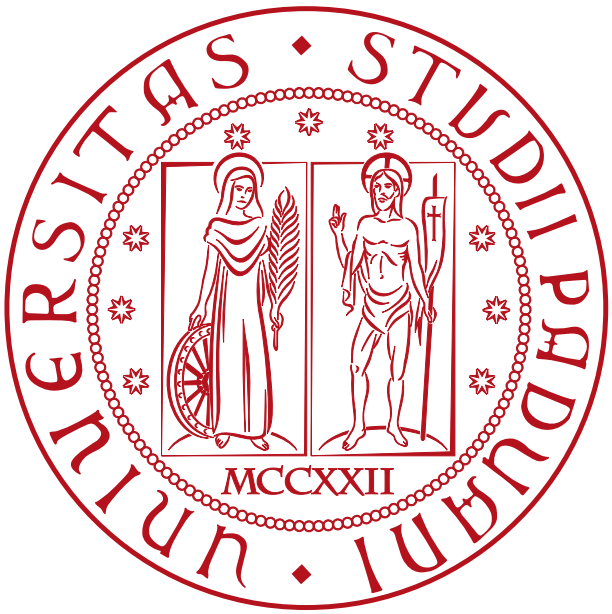


University of Padova

Department of Information Engineering

Master degree in Automation Engineering

Laboratory Report4: Longitudinal state–space control of the balancing robot

Group 2 Tuesday shift, components: Davide Baron, Alessio Bonetto, Marco Mustacchi, Luca Vittorio Piron

Academic Year 2021-2022

1 Introduction

1.1 Activity Goal

The goal of this laboratory activity is to design and test a longitudinal state–space controller for the two–wheeled balancing robot available in laboratory. The controller is designed to simultaneously stabilize the robot body to its upward vertical position, and the robot base to a desired longitudinal position set-point. The design is performed by resorting to a simplified model of the robot dynamics, obtained by assuming that the motion occurs along a straight line (i.e. the lateral or heading–angle dynamics is ignored). A nonlinear coupling term is usually neglected, whereas it has significant effects on the dynamic behavior of the system. Three different controller approach has been taken into account on this report:

- Nominal controller with state-space method
- Robust controller with state-space method
- Yaw PI controller

1.2 System and Model

The robot can be represent as a multi-body system composed by Wheels, Robot body and two DC motors driving the robot wheels. In order to describe the robot structure, it is possible to get the reference frames in figure 1 using the Denavit-Hartemberg convention.
The vector of generalized coordinates used is: $\boldsymbol{q} = [\gamma \quad \vartheta]^T$.
The Lagrangian approach has been used, so it is necessary to evaluate the kinetic and potential energy for the whole system, i.e. $\boldsymbol{\mathcal{L}} = \boldsymbol{T} - \boldsymbol{V}$ (more details for the Lagrangian function calculation can be found in the Appendix A.1.1). In this report, the analysis has been done by assuming that the two wheels' angles are always identical and hence the heading angle ψ is a constant, this is not a realistic assumption since it means having the two motor torque always identical.

Analysing the system dynamics is possible to obtain the so called Equation-of-motion (EoM) that can be described using

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

where $\boldsymbol{M}(\boldsymbol{q})$ is the inertia matrix obtained by determine the *Moment-of-Inertia* of all the robot's components, $\boldsymbol{C}(\boldsymbol{q})$ is the Coriolis matrix obtained by compute the *Christofell symbols*, $\boldsymbol{F_v}$ denote the matrix of viscous friction coefficients, while the $\boldsymbol{g}(\boldsymbol{q})$ term gives the torque contribution needed in order to compensate the effect of the gravity. The sum of all these terms is imposed to be equal to the input torque of the motor, i.e. $\boldsymbol{\tau} = [2\tau \quad -2\tau]^T$.
Since the system is non-linear (the dynamics depends on $cos\theta$ and $sin\theta$, which are clearly non-linear functions), it is necessary to linearize the system around an equilibrium point, which

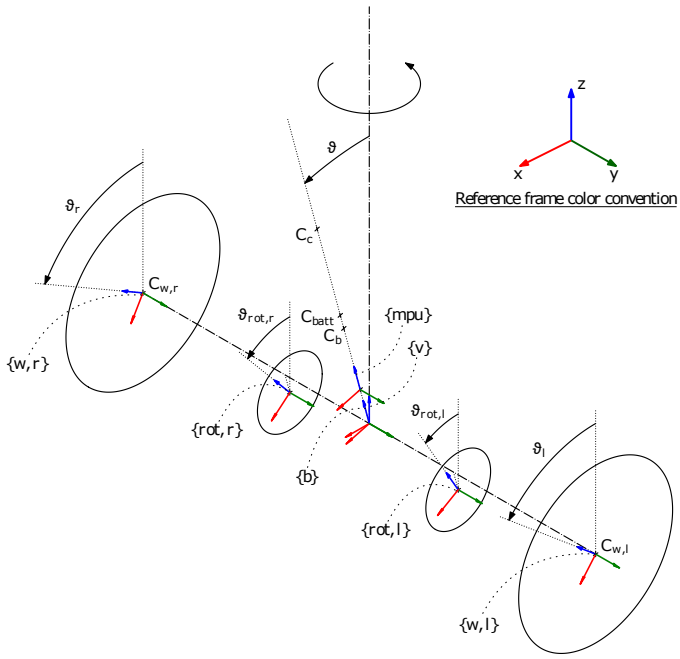


Figure 1: Robot structure and configuration

is for our purpose, the *upward vertical position*.
So the chosen equilibrium point is $P_0 = (\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \tau_0)$ where $\boldsymbol{q} = [\gamma_0 \quad 0]$, $\dot{\boldsymbol{q}} = \ddot{\boldsymbol{q}} = [0 \quad 0]$ and $\tau_0 = 0$, clearly this point is unstable.[1]
The linearization is beyond the scope of this report but the linearized version of (1) brings to the following (more details can be found on Appendix A.1.2):

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{F_v}\dot{\boldsymbol{q}} + \boldsymbol{G}\boldsymbol{q} = \boldsymbol{\tau} \tag{2}$$

which can be alternatively rewritten in the standard state-space form taking the state $\boldsymbol{x} = [\boldsymbol{q} \quad \dot{\boldsymbol{q}}]^T$, namely:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{3}$$

where the matrix $\boldsymbol{A}$ and $\boldsymbol{B}$ corresponds to:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{0}_{2\times2} & \boldsymbol{I}_{2\times2} \\ -\boldsymbol{M}^{-1}\boldsymbol{G} & -\boldsymbol{M}^{-1}\boldsymbol{F_v'} \end{bmatrix}, \qquad \boldsymbol{B} = \frac{2k_tN}{R_a}\begin{bmatrix} \boldsymbol{0}_{2\times2} \\ \boldsymbol{M}^{-1} \end{bmatrix}\begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{4}$$

in which all the symbols used and the matrices $\boldsymbol{M}$ and $\boldsymbol{F_v'}$ are defined in Appendix A.1.2.

[1] Note that the initial configuration of the robot will never coincides perfectly with this point, this things need to be taken into account for the stabilisation procedure

2 Tasks, Methodologies and Results

2.1 Nominal state-space controller

The aim of this assigment is to design a state-space controller that guarantees the perfect tracking of constant position set–points for the longitudinal displacement and the regulation problem for the vertical angular position in the nominal case.
The continous-time linearized model of the balancing robot has been obtained following the procedure explained in sec. 2 of Handout 4, linearizing the system around the vertical position. Since the controller is implemented through a microcontroller that works in discrete-time, to obtain a better feedback controller the direct digital design has been used. This type of design provides better results with respect to the emulation method, as seen in the laboratory experience number 2. Then, the matrices Φ, Γ, H and J are obtained through the $c2d$ routine of the Control System Toolbox, specifying a sampling time of $T = 0.01s$.

The structure of the nominal controller is shown in figure 2. The state and input feedforward gains are obtained by following the Handout 2 procedure:

$$\begin{bmatrix} N_x \\ N_u \end{bmatrix} = \begin{bmatrix} \Phi - I & \Gamma \\ H & J \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} \tag{5}$$

which provides : $N_x = [1, 0, 0, 0, 0]^T$ and $N_u = 0$.

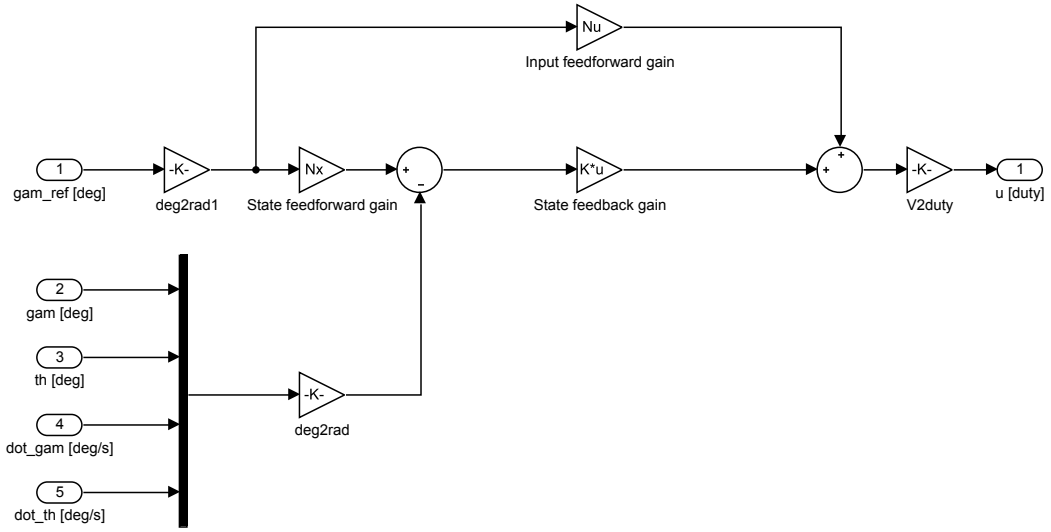


Figure 2: State space controller for nominal perfect tracking of constant set-points.

Then, the control law becomes:

$$u = N_u r - K(x - N_x r) = -Kx + (N_u + KN_x)r \tag{6}$$

Since the linearization works well only in a neighborhood area of the operating point, too much control may cause the robot to exit this area. Therefore, an LQR control has been used, in

order to limit the use of control by acting on the weight of u in such a way that the control effort will have an higher cost.
Obviously, also the more traditional eigenvalues positioning method still works, it is only needed to keep attention to the transient specifications to avoid that the balancing robot falls down due to much higher specified performances.
Then, the aim is to minimize the cost function:

$$J = \sum_{k=1}^{+\infty} x^T[k]\boldsymbol{Q}x[k] + \rho u^T[k]\boldsymbol{R}u[k] \tag{7}$$

Among the various methods to select Q and R, the Bryson's one has been choosen because it provides a physical intuition above the choice of these matrices. In fact, considering, at steady-state, these values:

$$|\gamma - \gamma^*| < \bar{\gamma} = \pi/18 \qquad |\theta| < \bar{\theta} = \pi/360 \qquad |u| < \bar{u} = 1V \tag{8}$$

which represent, respectively, the maximum displacement between the longitudinal position and its reference, the maximum regulation error of the vertical angular displacement, and the maximum input control provided to the motors. Then, the Bryson's rule suggest to use, as first trial, the matrices:

$$Q = diag\left\{\frac{1}{\bar{\gamma}^2}, \frac{1}{\bar{\theta}^2}, 0, 0\right\} \qquad R = \frac{1}{\bar{u}^2} \tag{9}$$

The extra weights ρ in (7) is used to adjust the relative weighting between the state and input contributions to the total cost function value. Since is not easy to provide a relation between ρ and the performances, usually this value is tuned by a trial-and-error approach. The following values are considered:

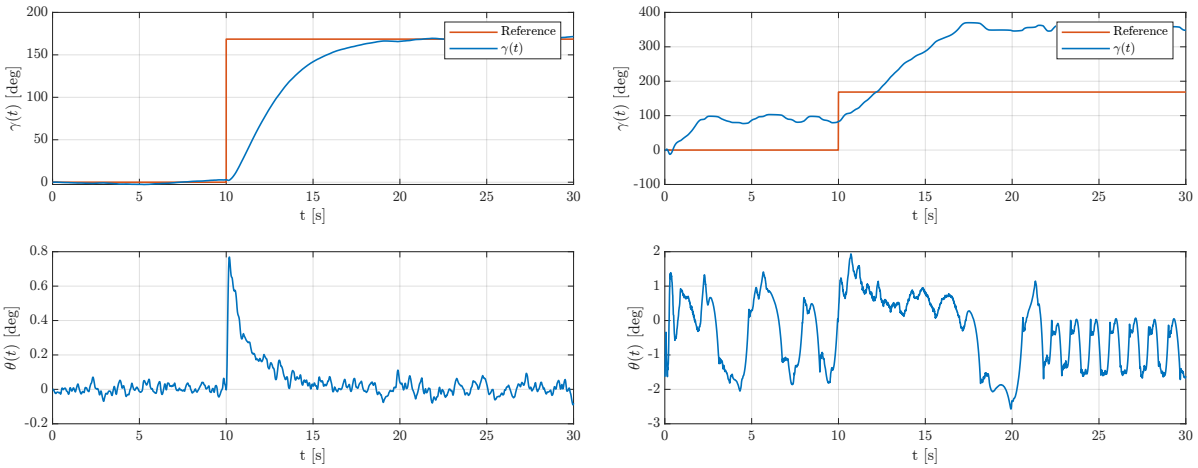
$$\rho \in \{500, 5000\} \tag{10}$$

The value of ρ is very important for the tradeoff of the amount control and the displacement respect to the reference.
The first test that has been done is the changing of the γ's reference, the step reference is applied at t=10s, in order to permit the balancing robot to stabilize in the vertical position. In figures 3 and 4 is shown a big difference between the nominal model and the real system. In fact, the nominal model reach correctly the reference with zero steady-state error, as expected from the theory, instead the real balancing robot do not follow the reference at all. This is probably due to the fact that the real system has the tendency of drifting laterally and hence one wheel rotates more than the other. Considering the relation:

$$\gamma = \frac{\theta_r + \theta_l}{2} \tag{11}$$

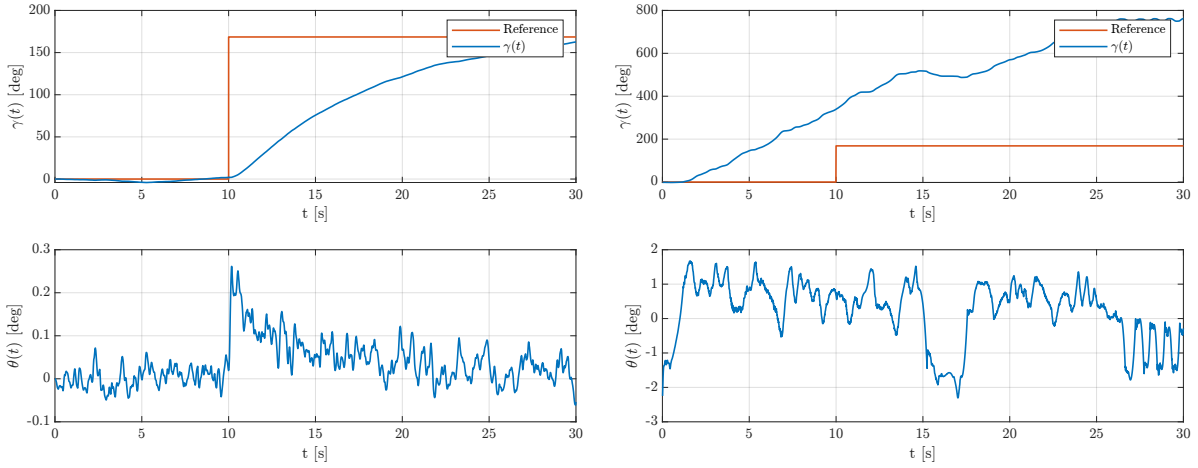
where θ_r and θ_l are the wheels angles derived from the encoders measurements, the displacement along the straight line is greatly influenced by the drift motion. This problem is caused by the fact that even if the two motors are driven by the same voltage command, they do not

necessarily move by the same angle, because of unavoidable differences in the motor parameters, and the presence of friction and backlash in the mechanical transmission.
A possible solution to avoid, or at least reduce, this problem is explained successively.



(a) Nominal model (b) Real balancing robot

Figure 3: Nominal tracking of longitudianl displacement reference with ρ=500.



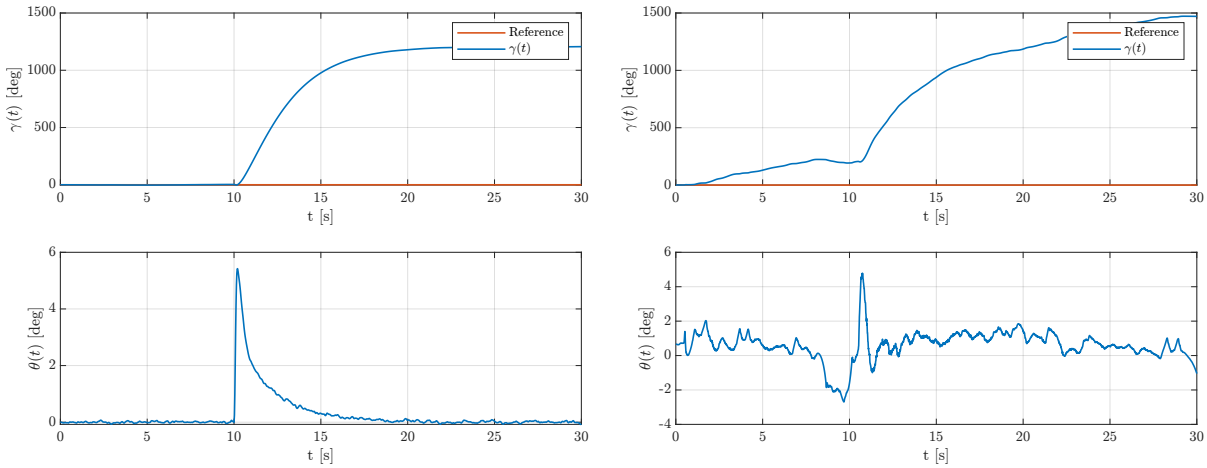
(a) Nominal model (b) Real balancing robot

Figure 4: Nominal tracking of longitudianl displacement reference with ρ=5000.

However, some considerations can be done. In particular, it is interesting to notice, especially in the nominal model, how the choice of ρ influences the dynamics of the response. Considering a low value of ρ the LQR alghoritm do not penalize too much the cost of the control in (7), instead with an higher value of ρ the algorithm try to reduce the control effort in order to minimize the cost function.
This fact is clearly observable in figures 3a and 4a, where is shown that with $\rho = 500$ the control effort is higher and the reference is reached in a shorter time, with respect to the controller obtained with $\rho = 5000$ which reduces the amount of control.

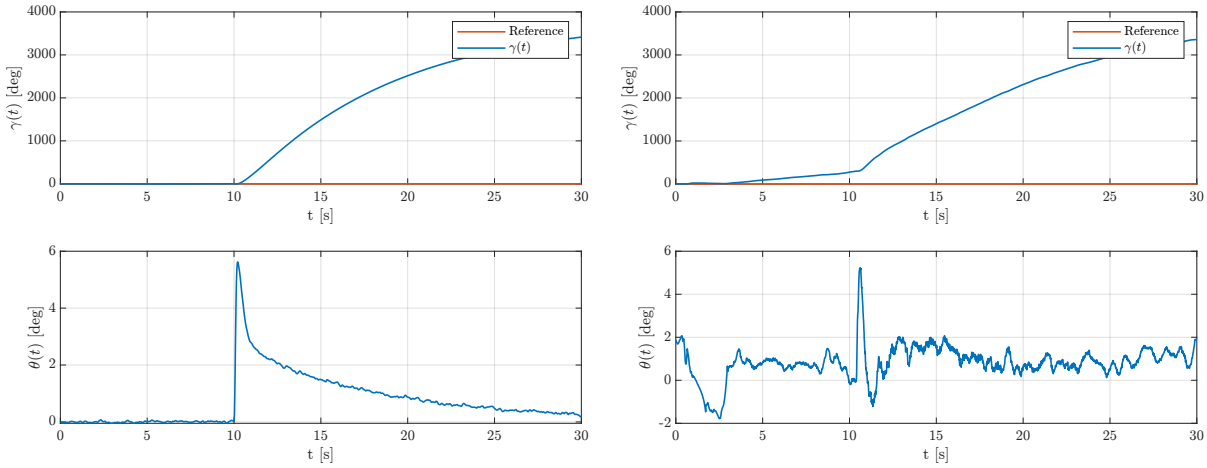
The second test that has been done is the introduction of a constant disturb in the control input of the two motors. Such disturb is applied at t=10s, in order to permit the balancing robot to stabilize in the vertical position.

In figures 5 and 6 is possible to observe how an higher controller compensate better, even if it doesn't erase, the applied disturb; this is expected from the theory. From the theory is also known as, in presence of disturb, a nominal controller does not provide a zero steady-state traking error, and this is what the experiment showed.
To compensate better the presence of the disturb, an integral control must be included.



(a) Nominal model (b) Real balancing robot

Figure 5: Nominal response to a disturb entering in the control input with ρ=500.



(a) Nominal model (b) Real balancing robot

Figure 6: Nominal response to a disturb entering in the control input with ρ=5000.

2.2 Robust state-space controller

In order to guarantee perfect steady state robust tracking of constant position reference and perfect rejection of constant load disturbances, like the static friction or the input control disturbance, an integral action has been introduced to the state-space controller in figure 2 . The new state vector becomes $x_e = [x_I \;\; x]^T$, where x is the state vector of the previous assignment and x_I is the new added component.

Referring to the control law (6) the integral action can be included as follows:

$$u = -Kx + (N_u + KN_x)r - K_I \int_0^t [y(\tau) - r(\tau)]d\tau \tag{12}$$

which decribes the block scheme in figure 7, in which the integral component has been approximated through forward Euler discretization method.

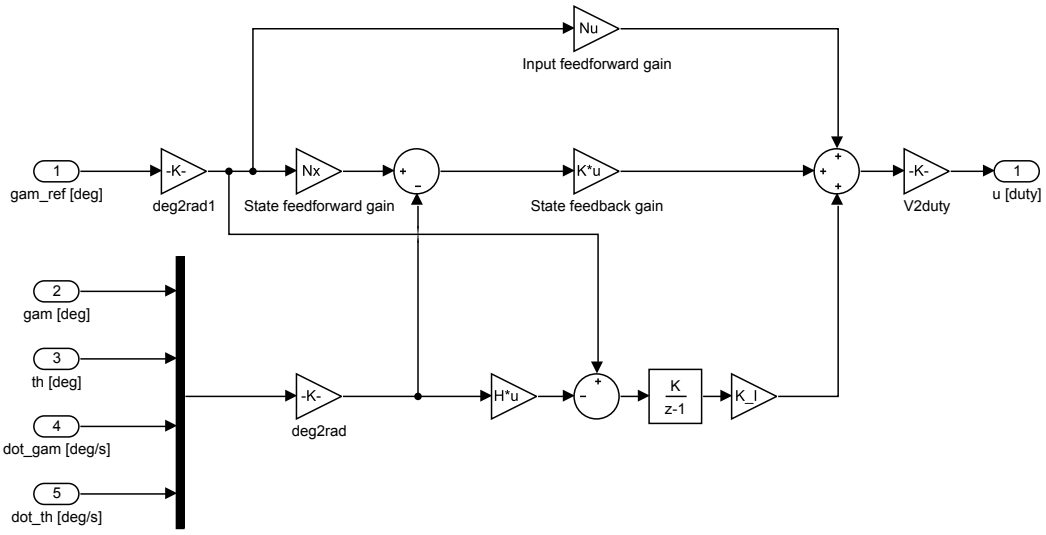


Figure 7: State space controller for robust tracking of constant set-points.

The integral component in some sense, try to compensate the constant disturbances in the model and the modelling errors.
Referring to the matrices Φ, Γ, H and J obtained in the previous point, the new extended state imposes the following state matrices:

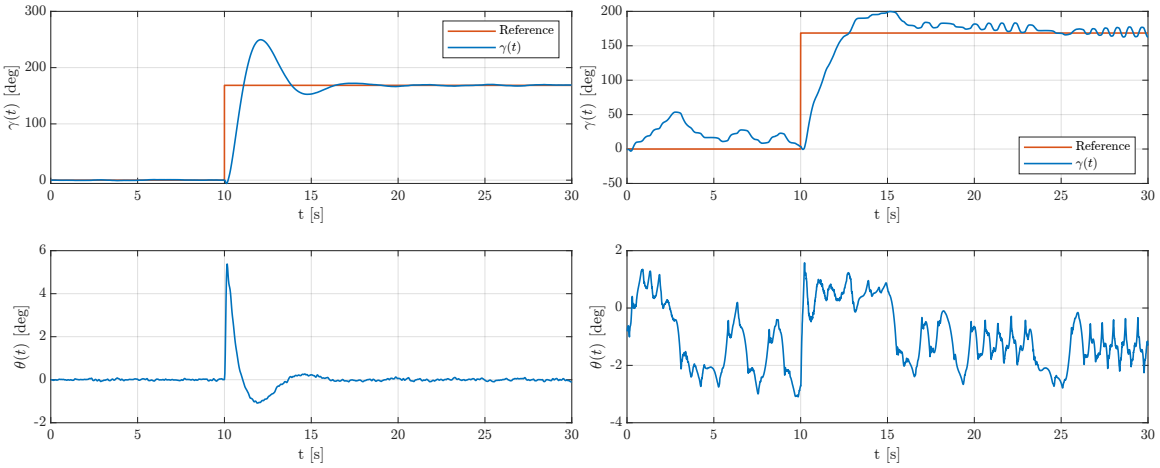
$$\Phi_e = \begin{bmatrix} 1 & H \\ 0 & \Phi \end{bmatrix} \quad \Gamma_e = \begin{bmatrix} 0 \\ \Gamma \end{bmatrix} \tag{13}$$

The feedforward matrixes N_x and N_u calculated in the previous point remain unchanged.
The feedback matrix has been evaluated always through the LQR method. In this case the R matrix remain unchanged, in the Q matrix a new weight has to be introduced, the so called q_{11}, the name derives from the position that it assumes in the Q matrix. This value is chosen by a trial-and-error approach considering the values:

$$q_{11} \in \{0.1, 1\} \tag{14}$$

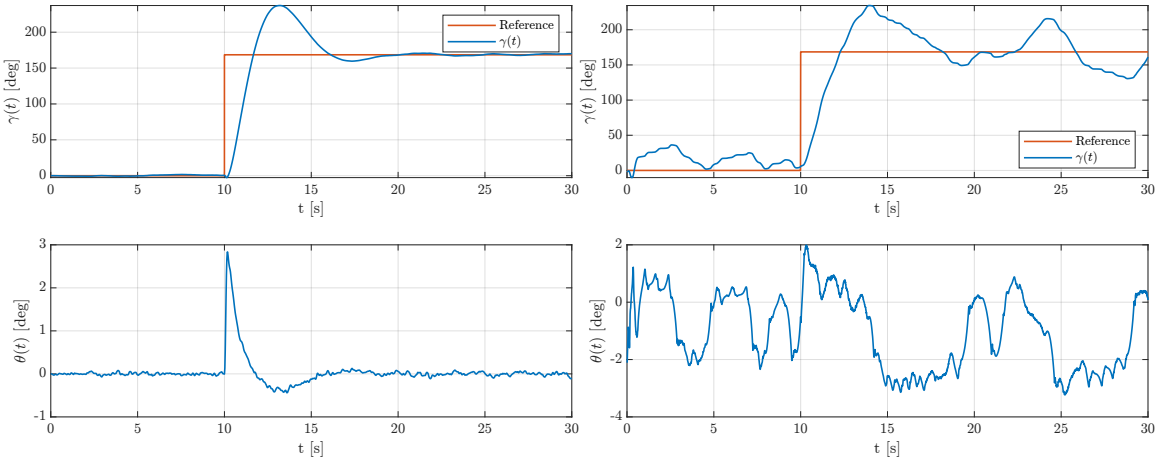
The remaining weights are chosen as in the previous point.
As before the first test that has been done is change the reference for the longitudinal position. In figures 8, 9, 10 and 11 it is possible to observe the behavior of the system respect to the choice of the free parameters ρ and q_{11}. The numerical values are reported in Table 1.
It is interesting to notice how, with the same value of q_{11}, the overshoot on the step response is higher considering ρ=500. In fact, an higher amount of control induces an higher longitudinal

speed and hence a smaller rise time but also higher overshoot caused by the inertia of the robot. This is observable having a look to the figures 8,9 and 10,11 respectively.
Another thing to observe is the value of θ in corrispondence of the step transition, in fact the initial error is such that initially the control has an high value and hence the robot tends to inclinate itself. This behavior is clearly accentuated by the choice of ρ which regulates the control effort, and hence with ρ=500 the peak of θ is higher.
The same type of behavior has been obtained changing q_{11} and maintaining the same value of ρ. It is obvious having a look to the figures 8,10 and 9,11 respectively. In this case, increasing q_{11} from 0.1 to 1 increase the integral gain K_I and hence the system reacts more quickly and the overshoot's amplitude increases.
Also in this case the shape of θ is influenced by the choiche of q_{11}, an higher value of the integral gain K_I induces a greater control and hence a bigger initial inclination.



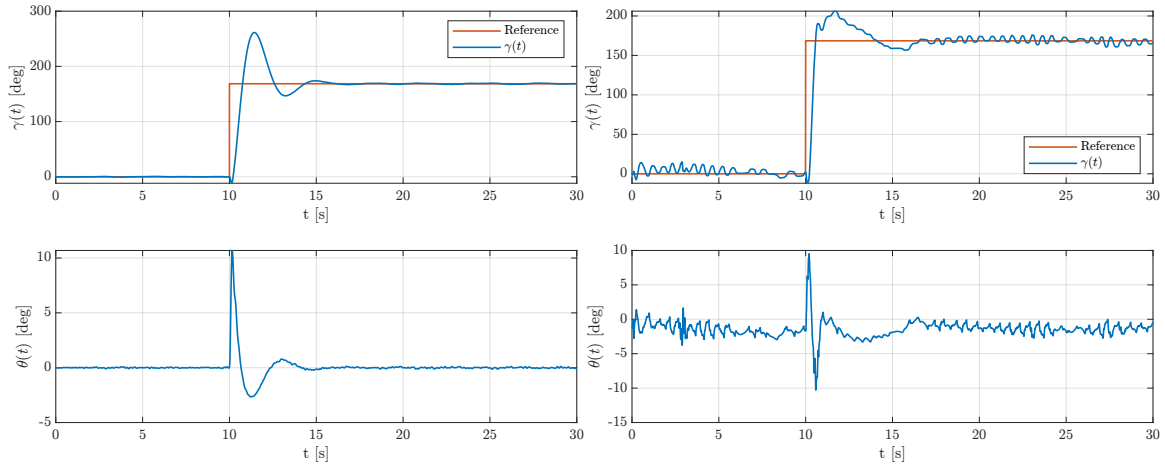
(a) Nominal model (b) Real balancing robot

Figure 8: Tracking of longitudinal displacement reference with ρ=500 and q_{11}=0.1.



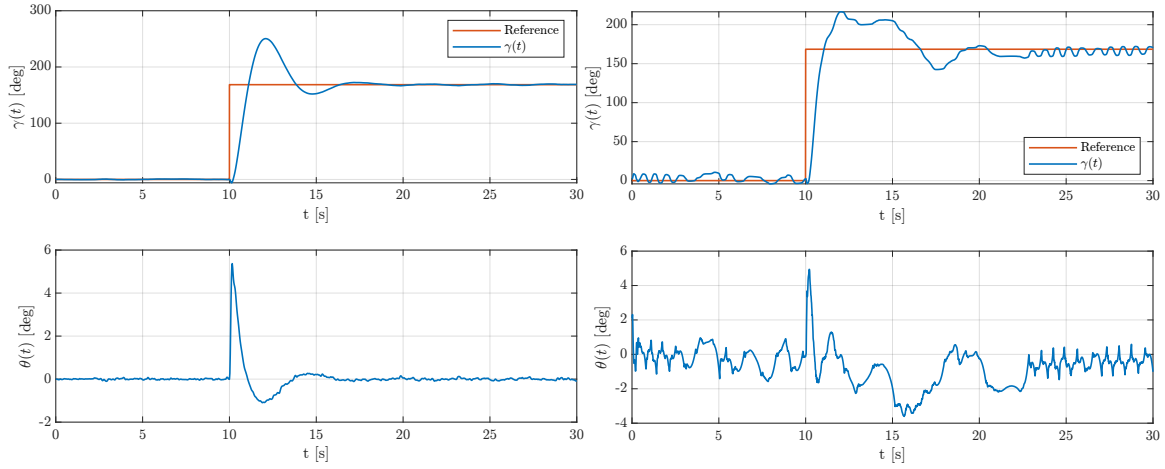
(a) Nominal model (b) Real balancing robot

Figure 9: Tracking of longitudinal displacement reference with ρ=5000 and q_{11}=0.1.



(a) Nominal model (b) Real balancing robot

Figure 10: Tracking of longitudinal displacement reference with ρ=500 and q_{11}=1.



(a) Nominal model (b) Real balancing robot

Figure 11: Tracking of longitudinal displacement reference with ρ=5000 and q_{11}=1.

In conclusion, the controller with ρ=500 and q_{11}=1 produces the faster response in terms of raise time and convergence time, the drawback is the fact that it induces an overshoot with higher value and a bigger initial inclination of the robot. On the other hand, the controller with ρ=5000 and q_{11}=0.1 induces a slower response which the advantage of reducing the overshoot and the inclination of the robot.

As before the second test that has been done is introducing a disturbance in the control input. In figures 12, 13, 14 and 15 it is possible to observe the behavior of the system respect to the different choice of the free parameters ρ and q_{11}.

In particular, it is interesting to notice how, with the same value of q_{11}, the overshoot caused by the disturb is attenuated of about 50%, from 250deg to 130 in figures 13a and 12a and from 130deg to 60 in figures 15a and 14a, decreasing the value of ρ and hence increasing the control effort. The same happens for the real balancing robot with a slightly different value. This make the system more reactive and hence it compensate the disturb before the

			t_r [ms]	$t_{s,5\%}$ [ms]	$M_p[\%]$
Nominal	ρ=500	q_{11}=0.1	0.77	15.65	48.10
	ρ=5000	q_{11}=0.1	1.25	17.54	40.73
	ρ=500	q_{11}=1	0.48	13.95	55.06
	ρ=5000	q_{11}=1	0.77	15.67	48.59
Real	ρ=500	q_{11}=0.1	2.06	23.47	18.54
	ρ=5000	q_{11}=0.1	1.86	-	39.11
	ρ=500	q_{11}=1	0.31	15.99	22.31
	ρ=5000	q_{11}=1	0.73	25.58	28.49

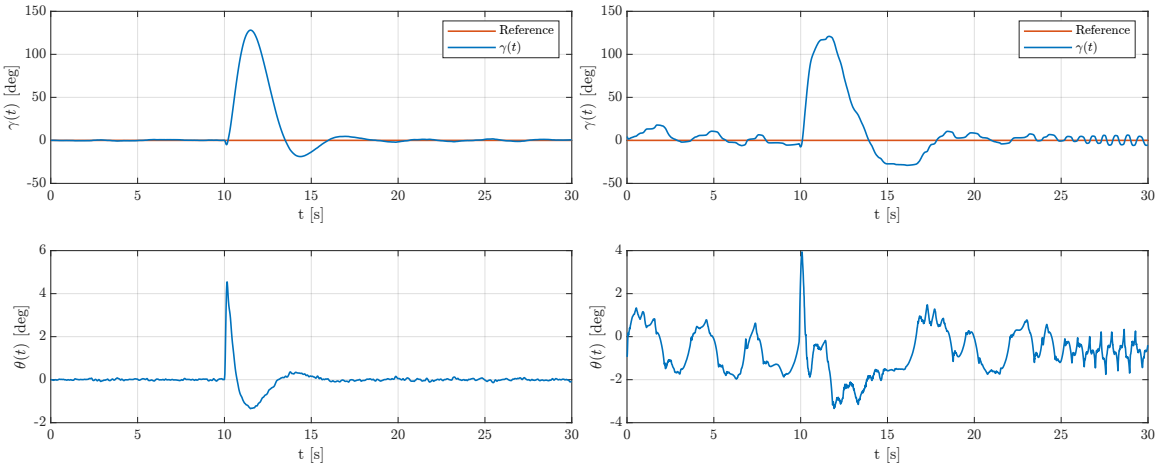
Table 1: Results obtained with the robust state-space controller for the reference step response.

error becomes too large. Clearly, reducing the overshoot permits the controller to stabilize the system in a shorter time.

The same type of behavior has been obtained changing q_{11} and maintaining the same value of ρ. It is obvious having a look to the figures 12,14 and 13,15 respectively. In this case, increasing q_{11} from 0.1 to 1 increase the integral gain K_I and hence the system reacts more quickly and the overshoot's amplitude increases.

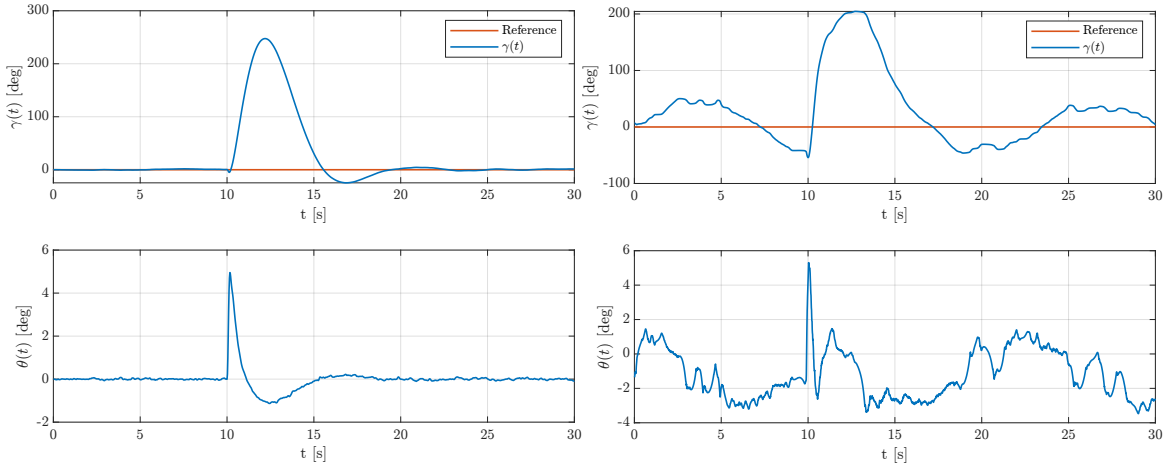
In conclusion, the controller with ρ=500 and q_{11}=1 produces the faster response in term of raise time and convergence time, and reduces the overshoot'amplitude. On the other hand, the controller with ρ=5000 and q_{11}=0.1 induces a slower response which the increasing the overshoot's amplitude.

A thing to keep in mind is the different behavior with respect to the reference step response. In fact, in that case the strong controller produces higher overshoot, while in this case the strong controller reduce the overshoot. This is another proof that the controller design is a tradeoff between different features, in this case the capability of follow quickly a reference and the capability to reacts quickly to a disturbance.



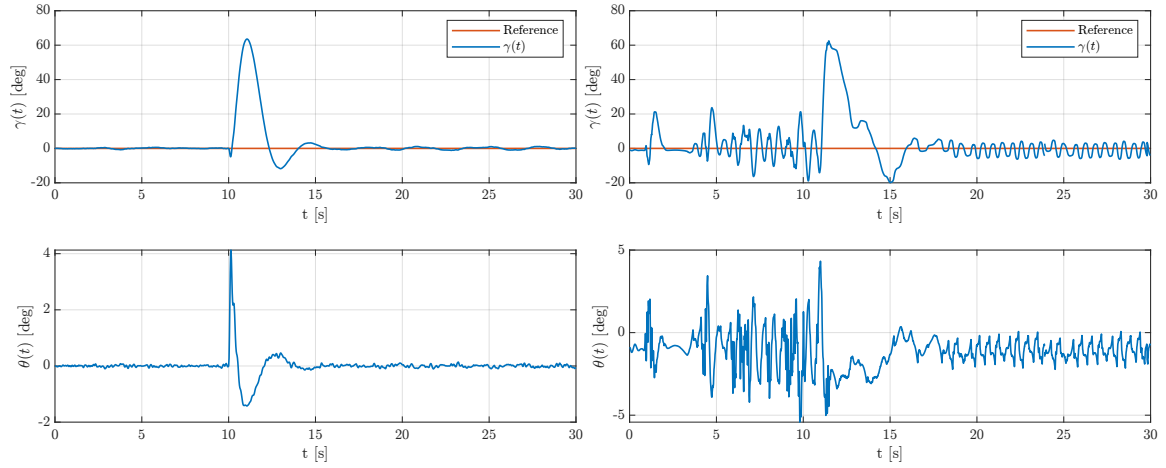
(a) Nominal model (b) Real balancing robot

Figure 12: Robust response to a disturb entering in the control input with ρ=500 and q_{11}=0.1.



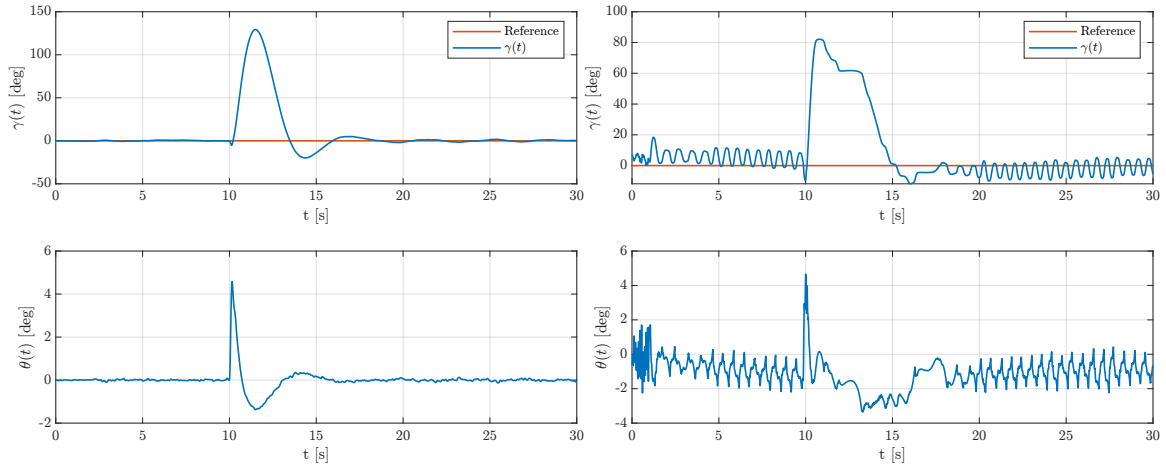
(a) Nominal model (b) Real balancing robot

Figure 13: Robust response to a disturb entering in the control input with ρ=5000 and q_{11}=0.1.



(a) Nominal model (b) Real balancing robot

Figure 14: Robust response to a disturb entering in the control input with ρ=500 and q_{11}=1.



(a) Nominal model (b) Real balancing robot

Figure 15: Robust response to a disturb entering in the control input with ρ=5000 and q_{11}=1.

2.3 Yaw PI controller

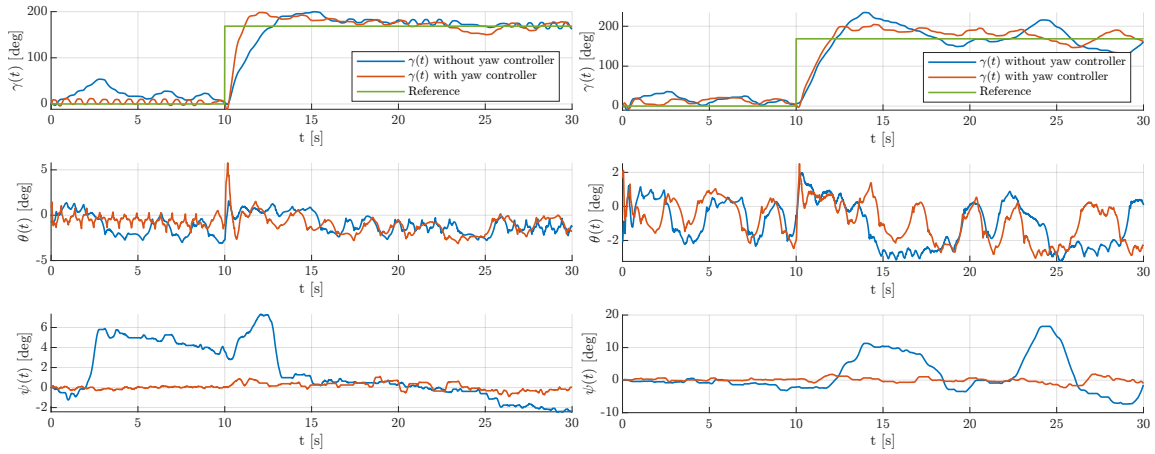
In the experimental tests on points 2.1 and 2.2 has been noticed the fact that the balancing robot has the tendency of drift laterally, does not following a perfectly straight line. This behavior influences the evolution and the trajectory of γ, especially with the nominal controller. In order to solve this problem a simple PI controller has been introduced:

$$C_\psi(s) = K_p + \frac{K_I}{s} \tag{15}$$

which works to regulate the angular rotation ψ. In practice, instead of control both motors with the same voltage input, now the whole controller controls the two motors indipendently, in order to compensate the mechanical differences between them.

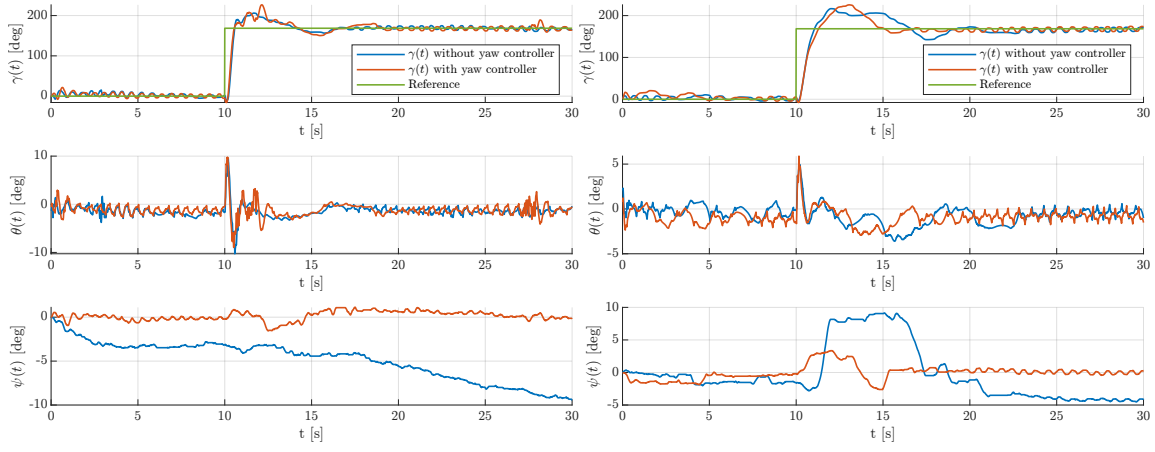
The controller implemented in Simulink is showed in figure 18, in which the controller in (15) has been discretized using the forward Euler approximation method.

In figure 16 is possible to observe the effectiveness of the control when the system is subjected to a change of reference. It is interesting to notice how, with the robust controller, the shape of γ and θ are not influenced by the introduction of yaw controller.



(a) ρ=500 and q_{11}=0.1

(b) ρ=5000 and q_{11}=0.1

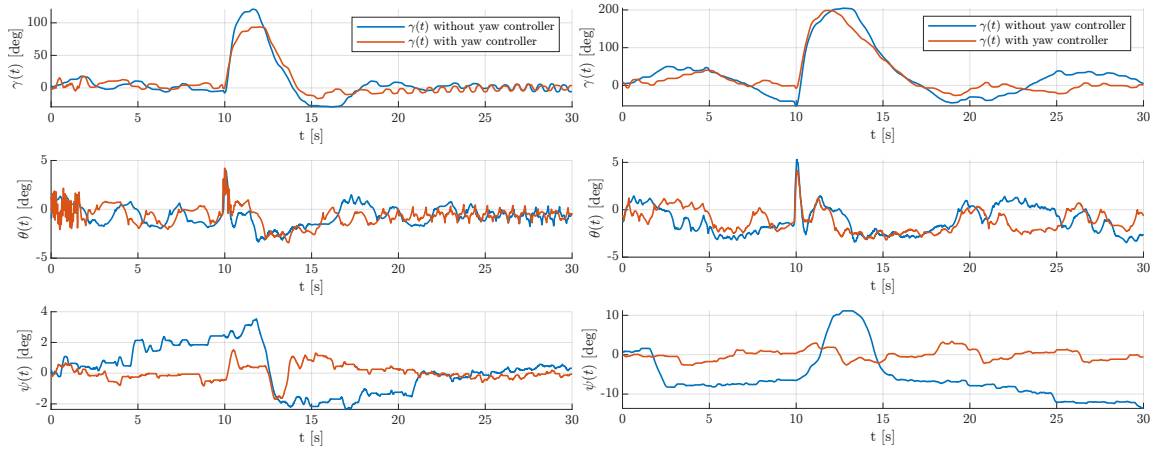


(c) ρ=500 and q_{11}=1

(d) ρ=5000 and q_{11}=1

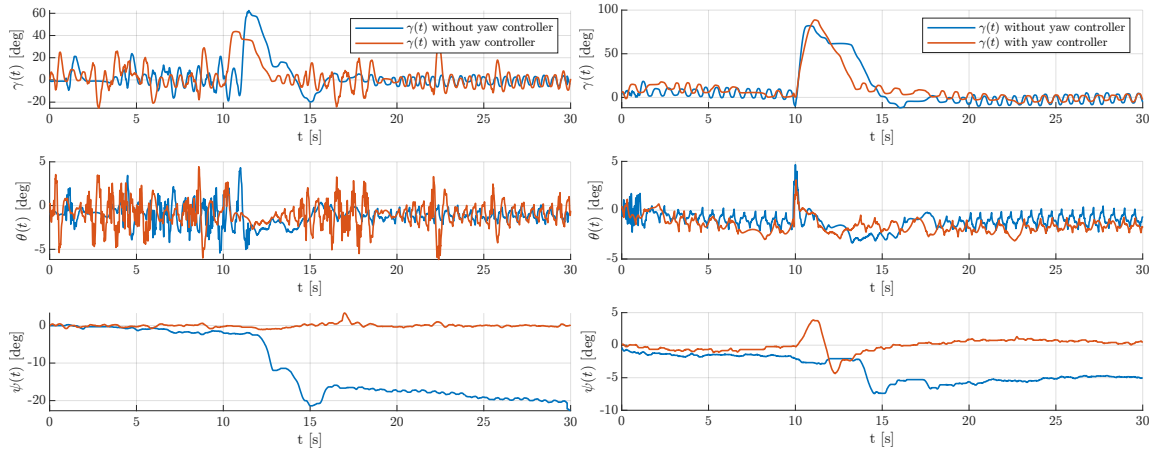
Figure 16: Tracking of longitudinal displacement reference with and without yaw controller, with real balancing robot.

In figure 17 is possible to observe the effectiveness of the control when the system is subjected to a constant disturb. Respect to the previous case the dynamics of γ sometimes has been influenced and improved in some sense, reducing the overshoot's amplitude caused by the application of the disturb, figures 17a and 17c.



(a) ρ=500 and q_{11}=0.1

(b) ρ=5000 and q_{11}=0.1



(c) ρ=500 and q_{11}=1

(d) ρ=5000 and q_{11}=1

Figure 17: Response to a disturb with and without yaw controller, with real balancing robot.

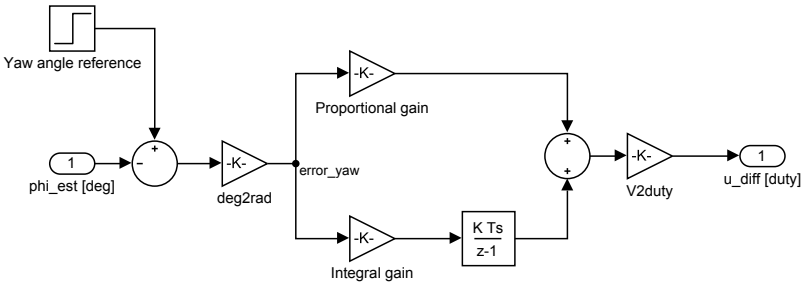


Figure 18: PI controller for yaw stabilization

3 Complementary filter

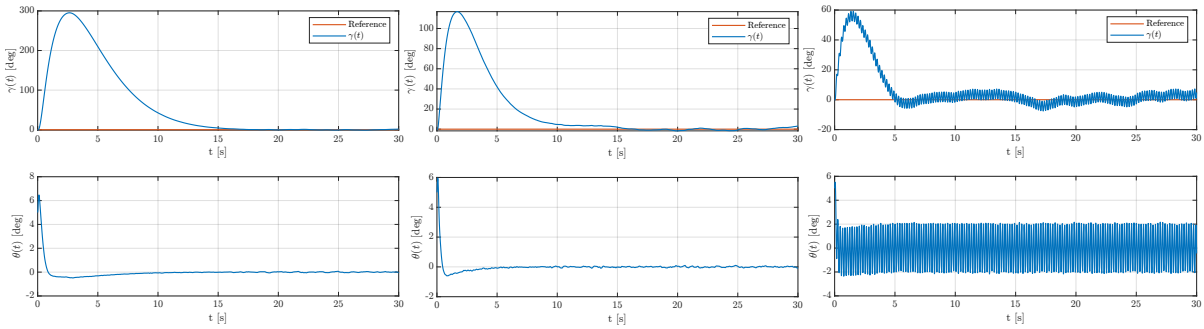
The complementary filter allows for a so-called frequency filtering in which it is possible to choose a certain frequency f_c, called the cut-off frequency, to change the properties of the filtered signal.

Initially, the value of the complementary filter cut-off frequency used for parameter estimation was 0.35Hz. The value of this parameter has been changed to see how the dynamics changes as it increases or decreases. The tests analyzed the stabilisation of the balancing robot, with initial condition $\pi/36$ of θ with respect to the vertical position.

Figures 19 and 20 show the evolutions of γ and θ with nominal and robust controller, specifically with cut-off frequencies of 0.1Hz, 0.35Hz and 1Hz.

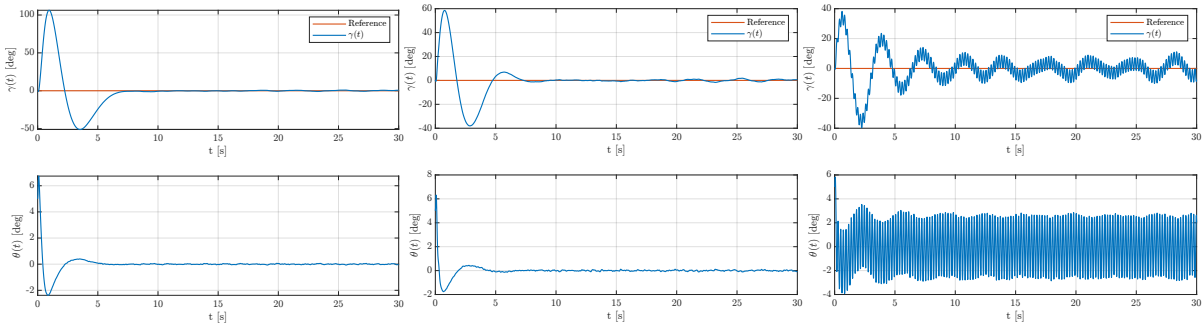
In accordance with theory, increasing f_c eliminates less high-frequency noise. So, as the cut-off frequency moves higher, the measurement of θ becomes noisier. Furthermore, given the noisier θ measurement, the controller attempts to position the robot in the vertical position by moving the wheels back and forth to stabilise itself. This behaviour becomes more and more evident as the cut-off frequency increases, see figures 19c and 20c, since the acceletometer's component is lesser attenuated by the complementary filter.

In conclusion, $f_c = 0.35$Hz can be considered a very good trade-off between speed convergency and a noisy evolution.



(a) f_c=0.1Hz (b) f_c=0.35Hz (c) f_c=1Hz

Figure 19: Different cut-off frequency for the complementary filter using nominal controller



(a) f_c=0.1Hz (b) f_c=0.35Hz (c) f_c=1Hz

Figure 20: Different cut-off frequency for the complementary filter using robust controller

4 Controller Performance Analysis

To assess numerically the performance of the different controllers respect to the various performed test, the Root Mean Square Error (RMSE) indicator has been used.
The values, reported in Tables 2, 3 and 4, have been obtained through the MATLAB code in Appendix A.2.7.
The data refer only to the experiments perfomed in laboratory, from them some observation can be done:

- In the tracking of the longitudinal position it can be noticed that the best performances are obtained, both for reference and disturb perturbances, with the controller characterized by ρ=500 among the nominal controllers and the controller with ρ=500 and q_{11}=1 among the robust controllers.
  On the contrary, the worst performances are obtained with the controller characterized by ρ=5000 among the nominal controllers and the controller with ρ=5000 and q_{11}=0.1 among the robust controllers.
  This is explanable thinking that in the fist case the controller provides a strong control, reducing a lot the settling time, reducing by consequence also the tracking error and hence the RMSE.
  The same reasoning can be applied in the second case, with the opposite conclusion.

- The behavior of the various controller respect to the evolution of θ is pratically the same for every controller and hence the RMSEs are very similar.

- For what concerns ψ, it is possible to observe the difference among the various typer of controllers, in fact the best performances are clearly obtained with the robust controllers equipped with the yaw one, the intermediate performances are obtained with the only robust controllers and the worst performance are provided by the nominal controllers.

	$\gamma(t)$	$\theta(t)$	$\psi(t)$
Yaw Robust reference $\rho = 500$ and $q_{11} = 0.1$	20.0095	1.3777	0.3560
Yaw Robust reference $\rho = 500$ and $q_{11} = 1$	16.6087	1.7242	0.6869
Yaw Robust reference $\rho = 5000$ and $q_{11} = 0.1$	26.4193	1.5050	0.8253
Yaw Robust reference $\rho = 5000$ and $q_{11} = 1$	23.7508	1.1879	1.0549
Yaw Robust disturbance $\rho = 500$ and $q_{11} = 0.1$	21.0171	1.0474	0.4636
Yaw Robust disturbance $\rho = 500$ and $q_{11} = 1$	10.4301	1.6221	0.5209
Yaw Robust disturbance $\rho = 5000$ and $q_{11} = 0.1$	52.6930	1.5033	1.1468
Yaw Robust disturbance $\rho = 5000$ and $q_{11} = 1$	19.8332	1.7124	1.0205

Table 2: Performance Analysis for robust controller with Yaw controller: RMSE

	$\gamma(t)$	$\theta(t)$	$\psi(t)$
Robust reference $\rho = 500$ and $q_{11} = 0.1$	25.8882	1.4670	2.9359
Robust reference $\rho = 500$ and $q_{11} = 1$	16.8712	1.7822	7.9979
Robust reference $\rho = 5000$ and $q_{11} = 0.1$	32.0787	1.5985	5.8698
Robust reference $\rho = 5000$ and $q_{11} = 1$	20.6896	1.1310	4.2127
Robust disturbance $\rho = 500$ and $q_{11} = 0.1$	30.4278	1.2326	1.4385
Robust disturbance $\rho = 500$ and $q_{11} = 1$	14.6622	1.6613	11.7500
Robust disturbance $\rho = 5000$ and $q_{11} = 0.1$	56.5151	1.6623	13.0066
Robust disturbance $\rho = 5000$ and $q_{11} = 1$	22.9044	1.3625	4.1924

Table 3: Performance Analysis for robust controller: RMSE

	$\gamma(t)$	$\theta(t)$	$\psi(t)$
Nominal reference $\rho = 500$	141.3367	1.0831	22.2117
Nominal reference $\rho = 5000$	373.8196	1.0183	32.1562
Nominal disturbance $\rho = 500$	994.5103	1.1451	75.4146
Nominal disturbance $\rho = 5000$	2.0582e+03	1.1930	38.4232

Table 4: Performance Analysis for nominal controller: RMSE

5 Additional tests

In laboratory, additional tests have been done, in order to test the behavior of the balancing robot in conditions that are different from the assigned ones.
In particular, three tests have been performed, in every test the usual LQR robust controller has been used, in addition with a PI controller for the yaw angle. The used parameter are $\rho = 500$ and $q_{11} = 0.1$ for the feedback controller, in addition to $K_P = 3.3$ and $K_D = 0.7$ for the yaw controller.
The tests were performed by subjecting the robot to the reference exchange and the application of the disturbance in the same test, respectively at t = 10s and t = 20s.

5.1 Change ψ reference

The first performed test was try to change the reference for the ψ angle.
Initially the test has been done with higher values of ψ_{ref}, like 90/180/360deg, but the controller, due to an higher output control, was not able to maintain the vertical position. To solve the problem, probably is sufficient to change the gains of the PI controller, reducing them.
In order to complete the test without implementing a trial-and-error procedure to change such parameters, it has been sufficient to reduce the reference value to ψ_{ref}=45deg, obtaining the response in figure 21.
The reference is reached in a good time and without overshoot in the ψ parameter.

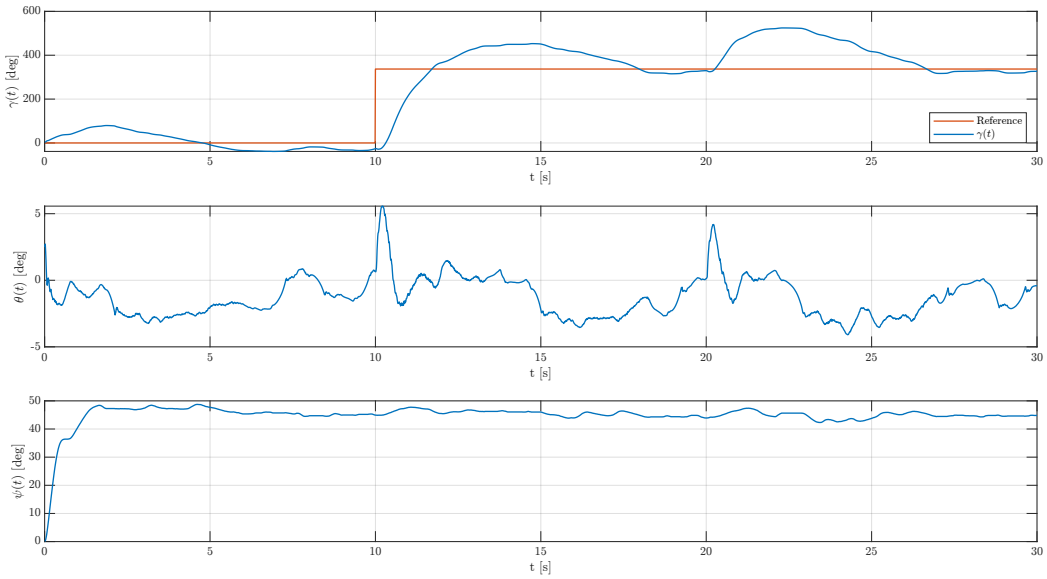


Figure 21: Evolution of the robot considering a change of ψ_{ref}

5.2 Inclined surface

In the second permformed test the balancing robot has been positioned over a inclined surface, obtained using some stuff found in laboratory.
This surface is characterized by 20cm of height and 100cm of lenght. The inclination angle is obtained by:

$$\alpha = arcsin\left(\frac{20}{100}\right) = 12.82\, deg$$

In figure 22 is possible to observer how, initially, the robot fall down along the surface. This happens because the controller is not expecting a sloped surface and therefore needs a moment to adjust the control.
It is interesting to observe the steady-state position of the angle θ. In fact the robot is not stabilized around the vertical position θ=0deg, it stabilize instead around θ=13deg, which is the inclination of the surface.
Another thing to note is the response to reference change and disturb application, which induces an upward motion along the sloped surface. In fact, the overshoots are reduced with respect to the same input in the orizontal plane, this is probably due to the gravity force that reduce the speed of the robot and the effect of the disturb.

5.3 Changing the center of mass

The third performed test was changing the center of mass of the balancig robot, in order to verify the behavior of the controller in a situation in which the model is not estimated correctly. To do that, a bottle of whater, 250g of weight, has been positioned over the robot. Unfortunately, it has not been possible to fix the bottle to the chassis of the robot, which would be the optimal hardware setup for the test.
For this reason is important to underline that the test results more difficult than with other

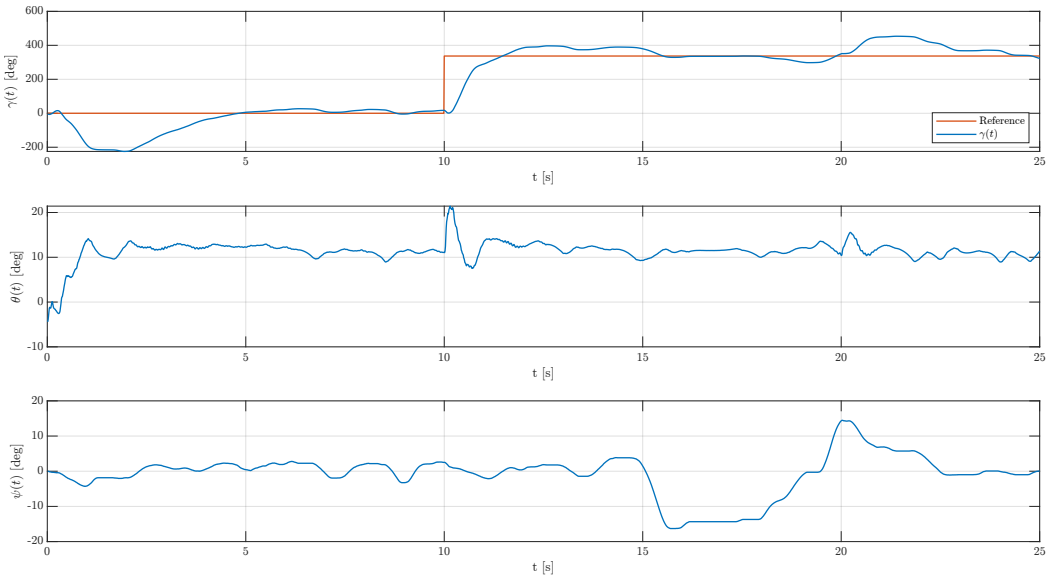


Figure 22: Evolution of the robot putting it in an sloped surface

objects, because the movement of the robot causes oscillations in the mass of water, making the conditions more critical.

In figure 23 is shown the behavior of the robot, which has good performances in terms of longitudinal and yaw control, not very good performance for the stabilization in vertical position. This is explanable thinking that the controller has been designed for a certain model and hence it controls a system that is very different from the one it thinks to control, and hence providing a not perfect action.

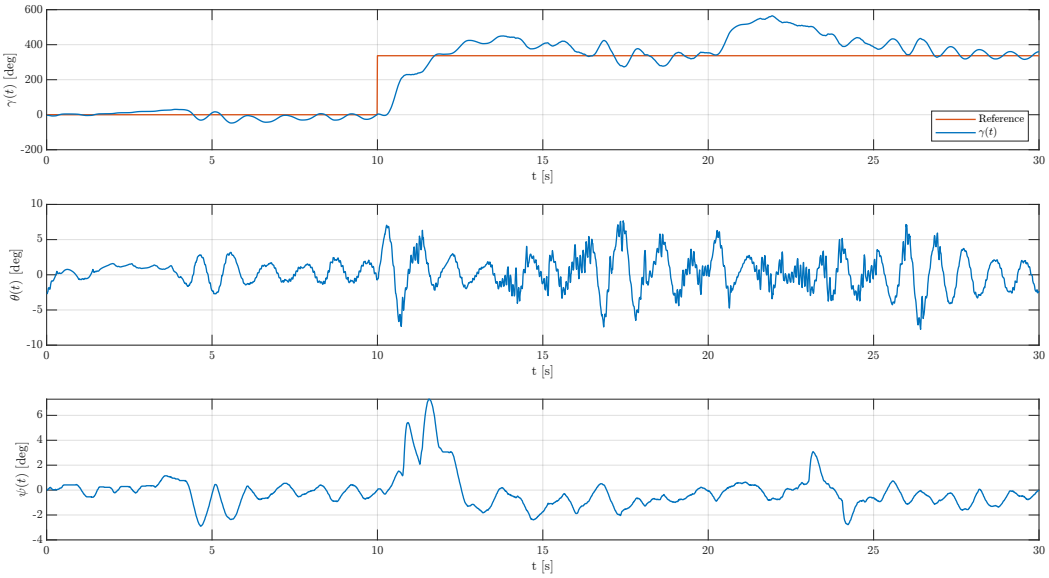


Figure 23: Evolution of the robot putting a bottle of water over it

6 General considerations

During simulations and tests in the laboratory, we noticed that the robot has the same behaviour as a non-minimum phase system. The typical behaviour of a non-minimum phase system is an initial displacement of the system to the opposite side of the reference. This is a typical characteristic that can also be found, for instance, in the so-called inverted pendulum, of which the balancing robot can be considered a more complex version.
It can be seen in the figures obtained in simulation and in the laboratory, particularly with respect to the parameter γ. In particular, it is more evident in the tests performed with respect to the tracking of the constant reference. At the arrival of the step signal, in fact, the response has an initial undershoot, which represents the wheels moving backwards, indicating that the robot initially moves from the opposite side to the prefixed one.
In figure 24, the behaviour just described can clearly observed.

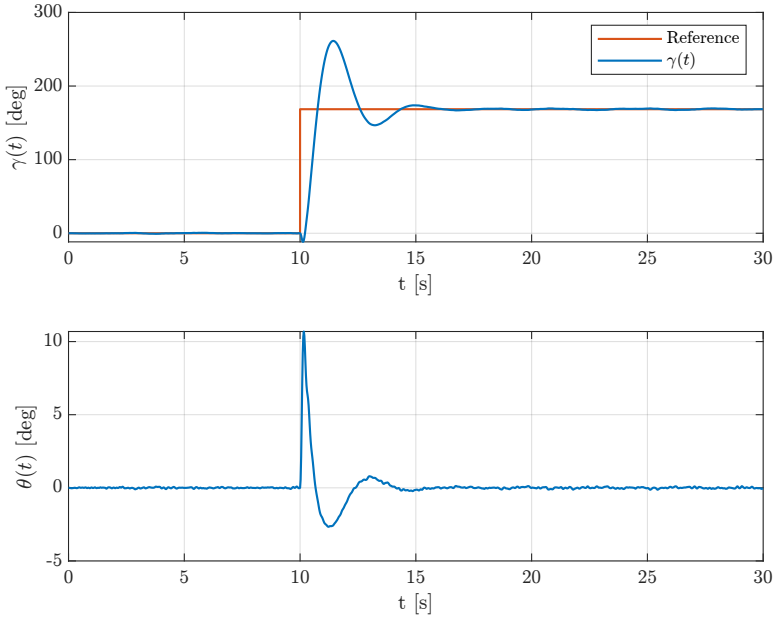


Figure 24: Minimum phase behaviour, observable in corrispondence of the reference change at t=10s.

A Appendix

A.1 Nominal model of the balancing robot

In the following table it is possible to see the geometrical and inertial parameters for each body.

• **Robot body**			
Center–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]
Center–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]
Center–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]
Center–of–Mass coords wrt body frame $\{b\}$	$x_{stat}^b,\ y_{stat}^b,\ z_{stat}^b$	$0,\ \pm 52.1,\ -7$	[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]
Center–of–Mass coords wrt body frame $\{b\}$	$x_{rot}^b,\ y_{rot}^b,\ z_{rot}^b$	$0,\ \pm 42.7,\ -7$	[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]
Center–of–Mass coords wrt body frame $\{b\}$	$x_w^b,\ y_w^b,\ z_w^b$	$0,\ \pm 100,\ 0$	[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]

Table 5: Geometrical and inertial nominal parameters

A.1.1 Lagrangian function

The contribution for the Lagrangian function $\boldsymbol{\mathcal{L}} = \boldsymbol{T} - \boldsymbol{V}$ are reported in this section, the kinetic energy is given by summing the following contributions:

$$T = T_b + T_w + T_{rot} \tag{16}$$

where T_b is the robot body kinetic and it is equal to:

$$T_b = \frac{1}{2} m_b r^2 \dot{\gamma}^2 + \frac{1}{2}(I_{b,yy} - m_b l^2)\dot{\theta}^2 + m_b \dot{\gamma}\dot{\theta} r l cos\theta \tag{17}$$

instead T_w is the kinetic of a single -wheel

$$T_w = (I_{w,yy} + m_w r^2)\dot{\gamma}^2 \tag{18}$$

and the rotor kinetic energy is:

$$\begin{aligned} T_{rot} = (N^2 I_{rot,yy} + m_{rot} r^2)\dot{\gamma}^2 + \big[(1-N)^2 I_{rot,yy} +_{rot} (z^b_{rot})^2\big]\dot{\theta}^2 + \dots \\ \dots + 2\big[N(1-N) I_{rot,yy} + m_{rot} r z^b_{rot} cos\theta\big]\dot{\gamma}\dot{\theta} \end{aligned} \tag{19}$$

And for the Potential energy we have:

$$U = U_b + U_w + U_{rot} \tag{20}$$

where:

$$\begin{aligned} U_b &= m_b g l cos\theta \\ U_w &= (2m_w) g z c_w = 0 \\ U_{rot} &= (2m_{rot}) g z^b_{rot} cos\theta \end{aligned} \tag{21}$$

So now it is possible to compute the Lagrangian function, which corresponds to:

$$\begin{aligned} \boldsymbol{\mathcal{L}} = \big[I_{w,yy} + N^2 I_{rot,yy} + \big(\tfrac{1}{2} m_b + m_w + m_{rot}\big) r^2\big]\dot{\gamma}^2 + \dots \\ \dots + \big[\tfrac{1}{2} I_{b,yy} + (1-N)^2 I_{rot,yy} + \tfrac{1}{2} m_b l^2 + m_{rot}(z^b_{rot})^2\big]\dot{\theta}^2 + \dots \\ \dots + \big[2N(1-N) I_{rot,yy} + \big(m_b l + 2 m_{rot} z^b_{rot}\big) r cos\theta\big]\dot{\gamma}\dot{\theta} - \dots \\ \dots - \big(m_b l + 2 m_{rot} z^b_{rot}\big) g cos\theta \end{aligned} \tag{22}$$

where all the symbols used can be found in Tab.5.

A.1.2 System linearization

For the linearization of (1) around P_0 it is necessary to assume small deviation for $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}$ that will be denote as $\delta\boldsymbol{q}, \delta\dot{\boldsymbol{q}}, \delta\ddot{\boldsymbol{q}}$. So we evaluate:

$$\boldsymbol{f}(P_0) + \frac{\partial \boldsymbol{f}(P_0)}{\partial \boldsymbol{q}}\delta\boldsymbol{q} + \frac{\partial \boldsymbol{f}(P_0)}{\partial \dot{\boldsymbol{q}}}\delta\dot{q} + \frac{\partial \boldsymbol{f}(P_0)}{\partial \ddot{\boldsymbol{q}}}\delta\ddot{q} + \frac{\partial \boldsymbol{f}(P_0)}{\partial \tau}\delta\tau \tag{23}$$

that bring to the linearized model seen in (2) and gives the following matrices for inertia and for Coriolis effect:

$$\boldsymbol{M} = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}, \qquad \boldsymbol{G} = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix}, \tag{24}$$

with all the terms described in the following:

$$\begin{aligned} M_{11} &= 2I_{w,yy} + 2N^2 I_{rot,yy} + (m_b + 2m_w + 2m_{rot})r^2 \\ M_{12} &= M_{21} = 2N(1-N)I_{rot,yy} + (m_b l + 2m_{rot} z^b_{rot})r \\ M_{22} &= I_{b,yy} + 2(1-N)^2 I_{rot,yy} + m_b l^2 + 2m_{rot}(z^b_{rot})^2 \end{aligned} \tag{25}$$

$$\begin{aligned} G_{11} &= G_{12} = G_{21} = 0 \\ G_{22} &= -\big(m_b l + 2m_{rot} z^b_{rot}\big)g \end{aligned} \tag{26}$$

.

Instead for the matrix F_v of viscous friction coefficients we have:

$$\boldsymbol{F_v} = \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix} \tag{27}$$

where:

$$\begin{aligned} F_{v,11} &= 2\big(B + B_w\big) \\ F_{v,12} &= F_{21} = -2B \\ F_{v,22} &= 2B \end{aligned} \tag{28}$$

that, after linearization, becomes:

$$\boldsymbol{F'_v} = \boldsymbol{F_v} + \frac{2N^2 k_t k_e}{R_a}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{29}$$

while the input torque becomes:

$$\boldsymbol{\tau}' = \frac{2Nk_t}{R_a}\begin{bmatrix} 1 \\ -1 \end{bmatrix} u_a \tag{30}$$

The Gearbox viscous friction coefficient B is equal to $25 \times 10^{-3} Nm/(rad/s)$, instead the wheel viscous friction coefficient namely B_w is $1.5 \times 10^{-3} Nm/(rad/s)$

Armature resistance	R_a	$2.4\,\Omega$
Armature inductance	L_a	n.a. (neglected)
Electric (BEMF) constant	k_e	$10.3 \times 10^{-3}\,\mathrm{V\,s/rad}$
Torque constant	k_m	$5.2 \times 10^{-3}\,\mathrm{Nm/A}$
Gearbox ratio	N	30

Table 6: DC gearmotor nominal parameters.

A.1.3 Actuator system modelling

By studying the electrical dynamic is possible to obtain the following equation:

$$L_a \frac{di_a}{dt} + R_a i_a + k_e \frac{d\Delta\theta_{rot}}{dt} = u_a \tag{31}$$

where R_a and L_a are, respectively, the resistence and inductance of the armature circuit, k_e is the BEMF, u_a is the supply voltate to the armature circuit and $\Delta\theta_{rot} = \theta_{rot} - \theta$ is the angular displacement of the rotor with respect to the stator.
By supposing that the time constant $\tau_d = R_a/L_a$ is negligible, the current i_a can be expressed as

$$i_a = \frac{1}{R_a}\big[u_a - k_e N(\dot{\gamma} - \dot{\theta})\big] \tag{32}$$

the same can be done with the torque τ

$$\tau = \frac{N k_t}{R_a} u_a - \frac{N^2 k_t k_e}{R_a}(\dot{\gamma} - \dot{\theta}) \tag{33}$$

with these results it is possible to obtain the matrices (4).

A.1.4 Complementary Filter Order

One of the main parameters useful for the controller is the tilt angle θ. This parameter need to be estimated from the data provided by the *Motion Processing Unit*. The best possible solution to have a good estimate of θ for every frequency, is to use both, the accelerometer and the gyroscope sensors. The former works well in low frequency [2] instead, the latter, works better with high frequency since it is affected by a bias and a linear ramp error term (*drift*) that are less relevant in high frequency. By means of a complementary filter is possible, in some sense, to select the best sensor based on the frequency. With a trial-and-error approach it is possible to select the best order for the complementary filter which transfer functions can be seen in the following.

[2] This happen because we assume that the robot motion in terms of θ and γ is slow, if the robot e.g. is falling or hits something, this assumption is no longer true. In addition the sensor has an high output noise.

- First order complementary filter transfer function:

$$H(s) = \frac{1}{T_c s + 1}, \qquad 1 - H(s) = \frac{T_c s}{T_c s + 1} \tag{34}$$

- Second order complementary filter transfer function:

$$H(s) = \frac{2T_c s + 1}{(T_c s + 1)^2}, \qquad 1 - H(s) = \frac{T_c^2 s^2}{(T_c s + 1)^2} \tag{35}$$

- Third order complementary filter transfer function:

$$H(s) = \frac{3T_c^2 s^2 + 3T_c s + 1}{(T_c s + 1)^3}, \qquad 1 - H(s) = \frac{T_c^3 s^3}{(T_c s + 1)^3} \tag{36}$$

where $T_c = 1/(2\pi f_c)$ and f_c denote a tentative value for the cut-off frequency of the filter (for the simulation it has been used $f_c = 0.35$Hz).

A.2 MATLAB code

A.2.1 Non-linear Electromechanical model

```
% Non-linear Electromechanical system for Simulink implementation
% input conversion
ua2tau1 = 2*gbox.N*mot.Kt/mot.R*[1; -1];

% inertia matrix
M11q = 2*wheel.Iyy + 2*mot.rot.Iyy*gbox.N^2 + (body.m + 2*(mot.rot.m+wheel.m))*wheel.r^2;
M21q = 2*(1-gbox.N)*gbox.N*mot.rot.Iyy + wheel.r*(body.zb*body.m + 2*mot.rot.m*mot.rot.zb)*cos(u);
M21q = M12q;
M22q = body.Iyy + 2*(1-gbox.N)^2*mot.rot.Iyy + body.m*body.zb^2 + 2*mot.rot.m*mot.rot.zb^2;

% matrix of centrifugal and Coriolis-related coefficients
C11qqdot = 0;
C12qqdot = -wheel.r*(body.m*body.zb + 2*mot.rot.m*mot.rot.zb)*sin(u)*dot_th; % dot_th is the derivative of the input u

% torque contribution due to gravity (gravity compensation)
g1q = 0;
g2q = -g*(body.m*body.zb + 2*mot.rot.m*mot.rot.zb)*sin(u);

% Accelerometer
% non-linear functions of the Simulink model, u is the input:
wheel.r*u(5)*cos(u(2))+sens.mpu.zb*u(6)+g*sin(u(2)) % x-axis
wheel.r*u(5)*sin(u(2))-sens.mpu.zb*u(4)^2-g*cos(u(2)) % z-axis
```

A.2.2 Linearized Electromechanical model

```
close all
clear all
clc

% loading parameters
balrob_params;

%% linearized model
% state q = [gamma, theta, dot_gamma, dot_theta]
M11q = 2*wheel.Iyy + 2*mot.rot.Iyy*gbox.N^2 + (body.m + 2*(mot.rot.m+wheel.m))*wheel.r^2;
M12 = 2*gbox.N*(1-gbox.N)*mot.rot.Iyy+(body.m*body.zb+2*mot.rot.m*mot.rot.zb)*wheel.r;
M21 = M12;
M22q = body.Iyy + 2*(1-gbox.N)^2*mot.rot.Iyy + body.m*body.zb^2 + 2*mot.rot.m*mot.rot.zb^2;
M = [M11,M12; ...
     M21,M22];

```

```
G11 = 0;
G12 = 0;
G21 = 0;
G22 = -(body.m*body.zb+2*mot.rot.m*mot.rot.zb)*g;
G = [G11,G12;...
     G21,G22];

% viscous friction
Fv11 = 2*(gbox.B + wheel.B);
Fv12 = -2*gbox.B;
Fv21 = Fv12;
Fv22 = 2*gbox.B;
Fv = [Fv11, Fv12;...
      Fv21, Fv22];
Fv1 = Fv + (2*gbox.N^2*mot.Kt*mot.Ke)/mot.R*[1,-1;-1,1];

% state space model formule (57) e (58) considering also actuators system
Minv = inv(M);
A21 = -Minv*G;
A22 = -Minv*Fv1;
A = [zeros(2,size(A21,2)),eye(size(A22,2));...
    A21,A22];
B = 2*gbox.N*mot.Kt/mot.R*[zeros(size(A,1)-size(Minv,1),size(Minv,2));Minv]*[1;-1];

sysC = ss(A,B,zeros(1,size(A,2)),zeros(1,size(B,2)));

%% check whether the system is controllable and observable
R = ctrb(sysC); % Controllability matrix
Ob = obsv(sysC); % Observability matrix

rankR = rank(R) % rank of controllability matrix
rankOb = rank(Ob) % rank of observability matrix

Nc = size(R,1) - rank(R) % number of uncontrollable states
No = size(Ob,1) - rank(Ob) % number of unobservable states

%% exact discretization
Ts = 0.01;
sysD = c2d(sysC, Ts, 'zoh');
[Phi,Gamma,~,~] = ssdata(sysD);
H = [1,0,0,0];
```

A.2.3 Simple State-space observer

```
% discrete-time complementary filter to merge sensors
fc = 0.35; % cut-off frequency [Hz]
Tc = 1/(2*pi*fc);

z = tf('z',Ts);

% Forward-Euler
s = (z-1)/Ts;
HFE = 1/(Tc*s+1);

% Backward-Euler
s = (z-1)/(Ts*z);
HBE = 1/(Tc*s+1);

% chosen low-pass
Hz = HFE;
Hcz = 1 - Hz;

% real-derivative filter to obtain angular speeds formula (89)
N = 3;
Hw = (1-z^-N)/(N*Ts);
```

A.2.4 Discrete LQR Nominal Controller

```
% feedforward
bb = zeros(size(Phi,1),1);
bb = [bb;ones(size(H,1))];
Nxu = [Phi - eye(size(Phi,1)), Gamma; H, zeros(size(H,1))]\bb;
Nx = Nxu(1:end-1);
Nu = Nxu(end);

% Bryson's rule
```

```
gammaBar = pi/18;
thetaBar = pi/360;
uBar = 1;

q11 = 1/gammaBar^2;
q22 = 1/thetaBar^2;
q33 = 0;
q44 = 0;
Q = diag([q11,q22,q33,q44]);

r11 = 1/uBar^2;
rho = 500;
R = r11*rho;

K = dlqr(Phi,Gamma,Q,R);
```

A.2.5 Discrete LQR Robust Controller

```
% extended state with integral action: xe = [xI, x]
Phie = [1,H;...
        zeros(size(Phi,1),1),Phi];
Gammae = [0;...
          Gamma];

% feedforward
bb = zeros(size(Phi,1),1);
bb = [bb;ones(size(H,1))];
Nxu = [Phi - eye(size(Phi,1)), Gamma; H, zeros(size(H,1))]\bb;
Nx = Nxu(1:end-1);
Nu = Nxu(end);

% Bryson's rule
gammaBar = pi/18;
thetaBar = pi/360;
uBar = 1;

q11 = 0.1;          % integral error weight
q22 = 1/gammaBar^2;
q33 = 1/thetaBar^2;
q44 = 0;
q55 = 0;
Q = diag([q11,q22,q33,q44,q55]);

r11 = 1/uBar^2;
rho = 500;
R = r11*rho;

Ke = dlqr(Phie,Gammae,Q,R);
Ki = Ke(1);
K = Ke(2:end);
```

A.2.6 Yaw PID controller

```
yaw_Kp = 3.3;
yaw_Ki = 0.7;
```

A.2.7 Performance Analysis - RMSE

```
% Compute RMSE
ref = data.out{1,1}(2,1:end-1);
traj = data.out{1,1}(1,1:end-1);

theta = data.out{2,1}(1,1:end-1);
stab = zeros(length(theta), 1);

psi = data.out{2,1}(2,1:end-1);
drift = zeros(length(psi), 1);

RMSE_traj = sqrt(mean(mean((ref-traj).^2)))
RMSE_stab = sqrt(mean(mean((stab-theta).^2)))
RMSE_drift = sqrt(mean(mean((drift-psi).^2)))
```