Laboratory activity 4: Longitudinal state–space control of the balancing robot

Group: 2, Shift: Friday
Franceschetti Lorenzo
Mengo Edoardo
Rettore Riccardo
Zaupa Alberto

1 Introduction

1.1 Activity Goal

The aim of this laboratory experience is to design and validate a control scheme for a two-wheeled balancing robot that, at the same time, keeps the robot around the upward equilibrium position and moves it to the desired longitudinal position.

The report is divided as follows: the first part introduces the physical model, presenting the equations that govern the robot under some simplifying assumptions. Then two control strategies based on LQR control are presented together with their simulation results on a Simulink model of the robot to assess some of their properties. At last, these numerical results are compared with the experimental ones obtained with the real device.

1.2 System and Model

The considered two-wheeled robot can be thought as made up of three different rigid bodies

- Left and right wheel
- DC gearmotor rotors, one for each wheel
- Robot body, subdivided into chassis, DC gearmotor stators and battery

All the parameters and diagrams of the robot can be found in Appendix A.1 and A.2.

1.2.1 Robot dynamics

Robot dynamics is described using the following coordinates:

- *robot position* $\boldsymbol{p}_v^o$, given by the coordinates of the vehicle frame origin with respect to the world frame

- *robot tilt angle* θ: pitch angle of the body frame with respect to the vehicle frame
- *wheel angles* θ_l, θ_r: pitch angle of the wheel frame with respect to the vehicle frame
- *robot heading angle* ψ: yaw angle of the vehicle frame with respect to the world frame
- *rotor angles* $\theta_{rot,l}, \theta_{rot,r}$: pitch angle of the rotor frames with respect to the vehicle frame

$\theta_{rot,\{l,r\}}$, $\theta_{\{l,r\}}$ and θ are coupled together by the gearbox, while ψ and $\theta_{\{l,r\}}$ can be related to each other via the pure-rolling and no side-slip wheel assumptions. To simplify the analysis, the dynamics is restricted to the longitudinal one, that is, ψ is assumed constant during motion, and without loss of generality set to $0°$. This means that $\theta_l = \theta_r$, so they can be denoted by the common value γ.

Robot dynamics is derived via Lagrange formalism, using as generalised coordinates the vector $\boldsymbol{q} = [\gamma, \theta]^T$. In matrix form, it can be written as

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{F}_v\boldsymbol{q} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau} \tag{1}$$

where $\boldsymbol{M}$ is the inertia matrix, $\boldsymbol{C}$ is the matrix of centrifugal and Coriolis-related terms, $\boldsymbol{F}_v$ the matrix of viscous frictions, $\boldsymbol{g}$ is the gravity contribution and $\boldsymbol{\tau} = [2\tau, -2\tau]^T$ the motor torque input.

After adding the electrical coupling to (1), by linearising the equations around the upward equilibrium and rewriting the model as a system of first-order differential equations with state vector $\boldsymbol{x} = [\boldsymbol{q}, \dot{\boldsymbol{q}}]^T$, the following linear state-space representation is obtained:

$$\begin{gathered}\boldsymbol{A} = \begin{bmatrix} \boldsymbol{0}_{2\times2} & \boldsymbol{I}_{2\times2} \\ -\boldsymbol{M}^{-1}\boldsymbol{G} & -\boldsymbol{M}^{-1}\boldsymbol{F'}_v \end{bmatrix}, \ \boldsymbol{B} = \frac{2Nk_t}{R_a} \begin{bmatrix} \boldsymbol{0}_{2\times2} \\ \boldsymbol{M}^{-1} \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \\ \boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}, \ \boldsymbol{D} = 0 \end{gathered} \tag{2}$$

with input u_a the supply voltage to the armature circuit. A more thorough dissertation and the expression of the matrices can be found in Section 2 of Handout 4.

1.2.2 Inertial measurement system

The output of a Motion Processing Unit (MPU) is used to obtain information about the tilt angle θ and it is comprised of an accelerometer and a gyroscope. The accelerometer output $\boldsymbol{y}_a$ is the sum of the actual linear acceleration experienced by the robot at the sensor centre and the gravity acceleration, while the gyroscope output y_g is the tilt angular velocity, namely

$$\boldsymbol{y}_a = [x_a^{mpu}, z_a^{mpu}]^T = \begin{bmatrix} r_w\ddot{\gamma}cos(\theta) + z_{mpu}^b\ddot{\theta} + gsin(\theta) \\ r_w\ddot{\gamma}sin(\theta) - z_{mpu}^b\dot{\theta}^2 - gcos(\theta) \end{bmatrix}, \ y_g = \dot{\theta} \tag{3}$$

2 Tasks, Methodologies and Results

The objective of the controller is to simultaneously keep the the robot in the upward position and guarantee the tracking of a constant reference in γ. In particular, the performances of the controller have been evaluated on the following 3 tasks:

1. Regulation of θ and γ from a tilted initial configuration, which was set to $\boldsymbol{x_0} = \begin{bmatrix} 0 & \frac{\pi}{36} & 0 & 0 \end{bmatrix}^T$ in the numerical experiments.
2. Regulation of θ and tracking of a constant reference $\gamma^* = 168.51°$ corresponding to a longitudinal displacement of $10cm$, starting from an initial configuration close to vertical.
3. The same as the previous point, with the addition of a constant disturbance of $5V$ applied at the motors input, with a delay of $5s$.

Two different control strategies have been analysed, a nominal and a robust one, both performed in discrete time with sampling period $T_s = 0.01s$. To achieve this, the state-space model has been discretised via the exact method, obtaining

$$\boldsymbol{\Phi} = \begin{bmatrix} 1 & 0.0022 & 0.0098 & 0.0002 \\ 0 & 1.0027 & 0.0001 & 0.0099 \\ 0 & 0.4345 & 0.9633 & 0.0349 \\ 0 & 0.5488 & 0.0192 & 0.9807 \end{bmatrix} \quad \boldsymbol{\Gamma} = \begin{bmatrix} 0.0002 \\ -0.0002 \\ 0.0472 \\ -0.0317 \end{bmatrix} \qquad (4)$$
$$\boldsymbol{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \quad \boldsymbol{J} = 0$$

Being the accelerometer more reliable at low frequencies and the gyroscope more reliable at high frequencies, a complementary filtering approach was used to obtain better estimates $\hat{\gamma}$ and $\hat{\theta}$ of the first two components of the state as presented in Section 3 of Handout 4. To do so, a low-pass filter $H(s)$ was designed to suitably combine the two outputs in the frequency regions where they are best. The three following filters were tested

$$H_1(s) = \frac{1}{T_{c,1}s + 1}, \; H_2(s) = \frac{2T_{c,2}s + 1}{(T_{c,2}s + 1)^2}, \; H_3(s) = \frac{3T_{c,3}^2 s^2 + 3T_{c,3}s + 1}{(T_{c,3}s + 1)^3} \qquad (5)$$

with $T_{c,\{1,2,3\}}$ suitably chosen. Since the whole design is in discrete time, the complementary filters were then discretised with Forward Euler method.

The estimates $\hat{\dot{\gamma}}$ and $\hat{\dot{\theta}}$ of the velocities were computed with the discrete-time derivative filter

$$H_{diff}(z) = \frac{1 - z^{-3}}{3T_s} \qquad (6)$$

The implementations of the full state observer and the complementary filtering approach can be found in Appendix A.4.

2.1 Laboratory assignments: numerical simulations

Numerical experiments were performed using a Simulink implementation of (1). The schemes of the robot model together with the controller implementations can be found in Appendix A.4.

2.1.1 Nominal State–Space Control

First a nominal control strategy was tested. The state-feedback gain matrix $\boldsymbol{K}$ has been derived using the `dlqr` MatLab method. The cost matrices $\boldsymbol{Q}$ and R were chosen according to Bryson's rule to obtain

$$\boldsymbol{Q} = \text{diag}\left\{\frac{1}{\bar{\gamma}^2}, \frac{1}{\bar{\ }^2}, 0, 0\right\}, \quad R = \frac{1}{\bar{u}^2}, \quad \bar{\gamma} = \frac{\pi}{18}, \quad \bar{\theta} = \frac{\pi}{360}, \quad \bar{u} = 1 \tag{7}$$

An additional parameter ρ has been used in order to balance the relative cost of $\boldsymbol{Q}$ and R. In particular, the tested values were 500 and 5000.

The feed-forward gains $\boldsymbol{N}_x$ and N_u were computed by solving the system

$$\begin{bmatrix} \boldsymbol{\Phi} - \boldsymbol{I} & \boldsymbol{\Gamma} \\ \boldsymbol{H} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{N}_x \\ N_u \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ 1 \end{bmatrix} \tag{8}$$

All the controller gains are reported in Tab. 1.

	$\rho = 500$	$\rho = 5000$
$\boldsymbol{K}$	$\begin{bmatrix} -0.2381 \\ -53.7057 \\ -1.1487 \\ -6.0768 \end{bmatrix}$	$\begin{bmatrix} -0.0754 \\ -51.9410 \\ -1.0856 \\ -5.9010 \end{bmatrix}$
$\boldsymbol{N}_x$	$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T$	$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T$
N_u	0	0

Table 1: Nominal gains with$\rho = 500$ and $\rho = 5000$

For the sensor fusing filters $H_{\{1,2,3\}}(z)$, a tentative value $T_{c,\{1,2,3\}} = \frac{1}{2\pi 0.35}s$ was initially chosen for all three filters.

In Fig. (1) are reported the responses in $\hat{\gamma}$ and $\hat{\theta}$ with the controller for $\rho = 500$ to assess the choice of T_c. While the first and second order behaved similarly in $\hat{\theta}$, with the second better in $\hat{\gamma}$, the third order had the worst response in $\hat{\theta}$. Analysing their Bode plots, reported on the left in Fig. (2), it was noticed that the three filters behave very differently at high frequencies. This suggested that it could be beneficial to adjust the values of $T_{c,\{2,3\}}$ so that the three graphs overlap in the high frequency region, as in the Bode plots on the right in Fig. (2). Having applied this correction, the final values were $T_{c,1} = \frac{1}{2\pi 0.35}s$, $T_{c,2} = \frac{1}{2\pi 0.17}s$ and $T_{c,3} = \frac{1}{2\pi 0.12}s$.

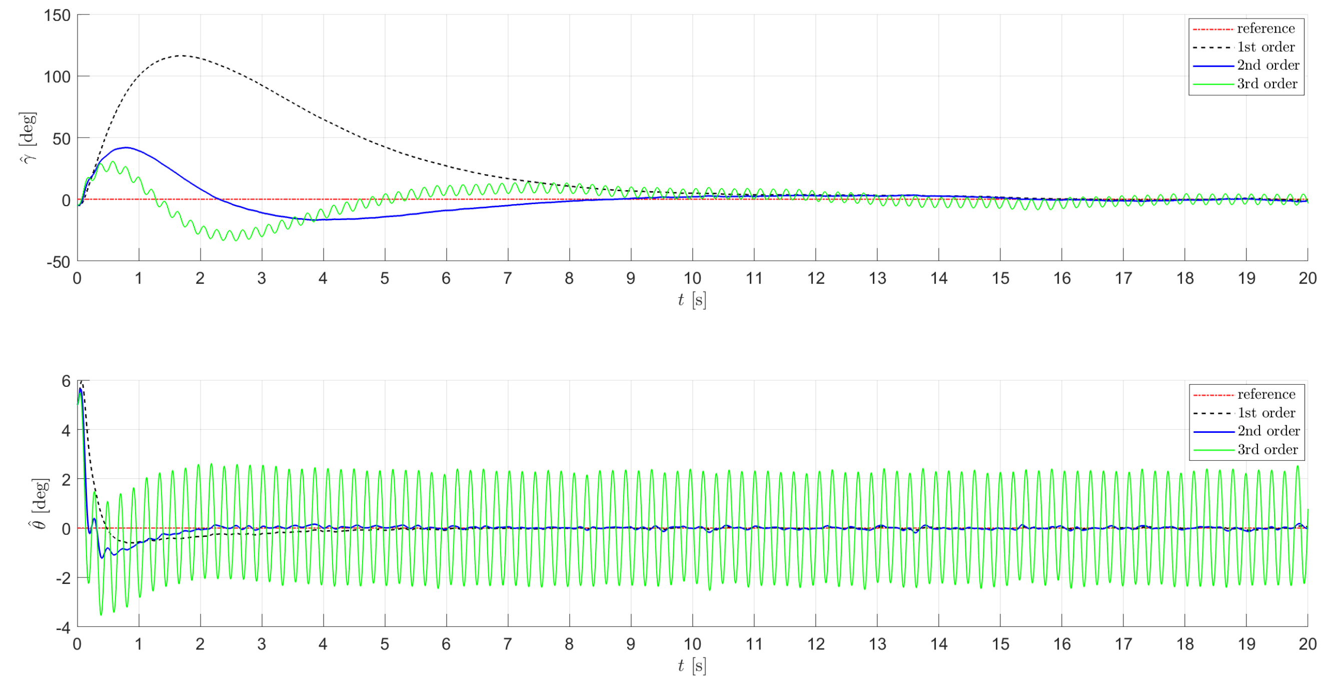


Figure 1: Comparison of nominal responses with the three filters with the same T_c

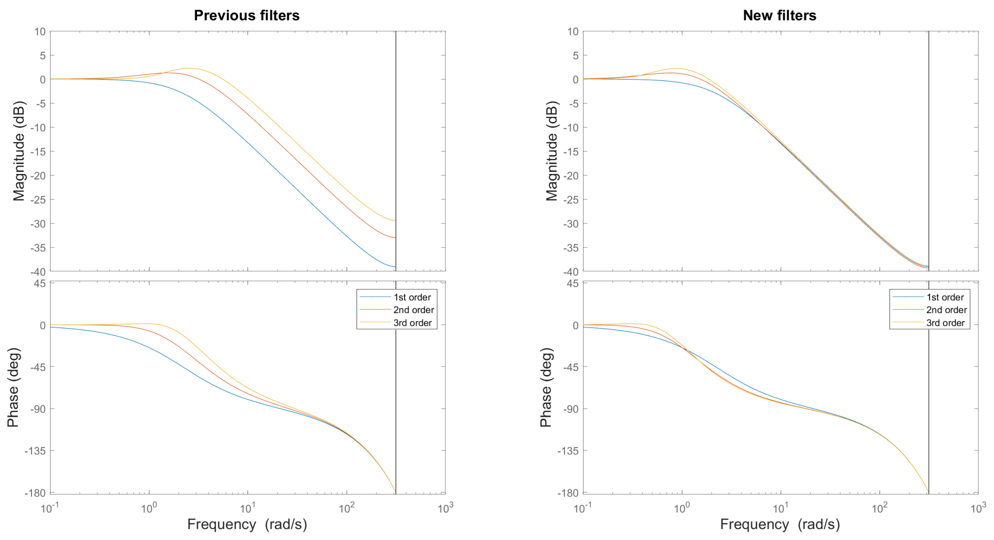


Figure 2: Filters Bode plots

The responses obtained in the stabilisation task are reported in Fig. (3). The system with the third order filter behaved better than in the previous case, but the second order still produced less oscillations around the reference. To reduce the number of degrees of freedom and simplify the analyses, in the following only the second order filter is employed, since it yields slightly better performances compared to the other two and it is able to reject the gyroscope bias while at the same time being simpler than the third order one.

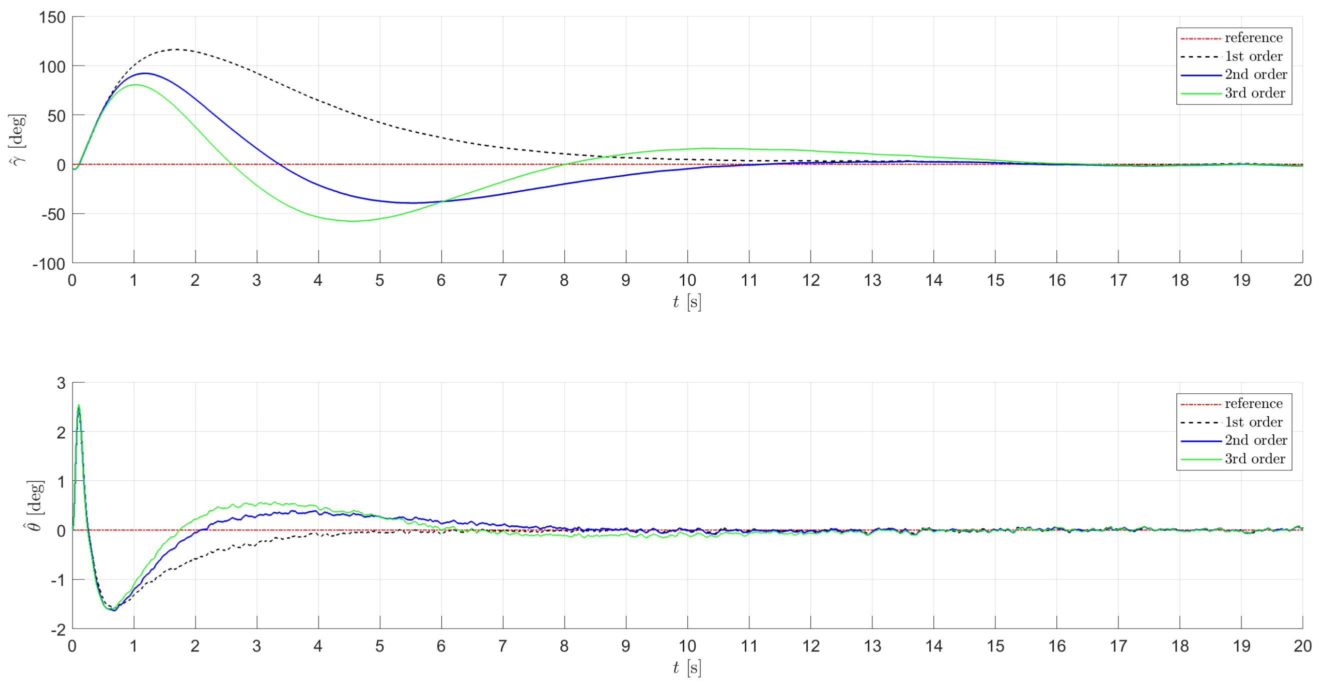


Figure 3: Comparison of nominal responses with the three filters with different T_c

In Figs. (4), (5) and (6) are reported the system responses in $\hat{\gamma}$ and $\hat{\theta}$ for the three tests described in the introduction to this Section for both values of ρ.

Both controllers are able to prevent the robot from falling and drive it to the original position with $\hat{\gamma} = 0°$. Between the two, the controller designed with $\rho = 500$ is faster in bringing $\hat{\gamma}$ to $0°$, with smaller overshoot but a slightly larger undershoot.

In the reference tracking case, the first controller is more reactive, driving the robot to the longitudinal position in less time and with smaller steady-state error.

As expected when dealing with nominal controllers, neither of the two is able to reject the constant $5V$ disturbance applied to the input. In the case $\rho = 500$, the system reaches a smaller steady-state value in $\hat{\gamma}$ with respect to the other case and brings $\hat{\theta}$ to zero faster.

In all three cases, the controller designed with $\rho = 500$ is more effective than the other. Characteristic values of the responses can be found in Tab. 2, where $\max|\hat{\gamma}|$ and $\max|\hat{\theta}|$ are the maximum absolute value of, respectively, the wheel angular displacement and the robot tilt angle, $\hat{\gamma}_{ss}$ is the steady-state value of $\hat{\gamma}$ computed as the average of the response in the last $5s$, $e_{ss} = \hat{\gamma}_{ss} - \gamma^*$ is the steady-state error, t_r is the rise time from 10% to 90% and M_p is the overshoot.

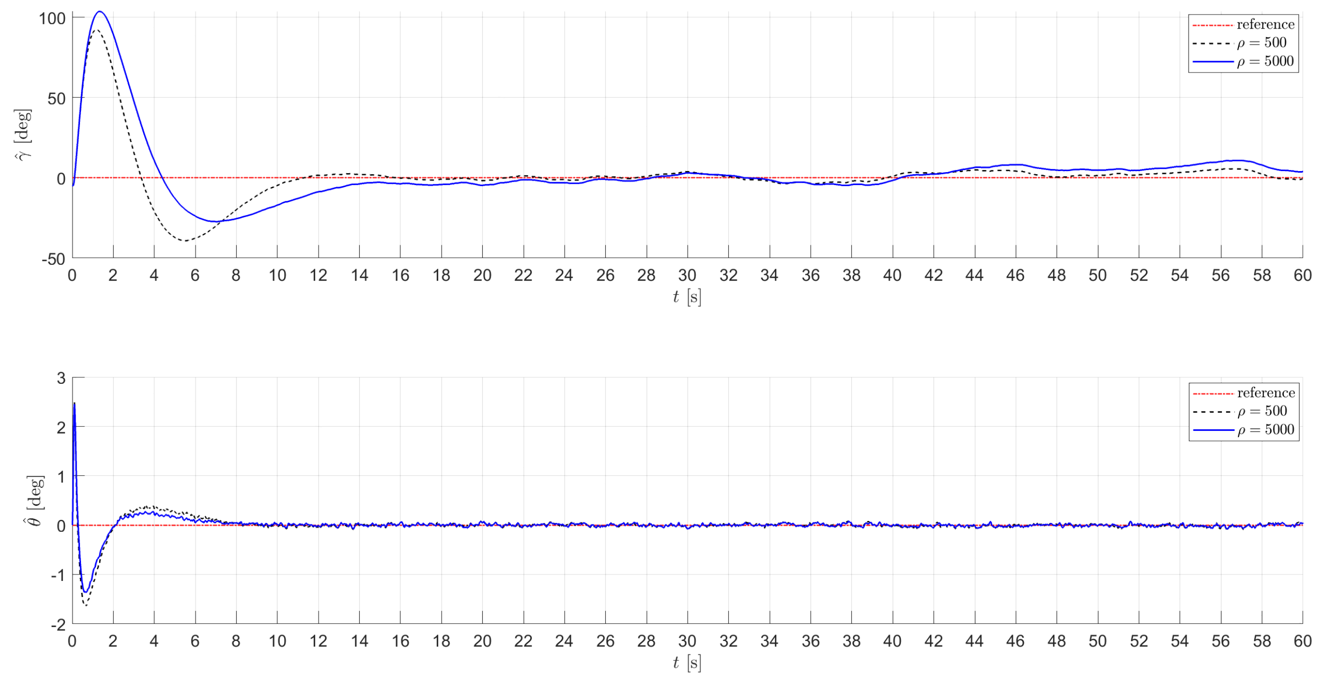


Figure 4: Nominal stabilisation

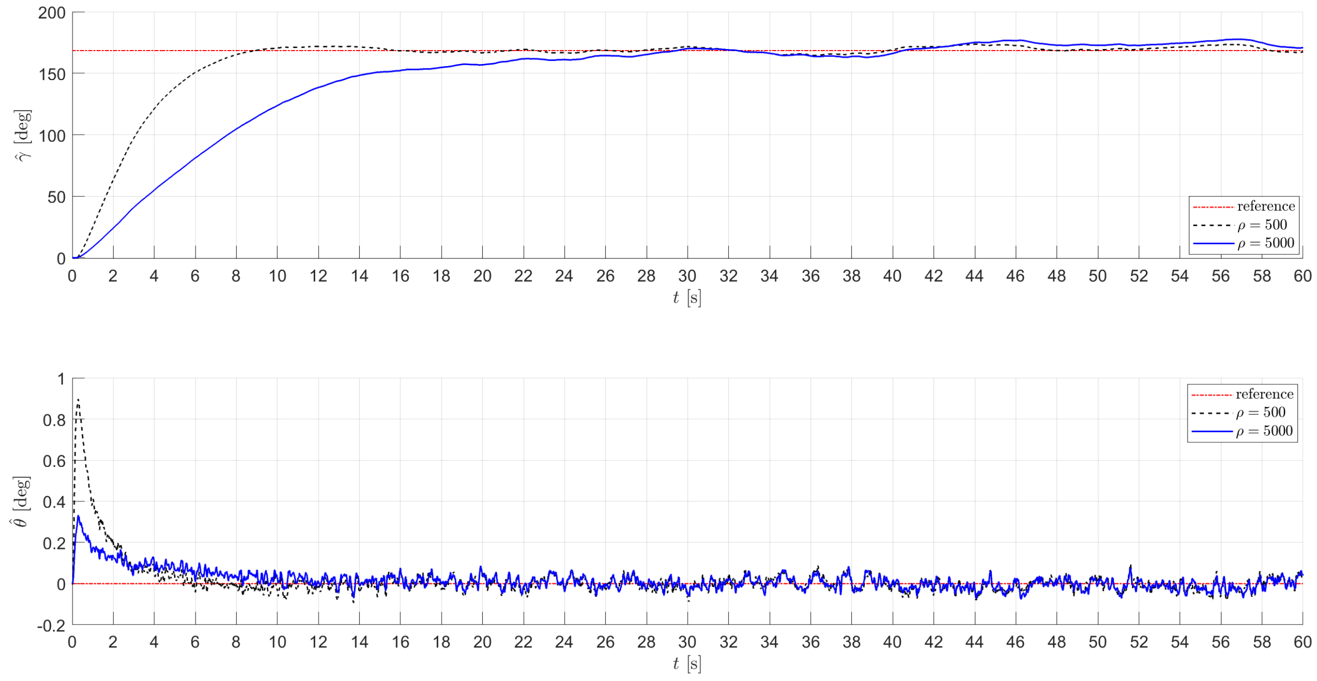


Figure 5: Nominal reference tracking

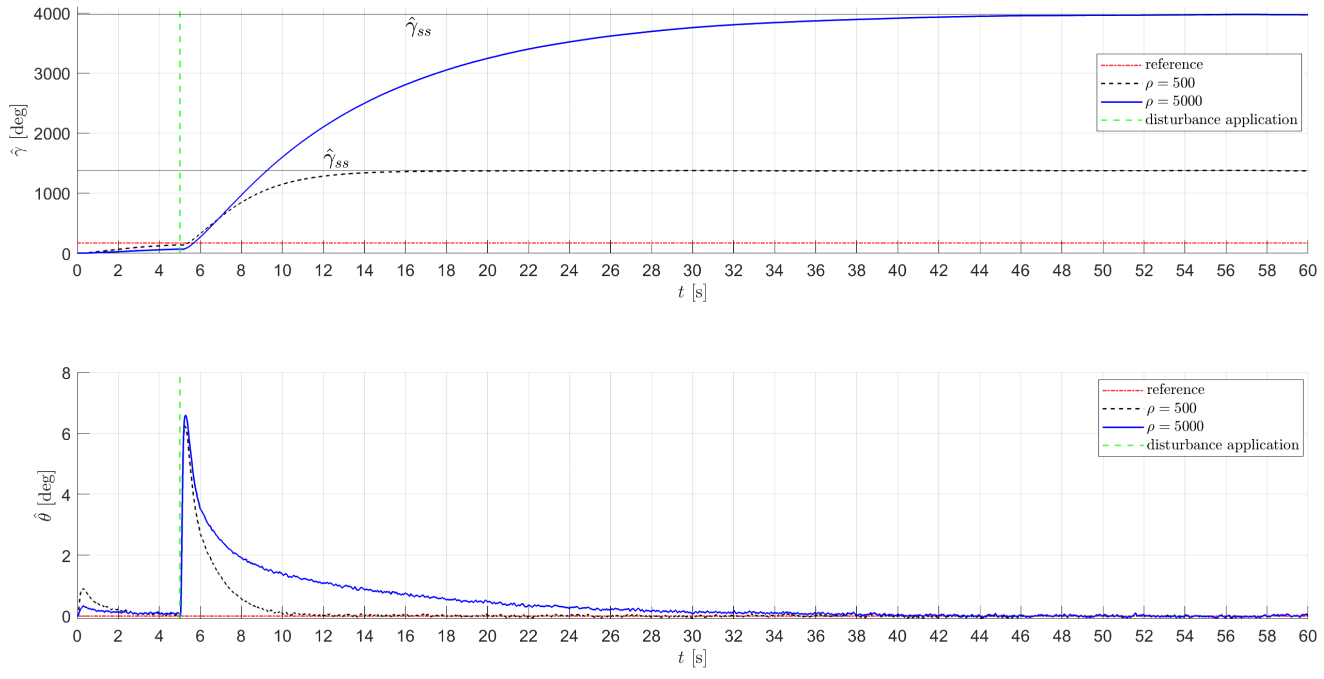


Figure 6: Nominal disturbance rejection

		$\rho = 500$	$\rho = 5000$
Test 1	$\max \lvert\hat{\gamma}\rvert$	$92.16°$	$103.6°$
	$\max \lvert\hat{\theta}\rvert$	$2.49°$	$2.45°$
Test 2	$\hat{\gamma}_{ss}$	$170.78°$	$174.9°$
	e_{ss}	$2.26°$	$6.38°$
	t_r	$5.3s$	$14.04s$
	M_p	3.14%	5.47%
	$\max \lvert\hat{\theta}\rvert$	$0.898°$	$0.33°$
Test 3	$\hat{\gamma}_{ss}$	$1374.4°$	$3971°$
	e_{ss}	$1205.9°$	$3802.5°$
	$\max \lvert\hat{\theta}\rvert$	$6.24°$	$6.6°$

Table 2: Nominal numerical performances

2.1.2 Robust State–Space Control

In order to achieve robust tracking of constant wheel angle position set-points, integral action has been added to the control law by performing state feedback on the usual augmented system $\Sigma_e = (\boldsymbol{A}_e, \boldsymbol{B}_e, \boldsymbol{C}_e, \boldsymbol{D}_e)$ with:

$$\boldsymbol{A}_e = \begin{bmatrix} 1 & \boldsymbol{H} \\ 0 & \boldsymbol{\Phi} \end{bmatrix} \quad \boldsymbol{B}_e = \begin{bmatrix} 0 \\ \boldsymbol{\Gamma} \end{bmatrix}$$
$$\boldsymbol{C}_e = \begin{bmatrix} 0 & \boldsymbol{C} \end{bmatrix} \quad \boldsymbol{D}_e = 0$$

The feedback matrix $\boldsymbol{K}_e = [K_I, \boldsymbol{K}]$ was again computed by solving a discrete-time LQR problem. For that purpose, the augmented cost matrix $\boldsymbol{Q_e} = \begin{bmatrix} q_{11} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Q} \end{bmatrix}$ was introduced and q_{11}, which weights the integrator state, was chosen between the two alternatives 0.1 and 1. The feed-forward gains $\boldsymbol{N}_x$ and N_u remained the same as in the nominal case. Tab.3 reports the values of K_I, $\boldsymbol{K}$, $\boldsymbol{N}_x$ and N_u for all possible combinations of parameters.

	$\rho = 500$ $q_{11} = 0.1$	$\rho = 500$ $q_{11} = 1.0$	$\rho = 5000$ $q_{11} = 0.1$	$\rho = 5000$ $q_{11} = 1.0$
K_I	-0.0130	-0.0409	-0.0041	-0.0130
$\boldsymbol{K}$	$\begin{bmatrix} -1.9197 \\ -65.1747 \\ -1.6809 \\ -7.3368 \end{bmatrix}^T$	$\begin{bmatrix} -4.1033 \\ -76.6611 \\ -2.2542 \\ -8.5964 \end{bmatrix}^T$	$\begin{bmatrix} -0.9348 \\ -58.6255 \\ -1.3841 \\ -6.6341 \end{bmatrix}^T$	$\begin{bmatrix} -1.8990 \\ -64.7419 \\ -1.6722 \\ -7.3042 \end{bmatrix}^T$
$\boldsymbol{N}_x$	$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T$	$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T$	$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T$	$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T$
N_u	0	0	0	0

Table 3: Robust Gains

As in the nominal case, the second order filter and the choice $\rho = 500$ lead to a more satisfactory behaviour, and therefore the following discussion focuses only on the effects due to q_{11}. In any case, Tab. 4 also reports some relevant metrics for the tests performed with $\rho = 5000$.

The results of the three experiments, shown in Figs. (7), (8) and (9), highlight how the robust controller is generally more effective than the nominal one, as it is able to bring both $\hat{\gamma}$ and $\hat{\theta}$ to the desired values more quickly. This is not surprising since the integral contribution is now added to the feed-forward action. Interestingly, however, a slight undershoot in $\hat{\gamma}$ is observed, as it is evident from Fig. (8).

Unlike in the nominal case, and as it is suggested by the internal model principle, the robust controller is able to reject the constant disturbance of $5V$ applied at the input of the motor, due to the additional feedback from the output to the integrator that allows to adjust the reference voltage once the disturbance is introduced.

The results of the experiments suggest that higher values of q_{11} lead to a more reactive response in both $\hat{\theta}$ and $\hat{\gamma}$. This is due to the fact that when a higher relative weight in the cost function is associated to the integrator state (i.e. the accumulated tracking error), the controller solving the LQ problem is constructed in such a way as to bring the error to zero faster, in order to keep the value of that state component smaller.

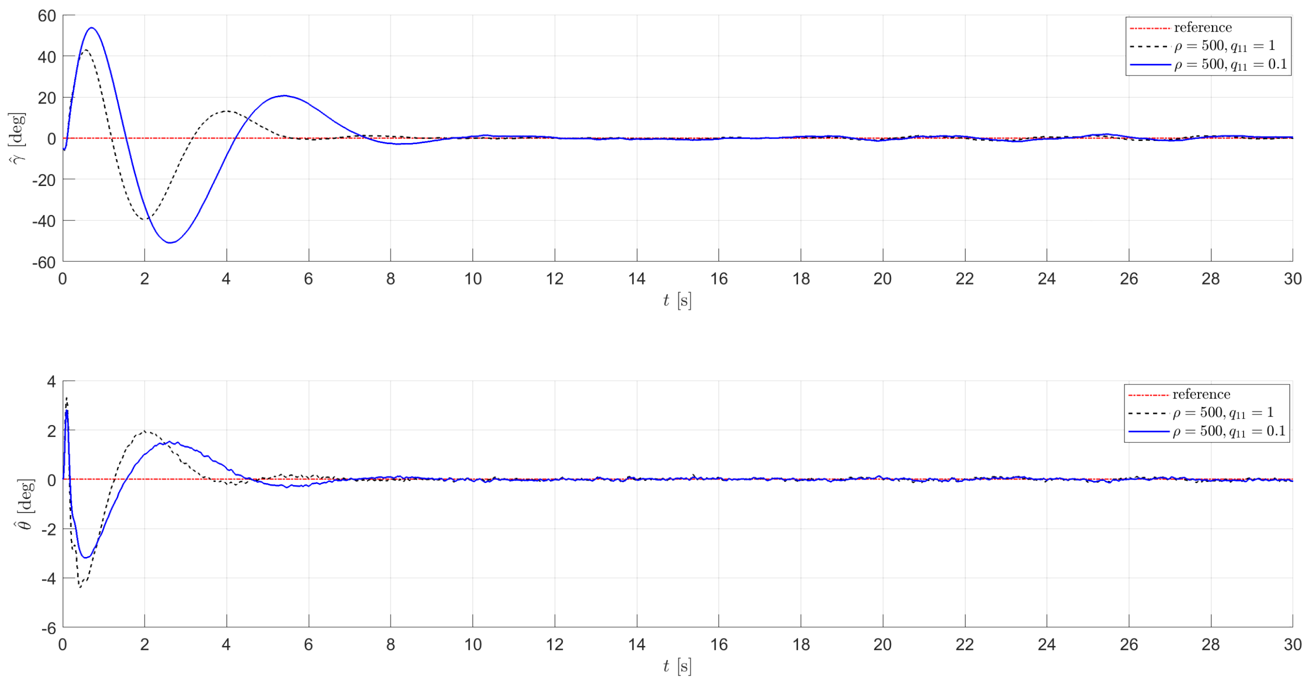


Figure 7: Robust stabilisation

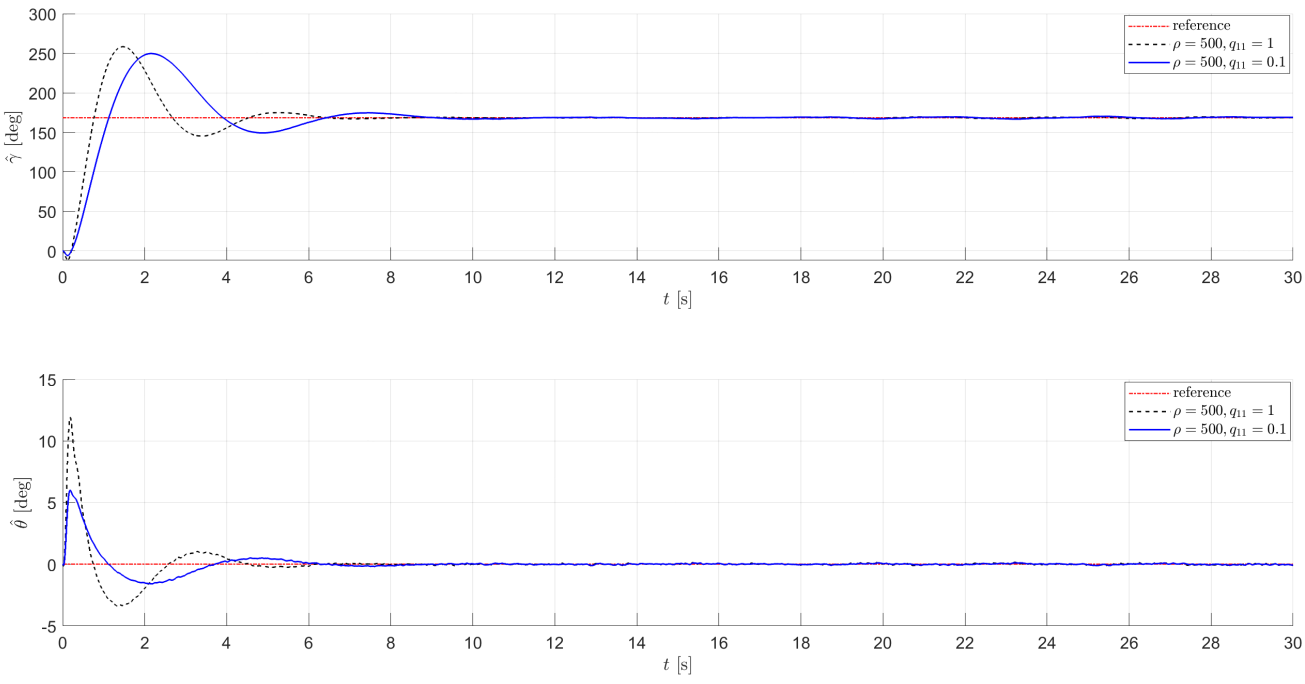


Figure 8: Robust reference tracking

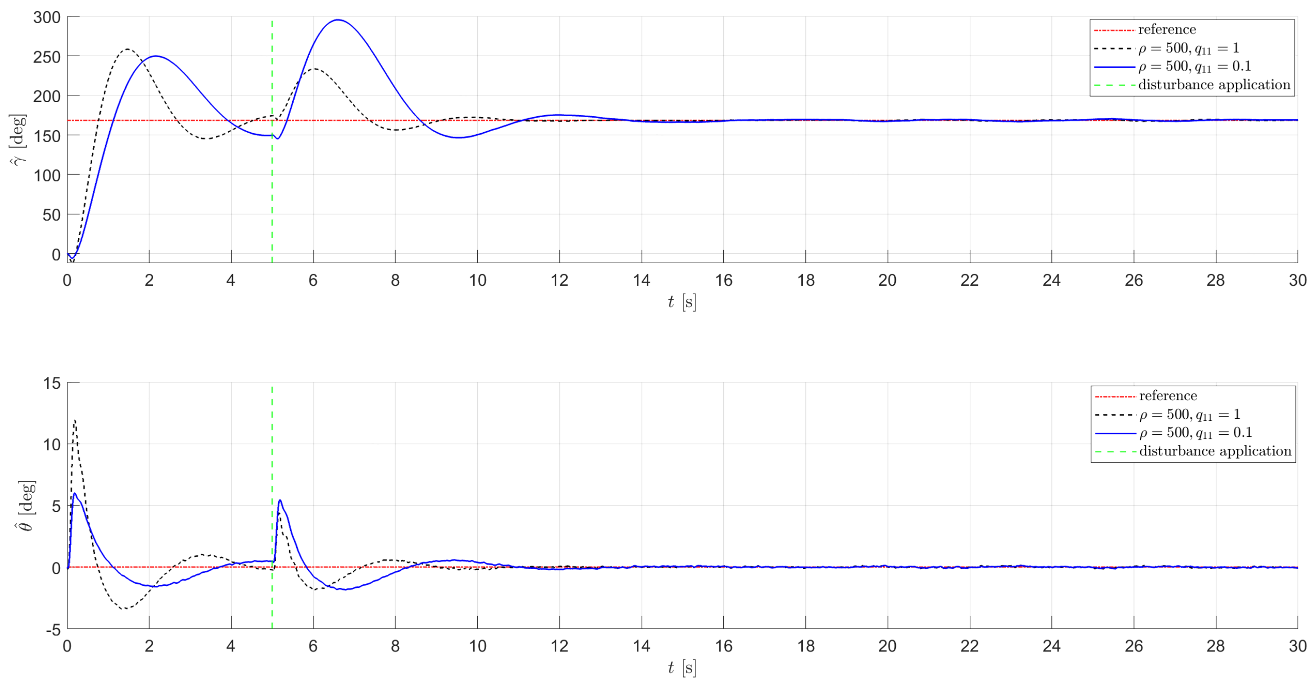


Figure 9: Robust disturbance rejection

		$\rho = 500$		$\rho = 5000$	
		$q_{11} = 0.1$	$q_{11} = 1.0$	$q_{11} = 0.1$	$q_{11} = 1.0$
Test 1	$\max \lvert\hat{\gamma}\rvert$	$53.79°$	$42.97°$	$67.64°$	$53.99°$
	$\max \lvert\hat{\theta}\rvert$	$3.19°$	$4.39°$	$2.61°$	$3.21°$
Test 2	$\hat{\gamma}_{ss}$	$168.07°$	$168.44°$	$167.60°$	$168.08°$
	e_{ss}	$-0.44°$	$-0.07°$	$-0.91°$	$-0.43°$
	t_r	$0.86s$	$0.43s$	$1.11s$	$0.69s$
	M_p	48.26%	53.44%	42.47%	48.60%
	$\max \lvert\hat{\theta}\rvert$	$6.0°$	$11.92°$	$3.19°$	$5.99°$
Test 3	$\hat{\gamma}_{ss}$	$168.07°$	$168.44°$	$167.63°$	$168.08°$
	e_{ss}	$-0.44°$	$-0.07°$	$-0.89°$	$-0.43°$
	$\max \lvert\hat{\theta}\rvert$	$6.00°$	$11.92°$	$5.53°$	$5.99°$

Table 4: Robust Numerical Performances

2.2 Laboratory assignments: experimental tests

Starting from the Simulink scheme built for the numerical assignments and following the procedure laid out in Section 6 of Handout 4, a new scheme was designed for real-time experiments in laboratory using the blocks in the Balancing Robot Toolbox (BRT) to be able to interface the controller with the robot. Hardware settings were configured based on *Laboratory guide 2*, with conversion for relevant signals to `single` so that transmission speed could be increased while maintaining satisfactory numeric precision for the measurements.

The push-button scheme proposed in the Handout to enable the controller could not be made to work, so it was deemed necessary to substitute it with a step signal activated after a certain delay to let first the robot connect via Bluetooth to the computer. In some cases, this meant that the robot started moving slightly before the connection was established, so some initial data were lost. While it was deemed acceptable for reference tracking and disturbance rejection since those signals were applied always after the robot had connected, for the stabilisation part the starting delay was increased to prevent losing some parts of the measurements.

Most of the experiments reported in the following were performed on `Robot 2`, although the stabilisation ones were done on `Robot 5` since the Bluetooth module was repeatedly failing to connect and measurements could not be taken.

The sensor interface was modified to take into account that the robot has one incremental encoder for each motor, so the angular displacement $\Delta\theta_{rot}$ is computed as the average of the two measurements $\Delta\theta_{rot} = \frac{\Delta\theta_{rot,r}+\Delta\theta_{rot,l}}{2}$.

The Simulink schemes used for the laboratory tests are reported in Appendix A.4.

2.2.1 Nominal state-space control

The same experiments performed on the numerical model were repeated in the laboratory on the real balancing robot. Since experimental results confirmed what was expected from numerical tests, that is, that $\rho = 500$ provided better performances, the tests were done using the corresponding controller in Tab. 1.

As before, the first experiment carried out was about the stabilisation of the robot in the upward position from a small enough initial angle θ. To have a clear picture of the behaviour of the robot, all three orders of the complementary filter were tested to assess the best among the three. In Fig. (10) are reported the responses of the system both in $\hat{\gamma}$ and $\hat{\theta}$ with the three filters.

All three controllers are able to bring and maintain $\hat{\theta}$ close to zero with some oscillations around the upward equilibrium position. Among the three designed filters, $\hat{\gamma}$ is driven to $0°$ only when the second order one is employed. In the other two cases, there is a non-zero constant error to the reference, as one should expect when using a nominal design on a real system, since there may be some violated assumptions that make the model not valid anymore.

When applying a reference of $\gamma^* = 168.51°$ to make the robot move forward by $10cm$, as reported in Fig. (11), the controller is still able to prevent the robot from falling, but it fails to track the step reference since the response has a non-zero error.

The last test reported in Fig. (12) is disturbance rejection, where a $5V$ constant signal was applied to both motors five seconds after having enabled the step reference to give the robot time to settle around the new position. As it is to be expected in the nominal case, the robot is not able to reject the disturbance and $\hat{\gamma}$ starts to drift away from the given reference. Still, this constant input does not prevent the controller from regulating $\hat{\theta}$ to $0°$.

Summarising results are reported in Tab. 5. For the Test 3 results, neither the steady-state value $\hat{\gamma}_{ss}$ nor the steady-state error e_{ss} are provided since the experiment had to be stopped before the robot had reached the final settling position. Also, since in Test 2 the robot position at the application of the reference signal was close to the desired set-point, it would make little sense to compute rise time and overshoot, so these values are not provided either.

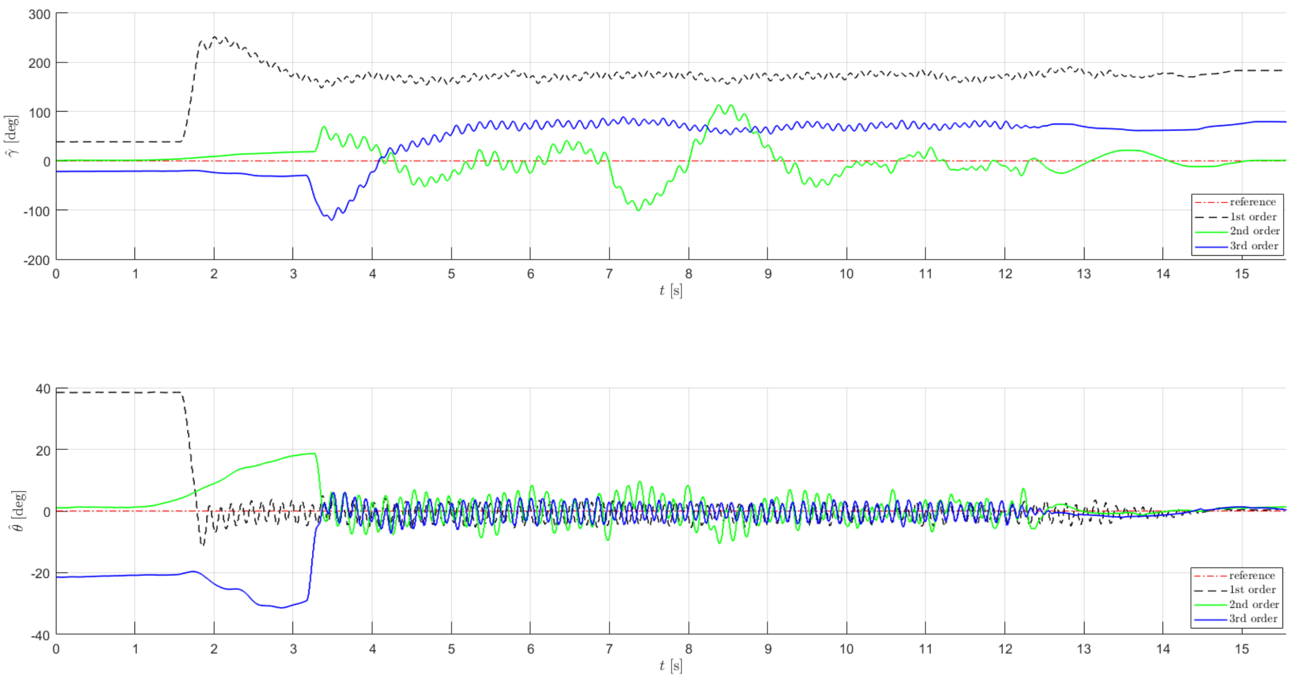


Figure 10: Stabilisation of the robot with nominal controller

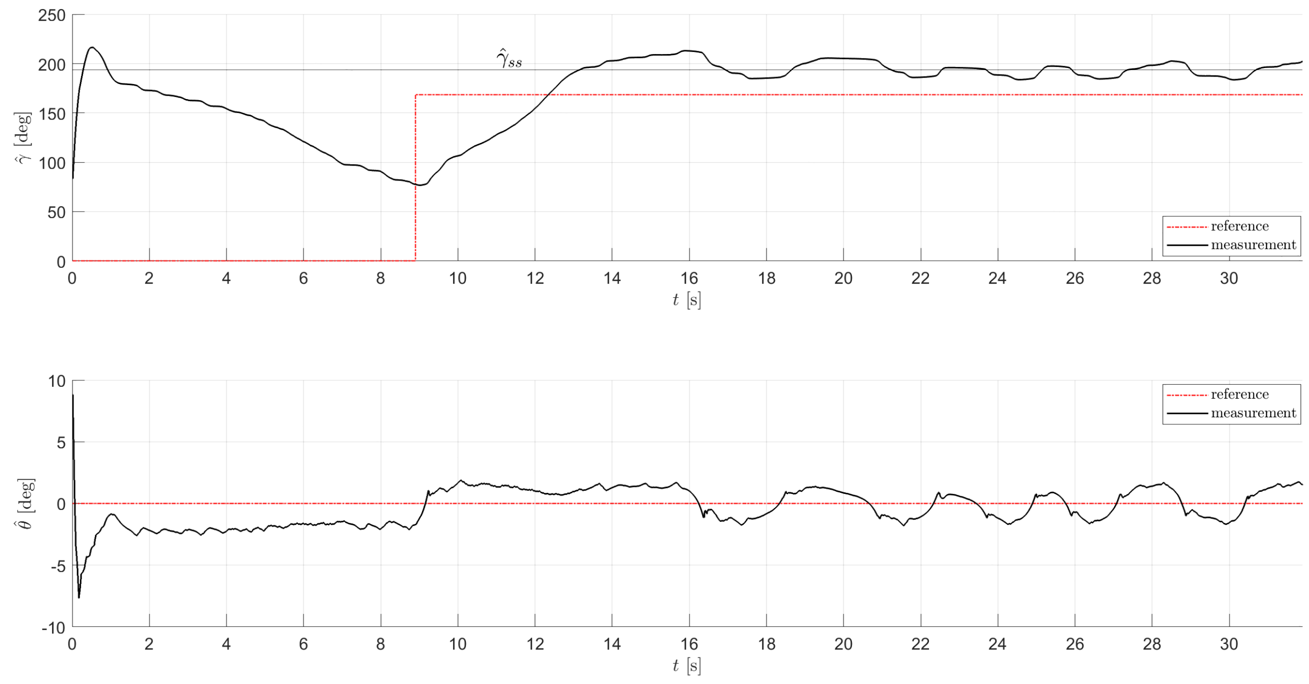


Figure 11: Reference tracking with nominal controller

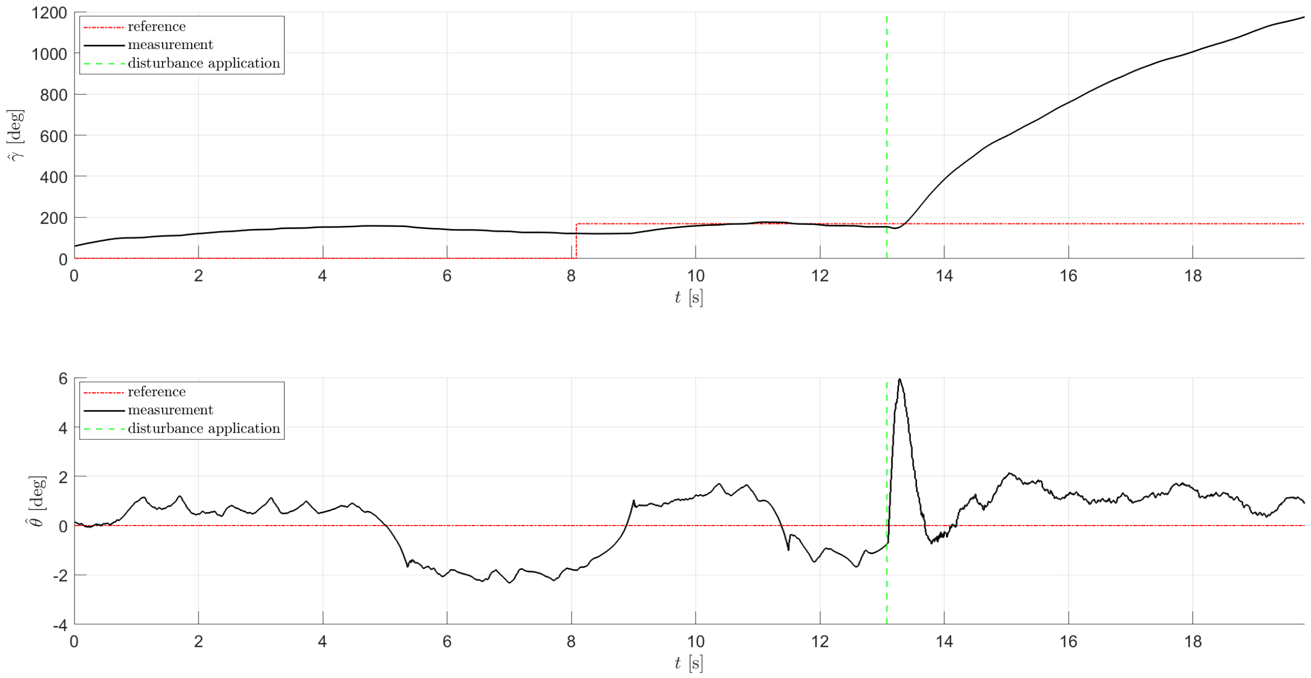


Figure 12: Disturbance rejection with nominal controller

		Numerical	Experimental
Test 1	$\max\lvert\hat{\gamma}\rvert$	$97.16°$	$113.61°$
	$\max\lvert\hat{\theta}\rvert$	$7.49°$	$18.69°$
Test 2	$\hat{\gamma}_{ss}$	$170.78°$	$193.64°$
	e_{ss}	$2.26°$	$25.13°$
	t_r	$5.3s$	-
	M_p	3.14%	-
	$\max\lvert\hat{\theta}\rvert$	$0.898°$	$8.83°$
Test 3	$\hat{\gamma}_{ss}$	$1374.4°$	-
	e_{ss}	$1205.9°$	-
	$\max\lvert\hat{\theta}\rvert$	$6.24°$	$5.96°$

Table 5: Performance metrics of the nominal controller with $\rho = 500$

2.2.2 Robust state-space control

The controllers tested in this section are those reported in Tab. 3. As discussed before, the second-order complementary filter turned out to be the most effective, and therefore all figures referenced in the following discussion are related to experiments carried out with that choice of filter.

In the balancing task, the parameter q_{11} turned out to have little effect on the robot's behaviour. This is not unexpected, since for stabilisation alone the contribution of the integrator is not significant. For this reason, in the experiments shown in Fig. (13) the value of q_{11} was fixed to $q_{11} = 1$, and the system's response was compared between the two possible choices of ρ.

Contrary to what observed in the nominal case, setting $\rho = 5000$ gave better performances. In fact, as it is apparent from the figure, the controller obtained with $\rho = 500$ induces aggressive oscillations around the vertical position, which are undesirable because they make any kind of interaction with the robot more difficult. Moreover, this type of behaviour would likely produce unnecessary wear on the physical components.

The tests related to tracking and disturbance rejection were performed keeping fixed $\rho = 5000$, in light of the discussion above, while instead varying q_{11}. The results are shown in Fig. (14) and Fig. (15) respectively. As can be seen from Fig. (15), introducing an integrator in the direct chain allows to successfully reject constant disturbances and track constant references, which is in agreement with what the internal model principle prescribes.

As expected from theoretical considerations and from the simulations, a higher value of q_{11} improves the reaction time of the controller, both in response to the input and to the disturbance, while however making the movement of the robot less smooth, as can be seen from the plot of $\hat{\theta}$. Therefore, the more appropriate choice of q_{11} would depend on the specific application and on the corresponding desired trade-off between responsiveness of the controller and smoothness of the robot's movement.

Tab. 6 reports some relevant metrics resulting from the three experiments performed with $\rho = 5000$ and compares them with those obtained from the corresponding numerical tests.

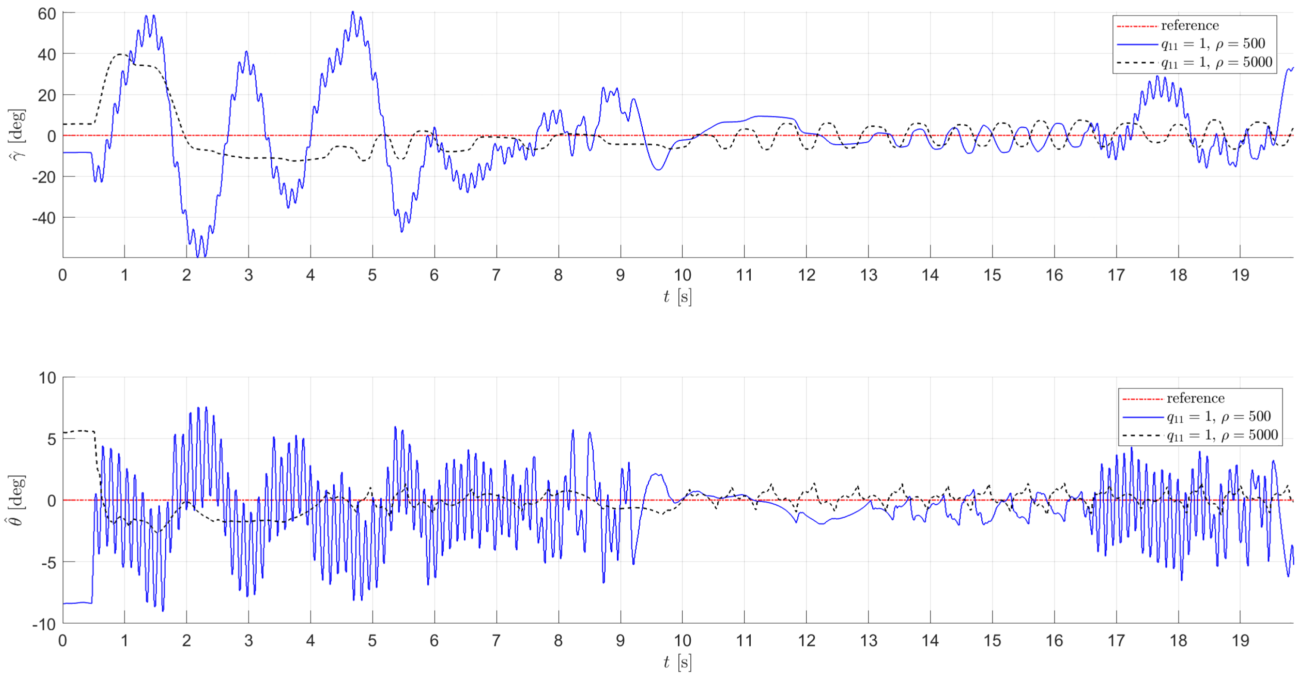


Figure 13: Comparison of the results of the stabilisation experiment with the robust controller.

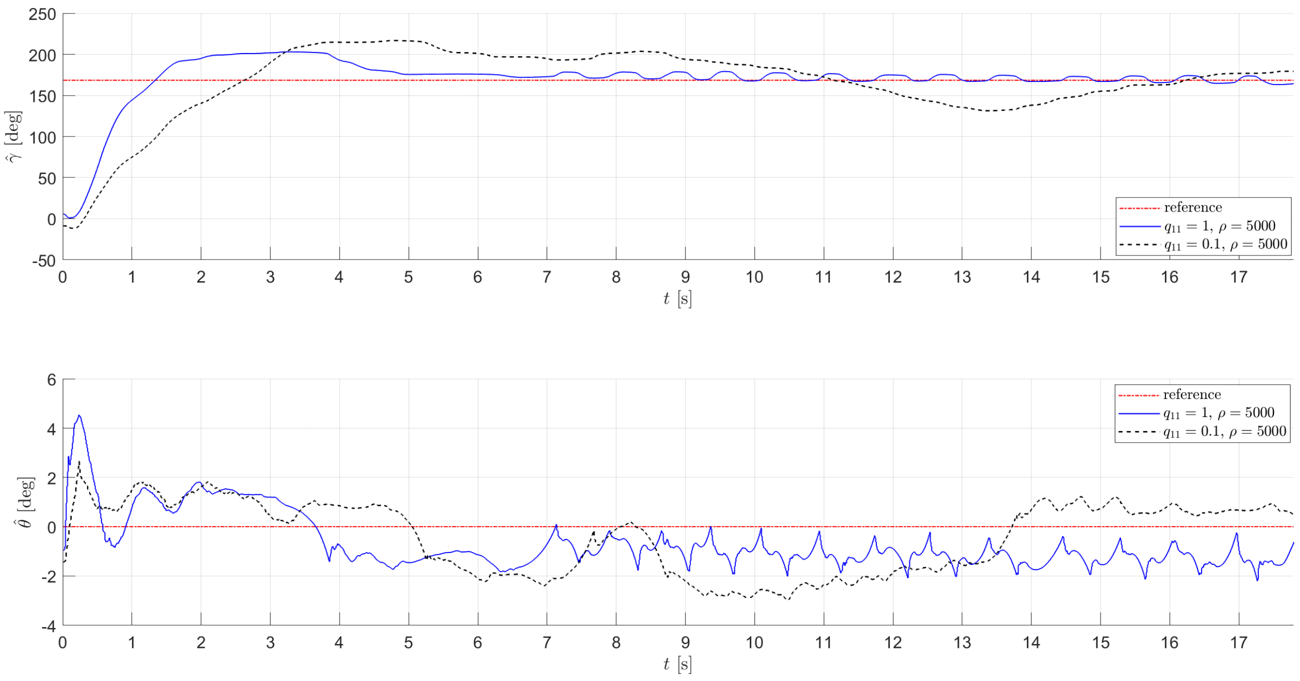


Figure 14: Comparison of the results of the tracking experiment with the robust controller.

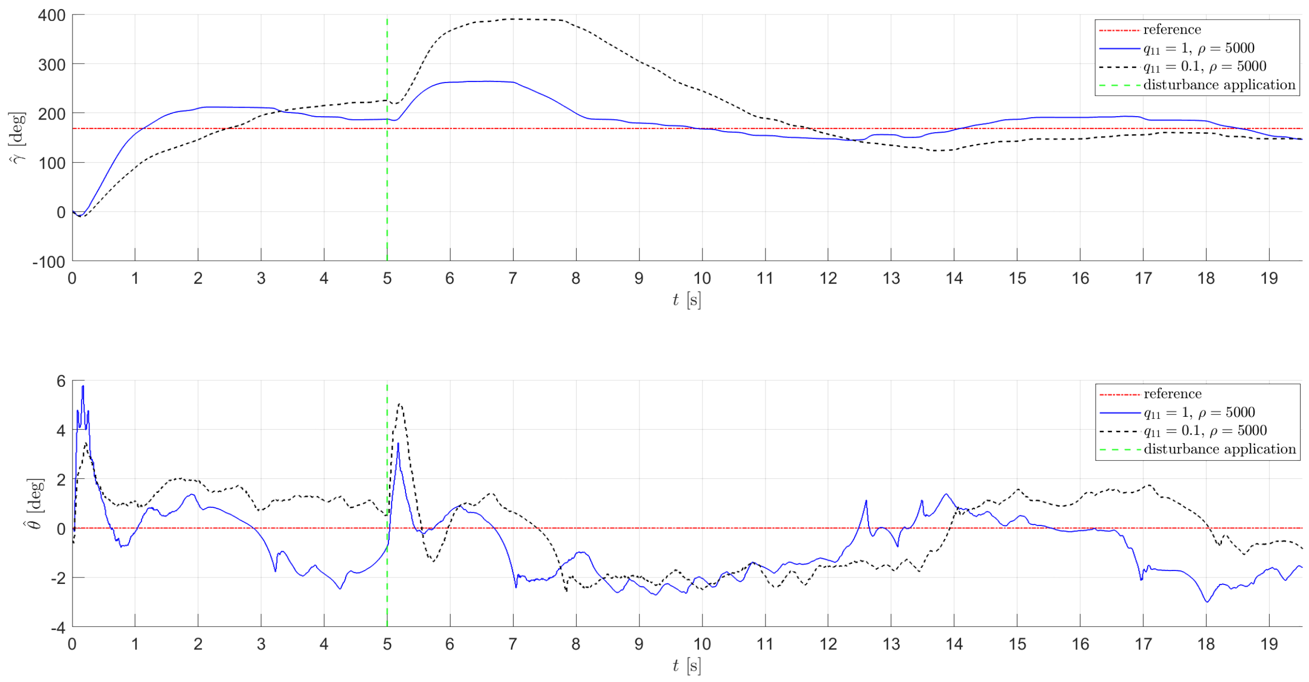


Figure 15: Comparison of the results of the disturbance rejection experiment with the robust controller.

2.2.3 Remarks

Looking at the plots showing the results of the laboratory experiments it is apparent how the robust controller was much more effective than the nominal one, with the difference being even more apparent than in simulation. The essential reason for such stark contrast is the fact that the nominal approach yields a controller that works partially in open loop, and therefore heavily relies on having a sound model of the system to be controlled. This was not the case in our context since a lot of simplifying assumptions were made in the derivation of the dynamics of the robot, and therefore to obtain a more satisfactory performance it was necessary to employ a fully closed loop strategy.

When comparing laboratory results with the corresponding simulations, it can be noticed an unexpected phenomenon in $\hat{\gamma}$ in the real responses. Indeed, differently from the simulations, after settling around a value, the robot started to oscillate back and forth without reaching a stable condition, most likely due to the presence of static friction, which was not modelled in the Simulink scheme for numerical experiments. This can be described as a dead-zone phenomenon, where control signals under a certain threshold do not have any effect, so the controller is forced to generate a larger action on the system which in this case produces the oscillation.

		$\rho = 5000$, $q_{11} = 1$		$\rho = 5000$, $q_{11} = 0.1$	
		Numerical	Experimental	Numerical	Experimental
Test 1	$\max\|\hat{\gamma}\|$	$53.99°$	$39.59°$	$67.64°$	$25.82°$
	$\max\|\hat{\theta}\|$	$3.21°$	$6.42°$	$2.61°$	$16.76°$
Test 2	$\hat{\gamma}_{ss}$	$168.08°$	$171.86°$	$167.60°$	$168.71°$
	e_{ss}	$-0.43°$	$3.35°$	$-0.91°$	$0.2°$
	t_r	$0.69s$	$0.81s$	$1.11s$	$1.82s$
	M_p	48.60%	20.56%	42.47%	28.75%
	$\max\|\hat{\theta}\|$	$5.99°$	$4.53°$	$3.19°$	$2.97°$
Test 3	$\hat{\gamma}_{ss}$	$168.08°$	$176.71°$	$167.73°$	$173.50°$
	e_{ss}	$-0.43°$	$8.2°$	$-0.89°$	$4.99°$
	$\max\|\hat{\theta}\|$	$5.99°$	$5.78°$	$5.53°$	$5.04°$

Table 6: Performance metrics of the robust controller.

2.2.4 Yaw compensation

Throughout the experience, all controllers were designed assuming that whenever the same voltage was applied to the two motors, the robot would move in a perfectly straight line. This however is not the case for a number of reasons, among which are the facts that the motors are not exactly equal and the wheels may slip in an asymmetric way.

In fact, although with an appropriate choice of controller all tasks were completed successfully, in the laboratory it was very apparent that the robot couldn't actually move in a straight line, since the yaw angle ψ would accumulate a certain amount of drift.

In order to compensate for this phenomenon, a PI controller providing a differential contribution was implemented as described in section 6.4 of the Handout for this experience. The resulting block scheme is reported in section A.4 of the Appendix.

Fig. (16) shows the improvements obtained in the disturbance rejection task by adding yaw compensation to the baseline robust controller with $\rho = 5000$ and $q_{11} = 1$, using the second order complementary filter. This task was chosen for the comparison since it is the most complex one.

The figures show that introducing yaw compensation not only allows to keep the robot's orientation close to the desired one at all times, but also leads to smaller oscillations around the desired set-point in $\hat{\theta}$ and, especially, in $\hat{\gamma}$. This is likely due to the fact that once the PI differential contribution is added to the architecture, the longitudinal approximation becomes reasonable and therefore the controller that was designed relying on that assumption is more effective.

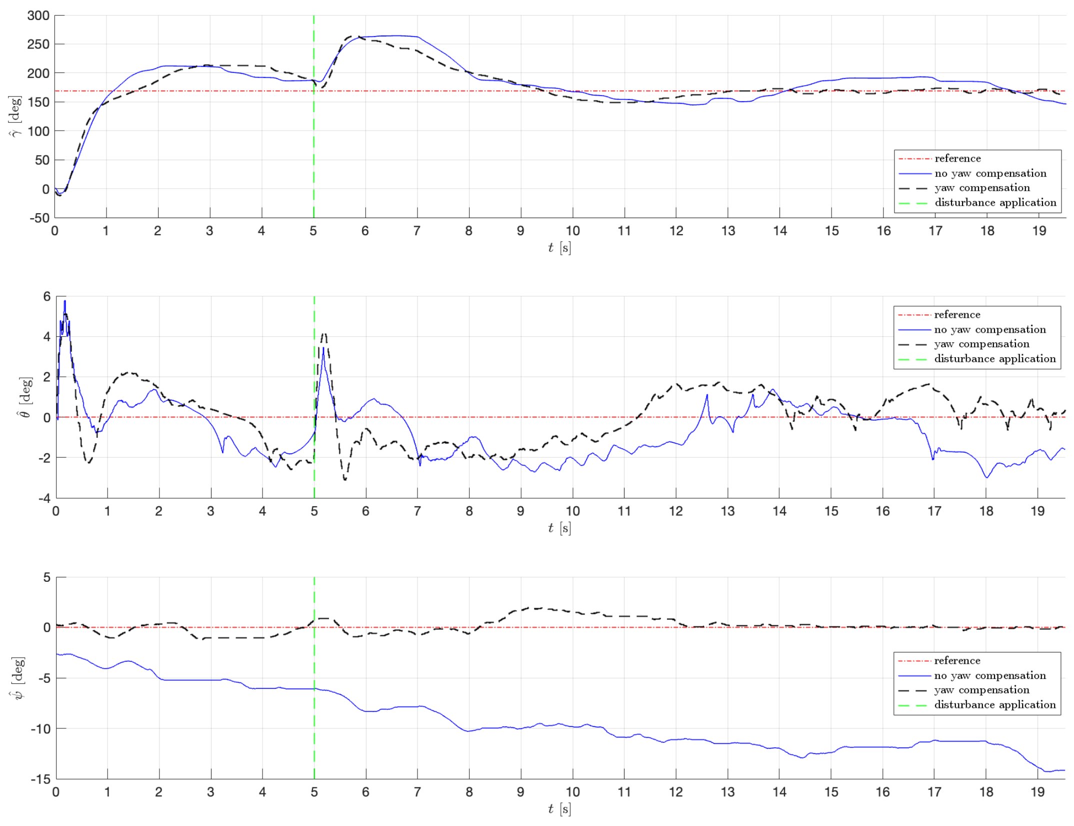


Figure 16: Disturbance rejection task. Evaluation of the improvements obtained adding yaw compensation.

3 Conclusion

The activity was divided into two main phases. During the design and testing phase, carried out in simulation, the nominal approach for tracking of constant references was compared to its robust counterpart, exploiting a simplified model of the robot. These numerical tests gave a first impression of the most appropriate choice of parameters and of the overall behaviour of the two controllers, which were then deployed on a real balancing robot in the second phase.

However, given its complexity, the real device presented a certain number of non-idealities, such as non symmetric motors or cambered wheels, that were not considered during the modelling and design process, and therefore the results of some experiments were quite different from what was observed in simulation.

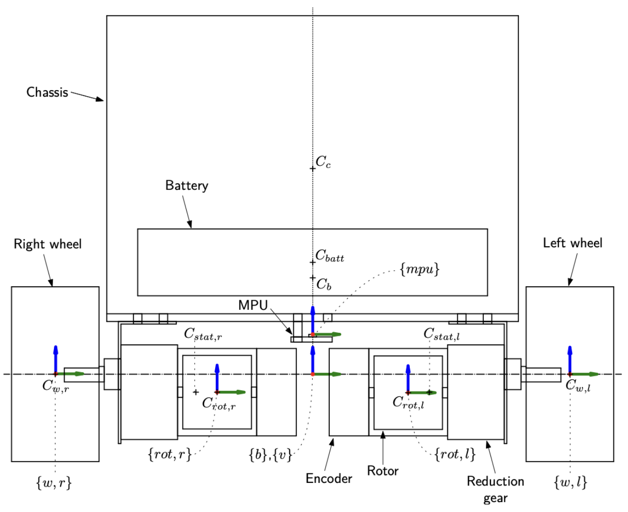
In summary, the nominal controller was unsuccessful in completing all three tasks, differently from the robust one whose performance was instead satisfactory, especially after yaw compensation was introduced.

A Appendix

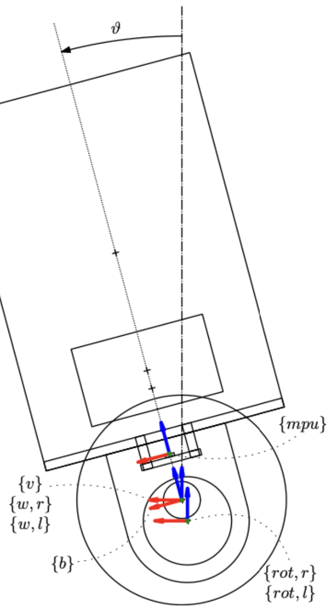
A.1 Data sheet

▪ **Robot body**			
Centre–of–Mass coords wrt body frame {b}	x_b^b, y_b^b, z_b^b	0, 0, 46.05	[mm]
Mass	m_b	1.06	[kg]
Principal Moments–of–Inertia	$I_{b,xx}, I_{b,yy}, I_{b,zz}$	4.22, 2.20, 2.65	[gm^2]
↳**Robot chassis**			
Dimensions (width, height, depth)	w_c, h_c, d_c	160, 119, 80	[mm]
Centre–of–Mass coords wrt body frame {b}	x_c^b, y_c^b, z_c^b	0, 0, 80	[mm]
Mass	m_c	456	[g]
Principal Moments–of–Inertia	$I_{c,xx}, I_{c,yy}, I_{c,zz}$	1.5, 0.78, 1.2	[gm^2]
↳**Battery**			
Dimensions (width, height, depth)	$w_{batt}, h_{batt}, d_{batt}$	136, 26, 44	[mm]
Centre–of–Mass coords wrt body frame {b}	$x_{batt}^b, y_{batt}^b, z_{batt}^b$	0, 0, 44	[mm]
Mass	m_{batt}	320	[g]
Principal Moments–of–Inertia	$I_{batt,xx}, I_{batt,yy}, I_{batt,zz}$	0.51, 0.07, 0.06	[gm^2]
↳**DC gearmotor stator**			
Dimensions (height, radius)	h_{stat}, r_{stat}	68.1, 17	[mm]
Centre–of–Mass coords wrt body frame {b}	$x_{stat}^b, y_{stat}^b, z_{stat}^b$	0, ±52.1, -7	[mm]
Mass	m_{stat}	139.75	[g]
Principal Moments–of–Inertia	$I_{stat,xx} = I_{stat,zz}, I_{stat,yy}$	0.064, 0.02	[gm^2]
▪ **DC gearmotor rotor**			
Dimensions (height, radius)	h_{rot}, r_{rot}	30.7, 15.3	[mm]
Centre–of–Mass coords wrt body frame {b}	$x_{rot}^b, y_{rot}^b, z_{rot}^b$	0, ±42.7, -7	[mm]
Mass	m_{rot}	75.25	[g]
Principal Moments–of–Inertia	$I_{rot,xx} = I_{rot,zz}, I_{rot,yy}$	0.01, 0.009	[gm^2]
▪ **Wheels**			
Dimensions (height, radius)	h_w, r_w	26, 34	[mm]
Centre–of–Mass coords wrt body frame {b}	x_w^b, y_w^b, z_w^b	0, ±100, 0	[mm]
Mass	m_w	50	[g]
Principal Moments–of–Inertia	$I_{w,xx} = I_{w,zz}, I_{w,yy}$	0.017, 0.029	[gm^2]
▪ **Viscous friction nominal parameters**			
Gearbox viscous friction coef.	B	$25 \cdot 10^{-3}$	$[\frac{Nm}{rad/s}]$
Wheel viscous friction coef.	B_w	$1.5 \cdot 10^{-3}$	$[\frac{Nm}{rad/s}]$
▪ **DC gearmotor nominal parameters**			
Armature resistance	R_a	2.4	$[\Omega]$
Electric (BEMF) constant	k_e	$10.3 \cdot 10^{-3}$	$[\frac{Vs}{rad}]$
Torque constant	k_t	$5.2 \cdot 10^{-3}$	$[\frac{Nm}{A}]$
Gearbox ratio	N	30	
▪ **MPU geometrical parameters**			
Sensor centre wrt body frame	$x_{mpu}^b, y_{mpu}^b, z_{mpu}^b$	$0, 0, 13.5$	[mm]

A.2 Robot scheme



(a) Front view.



(b) Side View

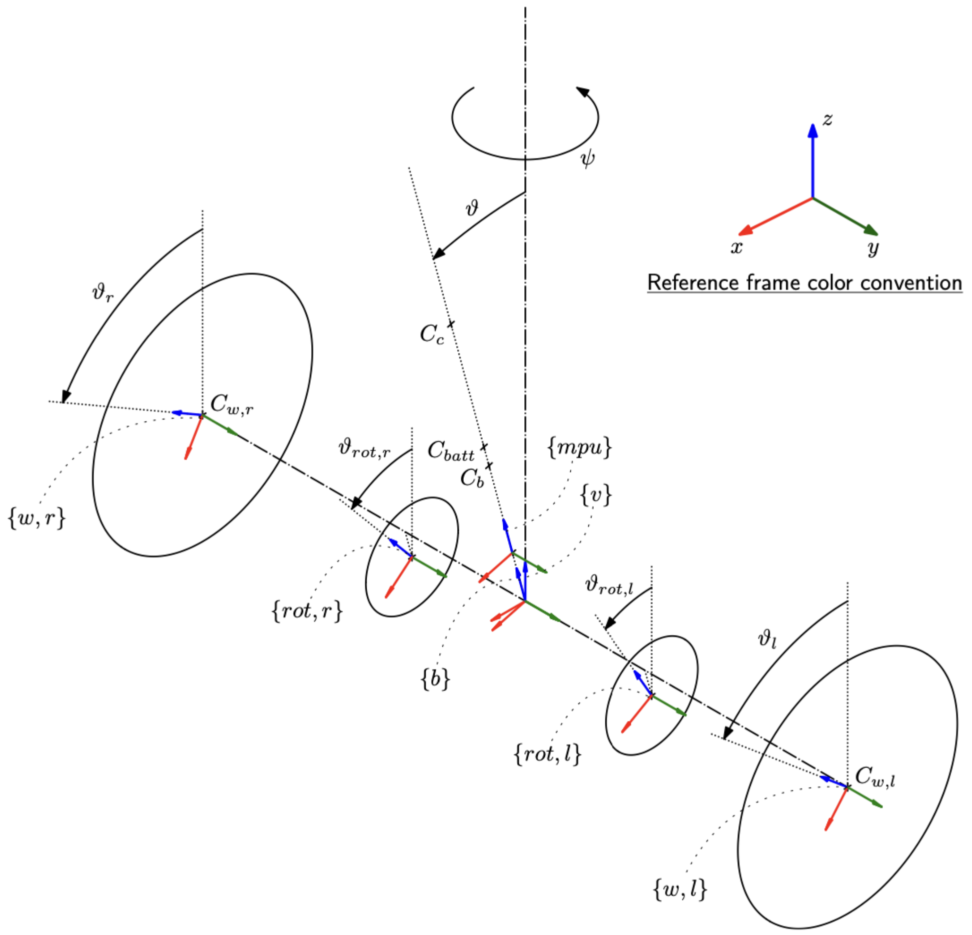


Figure 18: Simplified 3D view.

A.3 Matlab code

```
%% General parameters and conversion gains
% controller sampling time
Ts = 1e-2;

% gravity acc [m/s^2]
g = 9.81;

% conversion gains
rpm2rads = 2*pi/60; % [rpm] -> [rad/s]
rads2rpm = 60/2/pi; % [rad/s] -> [rpm]
rpm2degs = 360/60; % [rpm] -> [deg/s]
degs2rpm = 60/360; % [deg/s] -> [rpm]
deg2rad = pi/180; % [deg] -> [rad]
rad2deg = 180/pi; % [rad] -> [deg]
g2ms2 = g; % [acc_g] -> [m/s^2]
ms22g = 1/g; % [m/s^2] -> [acc_g]
ozin2Nm = 0.706e-2; % [oz*inch] -> [N*m]

% robot initial condition
x0 = [0, % gam(0)
      5 * deg2rad, % th(0)
      0, % dot_gam(0)
      0]; % dot_th(0)

%% DC motor data
% motor id: brushed DC gearmotor Pololu 30:1 37Dx68L mm
% electromechanical params
mot.UN = 12; % nominal voltage
mot.taus = 110 / 30 * ozin2Nm; % stall torque @ nom voltage
mot.Is = 5; % stall current @ nom voltage
mot.w0 = 350 * 30 * rpm2rads; % no-load speed @ nom voltage
mot.I0 = 0.3; % no-load current @ nom voltage

mot.R = mot.UN / mot.Is; % armature resistance
mot.L = NaN; % armature inductance
mot.Kt = mot.taus / mot.Is; % torque constant
mot.Ke = (mot.UN - mot.R * mot.I0) / (mot.w0); % back-EMF constant
mot.eta = NaN; % motor efficiency
mot.PN = NaN; % nominal output power
mot.IN = NaN; % nominal current
mot.tauN = NaN; % nominal torque

% dimensions
mot.rot.h = 30.7e-3; % rotor height
mot.rot.r = 0.9 * 17e-3; % rotor radius

mot.stat.h = 68.1e-3; % stator height
mot.stat.r = 17e-3; % stator radius

% center of mass (CoM) position
mot.rot.xb = 0; % (left) rot CoM x-pos in body frame
mot.rot.yb = 42.7e-3; % (left) rot CoM y-pos in body frame
mot.rot.zb = -7e-3; % (left) rot CoM z-pos in body frame

mot.stat.xb = 0; % (left) stat CoM x-pos in body frame
mot.stat.yb = 52.1e-3; % (left) stat CoM y-pos in body frame
mot.stat.zb = -7e-3; % (left) stat CoM z-pos in body frame

% mass
mot.m = 0.215; % total motor mass
mot.rot.m = 0.35 * mot.m; % rotor mass
```

```matlab
mot.stat.m = mot.m - mot.rot.m; % stator mass

% moment of inertias (MoI) wrt principal axes
mot.rot.Ixx = mot.rot.m/12 * (3*mot.rot.r^2 + mot.rot.h^2); % MoI along r dir
mot.rot.Iyy = mot.rot.m/2 * mot.rot.r^2; % MoI along h dim
mot.rot.Izz = mot.rot.Ixx; % MoI along r dir

mot.stat.Ixx = mot.stat.m/12 * (3*mot.stat.r^2 + mot.stat.h^2); % MoI along r dir
mot.stat.Iyy = mot.stat.m/2 * mot.stat.r^2; % MoI along h dir
mot.stat.Izz = mot.stat.Ixx; % MoI along r dir

% viscous friction coeff (motor side)
mot.B = mot.Kt*mot.I0/mot.w0;

%% Gearbox data
gbox.N = 30; % reduction ratio
gbox.B = 0.025; % viscous friction coeff (load side)

%% Battery data
% electrical data
batt.UN = 11.1; % nominal voltage

% dimensions
batt.w = 136e-3; % battery pack width
batt.h = 26e-3; % battery pack height
batt.d = 44e-3; % battery pack depth

% center of mass (CoM) position
batt.xb = 0; % CoM x-pos in body frame
batt.yb = 0; % CoM y-pos in body frame
batt.zb = 44e-3; % CoM z-pos in body frame

 % mass
batt.m = 0.320;

% moment of inertias (MoI) wrt principal axes
batt.Ixx = batt.m/12 * (batt.w^2 + batt.h^2); % MoI along d dim
batt.Iyy = batt.m/12 * (batt.d^2 + batt.h^2); % MoI along w dim
batt.Izz = batt.m/12 * (batt.w^2 + batt.d^2); % MoI along h dim

%% H-bridge PWM voltage driver data
drv.Vbus = batt.UN; % H-bridge DC bus voltage
drv.pwm.bits = 8; % PWM resolution [bits]
drv.pwm.levels = 2^drv.pwm.bits; % PWM levels
drv.dutymax = drv.pwm.levels-1; % max duty cycle code
drv.duty2V = drv.Vbus/drv.dutymax; % duty cycle code (0-255) to voltage
drv.V2duty = drv.dutymax/drv.Vbus; % voltage to duty cycle code (0-255)

%% Wheel data
% dimensions
wheel.h = 26e-3; % wheel height
wheel.r = 68e-3/2; % wheel radius

% center of mass (CoM) position
wheel.xb = 0; % (left) wheel CoM x-pos in body frame
wheel.yb = 100e-3; % (left) wheel CoM y-pos in body frame
wheel.zb = 0; % (left) wheel CoM z-pos in body frame

% mass
wheel.m = 50e-3;

% moment of inertias (MoI) wrt principal axes
```

```matlab
wheel.Ixx = wheel.m/12 * (3*wheel.r^2 + wheel.h^2); % MoI along r dim
wheel.Iyy = wheel.m/2 * wheel.r^2; % MoI along h dim
wheel.Izz = wheel.Ixx; % MoI along r dim

% viscous friction coeff
wheel.B = 0.0015;

%% Chassis data
% dimensions
chassis.w = 160e-3; % frame width
chassis.h = 119e-3; % frame height
chassis.d = 80e-3; % frame depth

% center of mass (CoM) position
chassis.xb = 0; % CoM x-pos in body frame
chassis.yb = 0; % CoM x-pos in body frame
chassis.zb = 80e-3; % CoM x-pos in body frame

% mass
chassis.m = 0.456;

% moment of inertias (MoI) wrt principal axes
chassis.Ixx = chassis.m/12 * (chassis.w^2 + chassis.h^2); % MoI along d dim
chassis.Iyy = chassis.m/12 * (chassis.d^2 + chassis.h^2); % MoI along w dim
chassis.Izz = chassis.m/12 * (chassis.w^2 + chassis.d^2); % MoI along h dim

%% Body data
% mass
body.m = chassis.m + batt.m + 2*mot.stat.m;

% center of mass (CoM) position
body.xb = 0; % CoM x-pos in body frame
body.yb = 0; % CoM y-pos in body frame
body.zb = (1/body.m) * (chassis.m*chassis.zb + ... % CoM z-pos in body frame
batt.m*batt.zb + 2*mot.stat.m*mot.stat.zb);

% moment of inertias (MoI) wrt principal axes
body.Ixx = chassis.Ixx + chassis.m*(body.zb - chassis.zb)^2 + ... % MoI along d dim
batt.Ixx + batt.m*(body.zb - batt.zb)^2 + ...
2*mot.stat.Ixx + ...
2*mot.stat.m*(mot.stat.yb^2 + (body.zb - mot.stat.zb)^2);

body.Iyy = chassis.Iyy + chassis.m*(body.zb - chassis.zb)^2 + ... % MoI along w dim
batt.Iyy + batt.m*(body.zb - batt.zb)^2 + ...
2*mot.stat.Iyy + ...
2*mot.stat.m*(body.zb - mot.stat.zb)^2;

body.Izz = chassis.Izz + batt.Izz + ... % MoI along h dim
2*mot.stat.Izz + 2*mot.stat.m*mot.stat.yb^2;

%% Sensors data - Hall-effect encoder
% Hall-effect encoder
sens.enc.ppr = 16*4; % pulses per rotation at motor side (w/ quadrature decoding)
sens.enc.pulse2deg = 360/sens.enc.ppr;
sens.enc.pulse2rad = 2*pi/sens.enc.ppr;
sens.enc.deg2pulse = sens.enc.ppr/360;
sens.enc.rad2pulse = sens.enc.ppr/2/pi;

%% Sensors data - MPU6050 (accelerometer + gyro)
% center of mass (CoM) position
sens.mpu.xb = 0;
sens.mpu.yb = 0;
sens.mpu.zb = 13.5e-3;

% MPU6050 embedded accelerometer specs
sens.mpu.acc.bits = 16;
sens.mpu.acc.fs_g = 16; % full-scale in "g" units
sens.mpu.acc.fs = sens.mpu.acc.fs_g * g2ms2; % full-scale in [m/s^2]
sens.mpu.acc.g2LSB = floor(2^(sens.mpu.acc.bits-1)/sens.mpu.acc.fs_g); % sensitivity [LSB/g]
sens.mpu.acc.ms22LSB = sens.mpu.acc.g2LSB * ms22g; % sensitvity [LSB/(m/s^2)]
sens.mpu.acc.LSB2g = sens.mpu.acc.fs_g/2^(sens.mpu.acc.bits-1); % out quantization [g/LSB]
sens.mpu.acc.LSB2ms2 = sens.mpu.acc.LSB2g * g2ms2; % out quantization [ms2/LSB]
sens.mpu.acc.bw = 94; % out low-pass filter BW [Hz]
sens.mpu.acc.noisestd = 400e-6*sqrt(100); % output noise std [g-rms]
sens.mpu.acc.noisevar = sens.mpu.acc.noisestd^2; % output noise var [g^2]

% MPU6050 embdedded gyroscope specs
sens.mpu.gyro.bits = 16;
sens.mpu.gyro.fs_degs = 250; % full scale in [deg/s (dps)]
sens.mpu.gyro.fs = sens.mpu.gyro.fs_degs * deg2rad; % full scale in [rad/s]
sens.mpu.gyro.degs2LSB = floor(2^(sens.mpu.gyro.bits-1)/sens.mpu.gyro.fs_degs); % sensitivity [LSB/degs]
sens.mpu.gyro.rads2LSB = sens.mpu.gyro.degs2LSB * rad2deg; % sensitivity [LSB/rads]
sens.mpu.gyro.LSB2degs = sens.mpu.gyro.fs_degs/2^(sens.mpu.gyro.bits-1); % out quantization [degs/LSB]
sens.mpu.gyro.LSB2rads = sens.mpu.gyro.LSB2degs * deg2rad; % out quantization [rads/LSB]
sens.mpu.gyro.bw = 98; % out low-pass filter BW [Hz]
sens.mpu.gyro.noisestd = 5e-3*sqrt(100); % output noise std [degs-rms]
sens.mpu.gyro.noisevar = sens.mpu.acc.noisestd ^2; % output noise var [degs^2]
```

A.4 Simulink model

A.4.1 Simulink scheme for simulations

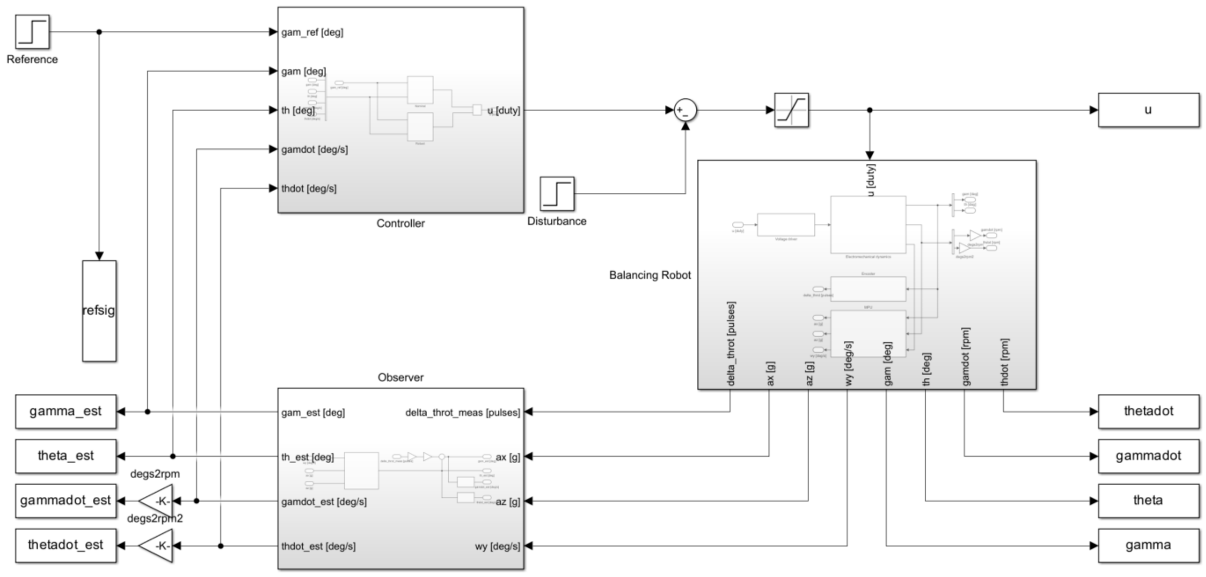


Figure 19: Simulation Simulink scheme

A.4.2 Simulink scheme of balancing robot

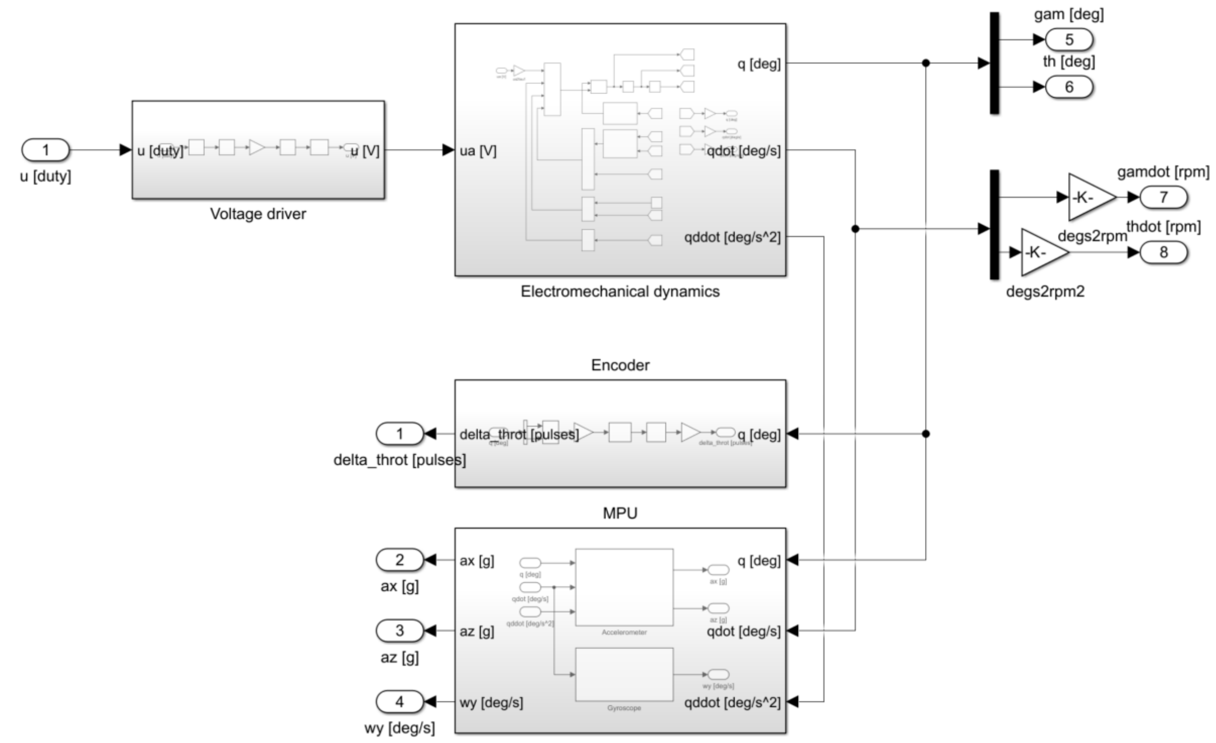


Figure 20: Balancing robot block scheme

A.4.3 Simulink model of the state observer

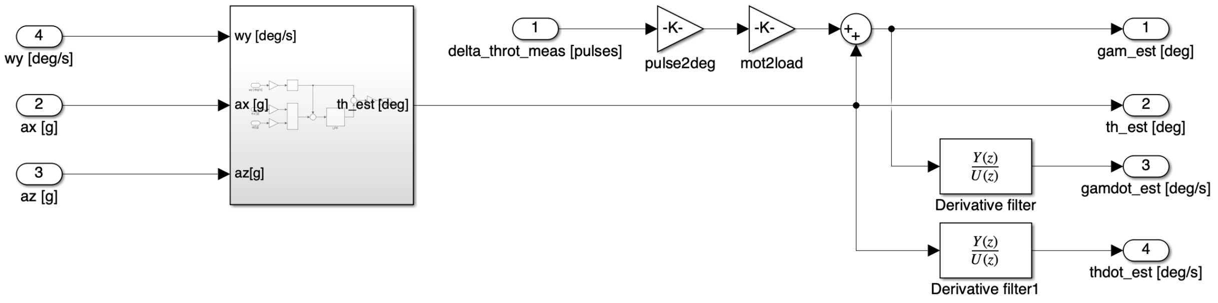


Figure 21: Observer

A.4.4 Simulink model of the complementary filter

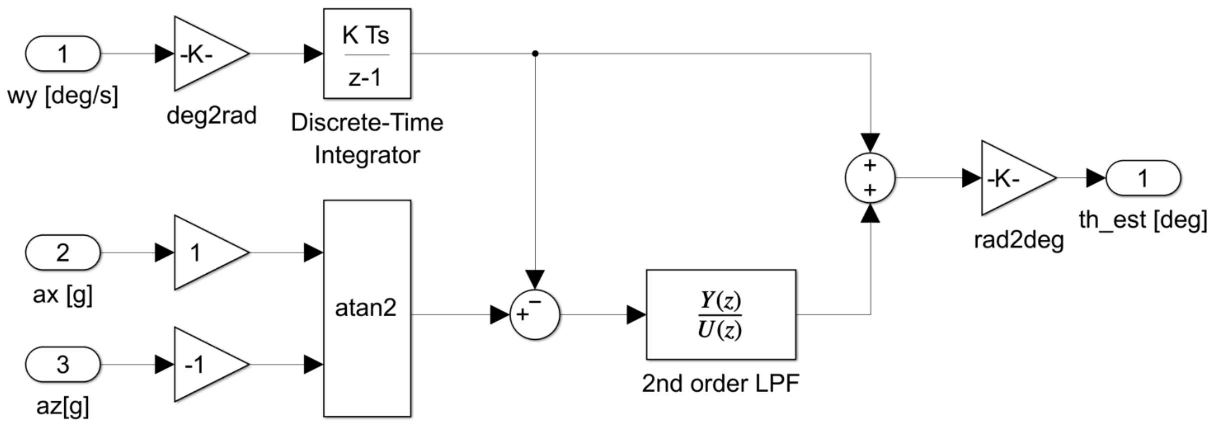


Figure 22: Complementary filters

A.4.5 Simulink model of the nominal controller

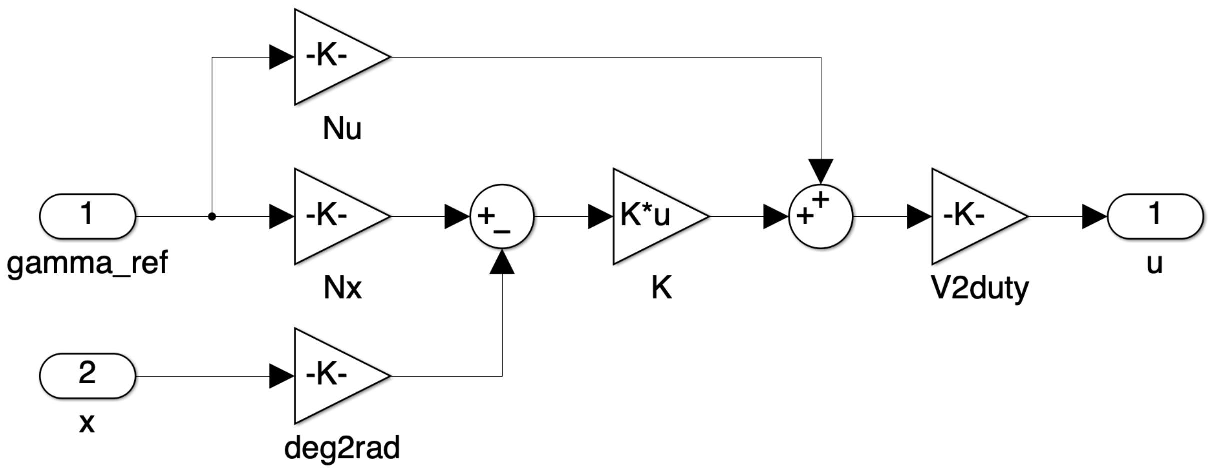


Figure 23: Nominal controller

A.4.6 Simulink model of the robust controller

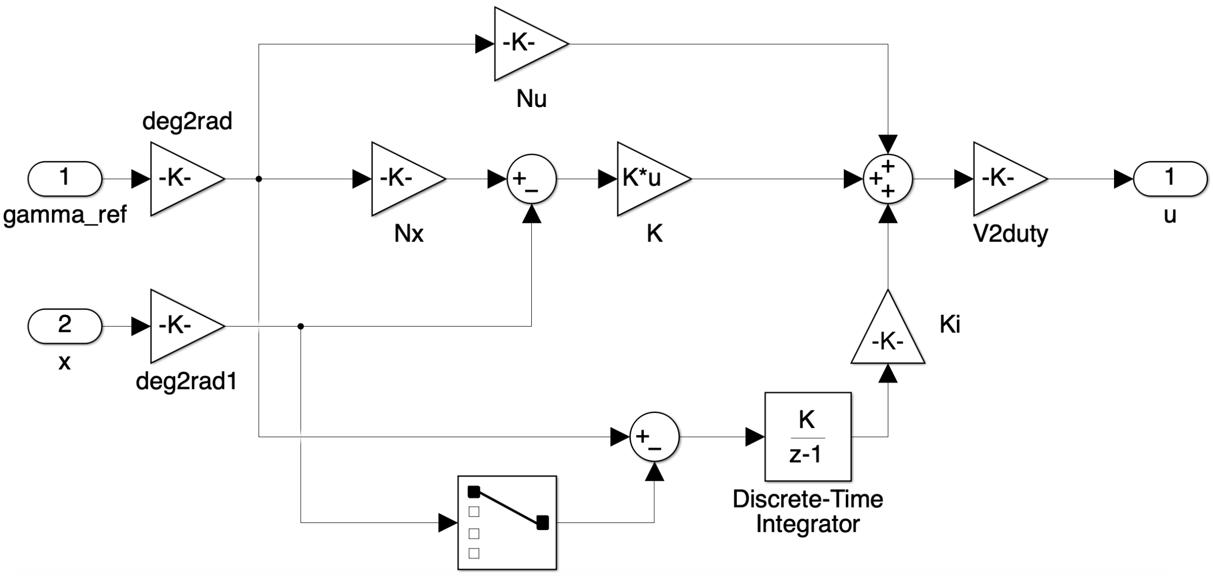


Figure 24: Robust controller

A.4.7 Simulink model of the yaw controller

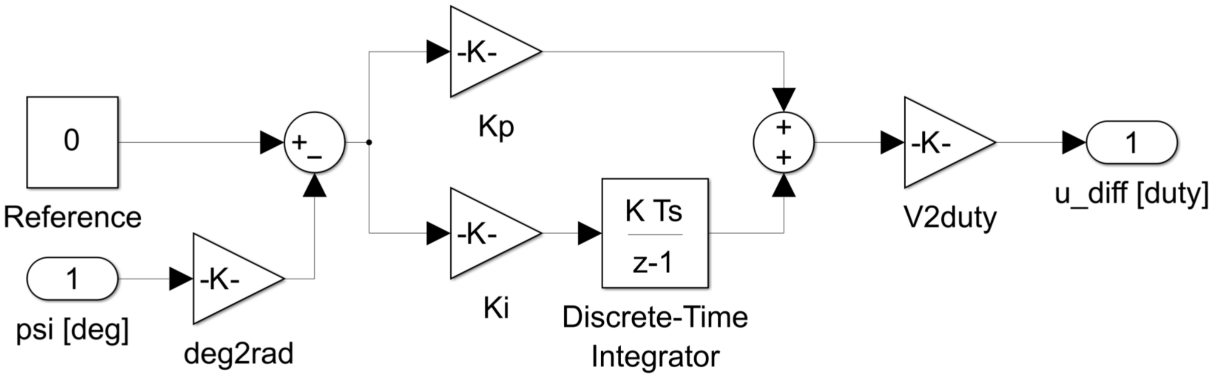


Figure 25: Yaw controller

A.4.8 Simulink scheme for real-time experiments

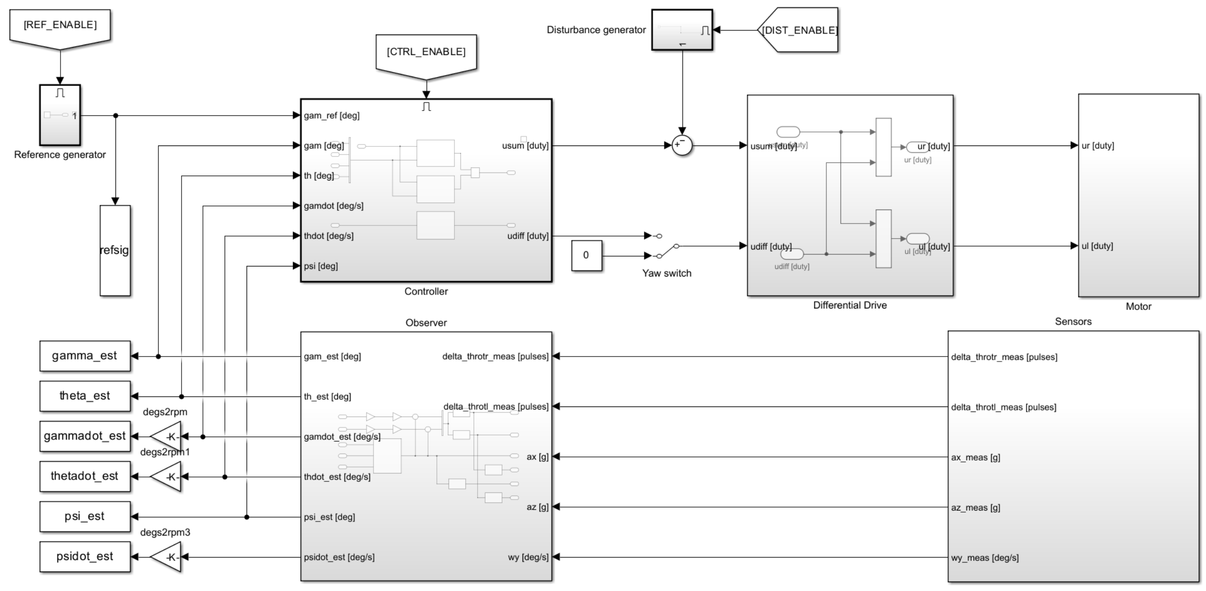


Figure 26: Real-time Simulink scheme

A.4.9 Simulink scheme for real-time observer

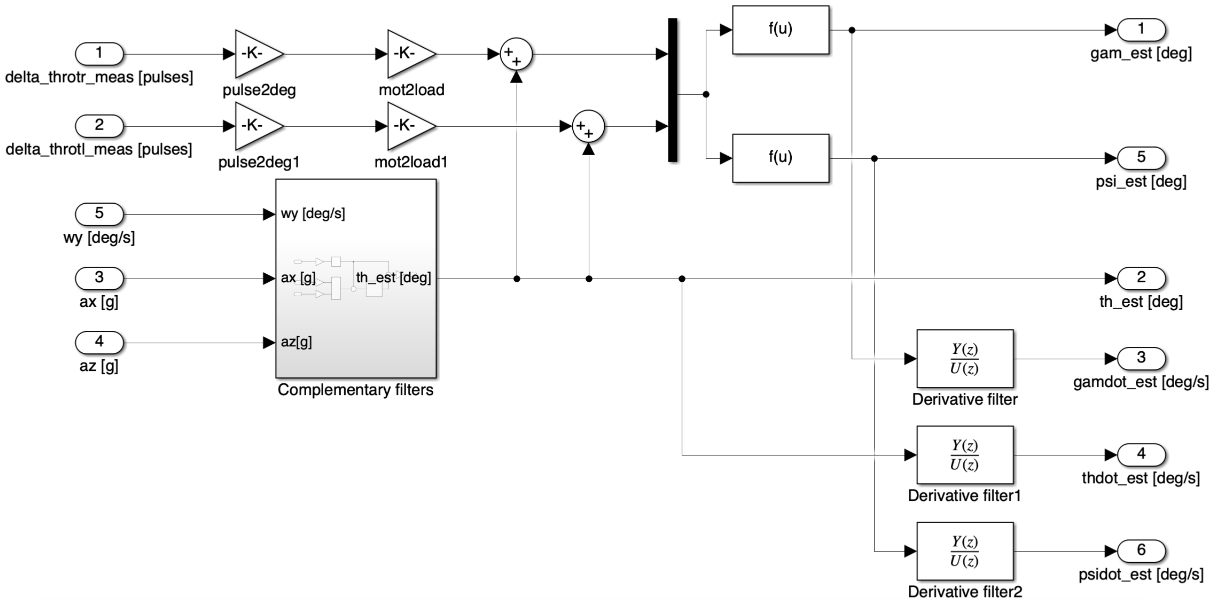


Figure 27: Real-time observer