Control design for the Pendubot system

Yohann Bonnavaud

Computer Control – Practice in automation

This century gave automation a huge position in almost all sectors and it is probably going to grow even more. Therefore, the improvement of control theory is really important for anything to improve the performances of any automated system. This goes along with the development of digital processing and computers, and by merging those improvements we can design some digital control systems to manage analog systems. Therefore, studying not only the theory but a real situation system is really important for further development of the same kind.

The system we will study right now is called Pendubot, a two links pendulum with one actuator. As we aim at controlling the position of each link in the space using only one input, we need to perform several steps, as finding the system dynamics, the identification of the system constants, the design of the control part and the results simulation and also the development of an LQR control system to perform complete movements. The first to movements to realize are two swing to reach a position straight to the top, and a middle position where the first link, pointing to the bottom, balance the second link, pointing to the top.

This project is made on the basis of the thesis *"Mechanical design and control of the Pendubot"*, Daniel Jerome Block, 1996. Although the global process is the same, we do everything in details and generate a code to simulate the movement of the Pendubot. The development of this project is based on Matlab and we used one model of the Pendubot which was available at the lab to perform some data gathering and trials.

Project report outline

I. The Pendubot system

1. Presentation

The Pendubot is a double pendulum system with only one torque input to move the first link around the two-dimensional space. A represented in figure I-1-1, the system variables are the angles measured by two encoders, that we will name q_1 for the angle of the first link regarding the horizontal and $q_2$for the angle of the second link regarding the first one.

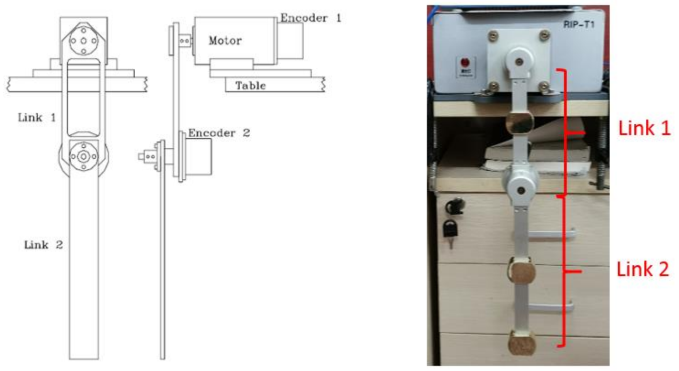


Figure I-1-1: a) Front and side view of the Pendubot in its initial state.
b) Real Pendubot system used at the lab.

To gather the data from the system, compute the force to input and give it back to the system, we are going to use a computer linked to digital/analog converter board, as we can see on the figure I-1-2. Matlab is used on the computer with a premade toolbox to perform control tuning easily (and more…).

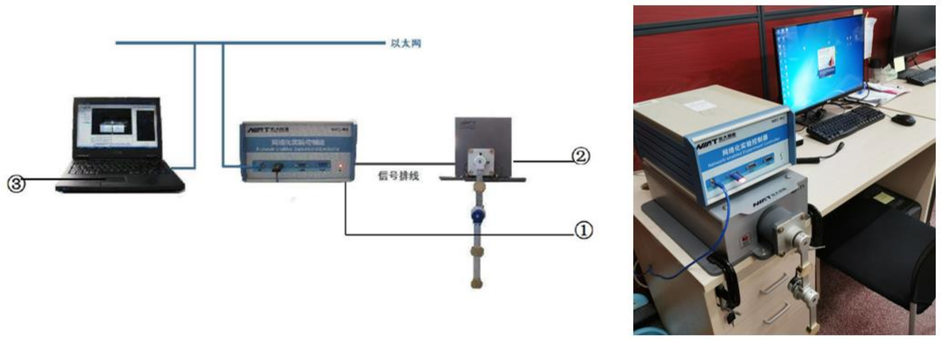


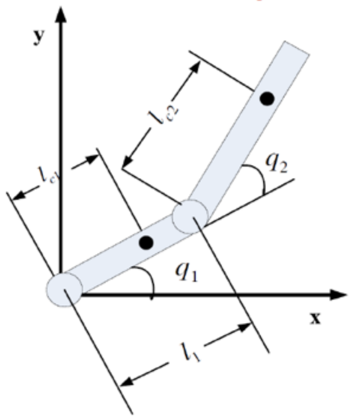
Figure I-1-2: Overall setup connections with real organization of the working area. The block labeled 2 is used to perform the analog/digital conversion and the amplification of the signal for the motor.

Regarding the system parameters (for instance the length of each length or their mass), we don't have any defined values. We are going to identify them in part I-3 running a simple test on the Pendubot.

2. System's dynamics

Now that we know more about our system, we are going to process it dynamics. From there, we will be able to use this result to simulate the Pendubot on a computer to perform some tests and have a better understanding of the overall system.

We are going to consider the figure I-2-1 below with the denomination of each variable or constant to find the system dynamics using Lagrange's equations.



$l_1$is the length of the first link.

l_{c1} and $l_{c2}$are the length from the joint to the center of mass of link 1 and link 2 respectively.

q_1 and q_2 are the angle of link 1 with the horizontal and link 2 with link1 respectively.

Figure I-2-1: Schematics of the double pendulum system giving the names of the Pendubot characteristics. Another non-presented variable of the system is the input torque τ.

First, we are going to look for the Lagrangian which is define by:

$$L = E_k - E_v$$

With $E_k$the total kinetic energy of the system and E_v the total potential energy of the system. We will proceed step by step to find each component of the potential and kinetic energy for each link.

For the potential energy, it can be described as:

$$E_v = mgh$$

With m the mass of the object, g the gravitational acceleration and h the height of the studied piece of the system, in other words its y position. For the first link, we have the following equation:

$$E_{v1} = m_1 g l_{c1} sin(q_1)$$

Since we can simulate the force of an object as applying on the object's center of mass, we use $l_{c1} sin(q_1)$ as the predefined height. The potential energy of the second link is a bit more complicated, since it is dependent of link 1. Using figure I-2-1, we can define the height h_2 and find the expression.

$$h_2 = f(q_1) + f(q_2)$$

$f(q_1)$ is the height of the point where link 1 and link 2 are attached. Its almost the same as h_1, but using l_1. $f(q_2)$ is the height link 2 center of mass if we consider the origin as being the extremity of link 2 which is attached to link 1. Therefore:

$$f(q_1) = l_1 sin(q_1)$$

$$f(q_2) = l_{c2} sin(q_1 + q_2)$$

We need to input the angle q_1 in $f(q_2)$ since q_2 is not an angle which is related to the horizontal (x-axis).

We have now the component to write the potential energy of link 2:

$$E_{v2} = m_2 g\big(l_1 sin(q_1) + l_{c2} sin(q_1 + q_2)\big)$$

Therefore, the system's potential energy is defined as:

$$E_v = E_{v1} + E_{v2}$$

$$E_v = m_1 g l_{c1} sin(q_1) + m_2 g\big(l_1 sin(q_1) + l_{c2} sin(q_1 + q_2)\big)$$

Now the kinetic energy is left. For the first link, the kinetic energy is generated by the rotation of the link around the point which is fixed to the table. This way there is no movement of this point and the link 1 can't move regarding translation. We only need to consider the rotational kinetic energy, which is defined by:

$$E_{k1} = \frac{1}{2} J_1\, {\omega_1}^2$$

With ω_1 the angular speed equals to $\dot{q_1}$, and J_1 the moment of inertia equals to $m_1 {l_{c1}}^2$ for a simple pendulum. Those modifications give us the following equation:

$$E_{k1} = \frac{1}{2}\, m_1\, {l_{c1}}^2\, \dot{q_1}^2$$

About the second link's kinetic energy, we need to consider this time not only the rotational kinetic energy, but also the translation one. Since link 1 is balancing, the point where link 2 is attached to is moving in the space. Let's start with the translation participation. First, we will express the cartesian coordinates (x_{c2}, y_{c2}) of the center of mass with our system's variable $(q_1,\ q_2)$. We already found the expression for y_{c2} when computing the potential energy, the process for x_{c2} is similar:

$$\begin{cases} x_{c2} = l_1 cos(q_1) + l_{c2} cos(q_1 + q_2) \\ y_{c2} = l_1 sin(q_1) + l_{c2} sin(q_1 + q_2) \end{cases}$$

Using definition, the kinetic energy for a translation of one point with mass m_2 at speed v_{c2} is given by:

$$E_{k2,T} = \frac{1}{2} m_2 {v_{c2}}^2$$

We also know the definition for the squared speed:

$${v_{c2}}^2 = \dot{x_{c2}}^2 + \dot{y_{c2}}^2$$

After several processing step:

$$\rightarrow \begin{cases} \dot{x_{c2}}^2 = {l_1}^2 \dot{q_1}^2 sin^2(q_1) + 2 l_1 l_{c2}\, \dot{q_1}\, (\dot{q_1} + \dot{q_2})\, sin(q_1)\, sin(q_1 + q_2) + {l_{c2}}^2 (\dot{q_1} + \dot{q_2})^2 sin^2(q_1 + q_2) \\ \dot{y_{c2}}^2 = {l_1}^2 \dot{q_1}^2 cos^2(q_1) + 2 l_1 l_{c2}\, \dot{q_1}\, (\dot{q_1} + \dot{q_2})\, cos(q_1)\, cos(q_1 + q_2) + {l_{c2}}^2 (\dot{q_1} + \dot{q_2})^2 cos^2(q_1 + q_2) \end{cases}$$

We can now sum those two equations and simplify it to:

$${v_{c2}}^2 = {l_1}^2 \dot{q_1}^2 + {l_{c2}}^2 (\dot{q_1} + \dot{q_2})^2 + 2 l_1 l_{c2}\, \dot{q_1}\, (\dot{q_1} + \dot{q_2})\, \big(sin(q_1)\, sin(q_1 + q_2) + cos(q_1)\, cos(q_1 + q_2)\big)$$

We can finish the simplification of the last term using trigonometric relation $\sin(a)\sin(b) + \cos(a)\cos(b) = \cos(a-b)$:

$$v_{c2}{}^2 = l_1{}^2\dot{q_1}{}^2 + l_{c2}{}^2(\dot{q_1} + \dot{q_2})^2 + 2l_1 l_{c2}\,\dot{q_1}\,(\dot{q_1} + \dot{q_2})\,cos(q_2)$$

$$E_{k2,T} = \frac{1}{2}m_2\big(l_1{}^2\dot{q_1}{}^2 + l_{c2}{}^2(\dot{q_1} + \dot{q_2})^2 + 2l_1 l_{c2}\,\dot{q_1}\,(\dot{q_1} + \dot{q_2})\,cos(q_2)\big)$$

About the rotational kinetic energy, we can use tensors to obtain the following matrixial expression:

$$E_{k2,R} = \frac{1}{2}(\dot{q_1} \quad \dot{q_2})\left\{\begin{bmatrix} I_1 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} I_2 & I_2 \\ I_2 & I_2 \end{bmatrix}\right\}\begin{pmatrix} \dot{q_1} \\ \dot{q_2} \end{pmatrix}$$

$$E_{k2,R} = \frac{1}{2}(\dot{q_1} \quad \dot{q_2})\begin{bmatrix} I_1 + I_2 & I_2 \\ I_2 & I_2 \end{bmatrix}\begin{pmatrix} \dot{q_1} \\ \dot{q_2} \end{pmatrix}$$

$$E_{k2,R} = \frac{1}{2}(\dot{q_1} \quad \dot{q_2})\begin{pmatrix} (I_1 + I_2)\dot{q_1} + I_2\dot{q_2} \\ I_2\dot{q_1} + I_2\dot{q_2} \end{pmatrix}$$

$$E_{k2,R} = \frac{1}{2}\Big((I_1 + I_2)\dot{q_1}{}^2 + 2I_2\dot{q_2}\dot{q_1} + I_2\dot{q_2}{}^2\Big)$$

We keep here I_1 and I_2 as parameters of the system.

This gives us the following equation for link 2 kinetic energy:

$$E_{k2} = \frac{1}{2}m_2\big(l_1{}^2\dot{q_1}{}^2 + l_{c2}{}^2(\dot{q_1} + \dot{q_2})^2 + 2l_1 l_{c2}\,\dot{q_1}\,(\dot{q_1} + \dot{q_2})\,cos(q_2)\big) + \frac{1}{2}\Big((I_1 + I_2)\dot{q_1}{}^2 + 2I_2\dot{q_2}\dot{q_1} + I_2\dot{q_2}{}^2\Big)$$

With those expression we can look for the Lagrangian expression:

$$L = \frac{1}{2}\,m_1\,l_{c1}{}^2\,\dot{q_1}{}^2 + \frac{1}{2}m_2\big(l_1{}^2\dot{q_1}{}^2 + l_{c2}{}^2(\dot{q_1} + \dot{q_2})^2 + 2l_1 l_{c2}\dot{q_1}\,(\dot{q_1} + \dot{q_2})\,cos(q_2)\big) + \frac{1}{2}\Big((I_1 + I_2)\dot{q_1}{}^2 + 2I_2\dot{q_2}\dot{q_1} + I_2\dot{q_2}{}^2\Big) - (m_1 l_{c1} + m_2 l_1)g sin(q_1) - m_2 g l_{c2} sin(q_1 + q_2)$$

We will now start the process to establish Lagrange equations regarding our variables $q_1$and q_2:

$$\begin{cases} \dfrac{d}{dt}\dfrac{\partial L}{\partial \dot{q_1}} - \dfrac{\partial L}{\partial q_1} = \tau_1 \\ \dfrac{d}{dt}\dfrac{\partial L}{\partial \dot{q_2}} - \dfrac{\partial L}{\partial q_2} = \tau_2 \end{cases}$$

In our situation, we only have one actuator regarding q_1 which means $\tau_1 = \tau$ and $\tau_2 = 0$.

We will process step by step. In addition, the expression on L has been developed at its maximum on paper to avoid mistakes. Let's start with the first equation:

$$\frac{\partial L}{\partial q_1} = -(m_1 l_{c1} + m_2 l_1)g cos(q_1) - m_2 g l_{c2} cos(q_1 + q_2)$$

$$\frac{\partial L}{\partial \dot{q_1}} = m_1\, {l_{c1}}^2\dot{q_1} + m_2{l_1}^2\dot{q_1} + m_2{l_{c2}}^2(\dot{q_1}+\dot{q_2}) + m_2l_1l_{c2}cos(q_2)(2\dot{q_1}+\dot{q_2}) + (I_1+I_2)\dot{q_1} + I_2\dot{q_2}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q_1}} = m_1\, {l_{c1}}^2\ddot{q_1} + m_2{l_1}^2\ddot{q_1} + m_2{l_{c2}}^2(\ddot{q_1}+\ddot{q_2}) + m_2l_1l_{c2}cos(q_2)(2\ddot{q_1}+\ddot{q_2}) \\ - m_2l_1l_{c2}sin(q_2)\left(2\dot{q_2}\dot{q_1}+{\dot{q_2}}^2\right) + (I_1+I_2)\ddot{q_1} + I_2\ddot{q_2}$$

We can now process the second equation:

$$\frac{\partial L}{\partial q_2} = -m_2l_1l_{c2}\dot{q_1}\,(\dot{q_1}+\dot{q_2})\,sin(q_2) - m_2gl_{c2}cos(q_1+q_2)$$

$$\frac{\partial L}{\partial \dot{q_2}} = m_2{l_{c2}}^2(\dot{q_1}+\dot{q_2}) + m_2l_1l_{c2}cos(q_2)\dot{q_1} + I_2(\dot{q_1}+\dot{q_2})$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q_2}} = (m_2{l_{c2}}^2+I_2)(\ddot{q_1}+\ddot{q_2}) + m_2l_1l_{c2}cos(q_2)\ddot{q_1} - m_2l_1l_{c2}sin(q_2)\dot{q_2}\dot{q_1}$$

We got the following Lagrangian equations:

$$\begin{cases} m_1\, {l_{c1}}^2\ddot{q_1} + m_2{l_1}^2\ddot{q_1} + m_2{l_{c2}}^2(\ddot{q_1}+\ddot{q_2}) + m_2l_1l_{c2}cos(q_2)(2\ddot{q_1}+\ddot{q_2}) - m_2l_1l_{c2}sin(q_2)\left(2\dot{q_2}\dot{q_1}+{\dot{q_2}}^2\right) \\ \quad +(I_1+I_2)\ddot{q_1} + I_2\ddot{q_2} + (m_1l_{c1}+m_2l_1)gcos(q_1) + m_2gl_{c2}cos(q_1+q_2) = \tau \\ (m_2{l_{c2}}^2+I_2)(\ddot{q_1}+\ddot{q_2}) + m_2l_1l_{c2}cos(q_2)\ddot{q_1} + m_2l_1l_{c2}{\dot{q_1}}^2\ sin(q_2) + m_2gl_{c2}cos(q_1+q_2) = 0 \end{cases}$$

However, this way of writing things is not really intuitive. What we want is to organize everything so that we can write the previous as:

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau$$

After developing and reorganizing everything, we can find the parameters of each matrices (inertia, centripetal/Coriolis, gravity) by identification. Therefore, we can write:

$$D = \begin{pmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{pmatrix} \quad with \quad \begin{aligned} d_{11} &= m_1\, {l_{c1}}^2 + m_2\left({l_1}^2 + {l_{c2}}^2 + 2l_1l_{c2}cos(q_2)\right) + I_1 + I_2 \\ d_{12} = d_{21} &= m_2\left({l_{c2}}^2 + l_1l_{c2}cos(q_2)\right) + I_2 \\ d_{22} &= m_2{l_{c2}}^2 + I_2 \end{aligned}$$

$$C = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix} \quad with \quad \begin{aligned} c_{11} &= k\dot{q_2} \\ c_{12} &= k\dot{q_2} + k\dot{q_1} \\ c_{21} &= -k\dot{q_1} \\ c_{22} &= 0 \end{aligned}, \quad k = -m_2l_1l_{c2}sin(q_2)$$

$$g = \begin{pmatrix} g_1 \\ g_2 \end{pmatrix} \quad with \quad \begin{aligned} g_1 &= (m_1l_{c1}+m_2l_1)gcos(q_1) + m_2gl_{c2}cos(q_1+q_2) \\ g_2 &= m_2gl_{c2}cos(q_1+q_2) \end{aligned}$$

We said earlier that we are having a situation where we can experiment with the plant, but we don't know its characteristics. That means we are going to led couple of experiments to compute the values of the different length and masses of the Pendubot. However, it is rather difficult for us to evaluate the values of all those unknown constants. To simplify our problem, we are going to look at the groups of constants that are the same to pass from seven to five unknown parameters.

Those new parameters are called θ_i and are defined as:

$$\boldsymbol{\theta_1} = m_1\, l_{c1}{}^2 + m_2 l_1{}^2 + I_1 \qquad \boldsymbol{\theta_3} = m_2 l_1 l_{c2} \qquad \boldsymbol{\theta_5} = m_2 l_{c2}$$
$$\boldsymbol{\theta_2} = m_2 l_{c2}{}^2 + I_2 \qquad \boldsymbol{\theta_4} = m_1 l_{c1} + m_2 l_1$$

We can now rewrite the previous matrices as:

$$D = \begin{bmatrix} \theta_1 + \theta_2 + 2\theta_3 cos(q_2) & \theta_2 + \theta_3 cos(q_2) \\ \theta_2 + \theta_3 cos(q_2) & \theta_2 \end{bmatrix}$$
$$C = \begin{bmatrix} -\theta_3 sin(q_2)\dot{q_2} & -\theta_3 sin(q_2)\dot{q_2} - \theta_3 sin(q_2)\dot{q_1} \\ \theta_3 sin(q_2)\,\dot{q_1} & 0 \end{bmatrix}$$
$$g = \begin{bmatrix} \theta_4 g cos(q_1) + \theta_5 g cos(q_1 + q_2) \\ \theta_5 g cos(q_1 + q_2) \end{bmatrix}$$

Finally, we can use those new matrices to reverse the system equations and obtain the dynamic of the system:

$$\begin{bmatrix} \ddot{q_1} \\ \ddot{q_2} \end{bmatrix} = D^{-1}(q)\tau - D^{-1}(q)C(\dot{q}, q)\begin{bmatrix} \dot{q_1} \\ \dot{q_2} \end{bmatrix} - D^{-1}(q)g(q)$$

We can now define our state variable system introducing a new variable x_i. We will have 4 state variables with the following definition:

$$\begin{matrix} x_1 = q_1 \\ x_2 = \dot{q_1} \\ x_3 = q_2 \\ x_4 = \dot{q_2} \end{matrix} \quad \rightarrow \quad \begin{cases} \dot{x_1} = x_2 \\ \dot{x_2} = \ddot{q_1} \\ \dot{x_3} = x_4 \\ \dot{x_4} = \ddot{q_2} \end{cases}$$

3. System's parameters

From now on, we have a good idea of our pendubot system. However, we still don't know it entirely since the parameters θ_i are not identified yet. In the thesis mentionned in introduction, three methods are used. For this project, we are only going to use one of them: the Energy Equation Method. The idea is to use the system as an open loop and study its response to a particular input. Using the energy theorem $\int_{t_1}^{t_2} T^T\, \dot{q}\; dt = L(t_2) - L(t_1)$ with T the torque vector of the applied forces and $L(t_i)$ the system lagrangian at the time t_i, we can define a system of equations we need to solve to find the different parameters. Here we are going to explain the process step by step. The file "calc_param.m" is a matlab function which is computing the folllowing mathes to return the different values of theta.

According to the energy theorem we wrote earlier, we need to define $L = E_K - E_V$.

The kinetic energy of the system is defined as:

$$E_K = \frac{1}{2}\dot{q}^T D(q)\dot{q}$$
$$= \frac{1}{2}\begin{pmatrix} \dot{q_1} & \dot{q_2} \end{pmatrix}\begin{pmatrix} \theta_1\dot{q_1} + \theta_2\dot{q_1} + 2\theta_3 cos(q_2)\dot{q_1} + \theta_2\dot{q_2} + \theta_3 cos(q_2)\dot{q_2} \\ \theta_2\dot{q_1} + \theta_3 cos(q_2)\dot{q_1} + \theta_2\dot{q_2} \end{pmatrix}$$
$$= \frac{1}{2}\left(\theta_1\dot{q_1}^2 + \theta_2\dot{q_1}^2 + 2\theta_3 cos(q_2)\dot{q_1}^2 + 2\theta_2\dot{q_2}\dot{q_1} + 2\theta_3 cos(q_2)\dot{q_2}\dot{q_1} + \theta_2\dot{q_2}^2\right)$$

$$= \left(\frac{1}{2}\dot{q_1}^2\right)\theta_1 + \left(\frac{1}{2}\dot{q_1}^2 + \dot{q_2}\dot{q_1} + \frac{1}{2}\dot{q_2}^2\right)\theta_2 + \left(cos(q_2)\dot{q_2}\dot{q_1} + cos(q_2)\dot{q_1}^2\right)\theta_3$$

For the potential energy of the pendubot, we already wrote the eqution when we were computing the lagrangian. It can be written using theta as:

$$E_V = \theta_4 gsin(q_1) + \theta_5 gsin(q_1 + q_2)$$

As we organize both equations to have a sum of the different theta multiply by some coefficient, we can rewrite E_K and E_V as the following sums:

$$E_K = \sum_{i=1}^{5} \frac{\partial E_K}{\partial \theta_i}\theta_i \quad and \quad E_V = \sum_{i=1}^{5} \frac{\partial E_V}{\partial \theta_i}\theta_i$$

For a easier reading, we will write $\frac{\partial E_K}{\partial \theta_i}$ as DK and $\frac{\partial E_V}{\partial \theta_i}$ as DV. This gives us the following:

$$\begin{aligned} DK_1 &= \frac{1}{2}\dot{q_1}^2 \\ DK_2 &= \frac{1}{2}\dot{q_1}^2 + \dot{q_2}\dot{q_1} + \frac{1}{2}\dot{q_2}^2 \\ DK_3 &= cos(q_2)\dot{q_2}\dot{q_1} + cos(q_2)\dot{q_1}^2 \\ DK_4 &= 0 \\ DK_5 &= 0 \end{aligned}$$

$$\begin{aligned} DV_1 &= 0 \\ DV_2 &= 0 \\ DV_3 &= 0 \\ DV_4 &= gsin(q_1) \\ DV_5 &= gsin(q_1 + q_2) \end{aligned}$$

We can do the same separation for L to obtain 5 equations regarding theta:

$$\begin{aligned} DL_1 &= \frac{1}{2}\dot{q_1}^2 \\ DL_2 &= \frac{1}{2}\dot{q_1}^2 + \dot{q_2}\dot{q_1} + \frac{1}{2}\dot{q_2}^2 \\ DL_3 &= cos(q_2)\dot{q_2}\dot{q_1} + cos(q_2)\dot{q_1}^2 \\ DL_4 &= -gsin(q_1) \\ DL_5 &= -gsin(q_1 + q_2) \end{aligned}$$

Back to our energy equation, T is the vector of torque applied to the system. For this project, we are not going to consider the friction forces around the joints. Therefore, $T = \binom{\tau}{0}$.

As we are going to get several measures of each state variables over the time, we are going to have a lot of different equations for eache DL_i. That means that, since the measure is made at a random time t, we need the integration interval to stop at this time. Since we need the have the whole history of the experiment (it stops at a moment so we don't have an inifite number of data to treat), we can combine all the equations in a $Ax = B$ like matrix system. If we consider the number of sample as k, it gives us k equations and we can define it as the following matrix system:

$$\left(\int_{t_0}^{t_k} T^T \dot{q} dt\right)_K = DL_k^T \theta$$

To gather data to solve this system, we used a step-input function with a signal gain of 0.3, 0.45 and 0.5. The idea is to run the code for each of these sets and the get an average value of each value of θ.

All the data is exported in a matlab file thanks to an additionnal algorithm that manage the pendubot. All we had to do is to set up the signal gain and chose a step input to gather all the data in couple of sets.

Once all the state variable and torque values were register, we establish all the different equations for each theta regarding the above expression and we solve the integral part using trapezoidal approximation method.

Once all the equation are set up, we just have to perform a non-negative least squares solution for the system of the regrouped k equations. To be able to raise accuracy of the result, we don't take a one sample time interval but ten sample time interval, which gives us a wider range and a lower sensibility to the sensors and methods' errors. After running the code, we find the following values for θ:

$$\theta_1 = 0.0102$$

$$\theta_2 = 0.0043$$

$$\theta_3 = 0.0041$$

$$\theta_4 = 0.0795$$

$$\theta_5 = 0.0293$$

NB: The value of gravity used for those results is $9.81\ m.s^{-2}$ and the different sets of data we used to obtain those results are available in the files called "expData_X.m" where X is the number of the set.

II. Performing a Swing

1. What movement does it describe?

As mentioned in the introduction, we want first to perform a swing, which means that by moving the first arm around we try to set a position to have the Pendubot arm in a predefined position. Basically? We are moving the system from is stable hanging position to an unstable equilibrium position. We are going to consider a swing up (swing to the upper position $(q_1 = \frac{\pi}{2}, q_2 = 0)$) and a swing mid (swing to middle

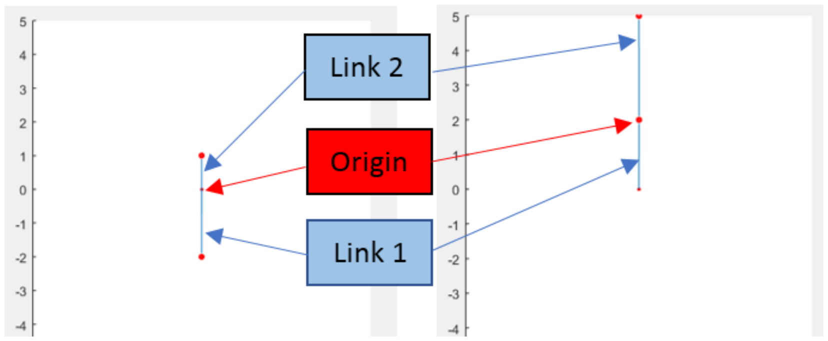


Figure II-1-1: Representation of both swing mid (on the left) and swing up (on the right) for the Pendubot.

position ($q_1 = -\frac{\pi}{2}, q_2 = \pi$)). As a reminder, the initial stable position of the Pendubot is ($q_1 = -\frac{\pi}{2}, q_2 = 0$). Both are represented in the following schemes:

To perform these swings, we need to design a controller which is going to be able to tune the torque command regarding the actual value of our state variable. However, the Pendubot is assimilable as a double-pendulum which is not linear. In fact, the linkage induces nonlinear effect of link 1 on link 2 and vice versa.

2. Control design for swing movement

To counter those nonlinear effects, we are going to use Partial Feedback Linearization. The idea behind this is to separate the system in two independent parts, one dealing with q_1 and the other one with q_2. Therefore, we can use the feedback of q_1 and q_2 to compute the first control signal without paying attention to $\ddot{q_2}$, and use this input and the feedback of q_1 and q_2 to compute the final control signal for the control of the swing, this time caring about $\ddot{q_2}$.

This idea is represented by the following block scheme, with an inner and outer loop feedback control:

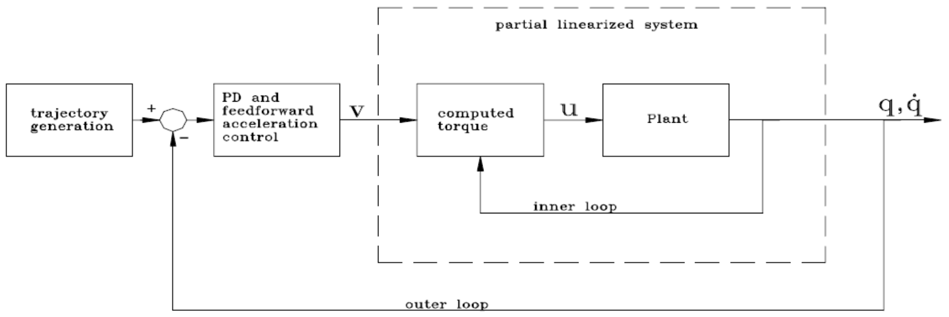


Figure II-2-1: Block diagram of the partial feedback linearization control.

According to this block diagram, we need to process in two steps. The first one is the computation of v, and the second one is the computation of u the real control input of the plant. It's called u in reference to the state relation $\dot{X} = Ax + Bu$, but we will continue to call it τ, the torque input (it's the same, it is just a better idea to stick with what we started with).

According to the system equation $D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(\dot{q}) = \tau$, we can write the following equations:

$$\begin{cases} d_{11}\ddot{q_1} + d_{12}\ddot{q_2} + c_{11}\dot{q_1} + c_{12}\dot{q_2} + g_1 = \tau \\ d_{21}\ddot{q_1} + d_{22}\ddot{q_2} + c_{11}\dot{q_1} + g_2 = 0 \end{cases}$$

Since the system is underactuated (we can command link 1 only), we are going to process one after the other since it is not possible to do both at the same time (as explained above). This way, we can use the second equation to find a value for $\ddot{q_2}$:

$$\ddot{q_2} = \frac{-d_{21}\ddot{q_1} - c_{11}\dot{q_1} - g_2}{d_{22}}$$

Then we replace in the first one to obtain an equation where we can have an expression for $\ddot{q_1}$:

$$\overline{d_{11}}\ddot{q_1} + \overline{c_{11}}\dot{q_1} + \overline{c_{12}}\dot{q_2} + \overline{g_1} = \tau$$

With:

$$\overline{d_{11}} = d_{11} - \frac{d_{12}d_{21}}{d_{22}}$$
$$\overline{c_{11}} = c_{11} - \frac{d_{12}c_{21}}{d_{22}}$$
$$\overline{c_{12}} = c_{12}$$
$$\overline{g_1} = g_1 - \frac{d_{12}g_2}{d_{22}}$$

We have now the expression to linearize $\ddot{q_1}$. We also are going to use this expression to compute τ later on, the inner loop control signal:

$$\overline{d_{11}} + \overline{c_{11}}\dot{q_1} + \overline{c_{12}}\dot{q_2} + \overline{g_1} = \tau$$

Which results in the following system:

$$\begin{cases} \ddot{q_1} = v \\ d_{22}\ddot{q_2} + c_{11}\dot{q_1} + g_2 = -d_{21}v \end{cases}$$

Now that we linearized $\ddot{q_1}$, all is left to do is to design this outer loop control to compute the value of v. According to the mentioned thesis, a simple PD controller is enough in this situation. Using a PID controller can work too, but it amplified the signal noise. This controller's job is to design a track for the motor to reach a position with link 1, which means that we just need to give it the value of the angle of link 1 we are aiming at (called ${q_1}^D$) for it to compute the signal to send. The equation of the outer loop controller is:

$$v = \ddot{q_1}^D + K_d\left(\dot{q_1}^D - \dot{q_1}\right) + K_p({q_1}^D - q_1)$$

Which can be simplified as:

$$v = -K_d\dot{q_1} + K_p({q_1}^D - q_1)$$

As we just mentioned, we now need to define a trajectory for the motor to follow. For the swing up, giving the angle q_1 to be its final position is enough. However, it is a little bit more complicated for the swing mid. It must first follow a sinusoidal function to give inertia to the second arm, and after some time when we consider it's enough, we just change the input so that it has to reach its final position. Indeed, since the final position of link1 is also it initial one for the swing mid, it makes no sense to proceed the same way as the swing up.

Table of the angle q_1 to reach for both swing movement	
Swing up	Swing mid

$q_1 = \frac{\pi}{2}$	$q_1 = \begin{cases} 1.4\,sin(-5t) - \frac{\pi}{2}, & t < \frac{2\pi}{5} \\ -\frac{\pi}{2}, & t \geq \frac{2\pi}{5} \end{cases}$

3. Real situation trials

After talking about the theory of everything, we led some experiment with a real Pendubot system. Although there was not much to do, the idea was to change the value of K_p and K_d to perform a swing Up and a swing MID. Actually, the controller we had to tuned was not only the PD controller we just saw that can balance the arm until one position, because it was able to balance the second arm around the final position. We are going to see more about this controller implementations in the following chapter.

It is important to mention that the Pendubot can become really dangerous. It is better to know what we are doing while tuning the controller, since it can accumulate a lot of inertia if the proportional gain is too high. Better have a good idea with the simulation first and still have an emergency shut down, just in case.

After several trials, we register the following curves for both swing up and swing mid. The controller coefficients are also display, but it is only valuable for our past experiment, it makes no sense to use them on another one.

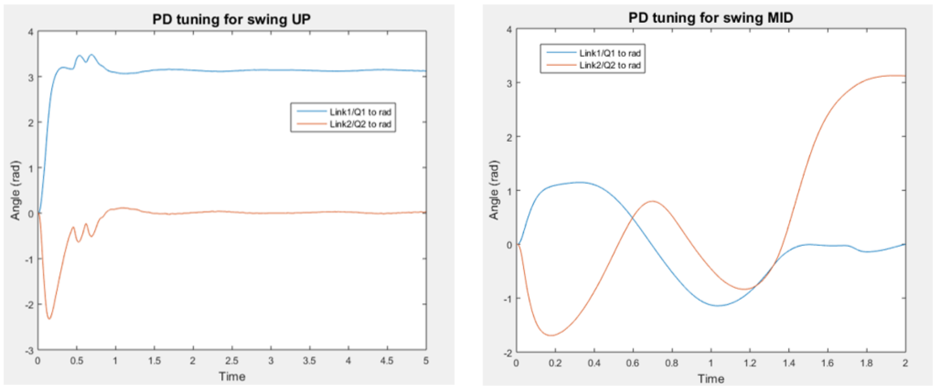


Figure II-3-1: Angular position gathered with real Pendubot system with tuned PD controller.

The coefficients of the controller were the following:

Swing UP	Swing MID
$K_p = -190, \qquad K_d = -20$	$K_p = 180, \qquad K_d = -20$

III. Balancing control

Now that we studied everything needed to bring the system to an unstable balance position, we need the second link to stick to it. In fact, in the actual situation, the unactuated link will just fall down after some time since link 1 is not moving after it reached its final position. Therefore, we need to implement another controller which is going to balance the second link after the first one did his job.

To do this, we are going to use a full state feedback controller using LQR method.

1. Total linearization of the system

The first step is to linearize totally the system. Compare to the previous situation, this is possible since we are studying the system around one equilibrium position, which means we can approximate the plant around this point using Taylor series:

$$f_a(x,u) = f_a(x_r,u_r) + \left.\frac{\partial f_a}{\partial x}\right|_{x_r,u_r}(x-x_r) + \left.\frac{\partial f_a}{\partial u}\right|_{x_r,u_r}(u-u_r)$$

In this equation, x_r and u_r are the equilibrium values of the states and control respectively. At the end, this situation is the same as getting the state equation form $\dot{X} = Ax + Bu$ at the equilibrium point. Therefore, in our situation, the function called $f_a(x,u)$ in the Taylor's decomposition is nothing else than our state equation. Since we want to study the system at its equilibrium point, we can already say that the all of the state variables will be equal to 0 at the equilibrium position, so $f_a(x_r,u_r) = 0$.

What about the equilibrium values x_r and u_r ? For both equilibrium positions after swing up and swing mid, we want x_2 and x_4 to be 0 since they describe the variations of the angles. Also, if we sum x_1 and x_3 for the final position, we realize that it's equal to $\frac{\pi}{2}$. This gives us the first equilibrium condition:

$$x_{r1} + x_{r2} = \frac{\pi}{2}$$

The second condition, on u_r, is given by the system dynamics ($D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau$). Using what we just saw, $\ddot{q}$ and $\dot{q}$ must be 0. This leave us with $g(q) = \tau$:

$$\begin{pmatrix}\tau_r \\ 0\end{pmatrix} = \begin{pmatrix}\theta_4 g cos(x_{1r}) + \theta_5 g cos(x_{1r}+x_{3r}) \\ \theta_5 g cos(x_{1r}+x_{3r})\end{pmatrix}$$

$$\tau_r = \theta_4 g cos(x_{1r}) + \theta_5 g cos(x_{1r}+x_{3r})$$

However, we just said that $x_{r1} + x_{r2} = \frac{\pi}{2}$, and we now $\cos\left(\frac{\pi}{2}\right) = 0$:

$$\tau_r = \theta_4 g cos(x_{1r})$$

We can gather these equilibrium conditions in the following system:

$$\begin{cases} x_{1r} + x_{3r} = {}^{\pi}/_{2} \\ x_{2r} = x_{4r} = 0 \\ \tau_r = \theta_4 g cos(x_{1r}) \end{cases}$$

Now, what's left is to compute the derivative matrices A $\left(\left.\frac{\partial f_a}{\partial x}\right|_{x_r,u_r}\right)$ and B $\left(\left.\frac{\partial f_a}{\partial u}\right|_{x_r,u_r}\right)$.

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \frac{\partial f_1}{\partial x_4} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} & \frac{\partial f_2}{\partial x_4} \\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_3} & \frac{\partial f_3}{\partial x_4} \\ \frac{\partial f_4}{\partial x_1} & \frac{\partial f_4}{\partial x_2} & \frac{\partial f_4}{\partial x_3} & \frac{\partial f_4}{\partial x_4} \end{bmatrix} \qquad B = \begin{bmatrix} \frac{\partial f_1}{\partial u} \\ \frac{\partial f_2}{\partial u} \\ \frac{\partial f_3}{\partial u} \\ \frac{\partial f_4}{\partial u} \end{bmatrix}$$

As a reminder, $f_i = \dot{x_i}$. The detailed expressions are given at the end of part I-2. Using this, we can simplify as the following:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{\partial f_2}{\partial x_1} & 0 & \frac{\partial f_2}{\partial x_3} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{\partial f_4}{\partial x_1} & 0 & \frac{\partial f_4}{\partial x_3} & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ \frac{\partial f_2}{\partial u} \\ 0 \\ \frac{\partial f_4}{\partial u} \end{bmatrix}$$

And it let us with six parameters to evaluate. For now, we are going to express the common formula, and we'll replace our x_1, x_3 and u with the right angles regarding if we perform swing up or swing mid at the end. Since there is a lot of process behind the final equations, only the main steps will be displayed. Anyways there is nothing difficult to understand at this point, it is just about going straight forward with the math. We use the expression of $\ddot{q_1}$ and $\ddot{q_1}$ developed earlier in part I-2.

For now, let's go through the A matrix parameters:

$$\ddot{q_1} = -\frac{d_{22}c_{11}\,\dot{q_1}}{d_{11}d_{22} - d_{12}d_{21}} - \frac{d_{22}c_{12}\,\dot{q_2}}{d_{11}d_{22} - d_{12}d_{21}} + \frac{d_{12}c_{21}\,\dot{q_1}}{d_{11}d_{22} - d_{12}d_{21}} + \frac{d_{12}c_{22}\,\dot{q_2}}{d_{11}d_{22} - d_{12}d_{21}} - \frac{d_{22}\,g_1}{d_{11}d_{22} - d_{12}d_{21}} + \frac{d_{12}\ \,g_2}{d_{11}d_{22} - d_{12}d_{21}}$$

$$\ddot{q_2} = \frac{d_{21}c_{11}\,\dot{q_1}}{d_{11}d_{22} - d_{12}d_{21}} + \frac{d_{21}c_{12}\,\dot{q_2}}{d_{11}d_{22} - d_{12}d_{21}} - \frac{d_{11}c_{21}\,\dot{q_1}}{d_{11}d_{22} - d_{12}d_{21}} - \frac{d_{11}c_{22}\,\dot{q_2}}{d_{11}d_{22} - d_{12}d_{21}} + \frac{d_{21}\,g_1}{d_{11}d_{22} - d_{12}d_{21}} - \frac{d_{11}\ \,g_2}{d_{11}d_{22} - d_{12}d_{21}}$$

One thing we can do right now is ignoring all the terms where $\dot{q}$ is involved. In fact, for our balancing conditions, we need them to be 0, so it means it is useless to compute the derivative of these terms since they will not contribute to the final result.

After the computation of the derivative for the last two terms left, we have the following expressions:

$$\boldsymbol{\frac{\partial f_2}{\partial x_1}} = \frac{\theta_2\theta_4 g sin(q_1) + \theta_2\theta_5 g sin(q_1+q_2) - \left(\theta_2 + \theta_3 cos(q_2)\right)\theta_5 g sin(q_1+q_2)}{\theta_2(\theta_1 + \theta_2 + 2\theta_3\cos(q_2)) - \left(\theta_2 + \theta_3 cos(q_2)\right)^2}$$

$$\boldsymbol{\frac{\partial f_4}{\partial x_1}} = \frac{-\left(\theta_2 + \theta_3 cos(q_2)\right)\left(\theta_4 g sin(q_1) + \theta_5 g sin(q_1+q_2)\right) + (\theta_1 + \theta_2 + 2\theta_3\cos(q_2))\theta_5 g sin(q_1+q_2)}{\theta_2(\theta_1 + \theta_2 + 2\theta_3\cos(q_2)) - \left(\theta_2 + \theta_3 cos(q_2)\right)^2}$$

$$\frac{\partial \boldsymbol{f_2}}{\partial \boldsymbol{x_3}} = -\frac{\{2\theta_2\theta_3 sin(q_2) - 2\theta_3 sin(q_2)(\theta_2 + \theta_3 cos(q_2))\}\{\theta_2\theta_4 g cos(q_1) + \theta_2\theta_5 g sin(q_1+q_2)\}}{\theta_2(\theta_1+\theta_2+2\theta_3\cos(q_2)) - (\theta_2+\theta_3 cos(q_2))^2}$$
$$-\theta_3\theta_5 g(sin(q_2)cos(q_1+q_2) + cos(q_2)sin(q_1+q_2))$$
$$+\frac{\{2\theta_2\theta_3 sin(q_2) - 2\theta_3 sin(q_2)(\theta_2 + \theta_3 cos(q_2))\}\{\theta_2\theta_5 g cos(q_1+q_2) + \theta_3\theta_5 g cos(q_2)cos(q_1+q_2)\}}{\theta_2(\theta_1+\theta_2+2\theta_3\cos(q_2)) - (\theta_2+\theta_3 cos(q_2))^2}$$

$$\frac{\partial \boldsymbol{f_4}}{\partial \boldsymbol{x_3}} = -\theta_4 g cos(q_1)\theta_3 sin(q_2)$$
$$+\frac{\{2\theta_2\theta_3 sin(q_2) - 2\theta_3 sin(q_2)(\theta_2 + \theta_3 cos(q_2))\}\{(\theta_2+\theta_3 cos(q_2))(\theta_4 g cos(q_1) + \theta_5 g cos(q_1+q_2))\}}{\theta_2(\theta_1+\theta_2+2\theta_3\cos(q_2)) - (\theta_2+\theta_3 cos(q_2))^2}$$
$$+\theta_1\theta_5 g sin(q_1+q_2) + \theta_3\theta_5 g(sin(q_2)cos(q_1+q_2) + cos(q_2)sin(q_1+q_2))$$
$$-\frac{\{2\theta_2\theta_3 sin(q_2) - 2\theta_3 sin(q_2)(\theta_2 + \theta_3 cos(q_2))\}\{(\theta_1+\theta_2+2\theta_3\cos(q_2))\theta_5 g os(q_1+q_2)\}}{\theta_2(\theta_1+\theta_2+2\theta_3\cos(q_2)) - (\theta_2+\theta_3 cos(q_2))^2}$$

For the B matrix, the equations of each part are way simpler:

$$\ddot{q_1} = \frac{d_{22}}{d_{11}d_{22} - d_{12}d_{21}}u$$
$$\ddot{q_2} = -\frac{d_{21}}{d_{11}d_{22} - d_{12}d_{21}}u$$

$$\frac{\partial \boldsymbol{f_2}}{\partial \boldsymbol{u}} = \frac{\theta_2}{\theta_2(\theta_1+\theta_2+2\theta_3\cos(q_2)) - (\theta_2+\theta_3 cos(q_2))^2}$$
$$\frac{\partial \boldsymbol{f_4}}{\partial \boldsymbol{u}} = -\frac{\theta_2+\theta_3 cos(q_2)}{\theta_2(\theta_1+\theta_2+2\theta_3\cos(q_2)) - (\theta_2+\theta_3 cos(q_2))^2}$$

We can now use those results to find the matrix for the swing mid balancing position and swing up balancing position. Actually, in the Matlab algorithm we directly use the expressions we have at this point, and selecting either we want one swing or the other we just have to input the value to reach for each angle.

<u>Swing up:</u> $\boldsymbol{q_1 = \frac{\pi}{2},\ q_2 = 0}$

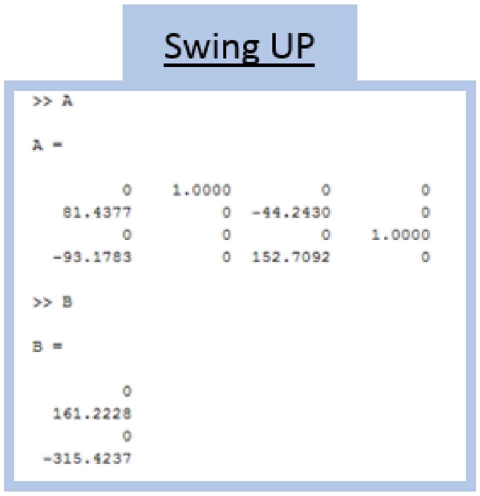
$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{\theta_2\theta_4 g - \theta_5\theta_3 g}{\theta_1\theta_2 - \theta_3^2} & 0 & \frac{-\theta_5\theta_3 g}{\theta_1\theta_2 - \theta_3^2} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{\theta_1\theta_5 g + \theta_5\theta_3 g - \theta_2\theta_4 g - \theta_3\theta_4 g}{\theta_1\theta_2 - \theta_3^2} & 0 & \frac{\theta_1\theta_5 g + \theta_5\theta_3 g}{\theta_1\theta_2 - \theta_3^2} & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ \frac{\theta_2}{\theta_1\theta_2 - \theta_3^2} \\ 0 \\ -\frac{\theta_2 + \theta_3}{\theta_1\theta_2 - \theta_3^2} \end{bmatrix}$$

Swing mid: $\boldsymbol{q_1} = -\frac{\pi}{2},\ \boldsymbol{q_2} = \boldsymbol{\pi}$

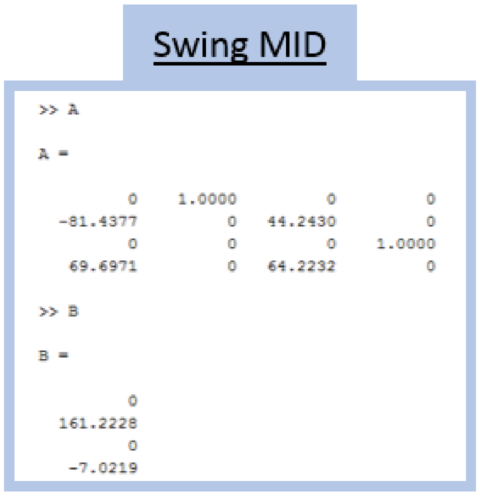
$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \dfrac{\theta_5\theta_3 g - \theta_2\theta_4 g}{\theta_1\theta_2 - {\theta_3}^2} & 0 & -\theta_5\theta_3 g & 0 \\ 0 & 0 & 0 & 1 \\ \dfrac{\theta_1\theta_5 g + \theta_2\theta_4 g - \theta_5\theta_3 g - \theta_3\theta_4 g}{\theta_1\theta_2 - {\theta_3}^2} & 0 & \theta_1\theta_5 g - \theta_5\theta_3 g & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ \dfrac{\theta_2}{\theta_1\theta_2 - {\theta_3}^2} \\ 0 \\ \dfrac{\theta_3 - \theta_2}{\theta_1\theta_2 - {\theta_3}^2} \end{bmatrix}$$

Once again, it is better not to use those particular matrices. Indeed, the more complete ones are useful since you can pick almost any position to balance the think 2 around. As we said already, our Matlab algorithm does not used those particular A and B matrix, but the basic ones.

Matlab gives us the following matrices after running "linearization.m" for both swing up and mid:



```
>> A

A =

         0    1.0000         0         0
   81.4377         0  -44.2430         0
         0         0         0    1.0000
  -93.1783         0  152.7092         0

>> B

B =

         0
  161.2228
         0
 -315.4237
```



```
>> A

A =

         0    1.0000         0         0
  -81.4377         0   44.2430         0
         0         0         0    1.0000
   69.6971         0   64.2232         0

>> B

B =

         0
  161.2228
         0
   -7.0219
```

Figure III-1-1: Presentation of the matrices A and B for our completely linearized system

2. Balancing control

Now that our system is linearized, we can use Linear Quadratic Regulation method to design our controller. What we are looking for are full state feedback controllers, where each state variable is affected by a coefficient to create the new command signal:

$$u - u_r = -K(x - x_r)$$

In our situation, $-u_r = 0$ since $cos\left(\frac{\pi}{2}\right) = 0$, so we can simplify the writing as:

$$\tau = -K(x - x_r)$$

The Matlab function "lqrd" solves this equation to minimize the following cost function:

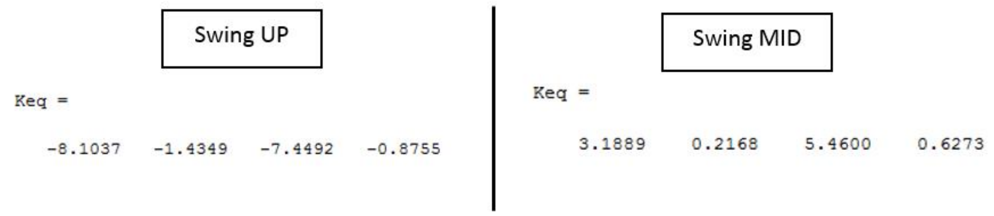
$$J = \int_0^\infty (x^T Q x + u^T R u) dt$$

For this process we can use the following matrices:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad R = [1]$$

Those matrices can actually be tuned regarding what kind of solution we want to our problem… Indeed, if we want to limit the uses of the input, for example to save energy, we can raise the value of R so that being bigger it'll have more weight in the cost function. We can also give more weight to each eigenvalues of the Q matrix to impact the state variables and therefore having a system that react faster or pay more attention to the position of the second link. In the one above, we have chosen not to pay attention about the variation of both angles.

By finding the minimum of this quadratic problem, the function "lqrd" returns the vector coefficient chosen for the controller. This gives us the following results:



Swing UP

```
Keq =

   -8.1037   -1.4349   -7.4492   -0.8755
```

Swing MID

```
Keq =

    3.1889    0.2168    5.4600    0.6273
```

Figure III-2-1: Presentation of the balancing control coefficients for both swing up and mid.

Now that we got both of our controller for the swing up and the balancing control, we need to build a way to know when to use the balancing control. In fact, if it starts to fast, link2 won't be at a position close enough to the final one and by stopping the swing we might fail the experiment.

Therefore, we have to choices. Either we tune a really good K_p and K_d combination and we look at the time for which the swing up is complete correctly, either we perform a pretty good but not perfect swing up and we look at what error is fine to consider the swing up is complete.

In the first situation, we should use the measured time to stop the swing up control and use the balancing control instead. In the second one, we must define a value for the error on q_1 and q_2 for which the swing process is over and that will give us the moment we need to switch for the balancing control. According to the model thesis, the second method is better for testing reasons. Actually, both of them works pretty well using simulations, but we kept the second one since it is better to refer to the system state variables until the end of the control design in my opinion. A minimum error of 0.1rad was choose for q_1 and one of 0.2rad was choose for q_2.

Those values can also be tuned to increase or vary the system response in reference to what the requirements are.

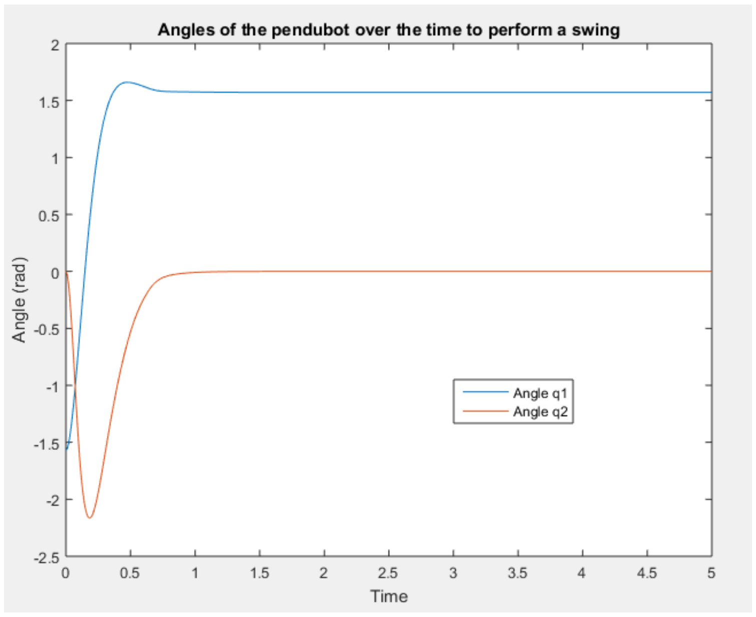
IV. Matlab simulation of the problem

To simulate the Pendubot system and apply everything we needed to control it, we processed in several steps. The idea is first to set up the known constants compute everything which is independent of the system states, and then run a simulation with ode45. The simulation function includes the computation of the system dynamics (inertia, centrifugal, and gravity matrices), the computation of the torque input and the process of the next step values of the state variables.

We can list the following steps:

- Setting up basic constants and choosing between swing UP and swing MID.
- Compute the value of the parameter theta using “calc_const.m”.
- Linearizing the system completely using “linearization.m”
- Finding the linear quadratic digital controller
- Simulate the next step of the system behavior with ode45 and “swing.m”. For each step:
    - Setting up the desired values for both angles.
    - Computation of the system matrices M, C and G using “calc_matrix.m”.
    - Setting up the PD controller
    - Compute the torque input regarding the actual position of the system.
    - Compute the new system’s state.
- Use the angles values over time to draw graph and to make a little animation of the Pendubot.

Running the program gives us the following results:



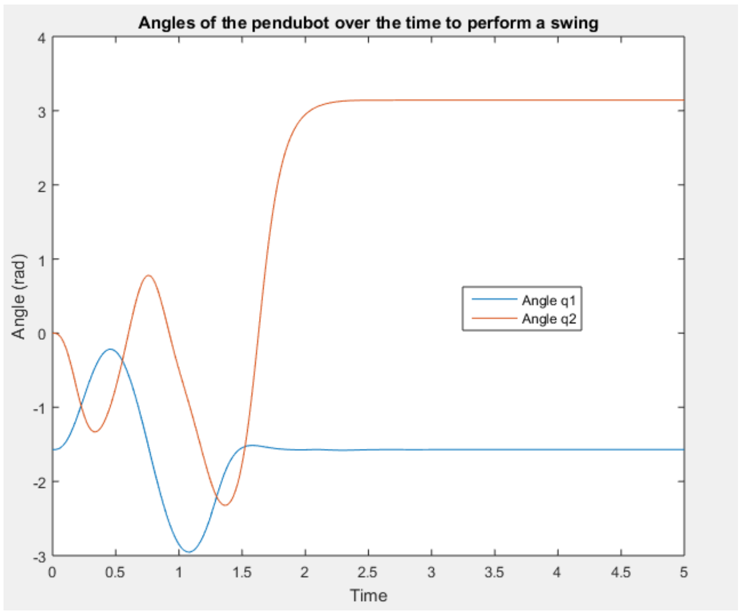


Figure IV-1: Plots of both system’s angles versus time for swing UP on the left and swing Mid on the right.

What we can notice right now is that using a simulation seems to get a pretty straight forward response. I don’t want to go through optimizing the system with only a model of the Pendubot, since as we can see there is not too much variations in the angles once we reach the steady area. Another thing to notice is the fact that the swing mid balancing control only starts once link 2 to enter the conditions we set up after the fourth “stage” of its movement, maybe it is possible to get a faster response with the system setting up to its final state when link 2 is on its third stage of movement.

Two videos of the animations are available attached to the document where we can see the Pendubot performing a swing UP and a swing MID in the (x,y) plan, as on the following picture:

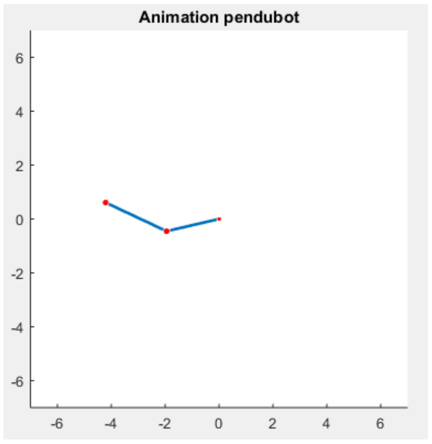


Figure IV-2: One image of the animation token during the simulation of swing MID

V. Conclusion

By splitting the job in one swing movement and one balancing movement, we managed to create a controller for a non linear system. This controller is able to bring the Pendubot in almost any state (the ones with $q_1 = 0$ or $q_1 = \pi$ are not controllable) by just giving it the initial and final values of those. In addition, everything was done without knowing about the system parameters, which means that the same method can be use in a case we don't know about the plant characteristics.

Simulation is a really interesting tool that allows us to test different things. Although that we kept it simple, by adding some perturbation to the system for example, we can tune the controller without even caring about messing with the plant. This is a huge test tool. However, we noticed that it has some limits because everything can't be simulated. In addition, it is also possible to match couple of specifications required by tuning several coefficients, especially the Q and R matrices that defines the cost function.

Overall, this project helped us to discover a real situation problem. Mixing the math and the trials, we learned how to apply those methods in a real case situation.