost 2 minutes and 30 seconds. The motorcycle balancing has been stopped due to motor safety.

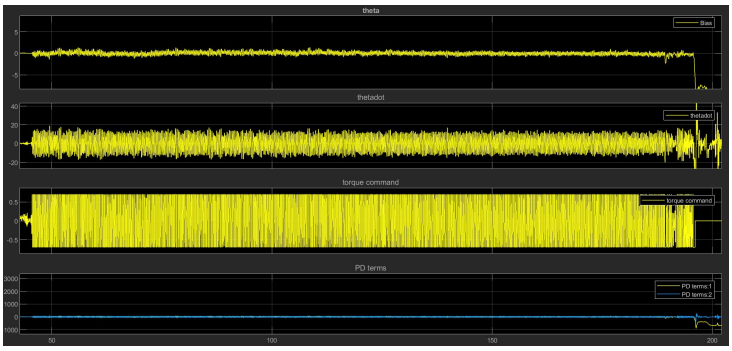

Figure 4.12: PID Data

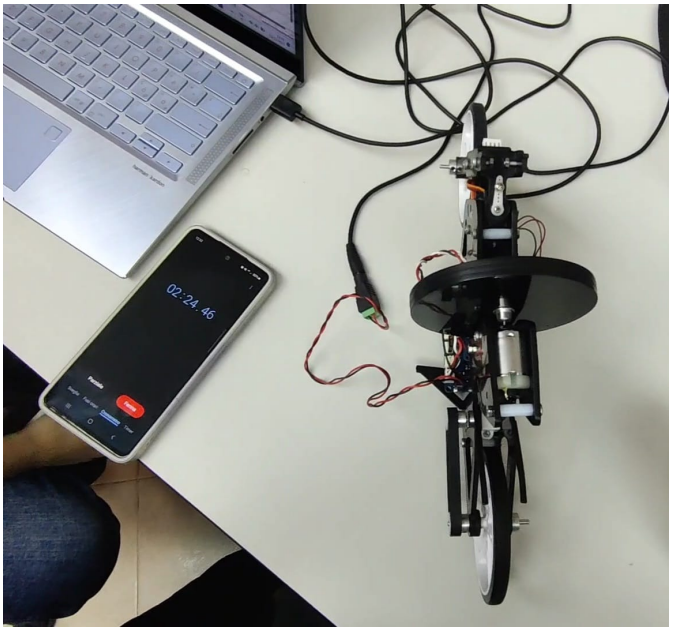

Figure 4.13: PID Final Result

Chapter 5

Feedback Linearization in place Control

The aim of feedback linearization control approach is the synthesis of global control strategies. It is based on the use of differential geometry and it is part of the wider area of geometric control approaches. The objective is to find a nonlinear control law $u = u(x, t)$ and a nonlinear state transformation $z = T(x)$ able to transform the nonlinear system of interest globally in a region of interest Ω into an input-state-output linear system. For this purpose, the motorcycle model can be rewritten in an affine form $\dot{x} = f(x) + g(x)u$. Defined $x_1 = \theta$, $x_2 = \dot{\theta}$, the input-state representation is

$$\begin{cases} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{M_M \cdot g \cdot h_{cm}}{I_M} \cdot \sin x_1 - \frac{1}{I_M} \cdot u \end{cases} \tag{5.1}$$

where

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} x_2 \\ \frac{M_M \cdot g \cdot h_{cm}}{I_M} \cdot \sin x_1 \end{bmatrix}$$

$$\mathbf{g}(\mathbf{x}) = \begin{bmatrix} 0 \\ -\frac{1}{I_M} \end{bmatrix}$$

Choosing the nonlinear state transformation $T(x) = x_1$, it is possible to verify that the conditions of the Input-state Linearization Theorem are satisfied, i.e. $\{g, ad_f(g)\}$ are linearly independent and $\{g\}$ is involutive. In this way, the feedback linearizing control input can be chosen as

$$u = \frac{1}{\mathcal{L}_g(\mathcal{L}_f(T))}(v - \mathcal{L}_f^2(T)) \tag{5.2}$$

where $\mathcal{L}_g(\mathcal{L}_f(T)) = -\frac{1}{I_M}$ and $\mathcal{L}_f^2(T) = \frac{M_M \cdot g \cdot h_{cm}}{I_M} \cdot \sin x_1$.

Selecting $v = -K_1 x_1 - K_2 x_2$, the control input is equal to

$$u = I_M(K_1 x_1 + K_2 x_2 + \frac{M_M \cdot g \cdot h_{cm}}{I_M} \cdot \sin x_1) \tag{5.3}$$

In this way, the system dynamics is

$$\ddot{x}_1 + K_2 \dot{x}_1 + K_1 x_1 = 0 \tag{5.4}$$

that is asymptotically stable as long as K_1 and K_2 are greater than zero.

The physical parameters of the motorcycle have been estimated assuming the system behaviour acts like a pendulum:

- $M_M = 0.4kg$
- $h_{cm} = 0.07m$
- $I_M = M_M \cdot h_{cm}^2 \approx 0.002kg \cdot m^2$

The resulting Feedback Linearization control scheme is depicted below

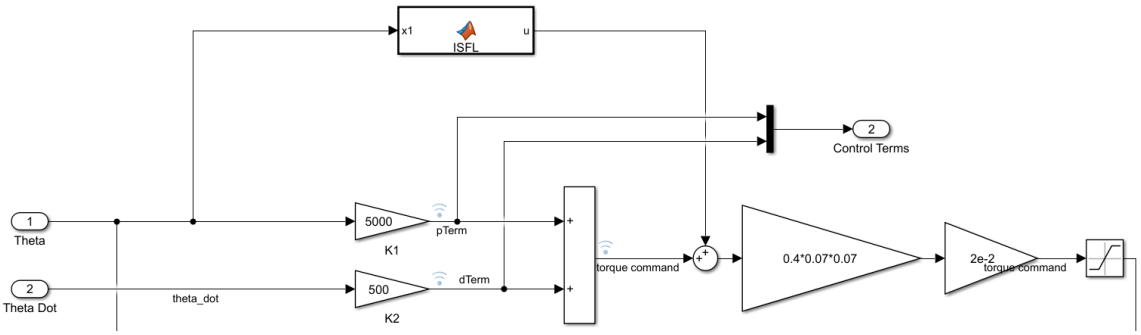

Figure 5.1: Feedback Linearization control scheme

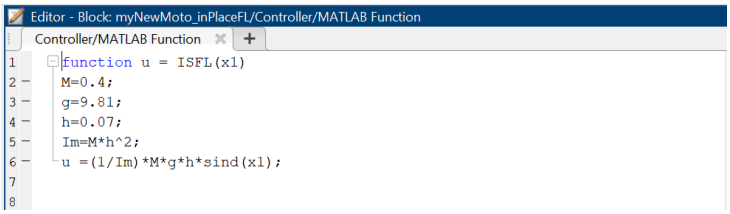

Figure 5.2: Zoom ISFL block

The gain values and the sample time have been respectively chosen as $K_1 = 5000, K_2 = 500, T_s = 0.01s$.

With this procedure, the motorcycle has been balanced for almost 1 minute and 50 seconds. The motorcycle balancing has been stopped due to motor safety.

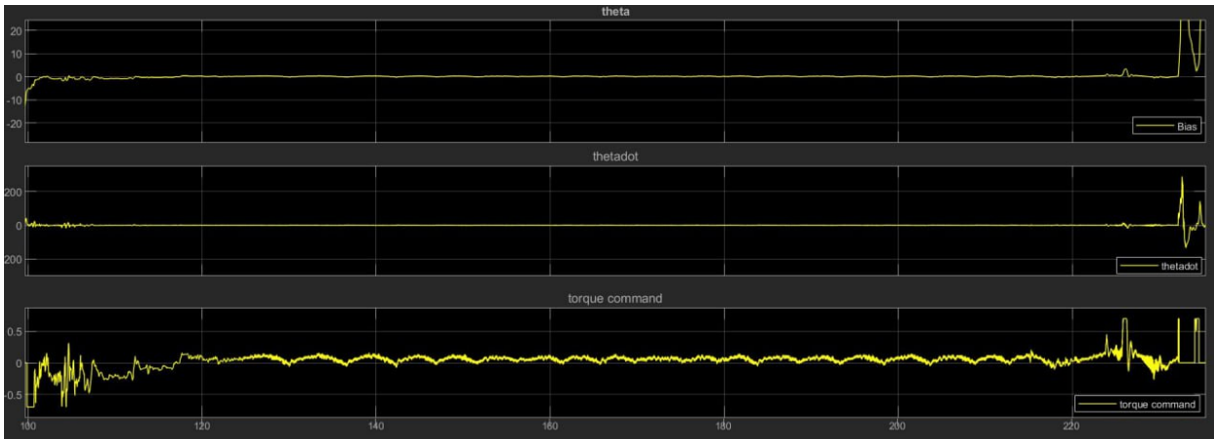

Figure 5.3: Feedback Linearization Data

Figure 5.4: Feedback Linearization Final Result

Chapter 6

Sliding Mode in place Control

The main drawback of the feedback linearization control is its lack of robustness against uncertainties or unmodeled dynamics. This problem can be overcome using switching or sliding control. The key idea behind sliding control is to consider controllers whose structure can change according to some conditions. In general the control "switches" from one configuration to the other in order to fulfil the control objectives. When a switching controller is used, the closed-loop system becomes a switched (or piecewise-smooth) dynamical system that has a Switching Manifold Σ. This type of system exhibits the so called "sliding solutions" where the system trajectories are forced onto the sliding surface Σ such that $\sigma(x) = 0$. The control idea is to develop a controller that induces sliding onto a Σ along which the system behaviour meets the design specifications. This approach is robust to perturbation because the trajectories will be stuck on Σ whatever the change of directions of the vector fields due to perturbations, as long as they point towards Σ.

To design a sliding mode controller, $\sigma(x)$ is defined as follow in order to ensure the transversality condition, i.e. $\mathcal{L}_g \neq 0$,

$$\sigma(x) = p_1 x_1 + p_2 x_2 \tag{6.1}$$

where $p_1, p_2 > 0$.
To ensure the attractivity on Σ, the control input u is set as

$$u = -\frac{K sign(\sigma) + \mathcal{L}_f(\sigma)}{\mathcal{L}_g(\sigma)} \tag{6.2}$$

where

- $K > 0$
- $\mathcal{L}_f(\sigma) = p_1 x_2 + p_2 \frac{M_M \cdot g \cdot h_{cm}}{I_M} \sin x_1$
- $\mathcal{L}_g(\sigma) = -\frac{p_2}{I_M}$

Once on Σ, the switching controller ensures that the system trajectories evolve according to the sliding vector field

$$\dot{x} = f_s(x) = f(x) + g(x) u_{eq} \tag{6.3}$$

where $u_{eq} = -\frac{\mathcal{L}_f(\sigma)}{\mathcal{L}_g(\sigma)}$.

Thanks to the chosen $\sigma(x)$, the closed loop system evolves as a first order system

$$\begin{cases} \dot{x}_1 &= -\frac{p_1}{p_2} x_1 \\ \dot{x}_2 &= -\frac{p_1}{p_2} x_2 \end{cases} \tag{6.4}$$

that goes asymptotically to zero as long as p_1 and p_2 are positive.

The resulting Sliding Mode control scheme is depicted below

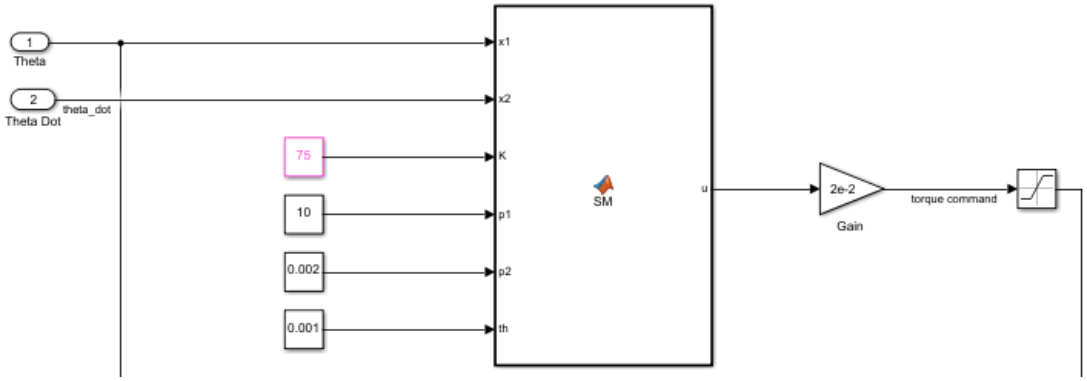

Figure 6.1: Sliding Mode control scheme

The gain values and the sample time have been respectively chosen as $p_1 = 10, p_2 = 0.002, K = 75, T_s = 0.01s$.

Ideally, the sliding on Σ requires an infinite switching frequency. In reality, the actuators can only provide a control input with a finite maximum frequency. This creates the chattering problem. To reduce this phenomena, a dead zone with a threshold $th = 0.001$ has been exploited.

Editor - Block: myNewMoto_inPlaceSM/Controller/MATLAB Function*

Controller/MATLAB Function*

```
1   function u = SM(x1,x2,K,p1,p2,th)
2   M=0.40;
3   h=0.07;
4   g=9.81;
5   Im=M*h^2;
6
7   sigma=p1*x1+p2*x2;
8
9   if(sigma>th)
10      u =Im/(p2)*(K+p1*x2+p2*M*g*h/Im*sind(x1));
11  elseif (sigma<-th)
12      u =Im/(p2)*(-K+p1*x2+p2*M*g*h/Im*sind(x1));
13  else
14      u =Im/(p2)*(p1*x2+p2*M*g*h/Im*sind(x1));
15  end
```

Figure 6.2: Zoom SM block

With this procedure, the motorcycle has been balanced for more then 1 minute 30 seconds. The motorcycle balancing has been stopped due to motor safety.

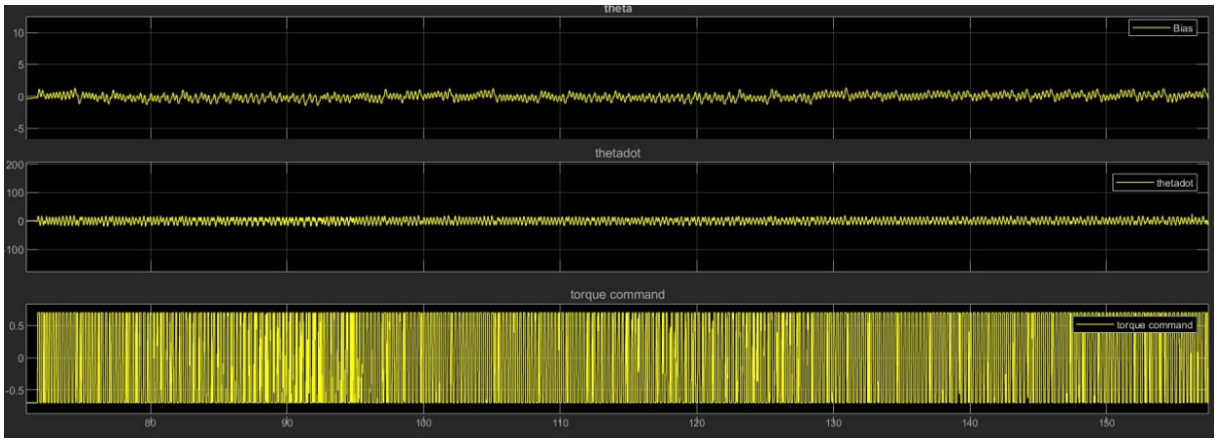

Figure 6.3: Sliding Mode Data

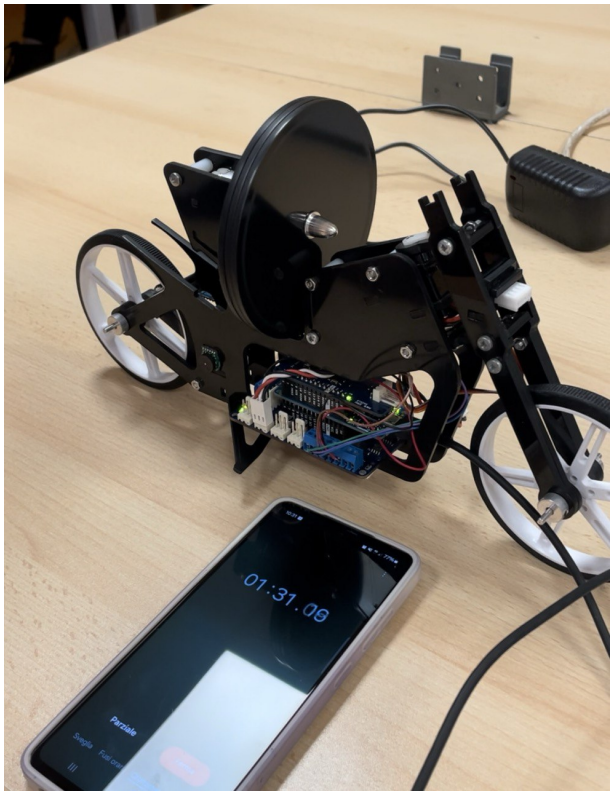

Figure 6.4: Sliding Mode Final Result

Chapter 7

PID straight motion Control

Once the system has been stabilized in vertical position, it's possible to make the motorcycle move in straight motion. To realize this, the original PID control scheme has been modified as shown below

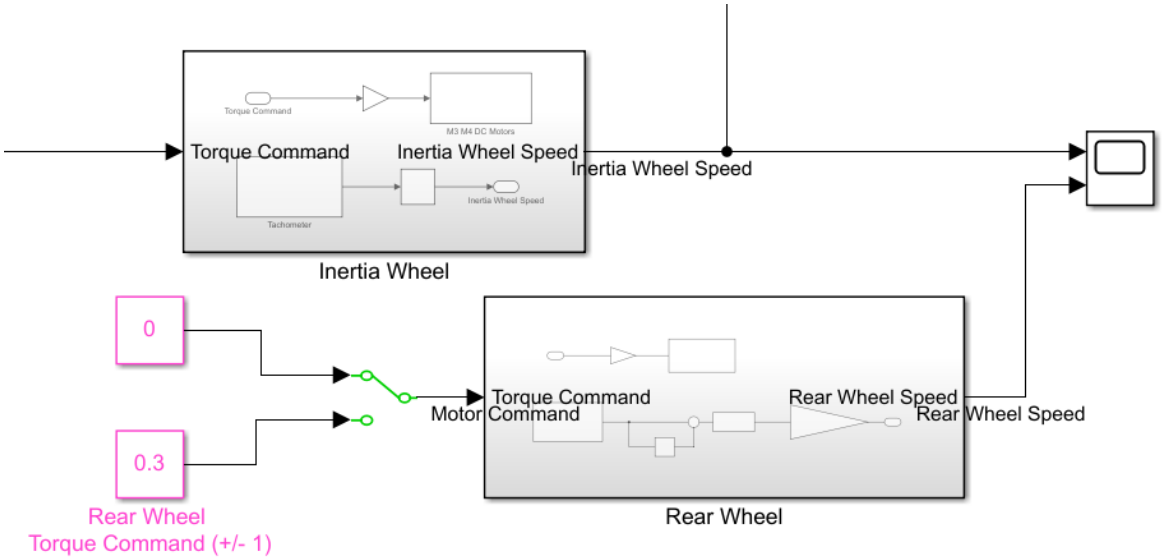

Figure 7.1: Zoom straight motion PID control scheme

To make the motorcycle move, the DC motor M4 on the rear wheel has to be activated. The embedded encoder allows to retrieve the linear velocity differentiating the encoder measurement. Finally, the resulting value has to be properly converted from count/s into m/s.

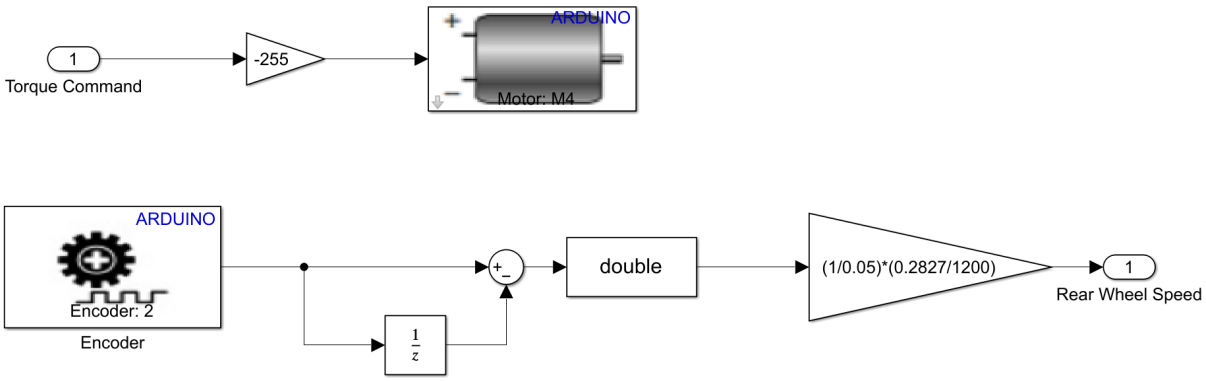

Figure 7.2: Zoom Rear Wheel block

With this procedure, the motorcycle has been balanced for almost 25s while covering approximatively 4m increasing gradually the torque command on the rear wheel from 0.25 to 0.4.

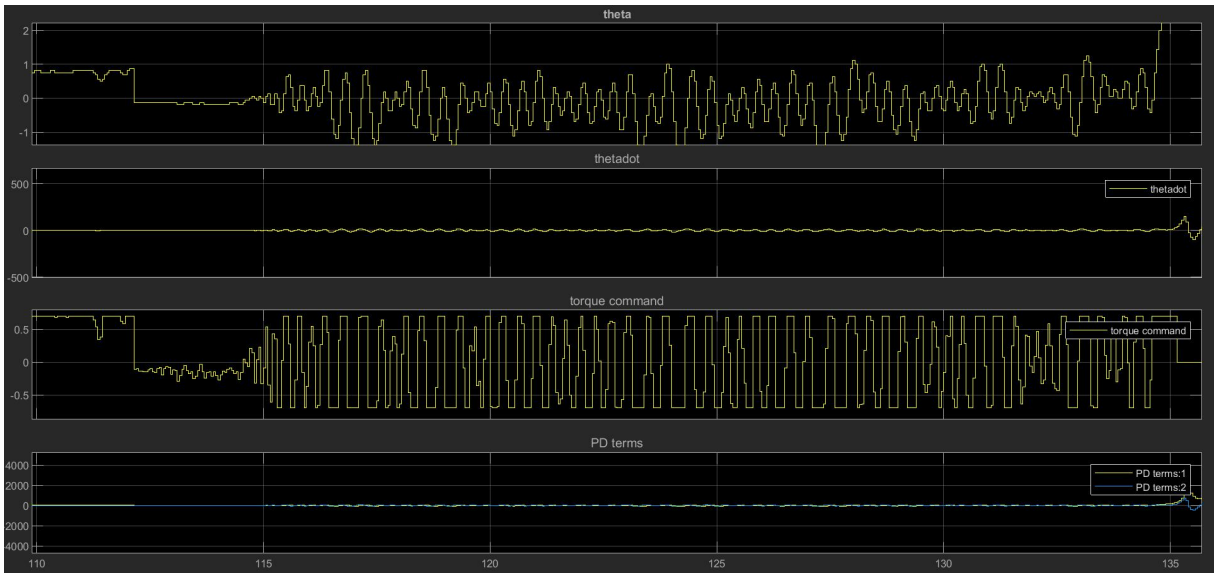

Figure 7.3: Straight Motion Data

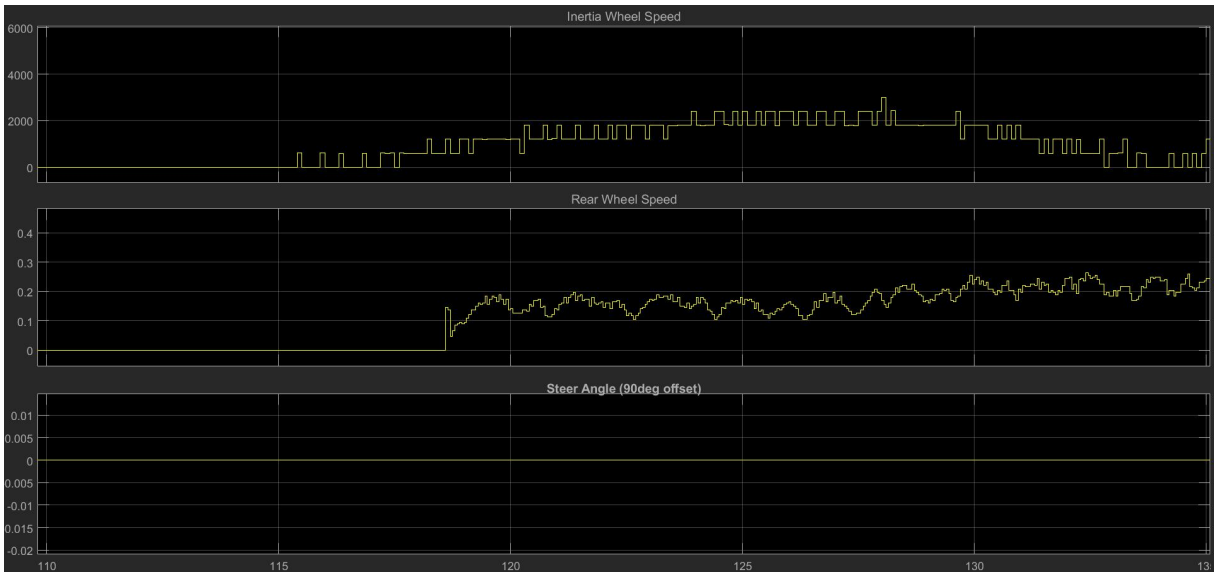

Figure 7.4: Straight Motion Data

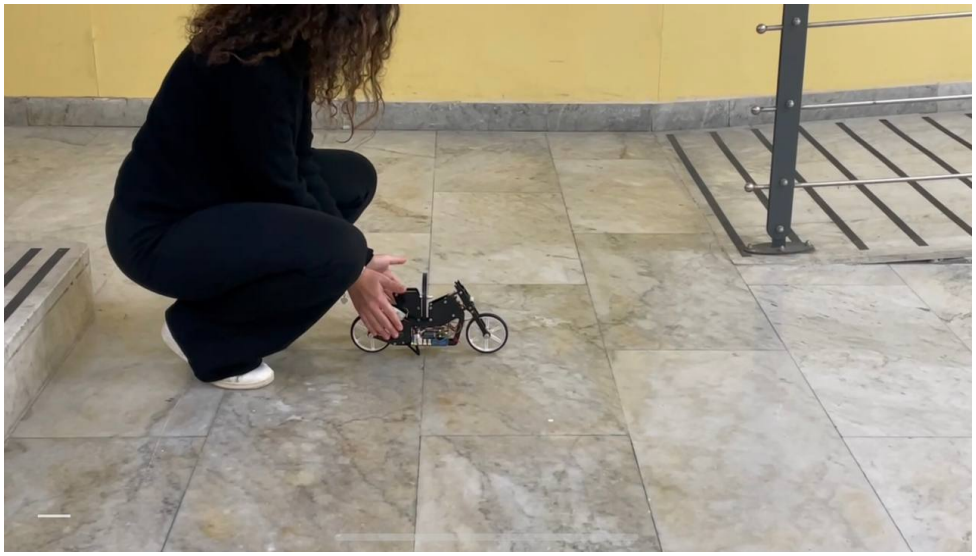

Figure 7.5: Straight motion PID Control Start

Figure 7.6: Straight motion PID Control Finish

In a similar way, it is possible to assign a negative torque command on the rear wheel to make the motorcycle move backwards.

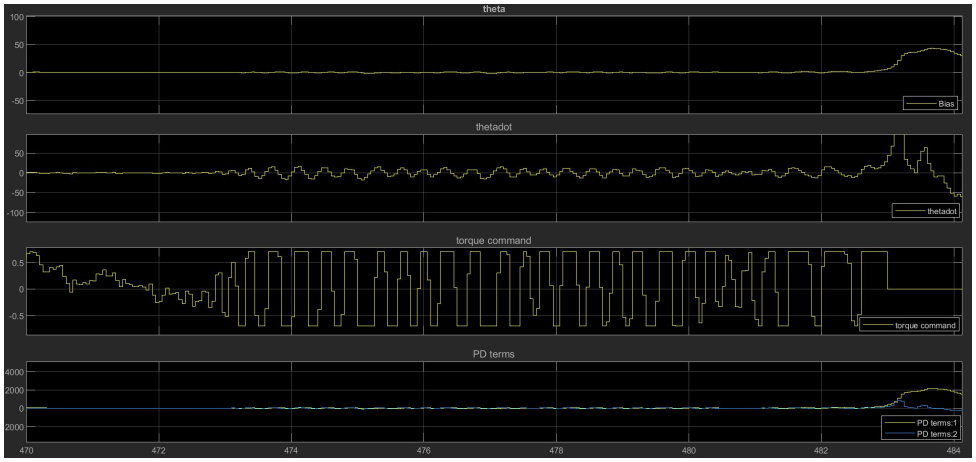

Figure 7.7: Back and forth Motion Data

Figure 7.8: Back and forth Motion Data

Chapter 8

Steering motion Control

Another possible test consists in making the motorcycle steer with a fixed angle. To realize this, there are two different ways to proceed:

- vertical position control
- optimal lean angle control

8.1 Vertical position PID Control

With this solution the motorcycle is still balanced in the vertical position while steering. In order to do this, the previous PID straight motion control scheme has been modified in the following way

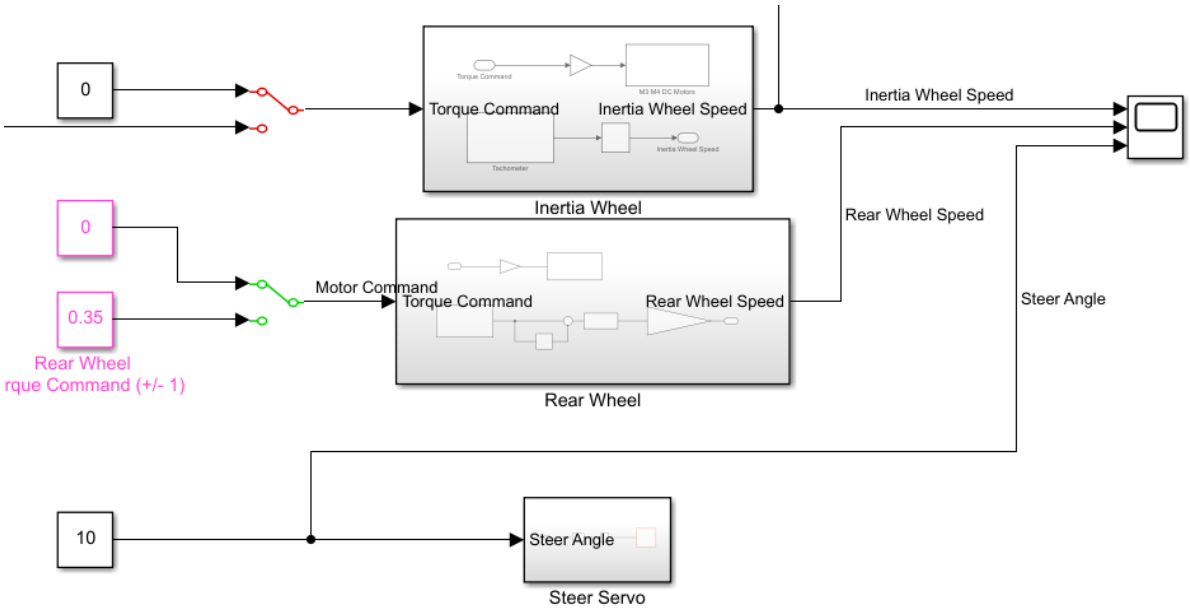

Figure 8.1: Zoom steering motion PID control scheme

The steering is realized using the servo motor equipped in the steer column.

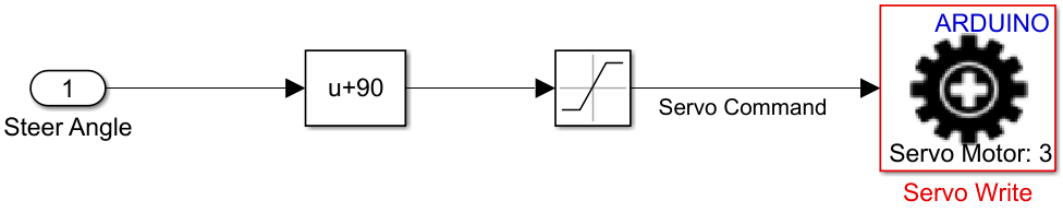

Figure 8.2: Zoom Steer Servo block

This servo motor can rotate its lever arm to any angle between 0 and 180 degrees, but due to mechanical bounds its range is between 60 and 120 degrees. These limits are ensured by the saturation block. The motorcycle is constructed such that the "straight" angle is approximately 90 degrees, but it is more intuitive to call this angle 0 and refer to left and right steer angles using respectively positive and negative angles with respect to the "straight" servo angle. In order to realize this, a Bias block of 90 degrees has been inserted.

Defining a relative Steer Angle of 10 degrees while increasing gradually the torque command on the rear wheel from 0.3 to 0.35, the motorcycle covers a distance of 1 m.

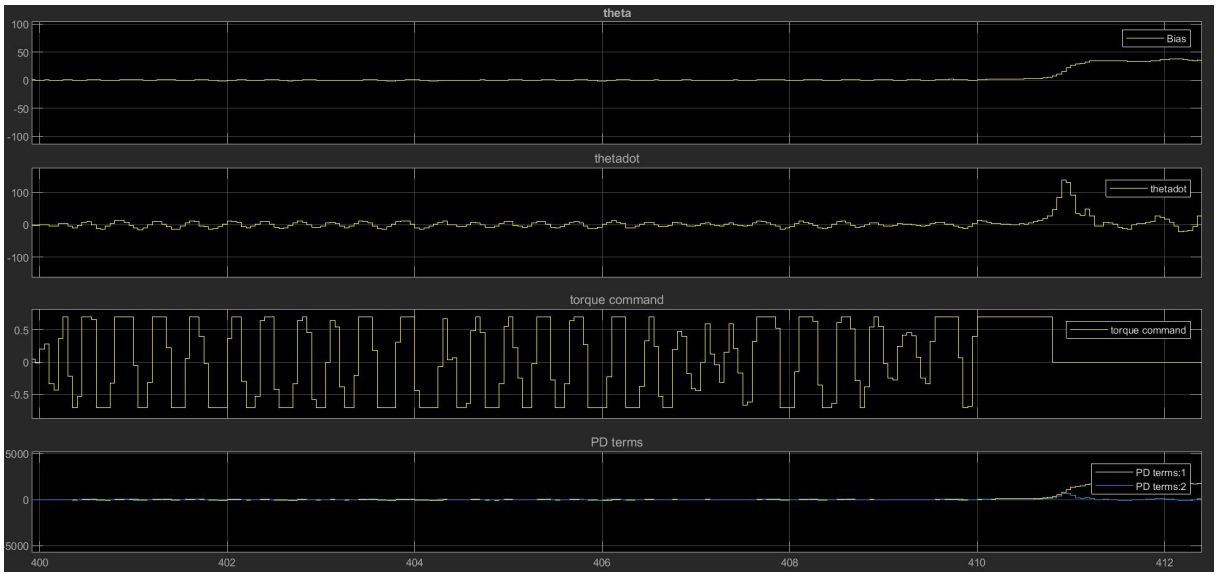

Figure 8.3: Steering Motion Data

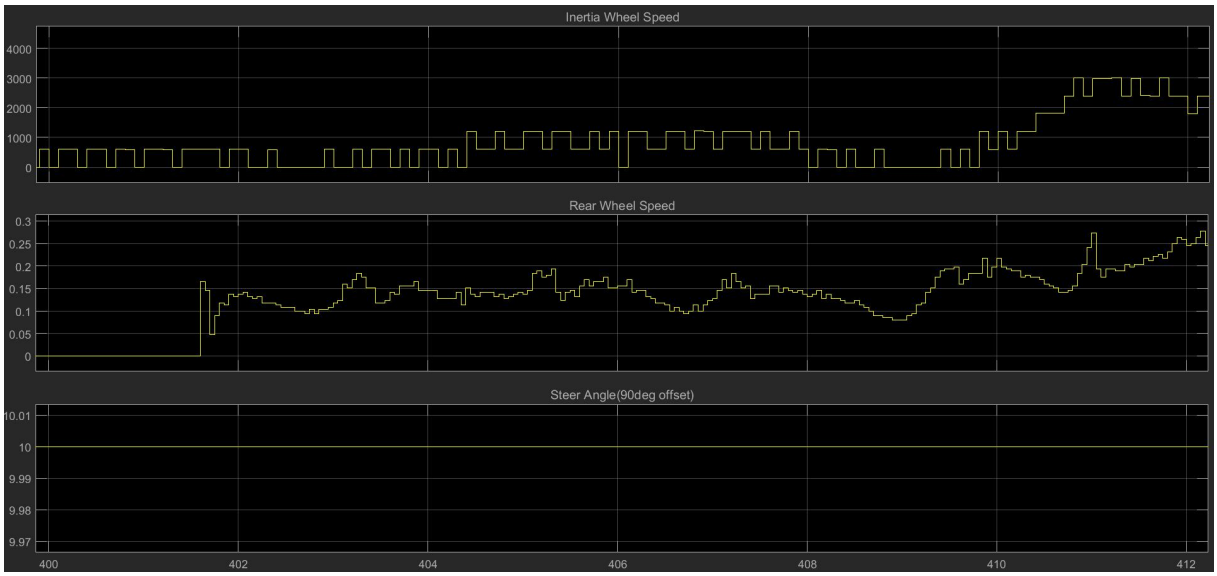

Figure 8.4: Steering Motion Data

Figure 8.5: Steering Motion Final Result

It is possible to notice that decreasing the Steer Angle, the distance covered by the motorcycle has been increased reaching a space of 2.5 m.

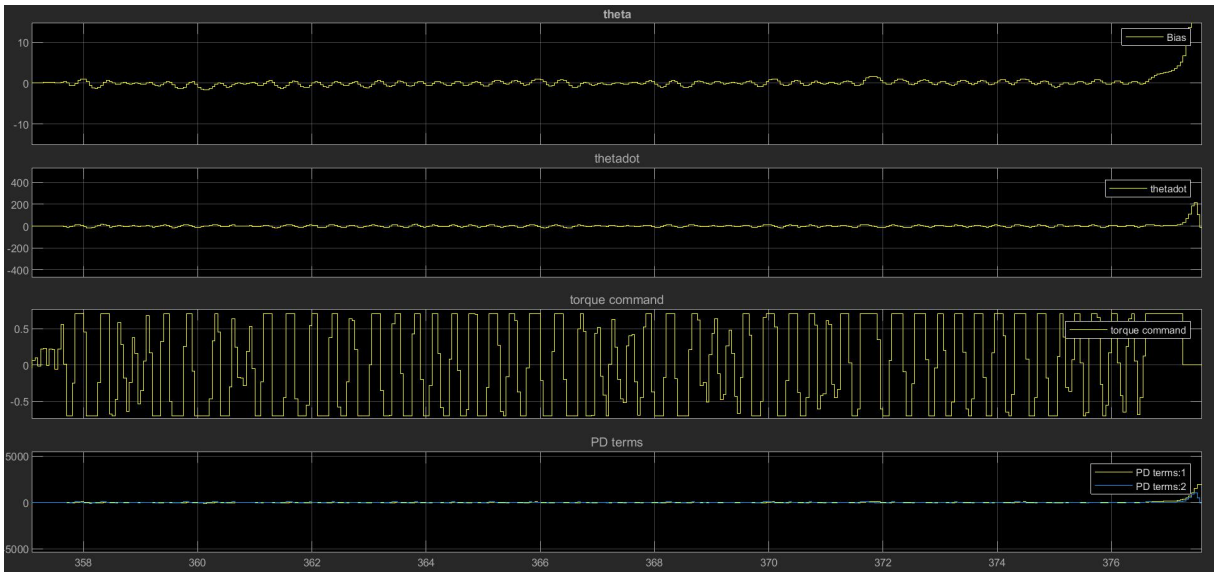

Figure 8.6: Steering Motion Data (small steer angle)

8.2 Optimal lean angle Sliding Mode Control

This alternative solution consists in modifying the previous PID steering Vertical position Controller so that the motorcycle leans inward as it turns. To achieve this, it is possible to identify the optimal lean angle for turns, and adjust the controller algorithm so that the motorcycle tends towards the optimal lean angle at all times. The PID algorithm works well with a linear model. Here, this assumption is true only with small lean angles, but when the motorcycle leans it can go out of this range. In this case, the Sliding Mode Control is more suitable since it directly works with the nonlinear model.
First of all, it is necessary to look at the torques acting on the motorcycle. The gravitational torque τ_g (with respect to the motorcycle frame) gets larger when the motorcycle leans farther out from the vertical plane since the force acts farther from the rotational axis. When the motorcycle turns, it also experiences a centrifugal force $\tau_{centrifugal}$ (with respect to the motorcycle frame), which pulls the motorcycle outward from the turn direction.

$$\tau_{centrifugal} = -M_M \cdot \frac{V^2}{R_{turn}} \cdot h_{cm} \cdot \cos\theta \tag{8.1}$$

where V is the linear speed of the motorcycle and R_{turn} is the turn radius, defined as the radius of the circle traced by the motorcycle as it turns.
To explain R_{turn}, the steer angle δ, the distance between the centers of the front and the rear wheel L_{FR} and the respective radii of the circles are R_F and R_R have been considered. The angle between the front and rear wheel radii is equal to the steer angle δ. Using trigonometric relationships, the center of mass radius R_{cm} has been computed as:

$$R_F \approx R_R \approx R_{cm} \approx \frac{L_{FR}}{\delta} \tag{8.2}$$

Finally R_{turn} can be approximate to R_{cm}.

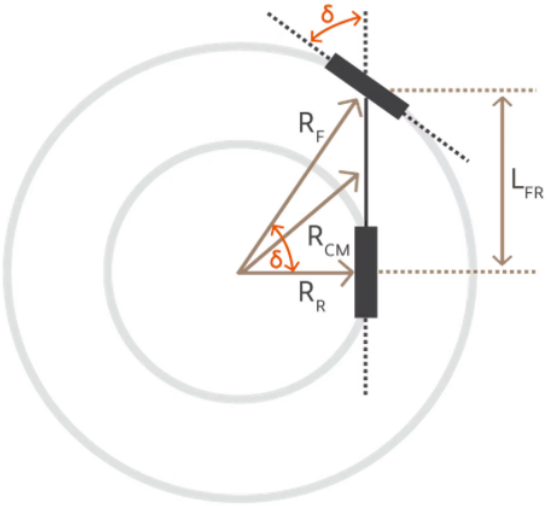

Figure 8.7: Focus on Steer Angle

To find the equilibrium lean angle, where the motorcycle is perfectly balanced, it is possible to assume the motor is turned off ($\tau_{IW} = 0$) and there is no external torque beyond the gravitational and centrifugal torques ($\tau_{ext} \approx 0$).
Based on these assumptions, the unmotorized net torque has been defined as follow

$$\tau_{net,nomotor} \approx \tau_g + \tau_{centrifugal} = M_M \cdot g \cdot h_{cm} \cdot \sin\theta - M_M \cdot \frac{V^2 \cdot \delta}{L_{FR}} \cdot h_{cm} \cdot \cos\theta \tag{8.3}$$

where L_{FR} has been estimated equal to $0.25m$.
When the motorcycle is at the equilibrium, the unmotorized net torque is equal to zero. In this case, it is possible to find the optimal lean angle as

$$\theta_{opt} = \arctan \frac{V^2 \cdot \delta}{L_{FR} \cdot g} \tag{8.4}$$

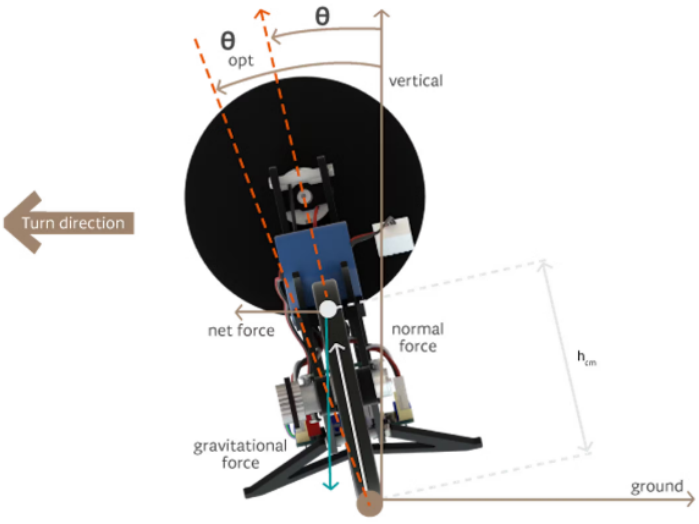

Figure 8.8: Motorcycle's free body diagram with θ_{opt}

This control algorithm is firstly tested with the Model In the Loop approach.

The Sliding Mode Control has been implemented as before, while the sliding surface Σ is changed as

$$\sigma(x) = p_1(x_1 - \theta_{opt}) + p_2 x_2 \tag{8.5}$$

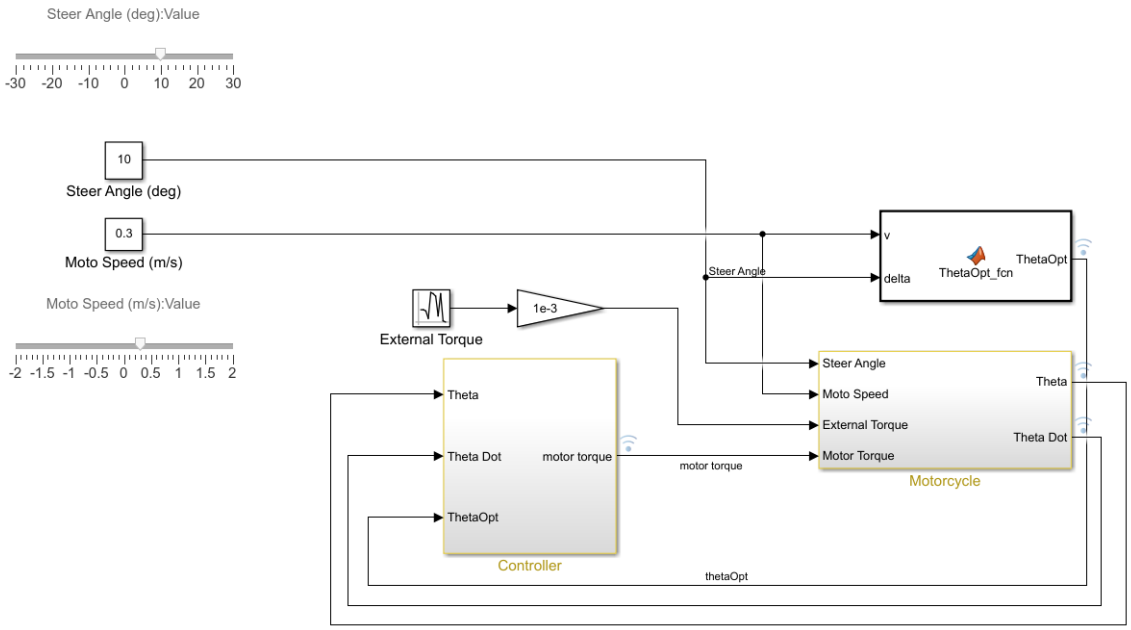

Figure 8.9: Simulated Steering motion SM control scheme

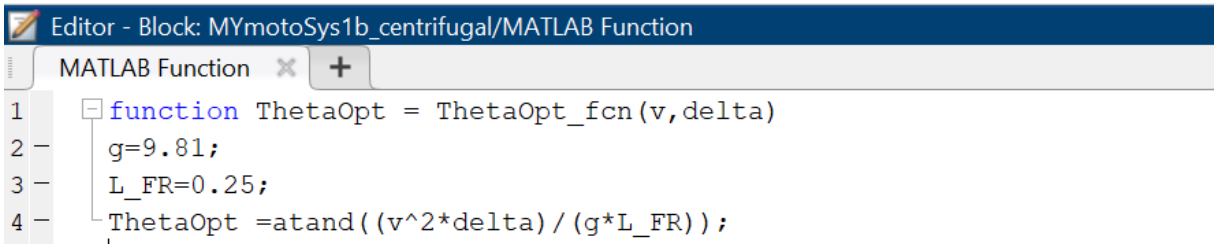

Figure 8.10: Zoom ThetaOpt_fcn

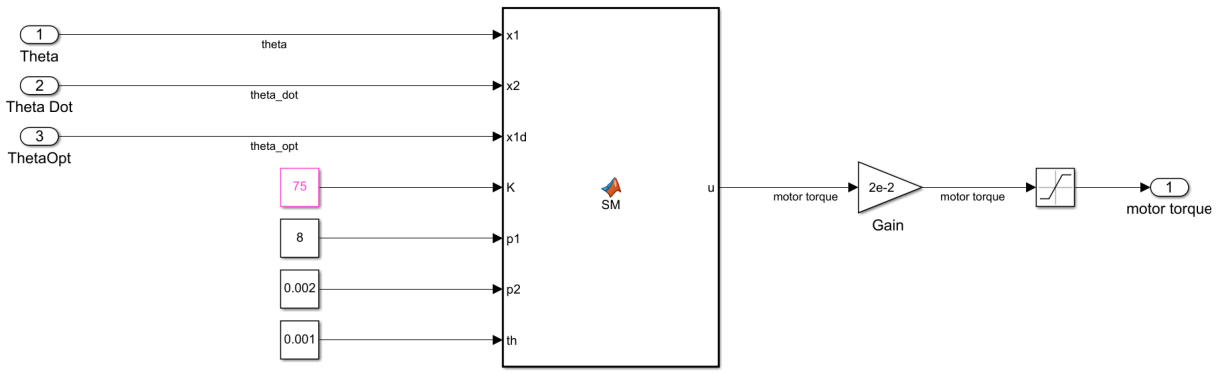

Figure 8.11: Zoom Controller

Editor - Block: MYmotoSys1b_centrifugal/Controller/MATLAB Function

Controller/MATLAB Function

```
function u = SM(x1,x2,x1d,K,p1,p2,th)
M=0.40;
h=0.07;
g=9.81;
Im=M*h^2;

sigma=p1*(x1-x1d)+p2*x2;

if(sigma>th)
    u =Im/(p2)*(K+p1*x2+p2*M*g*h/Im*sind(x1));
elseif (sigma<-th)
    u =Im/(p2)*(-K+p1*x2+p2*M*g*h/Im*sind(x1));
else
    u =Im/(p2)*(p1*x2+p2*M*g*h/Im*sind(x1));
end
```

Figure 8.12: Zoom steering motion SM Controller

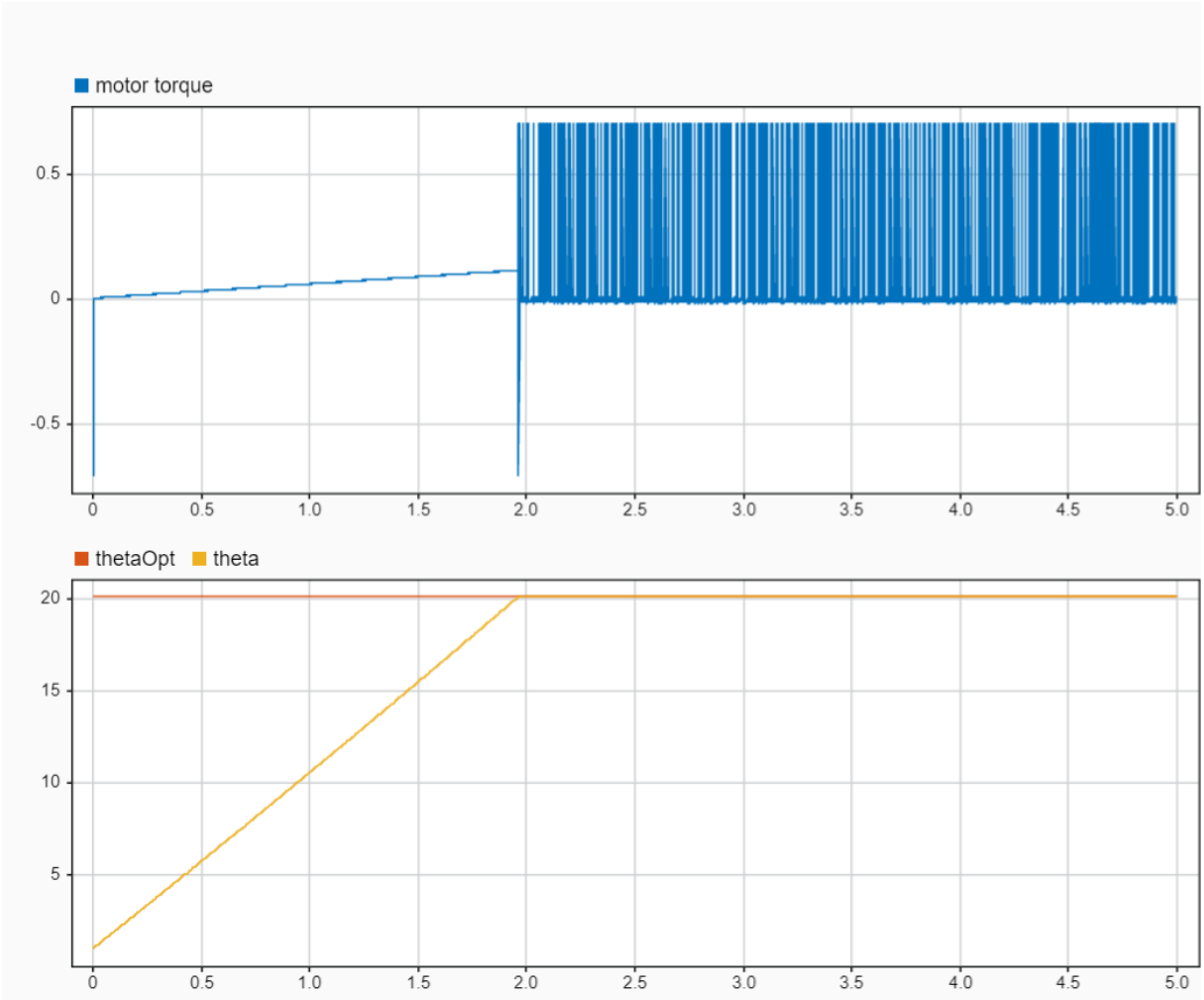

Figure 8.13: Simulated steering SM Control with θ_{opt} Data

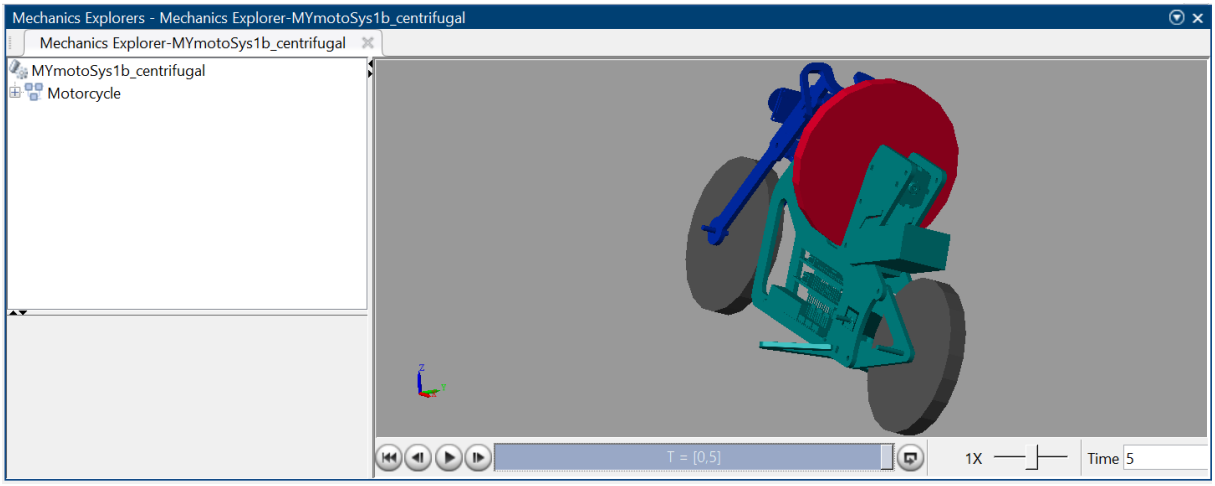

Figure 8.14: Simscape Multibody animation steering SM Control with θ_{opt}

As it is shown, when the motorcycle turns, the controller is able to make it lean such to reach the optimal lean angle without falling.

However, in the real case, due to physical limits of the motorcycle such as the contact surface between the ground and the wheel, the servo motor accuracy and the uncertain estimation of the parameters, the motorcycle is not able to lean at an excessive θ_{opt}.

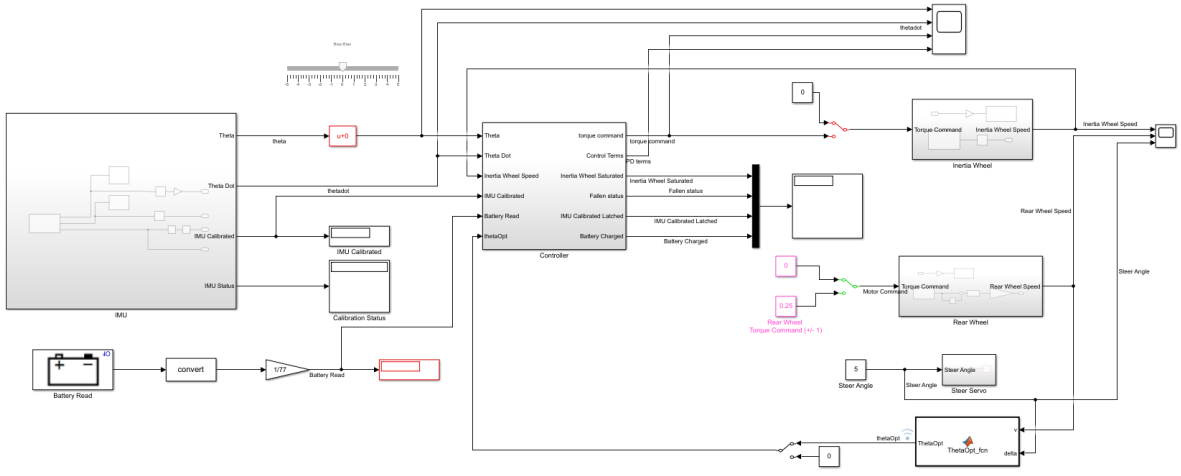

Figure 8.15: Steering motion SM control scheme

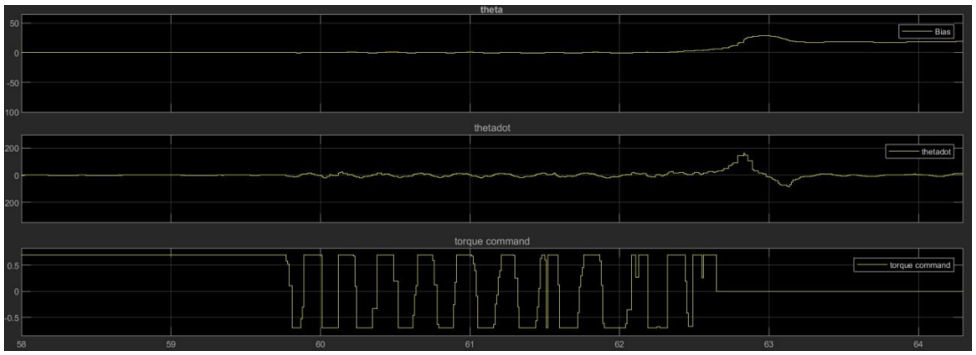

Figure 8.16: Steering SM Control with θ_{opt} Data

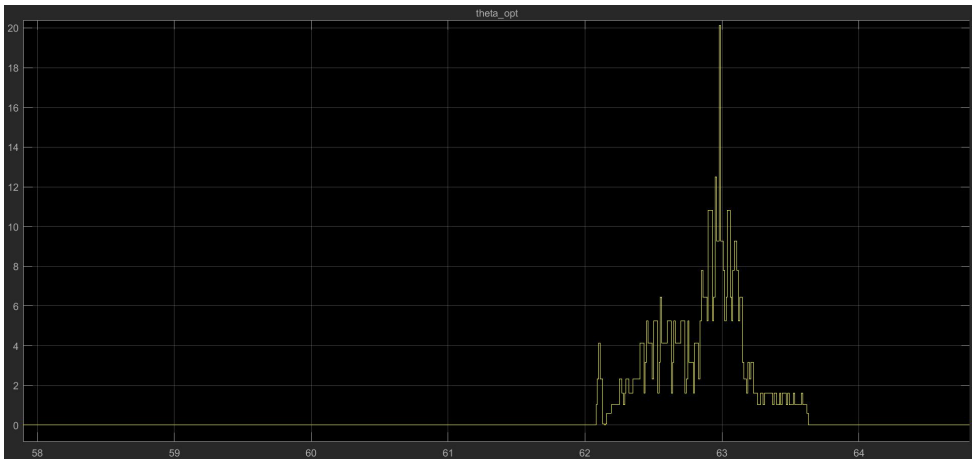

Figure 8.17: Steering SM Control with θ_{opt} Data

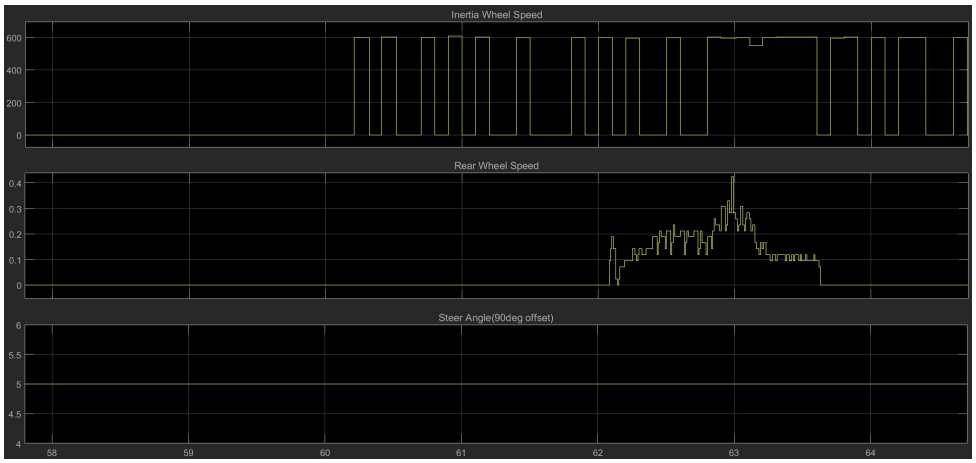

Figure 8.18: Steering SM Control with θ_{opt} Data

As it is shown, when the θ_{opt} reaches $20deg$, the motorcycle is not able to balance itself anymore.