Laboratoire d'Automatique

EPFL STI IGM LA
ME C2 399 (Bâtiment ME)
Station 9
CH-1015 Lausanne

Téléphone : + 41 21 693 38 38
Fax : + 41 21 693 25 74
E-mail : philippe.muellhaupt@epfl.ch
Site web : http://la.epfl.ch

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Lausanne, novembre 2016

Projet de semestre M3
Automne 2016

Candidat : **NGUYEN Duy Tam Frédéric**
Section de génie mécanique

Sujet : **Swing-up of a Triple Arm Inverted Pendulum using Time Reversal**

The LA recently upgraded its single arm inverted pendulum into a triple arm inverted pendulum. The mechanical part as well as the measurements part are now completed. The challenge of this project is to swing up the triple inverted pendulum using a time-reversal strategy. Numerical experiments using Matlab will be used to validate the approach and an implementation in Labview is also foreseen.

This project requires the following courses: Control Systems or Automatic and Commande Multivariable or equivalent. Nonlinear control will be a plus.

Le nombre de crédits réservés au plan d'études pour ce projet est de **dix.**

Dr. Philippe MULLHAUPT

Assistants responsables :
SARSHARI Ehsan

Abstract

In this report, the subject of the swing-up control of a triple arm inverted pendulum is approached. The swing-up strategy uses a time-reversal strategy. It is based on reproducing the reversed behaviour of a known desired trajectories of a falling triple pendulum. The focuses handled are the mathematical modeling of the triple pendulum, the elaboration of a controller for tracking a planned motion and the simulation of the swing-up phase by using the described strategy in MATLAB environment.
Keywords: Triple pendulum, swing-up, time-reversal strategy, LQR controller

Contents

1 Introduction

The triple arm inverted pendulum is a typical problem in the control system field. Since it has less actuators than degrees of freedom, this under-actuated system constitutes an excellent test bench for testing and evaluating many types of controllers. Moreover its non-linearity, a dynamics behaviour governed by coupled ordinary differential equations make the regulation task complicated.

When approaching the regulation of an inverted pendulum, there is a need to separate it in two tasks: the stabilization and the swing-up. Each of these tasks have seen numerous contributions and these contributions reflects the diversity of control techniques. For example the stabilization task is handled with a state feedback design by Medrano-Cerda (1997) or with a controller based on the LQR method designed by Gupta et al. (2014). The swing-up task has also seen noticeable improvements provided by the papers from Åström and Furuta (2000) and Glück et al. (2013).

This study aims to evaluate the feasibility of the swing-up strategy for the triple arm inverted pendulum. It will begin with a mathematical modeling of the system. Then some concepts such as controllability of a system and design of LQR controllers are discussed in this report. Finally, simulations from the MATLAB environment of the swing-up phase are presented. The project is restricted to pure simulations studies.

2 Principle of the swing-up strategy

The main focus of this project is the swing-up phase of the pendulum. This phase consists in leading the cart of the triple pendulum from the stable equilibrium to an unstable equilibrium where all the arms are in the upright position.

The principle is resumed in Figure 1. The first step (phases 1 and 2) is to let fall the triple pendulum from its unstable equilibrium to its stable equilibrium. A PD controller which applies to the cart acts as a damper. Meanwhile the falling of the three links, their angles and angular velocities are recorded (phases 2 and 3) as well as the position and velocity of the cart, and the input of the PD controller. The recording is stopped after a certain time or until the triple pendulum is stabilized. Then the logged input PD controller is reversed (phase 3) and implemented to the triple pendulum for the swing-up. The system have the same behaviour recorded previously but in the reversed time or way (phase 6).

3 Modeling of the triple pendulum

For the purpose of this project, it is required to develop the equations of motion of the triple pendulum system. A *Lagrangian approach* is used for the modeling of the triple pendulum. The obtained model is linearized around operating configuration such as the down and up configuration. In the following pages, vector and matrix quantities are identified in bold characters.

3.1 Description of the triple pendulum

The pendulum consists in three links mounted on a cart. The cart is driven on a rail track. It is the only actuated part of the system. The other parts are free to move in their degree of freedom. For each link, a length and a mass is defined. The links are supposed to be made of homogeneous material so the gravity center for each one are on the geometrical middle of each bars. The viscous frictions in the bears of the cart and the links are neglected for the modeling of the triple pendulum.

A schematic representation of the system is showed in Figure 2. The dispositions of the

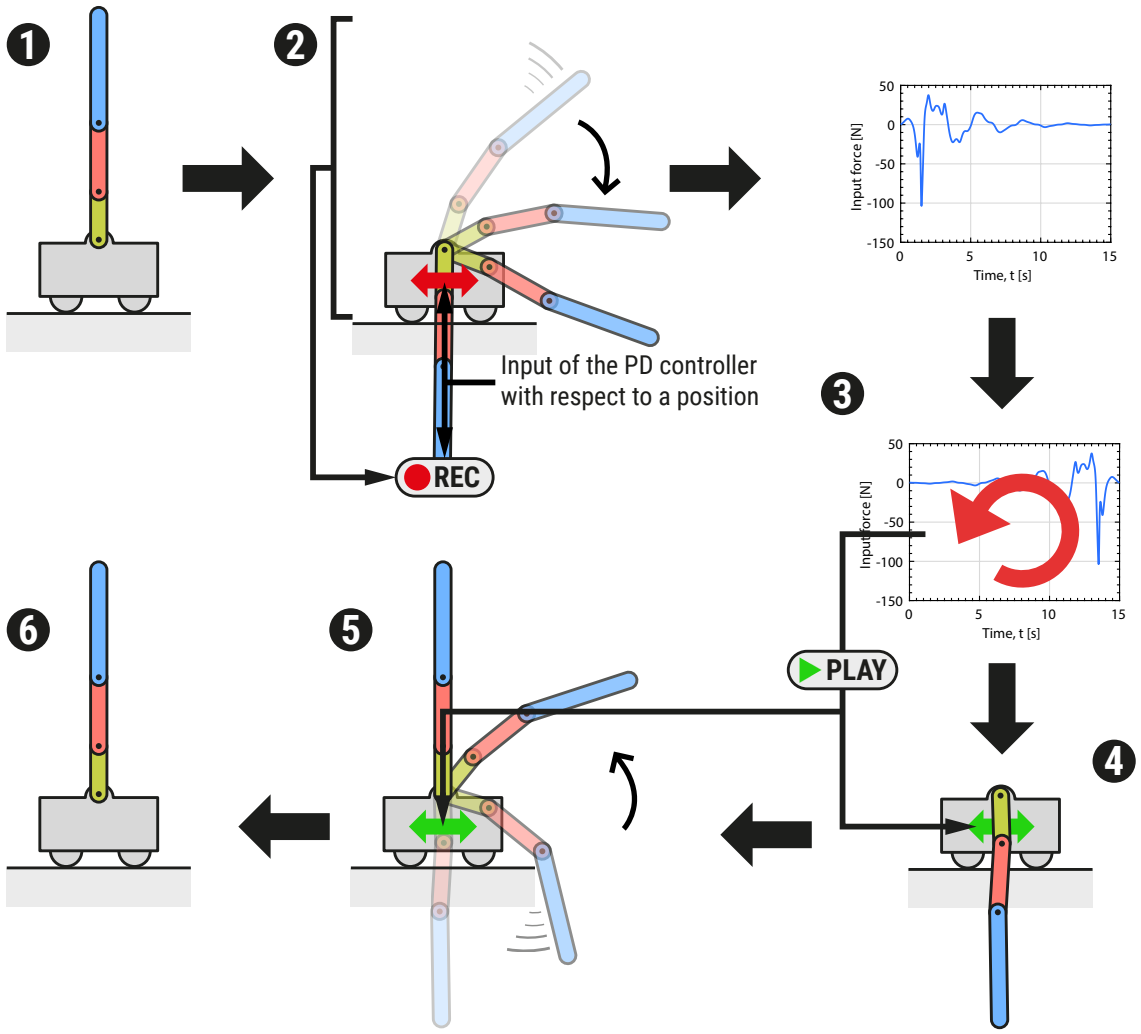


Figure 1 – Principle of the swing-up strategy

positions and angles, and the coordinate system are showed in Figure 2; they will be used for the modeling of the pendulum.

Since the triple pendulum is a 4-degree-of-freedom system, there are four generalized coordinates which are defined for this pendulum. The first one is the position of the cart x_{cart} with respect to a reference. The last three generalized coordinates are the angles $(\theta_1, \theta_2, \theta_3)$ of links at the joints. Their zero value are when the bars are in up position and they increase their values when turning counterclockwise. The assignment of each degree of freedom to a generalized coordinate is according to the following:

$$x_{cart} \rightarrow q_1, \qquad \theta_1 \rightarrow q_2, \qquad \theta_2 \rightarrow q_3, \qquad \theta_3 \rightarrow q_4$$

The parameters of the system are given values in Table 1 and will be used for every numerical simulation of the triple pendulum.

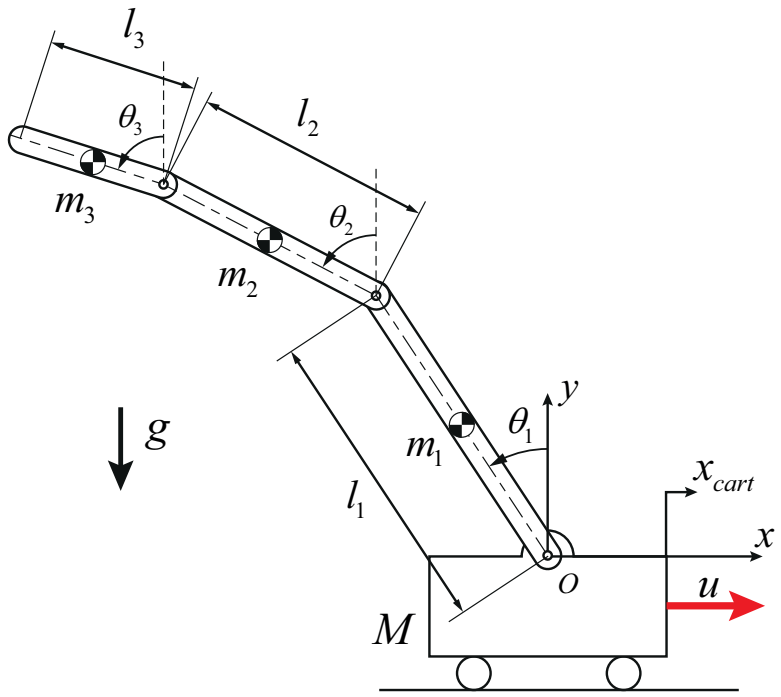


Figure 2 – Triple arm inverted pendulum with assumed coordinate systems, dimensions and angles

Table 1 – Parameters for the following simulations

Parameter	Symbol	Value	Unit
Mass of the cart	m_{cart}	1	kg
Mass of first arm	m_1	1	kg
Mass of second arm	m_2	1	kg
Mass of third arm	m_3	1	kg
Length of first arm	l_1	1	m
Length of second arm	l_2	1	m
Length of third arm	l_3	1	m
Inertia of first arm	$J_1 = \frac{m_1 l_1^2}{12}$	0.0416	kgm^2
Inertia of second arm	$J_2 = \frac{m_2 l_2^2}{12}$	0.0416	kgm^2
Inertia of third arm	$J_3 = \frac{m_3 l_3^2}{12}$	0.0416	kgm^2
Constant of gravity	g	9.81	m/s^2

3.2 Mathematical model

After defining the coordinate system and the generalized coordinates, the core of the modeling takes place here. The equations of motion for the system are derived from Lagrange's equations with a generalized force:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) - \frac{\partial \mathcal{L}}{\partial q_i} = u_i \quad \text{with } i = 1,2,3,4 \tag{1}$$

Knowing that there is only one force input which exert to the cart, the different u_i are defined such as $u_1 = u$ and $u_i = 0$ for $i = 2,3,4$. To derive these equations of motion, the Lagrangian

of the triple pendulum needs to be found:

$$\mathcal{L} = \mathcal{T} - \mathcal{V} \tag{2}$$

where $\mathcal{T}$ is the kinetic energy and $\mathcal{V}$ the potential energy. Furthermore, the vectors from the origin of the coordinate system to the center of mass of each arms are defined by:

$$\mathbf{p}_1 = \begin{bmatrix} x_{cart} - l_1 \sin(\theta_1)/2 \\ l_1 \cos(\theta_1)/2 \end{bmatrix}$$

$$\mathbf{p}_2 = \begin{bmatrix} x_{cart} - l_1 \sin(\theta_1) - l_2 \sin(\theta_2)/2 \\ l_1 \cos(\theta_1) + l_2 \cos(\theta_2)/2 \end{bmatrix}$$

$$\mathbf{p}_3 = \begin{bmatrix} x_{cart} - l_1 \sin(\theta_1) - l_2 \sin(\theta_2) - l_3 \sin(\theta_3)/2 \\ l_1 \cos(\theta_1) + l_2 \cos(\theta_2) + l_3 \cos(\theta_3)/2 \end{bmatrix}$$

The derivation for deriving kinetic and potential energies is borrowed from Glück et al. (2013). The kinetic energy of the whole system is therefore given by:

$$\mathcal{T} = \frac{1}{2} m_{cart} \dot{x}_{cart}^2 + \frac{1}{2} \sum_{i=1}^{3} m_i \dot{\mathbf{p}}_i^T \dot{\mathbf{p}}_i + \frac{1}{2} \sum_{i=1}^{3} J_i \dot{\theta}_i^2 \tag{3}$$

The potential energy is computed with the y components of each vector $\mathbf{p}_i$. Hence it is given by:

$$\mathcal{V} = g \cdot \sum_{i=1}^{3} m_i \, \mathbf{p}_{i,y} \tag{4}$$

The Lagrange's equations obtained with (1) are:

$$\begin{aligned} &-\ddot{\theta}_1 \left[(l_1 m_1 \cos(\theta_1))/2 + l_1 m_2 \cos(\theta_1) + l_1 m_3 \cos(\theta_1) \right] - \ddot{\theta}_2 \left[l_2 m_2 \cos(\theta_2)/2 + l_2 m_3 \cos(\theta_2) \right] \\ &\quad - \ddot{\theta}_3 l_3 m_3 \cos(\theta_3)/2 + \ddot{x}\left(M + m_1 + m_2 + m_3\right) + \dot{\theta}_1^2 l_1 \left(m_1/2 + m_2 + m_3\right) \sin(\theta_1) \\ &\quad + \dot{\theta}_2^2 l_2 \left(m_2/2 + m_3\right) \sin(\theta_2) + \dot{\theta}_3^2 l_3 m_3 \sin(\theta_3)/2 = u_1 \end{aligned} \tag{5}$$

$$\begin{aligned} &\frac{l_1}{6} \left[\ddot{x} \left(-3m_1 \cos(\theta_1) - 6m_2 \cos(\theta_1) - 6m_3 \cos(\theta_1)\right) + \ddot{\theta}_1 l_1 \left(2m_1 + 6m_2 + 6m_3\right) \right. \\ &\quad + \ddot{\theta}_2 l_2 \left(3m_2 + 6m_3\right) \cos(\theta_1 - \theta_2) + \ddot{\theta}_3 3 l_3 m_3 \cos(\theta_1 - \theta_3) \\ &\quad + \dot{\theta}_2^2 l_2 (3m_2 + 6m_3) \sin(\theta_1 - \theta_2) + \dot{\theta}_3^2 3 l_3 m_3 \sin(\theta_1 - \theta_3) \\ &\quad - 6g m_2 \sin(\theta_1) - 6g m_3 \sin(\theta_1) - 3g m_1 \sin(\theta_1) \\ &\quad \left. - g(3m_1 + 6m_2 + 6m_3) \sin(\theta_1) \right] = 0 \end{aligned} \tag{6}$$

$$\begin{aligned} &-\tfrac{1}{6} \left[l_2 (3g m_2 \sin(\theta_2) + 6g m_3 \sin(\theta_2) + 3m_2 \ddot{x} \cos(\theta_2) + 6m_3 \ddot{x} \cos(\theta_2) - 2 l_2 m_2 \ddot{\theta}_2 \right. \\ &\quad - 6 l_2 m_3 \ddot{\theta}_2 + 3 l_1 m_2 \dot{\theta}_1^2 \sin(\theta_1 - \theta_2) + 6 l_1 m_3 \dot{\theta}_1^2 \sin(\theta_1 - \theta_2) - 3 l_3 m_3 \dot{\theta}_3^2 \sin(\theta_2 - \theta_3) \\ &\quad \left. - 3 l_1 m_2 \ddot{\theta}_1 \cos(\theta_1 - \theta_2) - 6 l_1 m_3 \ddot{\theta}_1 \cos(\theta_1 - \theta_2) - 3 l_3 m_3 \ddot{\theta}_3 \cos(\theta_2 - \theta_3)) \right] = 0 \end{aligned} \tag{7}$$

$$\begin{aligned} &-\tfrac{1}{6} \left[l_3 m_3 (3 l_1 \sin(\theta_1 - \theta_3) \dot{\theta}_1^2 + 3 l_2 \sin(\theta_2 - \theta_3) \theta_2 d^2 - 2 l_3 \ddot{\theta}_3 + 3g \sin(\theta_3) + 3 \ddot{x} cos(\theta_3) \right. \\ &\quad \left. - 3 l_1 \ddot{\theta}_1 \cos(\theta_1 - \theta_3) - 3 l_2 \ddot{\theta}_2 \cos(\theta_2 - \theta_3)) \right] = 0 \end{aligned} \tag{8}$$

By arranging the four equations with the symbolic math toolbox from MATLAB, the expressions of second order derivative of each generalized coordinate are found.

$$\begin{cases} \ddot{q}_1 = y_1(q_1, \dot{q}_1, \ldots, q_4, \dot{q}_4, u_1) \\ \qquad\qquad \vdots \\ \ddot{q}_4 = y_4(q_1, \dot{q}_1, \ldots, q_4, \dot{q}_4, u_1) \end{cases} \tag{9}$$

Hence a system of second order differential equations is obtained. The system needs to be rewritten as an equivalent system of first order differential equations in order to be solved by the MATLAB ODE solver (ode45). The generic substitutions are used and the chosen state variables are resumed in Table 2.

Table 2 – Selection of the state variables

Object	Order	State variables
Cart	2	$z_1 = q_1, \quad z_2 = \dot{q}_1$
1st link	2	$z_3 = q_2, \quad z_4 = \dot{q}_2$
2nd link	2	$z_5 = q_3, \quad z_6 = \dot{q}_3$
3rd link	2	$z_7 = q_4, \quad z_8 = \dot{q}_4$

Hence the system obtained by the transformation is of the following form:

$$\begin{cases} \dot{z}_1 = z_2 = f_1(q_1, \dot{q}_1, \ldots, q_4, \dot{q}_4, u_1) \\ \dot{z}_2 = \ddot{q}_1 = f_2(q_1, \dot{q}_1, \ldots, q_4, \dot{q}_4, u_1) \\ \dot{z}_3 = z_4 = f_3(q_1, \dot{q}_1, \ldots, q_4, \dot{q}_4, u_1) \\ \dot{z}_4 = \ddot{q}_2 = f_4(q_1, \dot{q}_1, \ldots, q_4, \dot{q}_4, u_1) \\ \dot{z}_5 = z_6 = f_5(q_1, \dot{q}_1, \ldots, q_4, \dot{q}_4, u_1) \\ \dot{z}_6 = \ddot{q}_3 = f_6(q_1, \dot{q}_1, \ldots, q_4, \dot{q}_4, u_1) \\ \dot{z}_7 = z_8 = f_7(q_1, \dot{q}_1, \ldots, q_4, \dot{q}_4, u_1) \\ \dot{z}_8 = \ddot{q}_4 = f_8(q_1, \dot{q}_1, \ldots, q_4, \dot{q}_4, u_1) \end{cases} \tag{10}$$

with the second order derivative $\ddot{q}_i,\ i = 1,2,3,4$ obtained in (9).

3.3 State-space formulation and linearization

Taking into account the non-linearities showed by (10), the utilization of regulators such as LQR arises difficulties when regulating the triple pendulum. These difficulties can be avoided by doing a linear approximation of the model. For the needs of the LQR controllers, the non-linear system is linearized at certains positions such as the equilibrium points (stable, unstable). The linearization is derived following the method described in the lecture notes of the "Dynamical system" course[1]. Since the triple pendulum is a MIMO case, the matrices **A** and **B** are the Jacobian matrices of f_i (10) with respect to q_i and u_1, evaluated at a certain operating configuration of the pendulum $\bar{q}_i$ and $\bar{u}_1$:

$$\mathbf{A} = \left[\begin{array}{ccc} \frac{\partial}{\partial z_1} f_1(\bar{\mathbf{z}}, \bar{u}_1) & \cdots & \frac{\partial}{\partial z_8} f_1(\bar{\mathbf{z}}, \bar{u}_1) \\ \vdots & \ddots & \vdots \\ \frac{\partial}{\partial q_1} f_8(\bar{\mathbf{z}}, \bar{u}_1) & \cdots & \frac{\partial}{\partial q_8} f_8(\bar{\mathbf{z}}, \bar{u}_1) \end{array} \right] \Bigg|_{\bar{u}_1, \bar{z}_1, \ldots, \bar{z}_8}, \quad \mathbf{B} = \left[\begin{array}{c} \frac{\partial}{\partial u_1} f_1(\bar{\mathbf{z}}, \bar{u}_1) \\ \vdots \\ \frac{\partial}{\partial u_1} f_8(\bar{\mathbf{z}}, \bar{u}_1) \end{array} \right] \Bigg|_{\bar{u}_1, \bar{z}_1, \ldots, \bar{z}_8}$$

[1] ME-221 Systèmes dynamiques, EPFL 2014, D. BONVIN

Matrices **C** and **D** are parts of the output of the pendulum. The matrix **C** has 4 rows because the cart's position and the links orientation constitutes the output of the system:

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad \mathbf{D} = [0]$$

Hence the state-space form is obtained:

$$\begin{aligned} \dot{\mathbf{z}} &= \mathbf{A}\,\mathbf{z} + \mathbf{B}\,u_1 \\ \mathbf{y} &= \mathbf{C}\,\mathbf{z} + \mathbf{D}\,u_1 \end{aligned} \tag{11}$$

3.4 Verification of the Lagrange model

As a verification of the model obtained in subsection 3.2, it is proposed to check the variation of the total energy of the triple inverted pendulum. The definitions of the kinetic energy Equation 3 and potential energy Equation 4 are reused to obtain the total energy. The variation of the total energy should be nearly zero in the absence of external forces. In Figure 3, the variation of the total energy is of the order of magnitude 10^{-8} which ensure that the mathematical model is correct:

$$E_{total,k-1} - E_{total,k} \approx 0$$

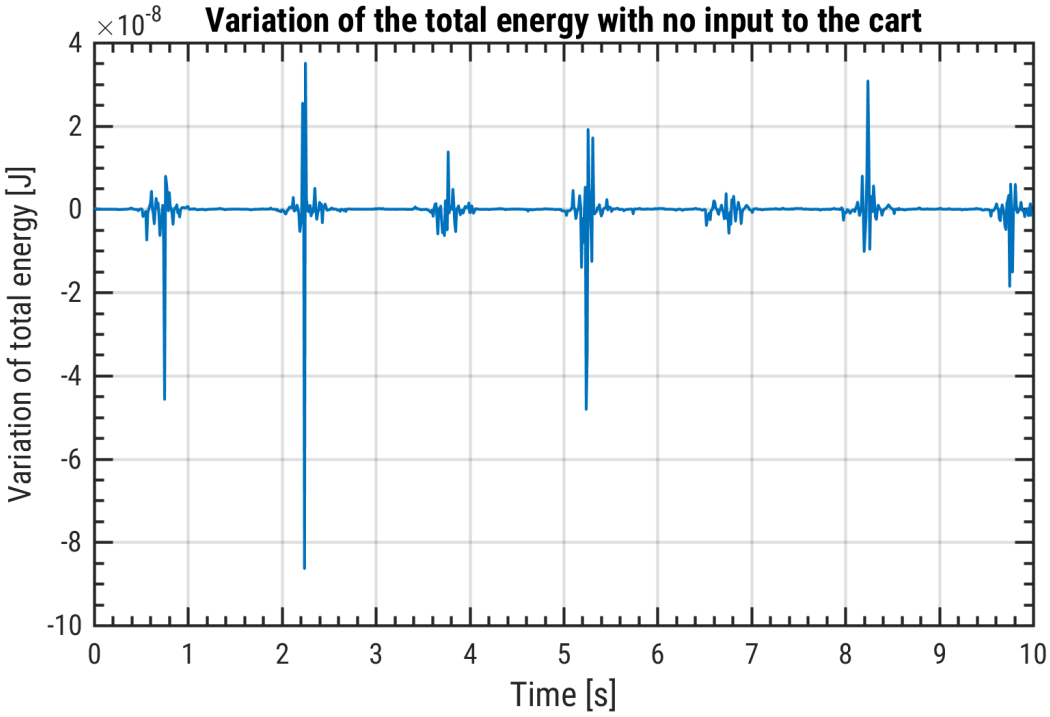


Figure 3 – Variation of the total energy with no external forces on the cart with initial conditions $\mathbf{z} = [0, 0, -\pi/2, 0, -\pi/2, 0, -\pi/2, 0]^{\mathrm{T}}$ simulated on MATLAB.

4 LQR controller

4.1 Brief overview of theory

The main focus of the LQR controller is to handle both the limitation of the control signal u and a fast transient phase for the system. A general block diagram of an LQR regulation is shown in Figure 4. It relies on minimizing a quadratic cost function J which contains the states $\mathbf{z}$ and input u. In other words, the optimal regulation is likely to find a control input $u_1(t)$, $t \in [0, \infty[$ to the system dynamics in continuous time

$$\dot{\mathbf{z}} = \mathbf{A}\mathbf{z} + \mathbf{B}\,u_1$$

that minimizes the cost function:

$$J = \int_0^\infty \left(z^T \mathbf{Q} z + u_1 R u_1\right) \mathrm{d}t$$

where $\mathbf{Q} \in \mathbb{R}^{8\times 8}$ is as a symmetric and positive-definite matrix weighting the states, and $R > 0$ is a scalar weighting factor since u_1 is also a scalar. The feedback control law minimizing the value of J is

$$u_1 = -\mathbf{K}\mathbf{z}$$

where $\mathbf{K}$ is given by $K = r^{-1}\mathbf{B}^T\mathbf{P}$. The matrix $\mathbf{P}$ is computed by solving the continuous time *Riccati* equation

$$\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} - \mathbf{P}\mathbf{B}\mathbf{B}^T\mathbf{P}\,R^{-1} + \mathbf{Q} = 0$$

The command `[K,P,E]=lqr(A,B,Q,R)` from MATLAB solves the Riccati equation and finds the optimal matrix gain $\mathbf{K}$.

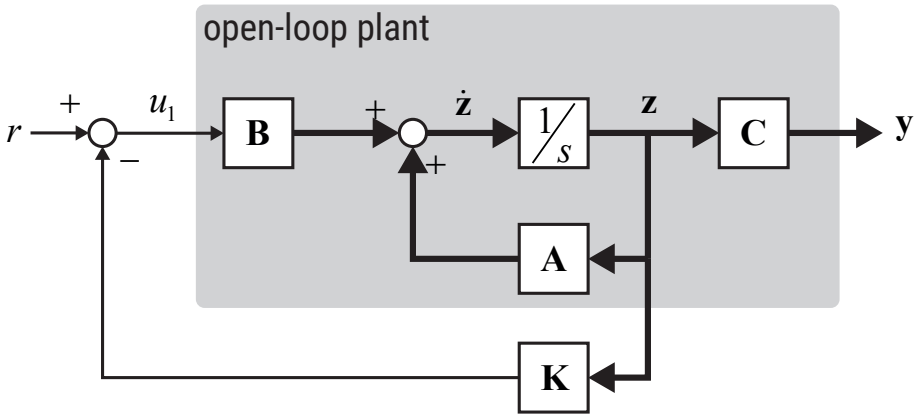


Figure 4 – Block diagram of an LQR control system type with full state feedback

4.2 Controllability

The satisfaction of the controllability property is crucial for designing a controller such as the LQR. It means that there exists a finite control input u_1 that the states of the system can be brought at any desired values $\mathbf{z}(t)$ from any initial values $\mathbf{z}_0$. For a system to be controllable, the rank of the controllability matrix $\mathcal{C} \in \mathbb{R}^{n\times n}$ must be equal to the number n of state variable of the system ($n = 8$ in the case of the triple pendulum).

$$\mathrm{rank}\,(\mathcal{C}) = \mathrm{rank}\Big(\left[\mathbf{B}|\mathbf{A}\mathbf{B}|\mathbf{A}^2\mathbf{B}|\cdots|\mathbf{A}^{n-1}\mathbf{B}\right]\Big) = n$$

The controllability matrix can be computed using the MATLAB function `Co=ctrb(A,B)`. Concerning the rank of the controllability matrix, it can be found by using the function `n=rank(Co)`. One can also compute its singular value decomposition and count the number of nonzero values in order to check the controllability of the system.

5 Proposed control strategy

The strategy chosen for the swing-up maneuver is based on reproducing the exact reversed behaviour of a falling triple pendulum.

5.1 Generation of a downward trajectory

The necessary step of the swing-up maneuver is the planning of a motion that has to be followed. The trajectory is generated by recording the variation of the generalized coordinates during a falling of the triple pendulum. Before the record begins, the pendulum is placed in the configuration desired at the end of the swing-up maneuver.

Since there is no friction in the pendulum, a proportional derivative controller is implemented to the cart. Figure 5 shows the implementation of the PD controller in the system. This controller will act as a spring and a damping component in order to dissipate the initial potential energy given by the upward initial configuration. The parameters of the PD controller and the initial configuration of the system are tuned such as during the falling, each link does not rotate several times on itself. With trial and error procedure, an ideal trajectory is found with appropriate parameters for the PD controller ($k_p = 0.3$ and $k_i = 6.5$) and the initial configuration ($\mathbf{z} = [0, 0, 0, -0.4, 0, -0.6, 0, -0.6]^{\mathrm{T}}$). At the end of the generation of a trajectory, the history of variations of the generalized coordinates and the force applied on the cart by the controller are saved. They will be used for the swing-up maneuver.

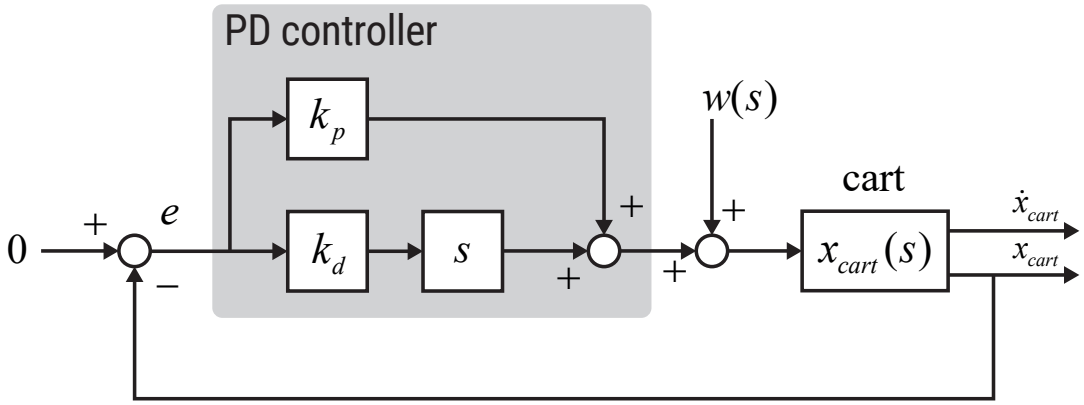


Figure 5 – Block diagram of the implementation of the PD controller in the triple pendulum. The $w(s)$ disturbance comes from the influences from the links to the cart.

5.2 Swing-up maneuver with exact initial conditions

As a first approach of the swing-up maneuver, the time-reversed input recorded is only provided as an input to the triple pendulum. Also the initial condition for the swing-up phase is equal to the last configuration N recorded with reversed speeds.

$$\mathbf{z}_{0,swing-up} = \left[x_{cart_N}, -\dot{x}_{cart_N}, \theta_{1_N}, -\dot{\theta}_{1_N}, \theta_{2_N}, -\dot{\theta}_{2_N}, \theta_{3_N}, -\dot{\theta}_{3_N}, \right]^T$$

At the end of this swing-up, the triple pendulum should have the same configuration as the one at the begining of the recorded trajectories.

$$\mathbf{z}_{N,swing-up} = \left[x_{cart_0}, -\dot{x}_{cart_0}, \theta_{1_0}, -\dot{\theta}_{1_0}, \theta_{2_0}, -\dot{\theta}_{2_0}, \theta_{3_0}, -\dot{\theta}_{3_0}, \right]^T$$

5.3 Swing-up maneuver with different initial conditions

As the triple pendulum exhibits a hyperchaotic behaviour, a slight variation of initial conditions for the swing-up phase can lead to another complete configuration at the end of the swing-up maneuver. An idea is to reduce the initial deviation during the first seconds so that it is possible to catch the planned motion. To solve this issue, the implementation of an LQR is considered in order to fulfil the task.

The operating point of the LQR controller is set at the stable equlibrium ($\mathbf{z}_{eq} = [0\,0\,0\,0\,0\,0\,0\,0]^T$) of the triple pendulum.

6 Numerical results

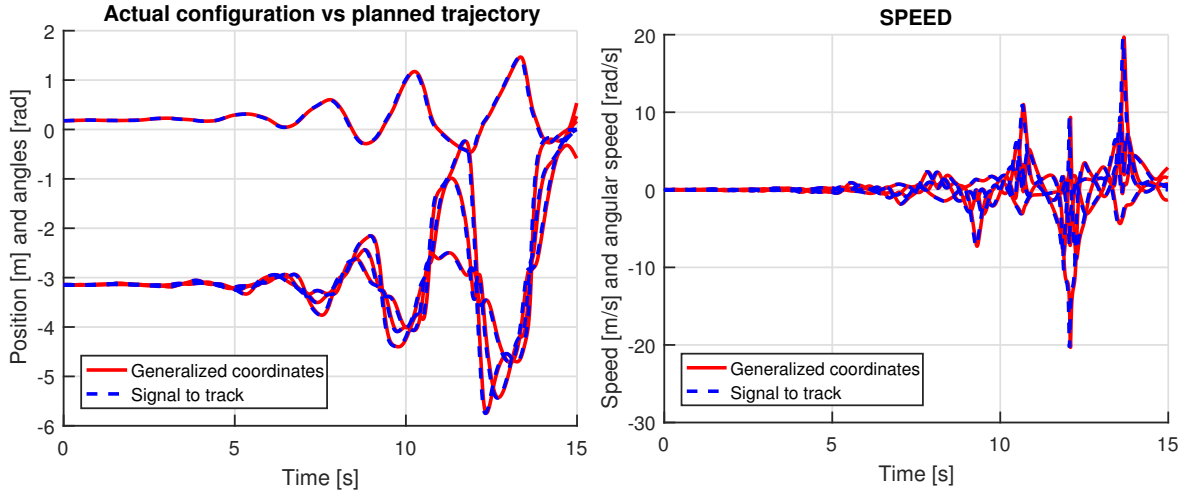
6.1 Generation of trajectories

A trajectory is generated by setting the initial condition found in subsection 5.1 and the mathematical model as input data to the function ode45 from MATLAB. The duration of the simulation is set to $T = 15\,\text{s}$ when the triple pendulum is nearly at rest. The results of the simulation are shown in Figure 6.

6.2 Swing-up maneuver simulation

Exact initial conditions

It is observed in Figure 7 that even the exact initial conditions are provided for the swing-up maneuver, there is a deviation at the end of the phase.



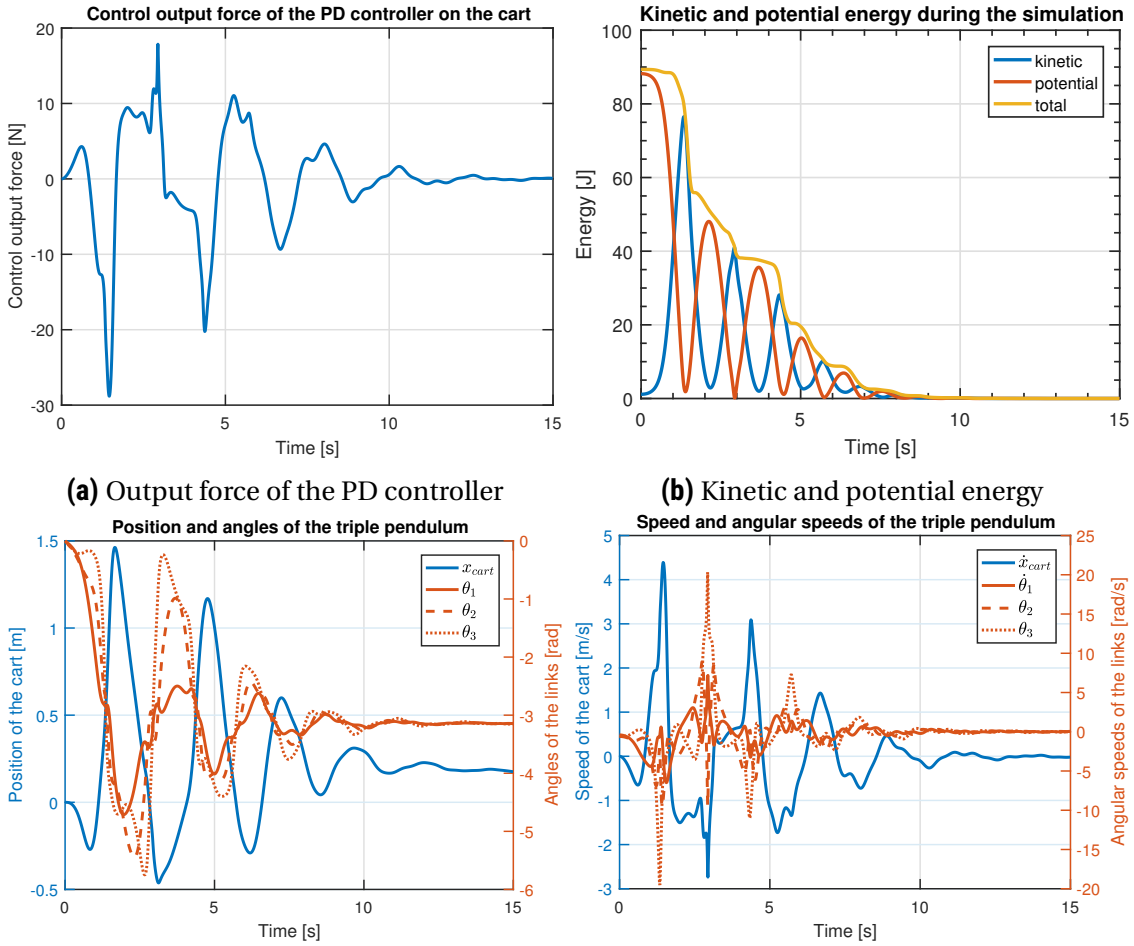
(a) Position and angles

(b) Speed and angular speed

Figure 7 – Actual vs planned trajectory

One can mesure the influence of the gap between the modified and planned initial conditions, a criterion is defined:

$$\Delta_{k,error}(t) = \left(z_k(t) - z_{k,planned}(t)\right)^2$$



(a) Output force of the PD controller

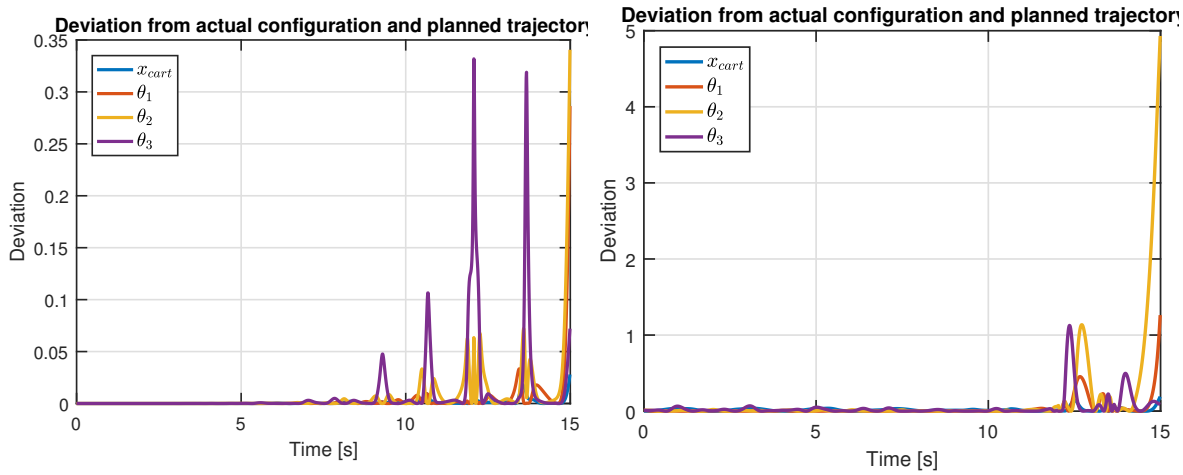
(b) Kinetic and potential energy

(c) Position of cart and angles of links

(d) Speed and angular speed

Figure 6 – Simulation of a falling triple pendulum with initial conditions $\mathbf{z} = [0,0,0,-0.4,0,-0.6,0,-0.6]^{\mathrm{T}}$

The trajectory resulted from modified initial conditions (by adding 0.1 rad at every angles of the links) differs completely from the planned trajectory. This can be observed in Figure 8.



(a) Exact initial conditions

(b) Modified initial conditions

Figure 8 – Comparison between trajectories with exact and modified initial conditions. The modified initial conditions are changed by adding 0.1 rad for all angles of the link.

Different initial conditions

In order to handle initial conditions different from the planned trajectory, an LQR is applied for the first 10 seconds. The linearized state-space model is done around $\mathbf{z}_{LQR} = [0, 0, -\pi, 0, -\pi, 0, -\pi, 0]^T$. Thus the matrices **A** and **B** obtained are:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 24.1343 & 0 & -3.9493 & 0 & 0.4388 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -70.2090 & 0 & 35.5433 & 0 & -3.9493 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 59.2388 & 0 & -65.8209 & 0 & 17.1134 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -19.7463 & 0 & 51.3403 & 0 & -35.1045 & 0 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 \\ 0.7761 \\ 0 \\ -0.9851 \\ 0 \\ 0.2687 \\ 0 \\ -0.0896 \end{bmatrix}. \tag{12}$$

The controllability condition is verified since $\text{rank}(\mathcal{C}) = 8$. Since there is no guideline for selecting the weighting coefficient **Q** and R, an adequate choice found is

$$\begin{aligned} \mathbf{Q} &= \text{diag}\begin{pmatrix} 100 & 0 & 100 & 0 & 100 & 0 & 100 & 0 \end{pmatrix} \\ R &= 0.5 \end{aligned} \tag{13}$$

The resulted gain matrix is given by

$$\mathbf{K} = \begin{bmatrix} 14.1421 & 12.6550 & -18.0235 & 2.6144 & -0.0900 & 1.2742 & 1.8248 & 0.3713 \end{bmatrix} \tag{14}$$

With initial conditions modified by adding 0.1 rad at each angle, the deviation in function of time is plotted in Figure 9. One can observe that for the first 3 seconds, the LQR regulator achieves to catch up the planned trajectory.

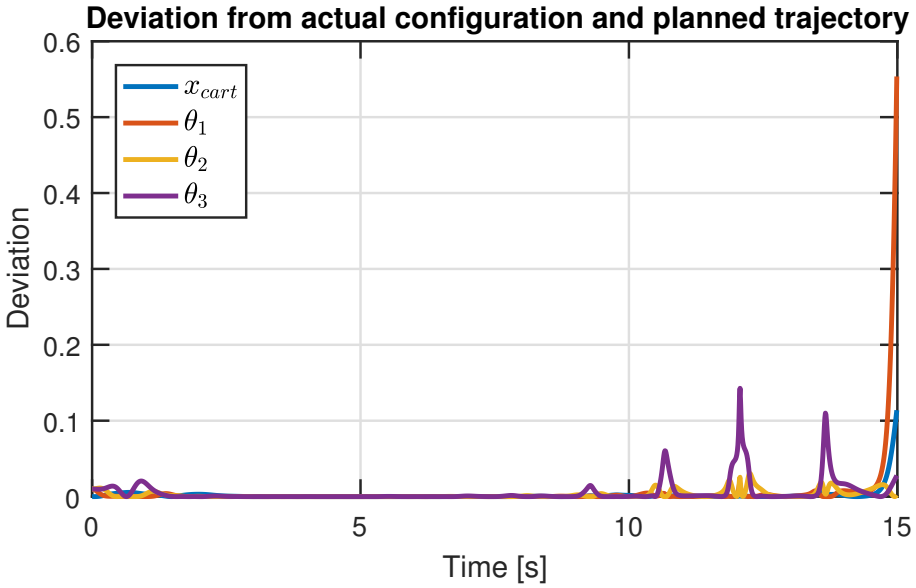


Figure 9 – Deviation from actual configuration and planned trajectory. The offset added to initial conditions is $\mathbf{z}_{0,offset} = [0, 0, 0.1, 0, 0.1, 0, 0.1, 0]^T$.

The sum of errors for each simulation are resumed in Table 3

Table 3 – Performance for each simulation

Initial conditions	Exact	Modified	Modified with LQR
Sum of errors between actual and planned	31.44	333.88	22.26

7 Conclusion and recommendations

In the light of the above, the time-reversal strategy provides an interesting method for the swing-up of the triple pendulum. The forces involved for the swing-up maneuver is limited. This can be suitable for systems which requires smooth accelerations. However the time needed for the pendulum to switch to the up equilibrium is equal to the time needed for the pendulum to fall and stabilize. It is much more longer than other methods such as the one treated by Glück et al. (2013). In their case, they achieved a transition time of 3.5 s.

One downfall of this method is that the generation of a trajectory is not done in an automatically way. Any changes in the pendulum parameters require to adapt the initial conditions for the generation of an ideal trajectory. Also period for which the LQR regulator is applied is set manually according to the behaviour of the pendulum. A trial and error method is required for an ideal trajectory.

The swing-up maneuver with exact initial conditions leads to a sgnificant deviation from the generated trajectory at the end of the phase. It should normally be exactly at the upright configuration at the end of the swing-up phase. However it is not the case and this might be due to numerical errors from the ODE function.

The single LQR applied for long periods in order to cope the deviations from planned trajectory is not adapted for this strategy. Indeed, since it is designed around an operating configuration, the linearized state-space model can not be correct during the first seconds. One mean of improvement is to design several LQRs along the trajectory with the purpose that several linearized state-space models along the time can approach the non-linear behaviour.

A MATLAB scripts and function files

File: `math_model_pendulum.m`

Description: Derivating the mathematical model of the triple pendulum using the symbolic math toolbox from MATLAB. It returns the expressions of the second order derivative of generalized coordinates.

Listing 1 – `math_model_pendulum.m`

```
syms M m1 m2 m3;
syms x xd xdd theta1 theta1d theta1dd theta2 theta2d theta2dd theta3 theta3d theta3dd;
syms l1 l2 l3;
syms u1 u2 u3 u4;
syms g;

%%%%%%%%%%%%%%%%%% POSITION OF CENTER OF MASS %%%%%%%%%%%%%%%%%%%%%%%%%%
p1 = [x - l1/2*sin(theta1); l1/2*cos(theta1)];
p2 = [x - l1*sin(theta1) - l2/2*sin(theta2); l1*cos(theta1) + l2/2*cos(theta2)];
p3 = [x - l1*sin(theta1) - l2*sin(theta2) - l3/2*sin(theta3); l1*cos(theta1) + l2*cos(theta2) + l3/2*cos(...
     theta3)];

%%%%%%%%%%%%%%%%%%%% SPEED OF CENTER OF MASS%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
p1d = diff(p1,x)*xd + diff(p1,theta1)*theta1d;
p2d = diff(p2,x)*xd + diff(p2,theta1)*theta1d + diff(p2,theta2)*theta2d;
p3d = diff(p3,x)*xd + diff(p3,theta1)*theta1d + diff(p3,theta2)*theta2d + diff(p3,theta3)*theta3d;

%%%%%%%%%%%%%%%%% LAGRANGIAN OF THE TRIPLE PENDULUM %%%%%%%%%%%%%%%%%%%%%
% Kinetic energy
KE = 0.5 * (M*xd^2 + m1*(p1d.')*p1d + m2*(p2d.')*p2d + m3*(p3d.')*p3d) + ...
    0.5 * (m1*l1^2/12*theta1d^2 + m2*l2^2/12*theta2d^2 + m3*l3^2/12*theta3d^2);
KE = simplify(KE);

% Potential energy
PE = g * (m1*p1(2) + m2*p2(2) + m3*p3(2));

% Lagrangian derivation
KExd = diff(KE,xd);
dtKExd = diff(KExd,xd)*xdd + ...
       diff(KExd,theta1)*theta1d + diff(KExd,theta1d)*theta1dd + ...
       diff(KExd,theta2)*theta2d + diff(KExd,theta2d)*theta2dd + ...
       diff(KExd,theta3)*theta3d + diff(KExd,theta3d)*theta3dd;
KEx = diff(KE,x);
PEx = diff(PE,x);

KEtheta1d = diff(KE,theta1d);
dtKEtheta1d = diff(KEtheta1d,theta1)*theta1d + diff(KEtheta1d,theta1d)*theta1dd + ...
       diff(KEtheta1d,xd)*xdd + ...
       diff(KEtheta1d,theta2)*theta2d + diff(KEtheta1d,theta2d)*theta2dd + ...
       diff(KEtheta1d,theta3)*theta3d + diff(KEtheta1d,theta3d)*theta3dd;
KEtheta1 = diff(KE,theta1);
PEtheta1 = diff(PE,theta1);

KEtheta2d = diff(KE,theta2d);
dtKEtheta2d = diff(KEtheta2d,theta1)*theta1d + diff(KEtheta2d,theta1d)*theta1dd + ...
       diff(KEtheta2d,xd)*xdd + ...
       diff(KEtheta2d,theta2)*theta2d + diff(KEtheta2d,theta2d)*theta2dd + ...
       diff(KEtheta2d,theta3)*theta3d + diff(KEtheta2d,theta3d)*theta3dd;;
KEtheta2 = diff(KE,theta2);
PEtheta2 = diff(PE,theta2);

KEtheta3d = diff(KE,theta3d);
dtKEtheta3d = diff(KEtheta3d,theta1)*theta1d + diff(KEtheta3d,theta1d)*theta1dd + ...
       diff(KEtheta3d,xd)*xdd + ...
       diff(KEtheta3d,theta2)*theta2d + diff(KEtheta3d,theta2d)*theta2dd + ...
       diff(KEtheta3d,theta3)*theta3d + diff(KEtheta3d,theta3d)*theta3dd;
KEtheta3 = diff(KE,theta3);
PEtheta3 = diff(PE,theta3);

%%%%%%%%%%%%%%%%%%%%%% EQUATIONS OF MOTION %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
eqx1 = simplify(dtKExd - KEx + PEx - u1);
eqx2 = simplify(dtKEtheta1d - KEtheta1 + PEtheta1);
```

```
eqx3 = simplify(dtKEtheta2d - KEtheta2 + PEtheta2);
eqx4 = simplify(dtKEtheta3d - KEtheta3 + PEtheta3);

Sol = solve(eqx1,eqx2,eqx3,eqx4,xdd,theta1dd,theta2dd,theta3dd)

syms z1 z2 z3 z4 z5 z6 z7 z8;
fx1 = subs(Sol.xdd, {x,xd,theta1,theta1d,theta2,theta2d,theta3,theta3d}, {z1,z2,z3,z4,z5,z6,z7,z8});
fx2 = subs(Sol.theta1dd, {x,xd,theta1,theta1d,theta2,theta2d,theta3,theta3d}, {z1,z2,z3,z4,z5,z6,z7,z8});
fx3 = subs(Sol.theta2dd, {x,xd,theta1,theta1d,theta2,theta2d,theta3,theta3d}, {z1,z2,z3,z4,z5,z6,z7,z8});
fx4 = subs(Sol.theta3dd, {x,xd,theta1,theta1d,theta2,theta2d,theta3,theta3d}, {z1,z2,z3,z4,z5,z6,z7,z8});

fx1 = simplify(fx1);
fx2 = simplify(fx2);
fx3 = simplify(fx3);
fx4 = simplify(fx4);

fx = [fx1 fx2 fx3 fx4];
```

File: `math_model_linear.m`

Description: Symbolic computation of the linearized state-space model by using the mathematical model from `math_model_pendulum.m`. Returns the expressions of jacobian matrices **A** and **B**.

Listing 2 – `math_model_linear.m`

```
syms u1 z1 z2 z3 z4 z5 z6 z7 z8;
feq1 = z2;
feq2 = subs(fx1, {x,xd,theta1,theta1d,theta2,theta2d,theta3,theta3d}, {z1,z2,z3,z4,z5,z6,z7,z8});
feq3 = z4;
feq4 = subs(fx2, {x,xd,theta1,theta1d,theta2,theta2d,theta3,theta3d}, {z1,z2,z3,z4,z5,z6,z7,z8});
feq5 = z6;
feq6 = subs(fx3, {x,xd,theta1,theta1d,theta2,theta2d,theta3,theta3d}, {z1,z2,z3,z4,z5,z6,z7,z8});
feq7 = z8;
feq8 = subs(fx4, {x,xd,theta1,theta1d,theta2,theta2d,theta3,theta3d}, {z1,z2,z3,z4,z5,z6,z7,z8});

A = jacobian([feq1; feq2; feq3; feq4; feq5; feq6; feq7; feq8], [z1 z2 z3 z4 z5 z6 z7 z8]);
B = jacobian([feq1; feq2; feq3; feq4; feq5; feq6; feq7; feq8], u1);

save('state_space_symb.mat','A','B')
```

File: `LQR_controller.m`

Description: The function verifies if the system is controllable at a certain operating configuration **z** and returns the gain matrix **K**. It uses the expressions of **A** and **B** from `math_model_linear.m` in order to find the linearized state-space model.

Listing 3 – `LQR_controller.m`

```
function K=LQR_controller(z,u)
  %%%%%%%% LINEARIZATION OF THE SYSTEM %%%%%%%%
  % Loading state-space model and parameters
  load('state_space_symb.mat')
  l1 = 1; l2 = 1; l3 = 1; % length of the links
  m1 = 1; m2 = 1; m3 = 1; % masses at the end of each link
  M = 1; % mass of the cart
  g = 9.8;
  u1 = u;

  % Linearizing around the following configuration
  z1 = z(1); z2 = z(2);
  z3 = z(3); z4 = z(4);
  z5 = z(5); z6 = z(6);
  z7 = z(7); z8 = z(8);

  A_lin = eval(A);
  B_lin = eval(B);
```

```
19
20    C = zeros(4,8);
21    C(1,1) = 1;
22    C(2,3) = 1;
23    C(3,5) = 1;
24    C(4,7) = 1;
25    D = 0;
26
27    %%%%%%%%%%%%%%% CONTROLABILITY OF THE LINEARIZED SYSTEM %%%%%%%%%%%%%%%
28    Co = ctrb(A_lin,B_lin);
29    r_Co = rank(Co);
30
31    if r_Co == 8
32      fprintf('The system is controllable. Rank of Co: %d \n',r_Co);
33    else
34      fprintf('The system is not controllable. Rank of Co: %d \n',r_Co);
35    end
36
37    %%%%%%%%%%%%%%%%%% LQR CONTROLLER %%%%%%%%%%%%%%%%%%
38    R = 1;
39    Q = C'*C;
40    Q = diag([1 0 1 0 1 0 1 0]);
41
42    [K,P,E] = lqr(A_lin,B_lin,Q,R);
43  end
```

File: `sim_generate_motion.m`

Description: This script simulates a falling triple pendulum with initial conditions `z0` in order to generate a trajectory. It calls the function `triple_pendulum_ODE.m` which defines the system of differential equations.

Listing 4 – `sim_generate_motion.m`

```
1  %%%%%%%%% DEFINITIONS OF THE PARAMETERS %%%%%%%%%
2  k1 = 0.3; k2 = 6.5; % constants for the PD controller
3  param = struct('l1',1,'l2',1,'l3',1, ... % length of the links
4         'm1',1,'m2',1,'m3',1,'M',1,.... % masses
5         'g',9.8);
6  u=0;
7
8  %%%%% SETTING PARAMETERS FOR THE ODE SOLVER %%%%%
9  init_t = 0;
10 final_t = 15;
11 dt = 0.01;
12 N = (final_t-init_t)/dt;
13 t_span = init_t:dt:final_t-dt;
14
15 %%%%%%%%%%%%% INITIAL CONDITIONS %%%%%%%%%%%%%%%%
16 z0 = [0 0 0 -0.4 0 -0.6 0 -0.6]';
17
18 zhistory1 = zeros(N,8);
19 uhistory1 = zeros(N,1);
20 t_history = zeros(N,1);
21
22 %%%%%%%%%%%%%%%%%% ODE SOLVER %%%%%%%%%%%%%%%%%%
23 options = odeset('abstol',1e-9,'reltol',1e-9);
24
25 zprec = z0;
26 u = 0;
27
28 for i =1:N
29     clc
30     fprintf('%d / %d',i,N);
31     [t1,z1] = ode113(@triple_pendulum_ODE, [0 dt 0.2], ...
32         zprec,options,u,param);
33     u = -k1*z1(2,1)-k2*z1(2,2);
34     uhistory1(i) = u;
35     t_history(i) = t1(2);
36     zhistory1(i,:) = z1(2,:);
```

```
    zprec = z1(2,:);
    if z1(2,1)>=4 || z1(2,1)<=-4
        fprintf('ODE113 terminated, out of bounds');
        break
    end
end
u0 = 0;
zhistory1 = [z0'; zhistory1];
zhistory1(end,:) = [ ];

save('trajectory_history.mat','zhistory1','uhistory1','t_history')
```

File: `triple_pendulum_ODE.m`

Description: MATLAB function defining the system of differential equations. The dynamics were derived in Equation 10 and from `math_model_pendulum.m`. *Note:* lines 24, 26 28 and 30 are shown partially in Listing 5.

Listing 5 – `triple_pendulum_ODE.m`

```
function dz = triple_pendulum_ODE(t,z,u,param)
    load('motion_equations.mat','fx');
        z1 = z(1);
        z2 = z(2);
        z3 = z(3);
        z4 = z(4);
        z5 = z(5);
        z6 = z(6);
        z7 = z(7);
        z8 = z(8);
        u1 = u;

        l1 = param.l1;
        l2 = param.l2;
        l3 = param.l3;
        m1 = param.m1;
        m2 = param.m2;
        m3 = param.m3;
        M = param.M;
        g = param.g;

        % These equations are derived from the file Symb_Development_3DOF
        dz1 = z2;
        dz2 = -(144*m2^2*u1*cos(2*z3 - 2*z5) - 144*m3^2*u1 - 240*m2^2*u1 + [...]
        dz3 = z4;
        dz4 = -(24*(6*m2^2*u1*cos(z3 - 2*z5) - 6*m3^2*u1*cos(z3) - 4*g*m2^3*sin(z3) - [...]
        dz5 = z6;
        dz6 = (24*(12*m2^2*u1*cos(z5) - 6*m3^2*u1*cos(2*z3 - z5) - 12*m2^2*u1*cos(2*z3 - z5) + [...]
        dz7 = z8;
        dz8 = (36*m2^2*u1*cos(2*z3 - z7) - 108*m2^2*u1*cos(2*z5 - z7) - 36*m2^2*u1*cos(z7) + [...]

        dz = [dz1 dz2 dz3 dz4 dz5 dz6 dz7 dz8]';
```

File: `sim_swingup_motion.m`

Description: This MATLAB script simulates the swing-up maneuver of the triple pendulum. The initial conditions `z0` and also the control strategy can be changed.

Listing 6 – `sim_swingup_motion.m`

```
%%%%%%%%% DEFINITIONS OF THE PARAMETERS %%%%%%%%%
param = struct('l1',1,'l2',1,'l3',1, ... % length of the links
       'm1',1,'m2',1,'m3',1,'M',1,.... % masses
       'g',9.8);
k1 = 0.5; k2 = 4; % constants for the PD controller
u = 0;

```

```
load('trajectory_history.mat')

%%%%% SETTING PARAMETERS FOR THE ODE SOLVER %%%%%
init_t = 0;
final_t = 15;
dt = 0.01;
N = (final_t-init_t)/dt;
t_span = init_t:dt:final_t-dt;

%%%%%%%%%%%%% INITIAL CONDITIONS %%%%%%%%%%%%%%%%
zhistory2 = [];
uhistory = [];

zhistory1_reversed = zeros(size(zhistory1));
uhistory1_reversed = zeros(size(uhistory1));
for i = 1:size(zhistory1_reversed,1)
    zhistory1_reversed(i,:) = zhistory1(end-i+1,:);
    uhistory1_reversed(i) = uhistory1(end-i+1);
end

zhistory1_reversed(:,2) = -zhistory1_reversed(:,2);
zhistory1_reversed(:,4) = -zhistory1_reversed(:,4);
zhistory1_reversed(:,6) = -zhistory1_reversed(:,6);
zhistory1_reversed(:,8) = -zhistory1_reversed(:,8);

x0 = zhistory1_reversed(1,:)' + [0;0;0.1;0;0.1;0;0.1;0]; % adding an offset to initial conditions

%%%%%%%%%%%%%% LQR CONTROLLER %%%%%%%%%%%%%%%%
x1 = [0 0 -pi 0 -pi 0 -pi 0]';
u1 = 0;
K1 = LQR_controller(x1,u1);

%%%%%%%%%%%%%%%%%% ODE SOLVER %%%%%%%%%%%%%%%%%%
options = odeset('abstol',1e-9,'reltol',1e-9);
% [t,z] = ode113(@three_dof_arm_cart_dyn_for_ODE_up, t_span, x0, options,u,l1,l2,l3,m1,m2,m3,M,g);

xprec = x0;
for i =1:N
    clc;
    fprintf('%d / %d',i,N);
    [t1,z1] = ode45(@three_dof_arm_cart_dyn_for_ODE_up, [0 dt 0.05], xprec,options,u,param);
    zhistory2 = [zhistory2; z1(2,:)];
    if i<= 900
        u = -K1*(z1(2,:)-zhistory1_reversed(i,:))' + uhistory1_reversed(i);
    else
        u = uhistory1_reversed(i);
    end
    xprec = z1(2,:);
    if z1(2,1)>=4 || z1(2,1)<=-4
        fprintf('ODE113 terminated, out of bounds');
        break
    end
end

zhistory2 = [x0'; zhistory2];
zhistory2(end,:) = [ ];
```

References

Dominique Bonvin. ME-221 Systèmes dynamiques. 2014.

Roland Büchi. *State Space Control, Lqr and Observer*. July 2010.

Tobias Glück, Andreas Eder, and Andreas Kugi. Swing-up control of a triple pendulum on a cart with experimental validation. *Automatica*, 49(3):801 – 808, 2013. ISSN 0005-1098. doi: http://dx.doi.org/10.1016/j.automatica.2012.12.006. URL http://www.sciencedirect.com/science/article/pii/S000510981200605X.

M. K. Gupta, K. Bansal, and A. K. Singh. Stabilization of triple link inverted pendulum system based on lqr control technique. In *International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014)*, pages 1–5, May 2014. doi: 10.1109/ICRAIE.2014.6909204.

G.A. Medrano-Cerda. Robust stabilization of a triple inverted pendulum cart. *Int. J. Control*, 68(4):849–865, 1997.

K.J. Åström and K. Furuta. Swinging up a pendulum by energy control. *Automatica*, 36(2):287 – 295, 2000. ISSN 0005-1098. doi: http://dx.doi.org/10.1016/S0005-1098(99)00140-5. URL http://www.sciencedirect.com/science/article/pii/S0005109899001405.