ELEC-E8101 Group project:
Lab A report
Group 11

HOUILLE Florent, MONTE SASTRE Enrique, SOULARD Yoann, WANG Tong

November 3, 2019

Instructions: *For this lab report there is no size limit on your report, but try to be concise.*

Reporting of Task 4.1

4.1.1 We chose the following x:

$$x = \begin{bmatrix} x_w \\ \dot{x}_w \\ \theta_b \\ \dot{\theta}_b \end{bmatrix}$$

4.1.2 Here are the EOM (20) re-arranged in order to establish γ, α and β matrices as suggested in **Hint 4.1**:

$$\begin{cases} m_b l_b \ddot{x}_w + (I_b + m_b l_b^2)\ddot{\theta}_b = m_b l_b g \theta_b - \dfrac{2K_t}{R_m} v_m + (\dfrac{2K_e K_t}{Rm} + b_f)(\dfrac{\dot{x}_w}{l_w} - \dot{\theta}_b) \\ (\dfrac{I_w}{l_w} + l_w m_b + l_w m_w)\ddot{x}_w + m_b l_b l_w \ddot{\theta}_b = \dfrac{2K_t}{R_m} v_m - (\dfrac{2K_e K_t}{Rm} + b_f)(\dfrac{\dot{x}_w}{l_w} - \dot{\theta}_b) \end{cases}$$

We have the following relation:

$$\begin{bmatrix} \gamma_{11} & \gamma_{12} \\ \gamma_{21} & \gamma_{22} \end{bmatrix} \begin{bmatrix} \ddot{x}_w \\ \ddot{\theta}_b \end{bmatrix} = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{13} & \alpha_{14} \\ \alpha_{21} & \alpha_{22} & \alpha_{23} & \alpha_{24} \end{bmatrix} \begin{bmatrix} x_w \\ \dot{x}_w \\ \theta_b \\ \dot{\theta}_b \end{bmatrix} + \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} v_m$$

Therefore:

$$\gamma_{11} = m_b l_b \qquad \gamma_{12} = I_b + m_b l_b^2$$
$$\gamma_{21} = \frac{I_w}{l_w} + l_w m_b + l_w m_w \qquad \gamma_{22} = m_b l_b l_w$$

$$\alpha_{11} = 0 \quad \alpha_{12} = \frac{1}{l_w}(\frac{2K_e K_t}{Rm} + b_f) \quad \alpha_{13} = m_b l_b g \quad \alpha_{14} = -(\frac{2K_e K_t}{Rm} + b_f)$$
$$\alpha_{21} = 0 \quad \alpha_{22} = -\frac{1}{l_w}(\frac{2K_e K_t}{Rm} + b_f) \quad \alpha_{23} = 0 \quad \alpha_{24} = \frac{2K_e K_t}{Rm} + b_f$$

$$\beta_1 = -2\frac{K_t}{K_e}$$
$$\beta_2 = 2\frac{K_t}{K_e}$$

Using second part of **Hint 4.1**, we can now deduce matrices A, B, C and D as following:

$$\gamma^{-1} = \frac{1}{\Delta_\gamma}\begin{bmatrix}\gamma_{22} & -\gamma_{12}\\ -\gamma_{21} & \gamma_{11}\end{bmatrix} = \frac{1}{\gamma_{11}\gamma_{22}-\gamma_{12}\gamma_{21}}\begin{bmatrix}\gamma_{22} & -\gamma_{12}\\ -\gamma_{21} & \gamma_{11}\end{bmatrix}$$

$$A = \frac{1}{\Delta_\gamma}\begin{bmatrix}0 & \Delta_\gamma & 0 & 0\\ \gamma_{22}\alpha_{11}-\gamma_{12}\alpha_{21} & \gamma_{22}\alpha_{12}-\gamma_{12}\alpha_{22} & \gamma_{22}\alpha_{13}-\gamma_{12}\alpha_{23} & \gamma_{22}\alpha_{14}-\gamma_{12}\alpha_{24}\\ 0 & 0 & 0 & \Delta_\gamma\\ -\gamma_{21}\alpha_{11}+\gamma_{11}\alpha_{21} & -\gamma_{21}\alpha_{12}+\gamma_{11}\alpha_{22} & -\gamma_{21}\alpha_{13}+\gamma_{11}\alpha_{23} & -\gamma_{21}\alpha_{14}+\gamma_{11}\alpha_{24}\end{bmatrix}$$

$$B = \frac{1}{\Delta_\gamma}\begin{bmatrix}\gamma_{22}\beta_1-\gamma_{12}\beta_2\\ -\gamma_{21}\beta_1+\gamma_{11}\beta_2\end{bmatrix}$$

$$C = \begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix}\ D = \begin{bmatrix}0\end{bmatrix}$$

4.1.3 And here are the numerical values computed with Matlab and rounded to 10^{-2}:

$$A = \begin{bmatrix}0 & 1 & 0 & 0\\ 0 & -773.79 & -6.57 & 16.25\\ 0 & 0 & 0 & 1\\ 0 & 3313.24 & 63.07 & -69.58\end{bmatrix}\ B = \begin{bmatrix}0\\ 36.60\\ 0\\ -156.71\end{bmatrix}$$

$$C = \begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix}\ D = \begin{bmatrix}0\end{bmatrix}$$

Reporting of Task 4.2

4.2.1 We provide state space's matrices to Matlab in order to compute the system's transfer function (numerical values rounded to 10^{-2}):

$$G(s) = \frac{-156.71s}{s^3 + 843.36s^2 - 63.07s - 2.70.10^4}$$

We can then compute zeroes and poles to obtain this factorized form:

$$G(s) = \frac{-156.71s}{(s+843.40)(s-5.68)(s+5.64)}$$

4.2.2 When we searched by hand the zeroes and poles, we were expecting only one zero (order 2). But with Matlab we had two zeroes which prevented us from cancelling some poles. These unexpected zeroes came from very small (10^{-14} to 10^{-11}) coefficients in the numerator, themselves produced by artithmetic errors in Matlab. Once we deleted them, we obtained the wanted transfer function.

Reporting of Task 4.3

4.3.1 We have the following system:

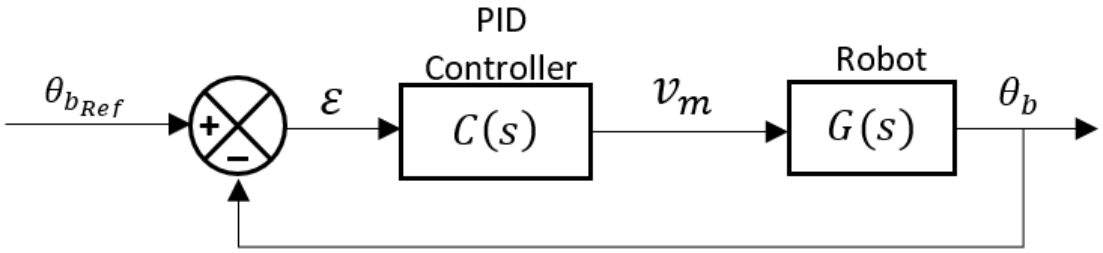


Figure 1: Closed loop continuous system

With the following system and controller transfer functions:

$$G(s) = \frac{Ks}{(s-p_1)(s-p_2)(s-p_3)}$$

$$C(s) = K_p + K_i\frac{1}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s}$$

Here is the closed-loop transfer funtion:

$$H_{CL}(s) = \frac{C(s)G(s)}{1 + C(s)G(s)} = \frac{K(K_d s^2 + K_p s + K_i)}{(s-p_1)(s-p_2)(s-p_3) + K(K_d s^2 + K_p s + K_i)}$$

Therefore the developped denominator of H_{CL} is:

$$P(s) = s^3 + s^2(KK_d - p_1 - p_2 - p_3) + s(KK_p + p_1p_2 + p_2p_3 + p_1p_3) + KK_i - p_1p_2p_3$$

We already know that our plant has got 3 poles, one of which is unstable ($p_2 > 0$). Therefore, we will try to replace this pole with a stable one and end up with the following closed-loop denominator:

$$P(s) = (s-p_1)(s-Q)(s-p_3) = s^3 + s^2(-p_1 - Q - p_3) + s(p_1Q + p_3Q + p_1p_3) - p_1p_3Q$$

With a purely real negative Q we will have a stable system without oscillations. Then we can identify the terms between the polynom we have and the one we want to obtain the following equations system and therefore deduce the PID's parameters:

$$\begin{cases} KK_d - p_1 - p_2 - p_3 = -p_1 - Q - p_3 \\ KK_p + p_1p_2 + p_2p_3 + p_1p_3 = p_1p_3 + p_1Q + p_3Q \\ KK_i - p_1p_2p_3 = -p_1p_3Q \end{cases} \quad \Longrightarrow \quad \begin{cases} K_d = \dfrac{p_2 - Q}{K} \\ K_p = \dfrac{p_1Q + p_3Q - p_1p_2 - p_2p_3}{K} \\ K_i = \dfrac{p_1p_2p_3 - p_1p_3Q}{K} \end{cases}$$

We want the impulse response to asymptotically reach the null angle as this is our reference angle. The transient response is not important as the impulse input itself is not physically meaningful.

4.3.2 With $Q = -30$ we obtain the following PID parameters:

$$\begin{cases} K_p = -193.31 \\ K_i = -1.1.10^3 s \\ K_d = -0.23 s \end{cases}$$

4.3.3 And this closed-loop transfer function:

$$H_{CL}(s) = \frac{35.68s^4 + 3.029.10^4 s^3 + 1.7.10^5 s^2}{s^5 + 879s^4 + 3.023.10^4 s^3 + 1.4.10^5 s^2} \approx \frac{1}{1 + 0.3s}$$

The first order system is obtained after approximating and canceling zeroes and poles using Matlab's *minreal* tool. It helps us to predict the overall system's behaviour (it should be close to a regular first order system).
We keep the full system and obtain the following impulse response:

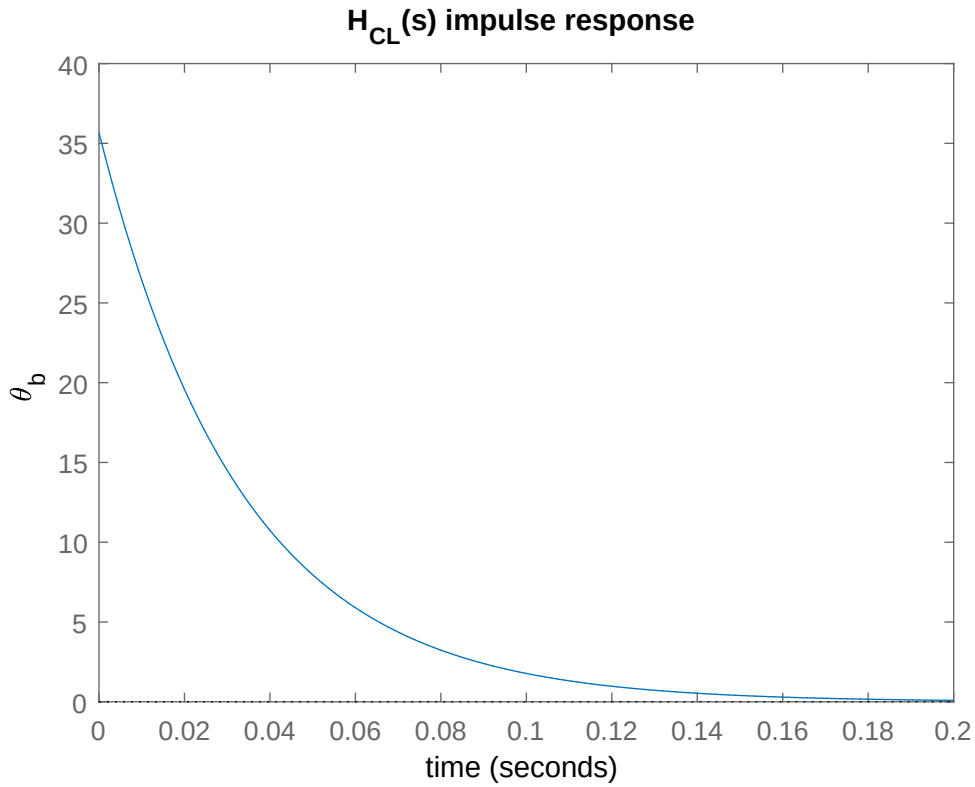


Figure 2: $H_CL(s)$ impulse response

The system asymptotically reaches 0, this is what we want.

Reporting of Task 4.4

4.4.1 We start with adding the disturbance d to Fx:

$$Fx + d = m_b \ddot{x}_b$$

The updated EOM in parametric form are:

$$\begin{cases} m_b l_b \ddot{x}_w + (I_b + m_b l_b^2)\ddot{\theta}_b = m_b l_b g \theta_b - \dfrac{2K_t}{R_m} v_m + (\dfrac{2K_e K_t}{Rm} + b_f)(\dfrac{\dot{x}_w}{l_w} - \dot{\theta}_b) + d l_b \\ (\dfrac{I_w}{l_w} + l_w m_b + l_w m_w)\ddot{x}_w + m_b l_b l_w \ddot{\theta}_b = \dfrac{2K_t}{R_m} v_m - (\dfrac{2K_e K_t}{Rm} + b_f)(\dfrac{\dot{x}_w}{l_w} - \dot{\theta}_b) + d l_w \end{cases}$$

4.4.2 We can now compute the new state space model:

$$\begin{bmatrix} \dot{x}_w \\ \ddot{x}_w \\ \dot{\theta}_b \\ \ddot{\theta}_b \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -773.79 & -6.57 & 16.25 \\ 0 & 0 & 0 & 1 \\ 0 & 3313.24 & 63.07 & -69.58 \end{bmatrix} \begin{bmatrix} x_w \\ \dot{x}_w \\ \theta_b \\ \dot{\theta}_b \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 36.6 & 1.88 \\ 0 & 0 \\ -156.71 & 1.08 \end{bmatrix} \begin{bmatrix} v_m \\ d \end{bmatrix}$$

$$\theta_b = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_w \\ \dot{x}_w \\ \theta_b \\ \dot{\theta}_b \end{bmatrix}$$

Reporting of Task 4.5

4.5.1 Here is the simulink scheme we implemented with the new A, B, C and D matrices:

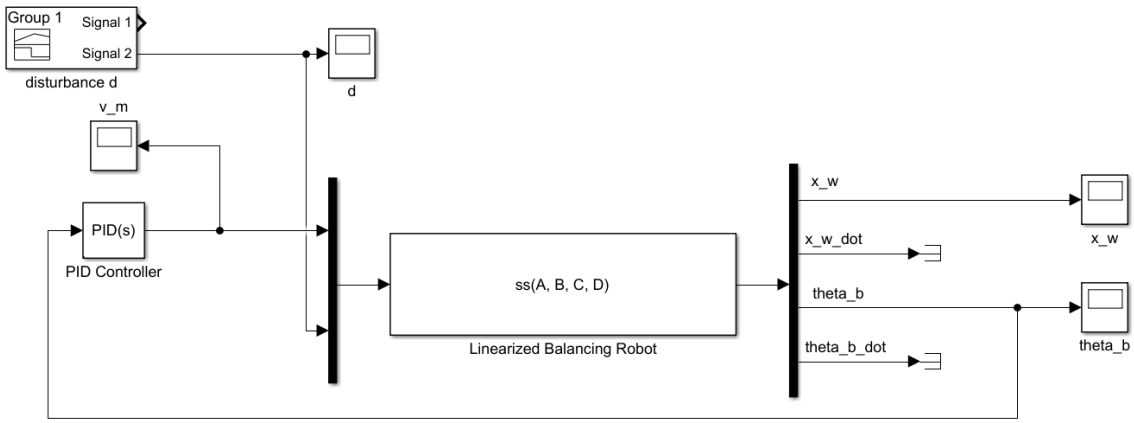


Figure 3: Simulink model

The unused Signal 1 is a delayed disturbance, to ensure that the system is stable even before applying the d force.

4.5.2 Here are the results of the simulation:

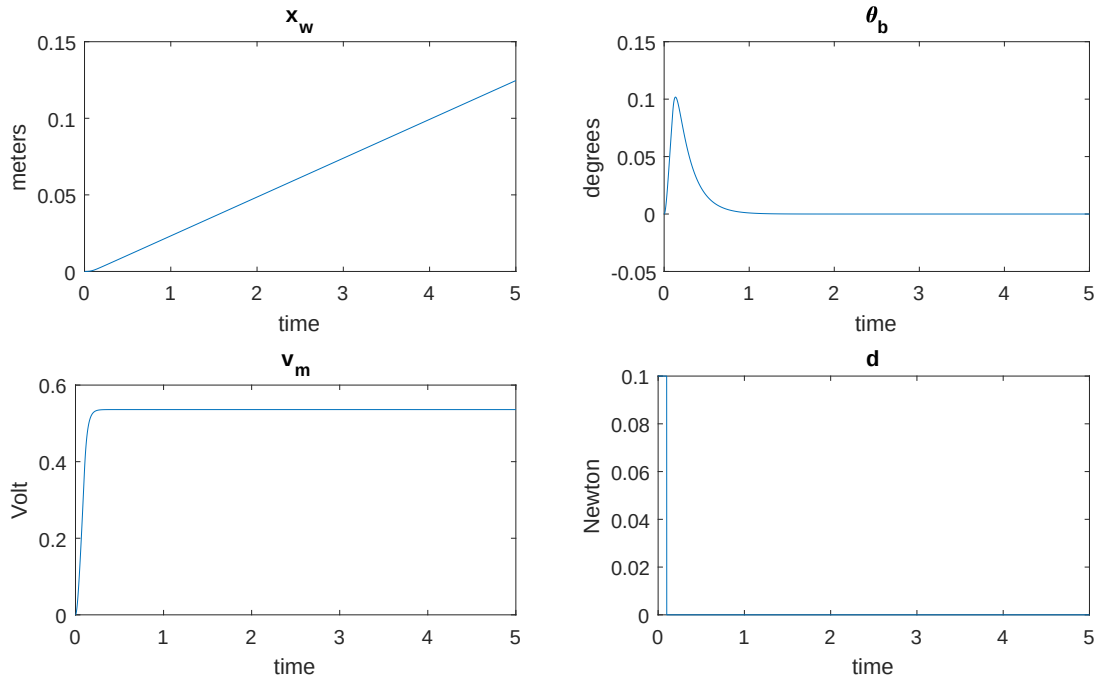


Figure 4: Simulation results

One can see that the robot moves forward to prevent itself from falling, but it keeps moving forward after (i.e. x_w is always increasing, it is not stable). The angle θ_b is stabilized to 0° despite the initial disturbance d. The command v_m is stabilized at 0.5 Volts. This is a realisitic value that could be provided by the battery.

Reporting of Task 4.6

4.6.1 Using Matlab tools, we compute $H_{CL}(s)$'s bandwidth and find $\omega_{max} = 30 rad.s^{-1}$.

4.6.2 Therefore, to comply with Nyquist-Shannon criterion, we must chose a sampling time h as follow:

$$\omega_{ech} > 2\omega_{max} \Leftrightarrow h < \frac{2\pi}{2\omega_{max}}$$

To increase the margin we will take:

$$h = \frac{2\pi}{5\omega_{max}} = 42ms$$

This value gives us more margin but should be high enough to be supported by the Arduino board.

4.6.3 We were not sure which bandwidth we should compute between $G(s)$ alone or $H_{CL}(s)$. Because some of us argued that we will sample only signal coming from outside $G(s)$ via the feedback loop, therefore we should comply to $G(s)$ bandwidth. Others said that the controller is part of the system, therefore it will produce frequencies that we should be able to sample correctly. Not knowing how to solve this issue, we decided to comply to the worst case in order to be sure that our sampling time will comply to both hypothesis. We had $\omega_{maxG(s)} = 13.6 rad.s^{-1}$ and $\omega_{maxH_{CL}(s)} = 30 rad.s^{-1}$. That's why we chose the closed-loop.

4.6.4 We can now discretize the controller using ZOH method. Here are the equivalences of $C(s)$'s proportionnal, integrative and derivative components between Laplace domain and Z domain:

$$\begin{cases} P(s) = K_p(\Theta_{b_{Ref}}(s) - \Theta_b(s)) \\ I(s) = \dfrac{K_i}{s}(\Theta_{b_{Ref}}(s) - \Theta_b(s)) \\ D(s) = -K_d s \Theta_b(s) \end{cases} \Rightarrow \begin{cases} P(z) = K_p(\Theta_{b_{Ref}}(z) - \Theta_b(z)) \\ I(z) = \dfrac{K_i h}{z-1}(\Theta_{b_{Ref}}(z) - \Theta_b(z)) \\ D(z) = -\dfrac{K_d}{h}\dfrac{z-1}{z}\Theta_b(z) \end{cases}$$

Here, $\Theta_{b_{Ref}}$ is always the null angle. Therefore, in discrete time domain we will implement this differential equation:

$$v_m(k) = -K_p\theta_b(k) - K_i \sum_{n=-\infty}^{k-1} \theta_b(n) - \frac{K_d}{h}(\theta_b(k) - \theta_b(k-1))$$

Reporting of Task 4.7

4.7.1 We tried our **continuous controller** over the linearized and non-linearized simulators. In the non-linear system, the robot falls at $d = 1.4N$. Then we tried our **discrete controller**. In the non-linear system, the robot falls at $d = 1.36N$.

4.7.2 Here are simulations of both controllers on each simulator (the disturbance is set just under the falling value):

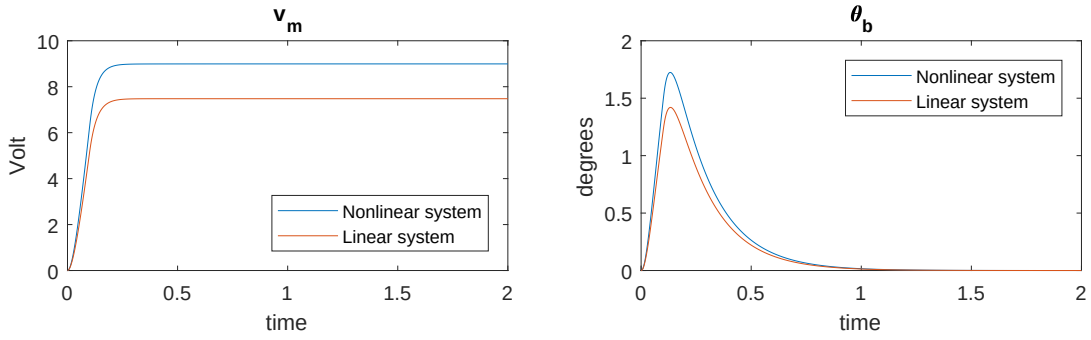


Figure 5: Simulation with continuous time controller $C(s)$ and $d = 1.395N$

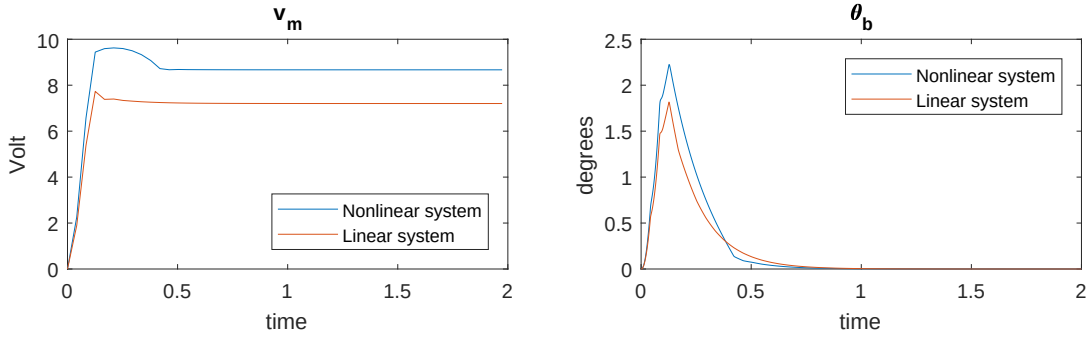


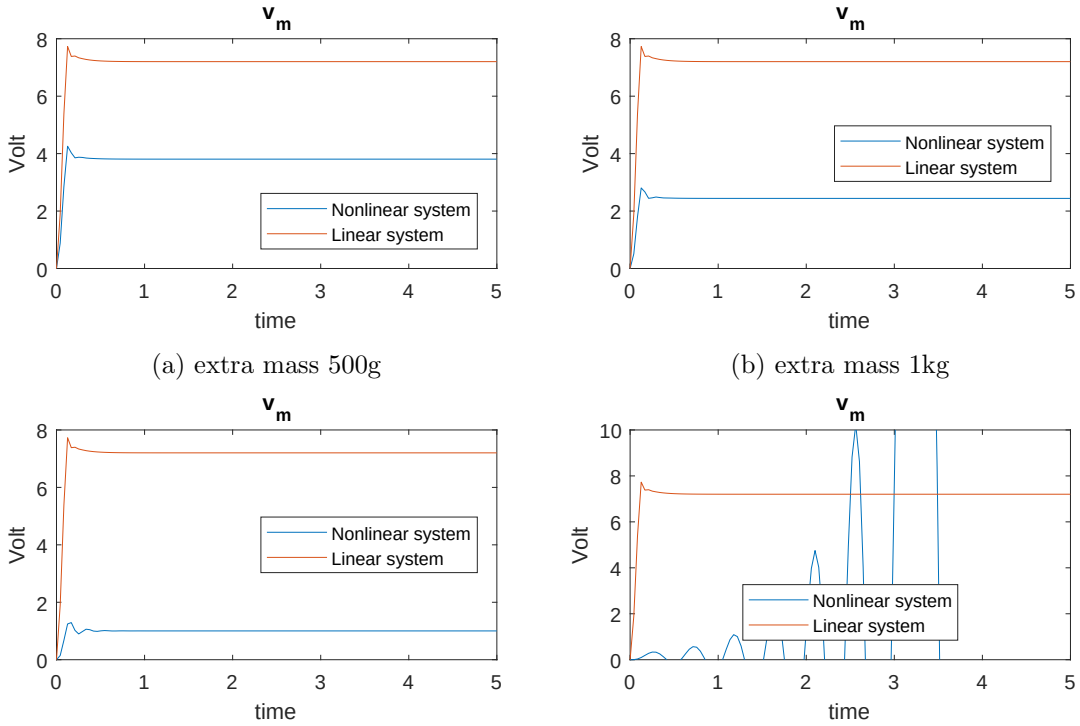
Figure 6: Simulation with discrete time controller $C(z)$ and $d = 1.35N$

4.7.3 In both cases, we have $\theta_b^{lin} < \theta_b$. Therefore the controller is forced to provide a higher command to stabilize the nonlinear system, because of new forces that were not in

the linear model. In reality, this may not work because we are exceeding the nominal value that batteries are able to provide (7.2V), i.e. the robot may fall with a smaller disturbance. Apart from this, one can see that with the discretized controller the responses are very similar to the full-continuous system. This is due to our sampling time that is quite short.

The worst sampling time we can take while staying in the Shannon-allowed range is $h = 100ms$. In that case, the system is already not stable anymore. The robot doesn't fall but ocillates around the null angle (marginal stability). $h = 70ms$ is approximately the higher sampling time we can choose whithout having unstable system, more than that makes the robot oscillating after several seconds.

Then it is interesting to see that the extra mass can have the effect to slow down the robot. As we can see in figure 7, the more mass we put and lower is the needed command voltage. Of, course there is a limit to this and for too heavy masses, the robot is not stable anymore (anyway in reality such masses can't be held by the robot).



(a) extra mass 500g

(b) extra mass 1kg

(c) extra mass 3kg

(d) extra mass 25g

Figure 7: Simulation with $C(z)$, $d = 1.35N$ and extra masses

Lab A conclusion

After establishing the state space representation of the system, we were able to design a continuous-time controller thanks to pole placement method. Then we simulated this controller (continuous and discretized) along with linear and nonlinear systems to test its robustness. During this lab we had lot of troubles to find the initial ss representation because we took a unconvenient x vector. Then the other part that took us lot of time was the PID design with pole placement. We had several ideas but none of them worked (we tried root-locus method, poles cancellation and even the full-state feedback method...). Finally after re-thinking our solution and asking some hints to the teachers, we found a PID that stabilizes the system using a realistic command.