# Simulating Jack in a Box – ME-314 Final Project

**Brief Description of the Project:**

In this project, I simulated the dynamics of a planar jack-in-a-box system. I simulated and animated the dynamics of the multi-body system in time, with external forces applied on the box and impact occurs between the two rigid bodies (the jack and the box's walls).

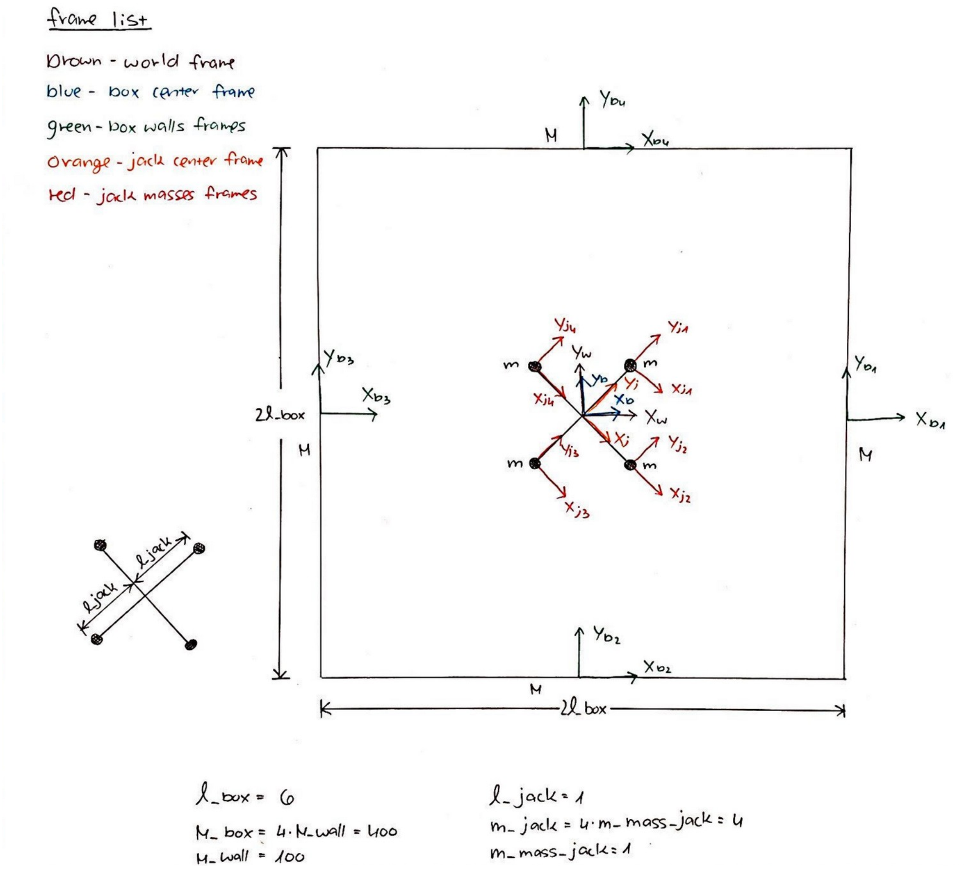
Both bodies in the planar systems are in gravity (in the -y direction of the world frame). In addition, two external forces are applied to the box – the first is an equal and opposite force to gravity (to ensure the box stays at the same position), and the second is a torque to rotate the box. The applied torque has a sinusoidal form to make the box rotate back and forth in a constant frequency.

The system can be define using the following configuration valuables:

$$q = [\ x_{\text{box}},\ y_{\text{box}},\ \theta_{\text{box}},\ x_{\text{jack}},\ y_{\text{jack}},\ \theta_{\text{jack}}\ ]^T$$

In the projects, both the jack and the box started from rest, in the initial position of (0, 0, 0) relative to the world frame. The world frame is fixed, while the two bodies can translate and rotate freely in the plane. In addition, I added 8 extra frames on the center of each mass to help simulate the impact's dynamics – 4 frames on the masses of the jack and 4 frames on the center of each wall.

You can see the different frames in the following drawing, that shows the modeling of the system:



A drawing of the system

Note that in the initial state, the frames of the world, the box, and the jack are aligned. In the drawing, I rotated the jack by 45° to get a better view on the frames.

The second view of the jack is for dimensions purposes.

## The Rigid-Body Transformations:

In order to transform between frames, I used several rigid-body transformations. The transformations where calculated using the standard form of rigid-body transformation:

$$g(x, y, \theta) = \begin{matrix} [\cos(theta), & -\sin(theta), & 0, & x], \\ [\sin(theta), & \cos(theta), & 0, & y], \\ [\quad 0, & 0, & 1, & 0], \\ [\quad 0, & 0, & 0, & 1] \end{matrix}$$

Transformation between the fixed world frame to the two body frames:

- Transformation between world to box – $g_{w\text{-}b}(x_{box}, y_{box}, \theta_{box})$ – contains rotation and translation.
- Transformation between world to jack – $g_{w\text{-}j}(x_{jack}, y_{jack}, \theta_{jack})$ – contains rotation and translation.

Transformation between the body frames and the mass frames:

- Transformation between box center to box wall – $g_{b\text{-}bi}$ – contains translation only (where i is the wall's number, according to drawing).
- Transformation between jack center to jack mass – $g_{j\text{-}ji}$ – contains translation only (where i is the mass's number, according to drawing).

In order to find the transformation between the world frame to every mass frame, I used matrix multiplication:

- Transformation between world to box wall – $g_{w\text{-}bi} = g_{w\text{-}b} \cdot g_{b\text{-}bi}$ (where i is the wall's number, according to drawing).
- Transformation between world to jack's mass – $g_{w\text{-}ji} = g_{w\text{-}j} \cdot g_{j\text{-}ji}$ (where i is the mass's number, according to drawing).

In order to find the transformation between every mass frame of the box to every mass frame of the jack, I used matrix multiplication:

- Transformation between box mass to jack mass – $g_{bi\text{-}jk} = g_{w\text{-}bi}^{-1} \cdot g_{w\text{-}jk}$ (where i is the wall's number and k in the mass's number, according to drawing).

  To calculate the inverse of the transformation matrix, I wrote a function called mat_inverse(g), Which gets a transformation matrix and returns its inverse.

You can see all the transformation calculations on the attached code.

## Process Explanation:

Before starting any calculations, I defined the **external forces** applied on the system. I didn't want the box to "fall" with gravity (for animation purposes), so I applied an equal and opposite force in the $y_w$ direction. Moreover, I applied sinusoidal torque on the box to rotate it back and forth in a constant frequency.

I defined the torque to be:

$$F_{\theta\text{-}box} = k \cdot \theta_{d\text{-}box}$$

$$k = 30{,}000$$

$$\theta_{d\text{-}box} = \sin(2\pi t/5)$$

where k in the amplitude of the force, and $\theta_{d\text{-}box}$ is the desired sinusoidal form of the force.

In order to compute the **Euler-Lagrange equations**, I calculated the transformation matrices describe above and used them to calculate the velocities and the inertia matrices of the rigid-bodies:

The bodies velocity were calculated using the bodies transformations (for every body):

$$V = \text{unhat}(g_{w\text{-}body}^{-1} \cdot (g_{w\text{-}body}^{-1})')$$

The inertia matrices were calculated using the bodies mass and inertia (for every body):

$$I = \begin{bmatrix} m, & 0, & 0, & 0, & 0, & 0 \\ 0, & m, & 0, & 0, & 0, & 0 \\ 0, & 0, & m, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0, & 0, & 0 \\ 0, & 0, & 0, & 0, & 0, & J \end{bmatrix}$$

Where m is the body's mass and J is the body's inertia. In a planar system, $J_x$=$J_y$=0.

Then, I calculated the kinetic and potential energy of the system to get the Lagrangian:

$$KE = 1/2 \cdot \Sigma \, (V_i^T \cdot I_i \cdot V_i)$$

$$PE = \Sigma \, (g \cdot m_i \cdot y_i)$$

$$L = KE - PE$$

Then, I took a derivative of the Lagrangian with respect to the configuration vector q to get the left side of the E-L equations, and compared that to the force vector:

$$(\partial L/\partial q')' - \partial L/\partial q = F$$

To compute the impacts, I defined 16 **impact constraints**. The impact constraints were the distance between the jack's mass to the wall, for every wall and mass – when the distance is 0, an impact occurs.

For the right/left wall, the constraints depended only on the x axis:

$\varphi_{b1/b3\text{-}ji} = g_{b1/b3\text{-}ji}[x]$ (The x coordinate of the distance, where i is the jack's mass number)

For the top/bottom wall, the constraints depended only on the y axis:

$\varphi_{b2/b4\text{-}ji} = g_{b1/b3\text{-}ji}[y]$ (The y coordinate of the distance, where i is the jack's mass number)

To compute the **impact equations**, I calculated the Hemiltonian:

$$H = \partial L/\partial q' \cdot q' - L$$

Then, I calculated the impact equations for every one of the possible impacts (16 possibilities):

$$(\partial L/\partial q')^{+} - (\partial L/\partial q')^{-} = \lambda \cdot \nabla\varphi$$

$$H^{+} - H^{-} = 0$$

To check if an impact occurs, I defines a function called "impact_condition()", that gets the current state, the phi matrix (combines the 16 impact constraints we calculated) and a threshold for impact (from which distance two bodies considered to be in impact with each other), and returns if there is an impact in the system. If an impact is detected, the function also returns the impact constraint's number.

To update the system configuration after impact, I defined another function called "impact_update()", that gets the current state and the impact equations, and returns the updated configuration after impact for the relevant variables (updates the velocities of the jack and the box for the impacted frames).

The impact_condition() function is called every iteration of the simulation to check for impacts. The impact_update() function is called only when an impact detected.

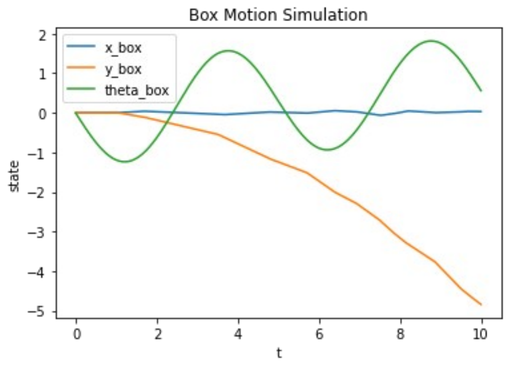
**Simulation:**

For the simulation, I wrote a dynamics function, that gets the current state and time stamp, and returns the time-derivative of the state.

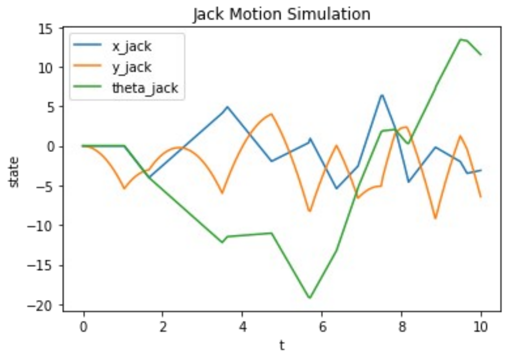
In the simulation process, I called a simulation function, that gets the dynamics function, an initial state, the time conditions (dt = 0.01 for 10 seconds), and an integration function. My initial state is all zeros (the two bodies stats from rest on (0,0) in the world frame), besides the box's angular velocity – which starts in the middle of the force amplitude.

The simulation function check for impacts using the impact – condition() function every time step dt, and if an impact is detected, the impact – update() function is called in order to update the configuration.

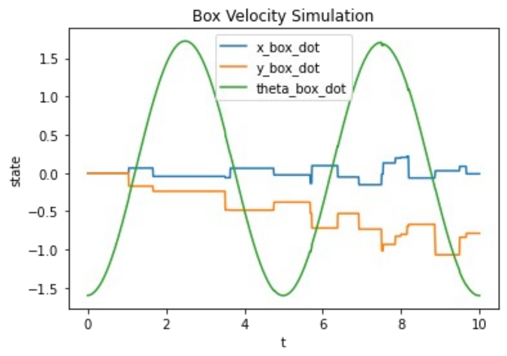
For visualization, I simulated the system and plotted 4 graphs for the box and jack configurations and the box and jack velocities:



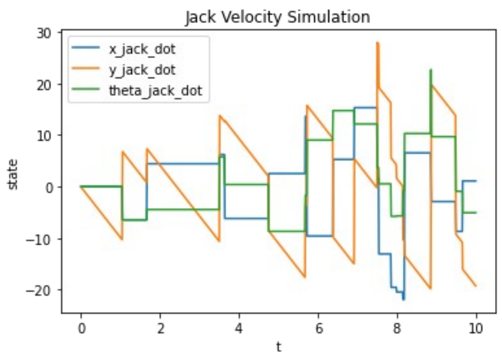
The box configuration



The jack configuration



The box velocity



The jack velocity

From those graphs, we can detect the sinusoidal change in the $\theta_{\text{box}}$ configuration and velocity. We can barely see the effect of the gravity because of the impacts, but without impact we could clearly see the $y_{\text{jack}}$ decreasing.

We can detect the impacts in the graphs by the instant changes in the curves. We can see that the motion graphs are continuous – after impact, the jack and box stay in the same place and change the motion direction. In contrast, the velocities curves are on continuous, and we can see “jumps” when the impacts occurs – as we expected.

We can see that the jack’s total mass is smaller that the box’s total mass, so the impacts’ effect on its motion and velocity is greater.