FRA501 Class Project: 1-Dimension Jumping Robot

Chawakorn Chaichanawirote, Kanut Thummaraksa,
and Noppasorn Larpkiattaworn

December 16, 2019

We model and simulate the dynamics and control of a 1-dimension Jumping robot with a linear actuator.

Robot Model

Continuous Dynamics

The robot consists of two links; a linear actuator with mass m_a and a moving thrust rod m_r. The bottom of the rod is connected to a massless spring, with a stiffness of k and a drag coefficient b. Figure 1 shows the system diagram.

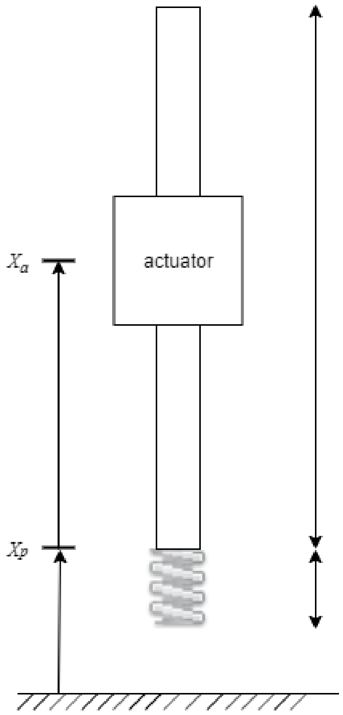

Figure 1: The jumping robot.

The joint space of the robot is defined by

$$\vec{q} = \begin{bmatrix} x_p \\ x_a \end{bmatrix} \tag{1}$$

We use Lagrangian dynamics to model the system's behaviour. Note that for convenience's sake, we will consider the spring force as an external force, as the links in motion must be rigid bodies

The Lagrangian $\mathcal{L}$ of a mechanical system is defined by a the difference between its kinetic and potential energy

$$\mathcal{L} = K - P \tag{2}$$

$$\mathcal{L} = \frac{1}{2}m_a(\dot{x_a} + \dot{x_p})^2 + \frac{1}{2}m_r\dot{x_p}^2 - m_a g(x_a + x_p) - m_r g(x_p + \frac{l_r}{2}) \tag{3}$$

The Euler-Lagrange equations are then

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\vec{q}}} - \frac{\partial L}{\partial \vec{q}} = Su \tag{4}$$

where, S is the selection matrix of the input u. For this robot, $S = \begin{bmatrix} 0 & 1 \end{bmatrix}$

The input force given by the actuator is $u = F_a$. From (4) and the Lagrangian, we have

$$\begin{bmatrix} m_a + m_r & 0 \\ 0 & m_a \end{bmatrix} \begin{bmatrix} \ddot{x_p} \\ \ddot{x_a} \end{bmatrix} + \begin{bmatrix} m_a + m_r \\ m_a \end{bmatrix} g = \begin{bmatrix} 0 \\ 1 \end{bmatrix} F_a \tag{5}$$

With no external force acting on the system, (5) represents the dynamics of the robot in mid-air. Once the spring touches the ground($x_p < l_s$)and starts to compress, we can add its external force $f_s = -(k(l_s - x_p) + b\dot{x_p})$ to the system via its transposed Jacobian $J_s^T = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ to get

$$\begin{bmatrix} m_a + m_r & 0 \\ 0 & m_a \end{bmatrix} \begin{bmatrix} \ddot{x_p} \\ \ddot{x_a} \end{bmatrix} + \begin{bmatrix} m_a + m_r \\ m_a \end{bmatrix} g = \begin{bmatrix} 0 \\ 1 \end{bmatrix} F_a - \begin{bmatrix} 1 \\ 0 \end{bmatrix} (k(l_s - x_p) - b\dot{x_p}) \tag{6}$$

(6) represents the dynamics of the system when the robot is on the ground. This can be generalized into the robot equation

$$M\vec{a} + G = S\vec{u} + J_s^T f_s \tag{7}$$

Discrete States

The robot's actuation of x_a has a limit of $[0, l_r]$, at the limits, we can classify the sytem's behaviour into four constrained states. At this point, we introduce a discrete valued state variable $x_d \in \{0, 1, 3, 4, 5\} \subset \mathbb{Z}$ that encodes the contact situations as assigned in Figure 2.

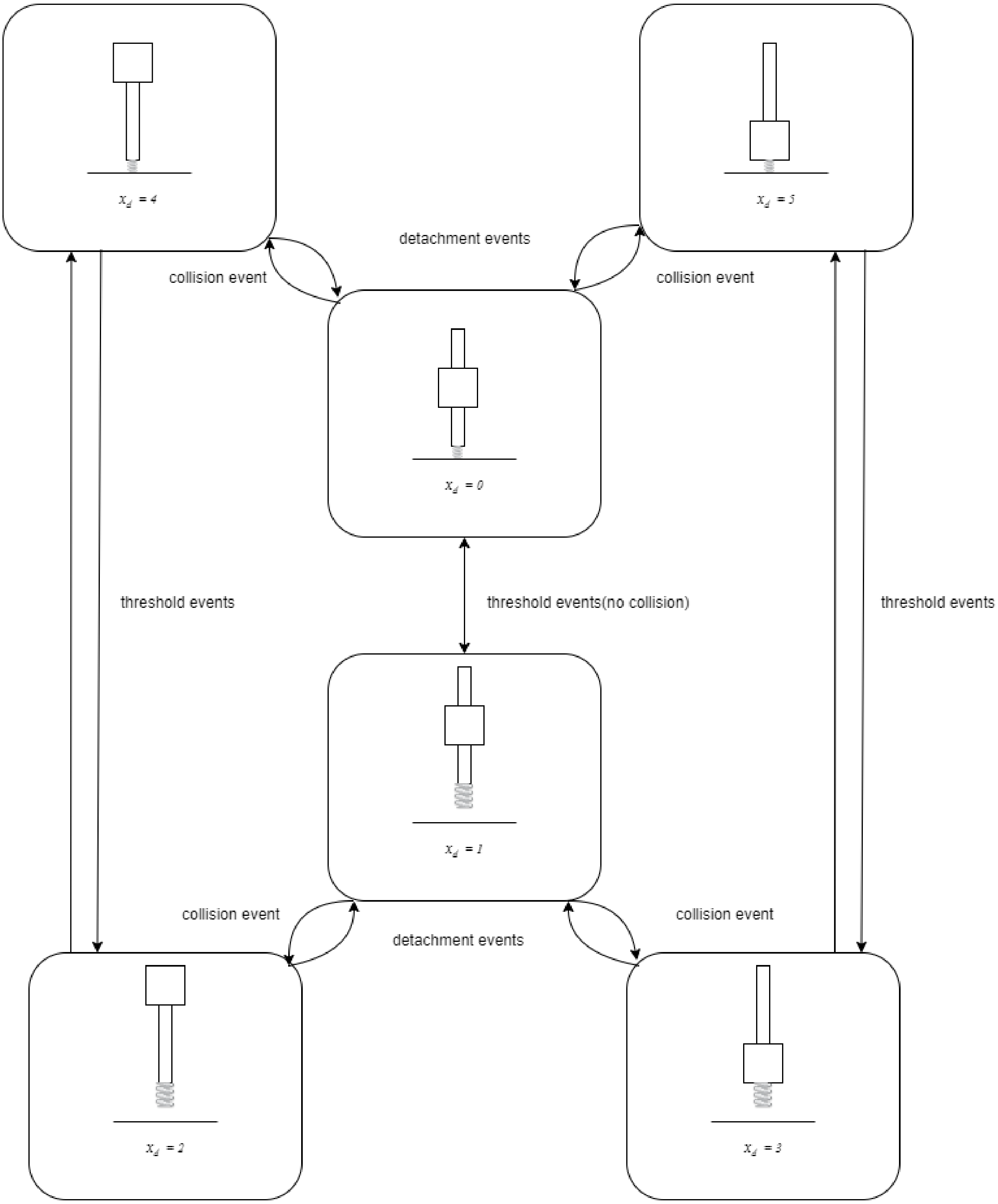

Figure 2: Transition graph of the robot.

The contact force for each contact situation is

$$F_c = (J_c M^{-1} J_c^T)^{-1} (J_c M^{-1} G - J_c M^{-1} S\vec{u} - J_c M^{-1} J_s^T f_s) \tag{8}$$

Which allows us to find the acceleration

$$\vec{a} = M^{-1}(-G + S\vec{u} + J_s^T f_s + J_c F_c) \tag{9}$$

The Jacobian for all contacts are $J_c = \begin{bmatrix} 0 & 1 \end{bmatrix}$. Note that f_s is only present for states with $x_p < l_s$.

Next, we find the the collision force of each collision, F_i, as well as the velocity after impact of the actuator v^+

$$F_i = -(J_c M^{-1} J_c^T)^{-1} J_c v^- \tag{10}$$

$$v^+ = v^- + M^{-1}(J_c^T F_i) \tag{11}$$

With these equations, we can simulate the dynamics of the robot.

Simulation

We implement the simulation and control in MATLAB, using according to the following control diagram.

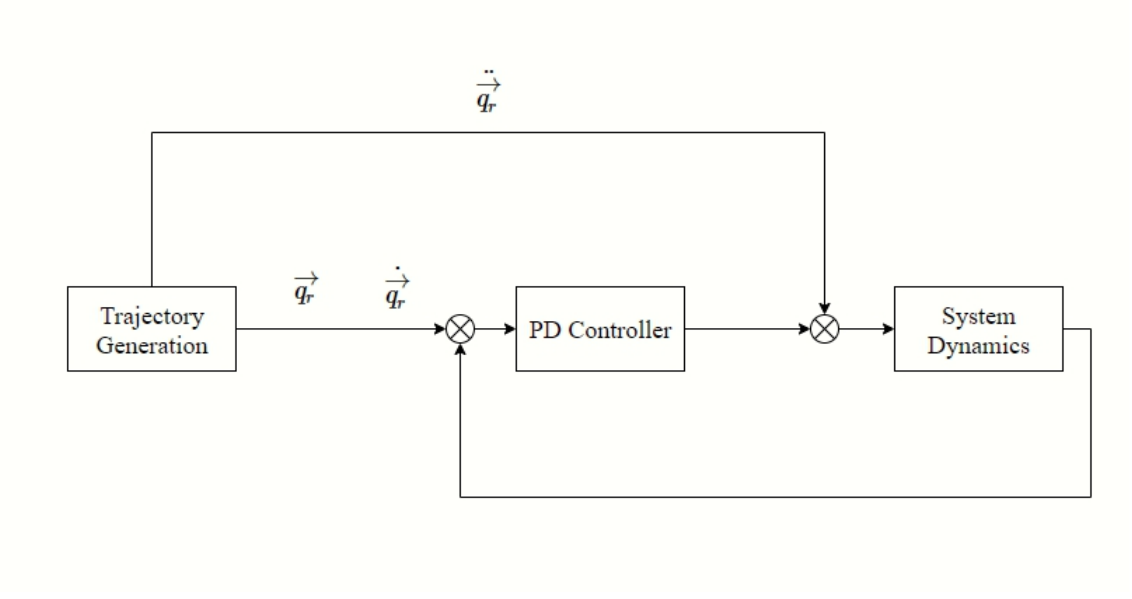

Figure 3: Control of the robot.

We run the simulation using the following input trajectory.

$$\vec{q} = \frac{l_r}{2}(2\pi t) + \frac{l_r}{2}$$

$$\dot{\vec{q}} = 2\pi\cos(2\pi t)$$

$$\ddot{\vec{q}} = -4\pi^2(\frac{l_r}{2})\sin(2\pi t)$$

The PD constants are $K_p = 1000, K_d = 10$

The system is simulated for total time $T = 30$ seconds. The initial values and constants are as follows(all units are SI):

$$x_a = 0.25$$

$$x_p = 10.2$$

$$x_d = 1$$

$$g = 9.80665$$

$$m_r = 0.2$$

$$m_a = 1$$

$$l_s = 0.2$$

$$l_r = 0.5$$

The results of the simulation are shown in Figure 4.

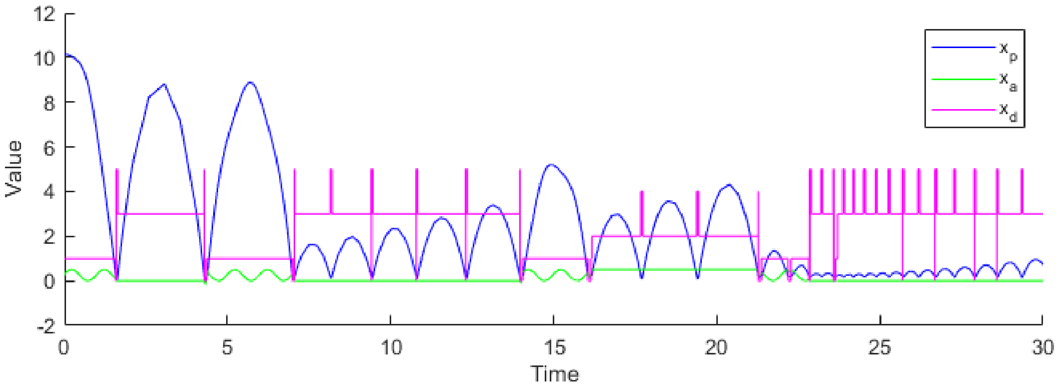

Figure 4: The result of the simulation.

Notice that the robot achieves a jumping motion, while the values for x_a are still within its bounds.