Ecole Centrale de Nantes

Humanoid Robots Report Lab 2

Roberto Canale

December 2, 2020

Contents

List of Figures

1 Introduction

The goal of this laboratory is to model a compass-walking robot walking, similarly to the first lab. The robot is considered in single support, but then the second leg swings and performs a step. The known quantities are:

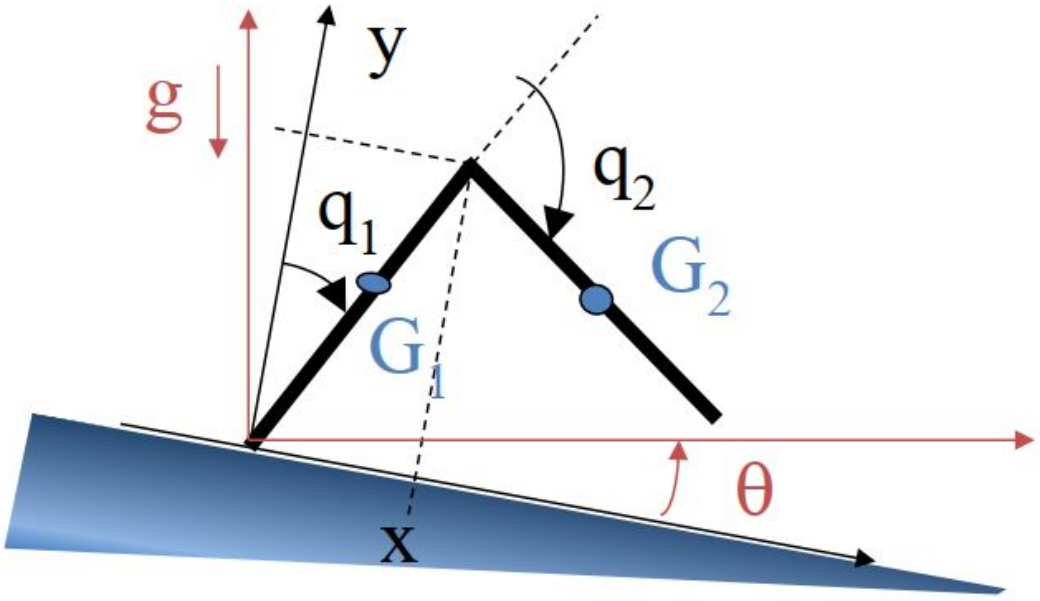

Figure 1: Compass Robot

- Link Length $l = 0.8m$
- Link Mass $m = 2kg$
- Inertia $I = 0.1kgm^2$
- Center of Mass (G1, G2)
- distance from the hip $S = 0.5m$
- Gravity $g = 0.8m * s^{-1}$
- Slope Angle $\theta = rad$

In this report, we present the steps followed to obtain the required results and visualization implemented with **MATLAB** files. A structure of the periodic passive motion **MATLAB** code can be found in the Appendix. We created a dynamic model of the robot in single support, a reaction force model an impact model.

2 Simulation of the Passive Gait

To run this part and check that the calculations are correct, it is sufficient to run the "*Main.m*" file that comes along with the report. The state of the robot is:

$$z = \begin{bmatrix} q_1 \\ q_2 \\ \dot{q_1} \\ \dot{q_2} \end{bmatrix} \tag{1}$$

and the initial conditions are, for the first case:

$$\begin{bmatrix} q_1 \\ q_2 \\ \dot{q_1} \\ \dot{q_2} \end{bmatrix} = \begin{bmatrix} 0.1860rd \\ 2.7696rd \\ -1.4281rd/s, \\ 0.3377rd/s \end{bmatrix} \tag{2}$$

The simulation has been done with the *Ode45* function. To properly implement it, it is required to properly set the options using the "*PEvents.m*" function, with the conditions specified in the assignment. Here, the value is equal to to the height of the swing tip if x>0.1, 1 otherwise; direction is = 0 and is_terminal_value= 0. Secondly, the function requires the derivation of the states, which is performed in the function "*SS_ passif.m*". The starting point is **x0**=[0 0] here, we need to extract then acceleration of the states, which is possible to do from the first lab and the "*function_ dyn.m*", using the formula:

$$\ddot{q} = A^{-1}[-H - Q] \tag{3}$$

Then, we perform the visualization of the first leg swing via the "*animation.m*", which just computes lines between the hip(green star) and the feet (red stars). Then, we need to change the leg of support for the next swing. Our new initial state is the **ze** extracted from the integration. This can be done by relabelling the states, according to the function "*relabel_ equation.m*", which performs the following angle transformations, which have been deduced geometrically from the intersection of parallel lines and segments:

$$\begin{bmatrix} q_{1n} \\ q_{2n} \\ \dot{q_{1n}} \\ \dot{q_{2n}} \end{bmatrix} = \begin{bmatrix} -\pi + q_1 + q_2 \\ -q_2 \\ \dot{+}\dot{q_1} + \dot{q_2} \\ -\dot{q_2} \end{bmatrix} \tag{4}$$

According to the guide, we need now to calculate the state before impact, calculate the velocity after impact, change the labels and then go back to step 1 to simulate the next half step. This can be done by implementing the impact model of Lab1:

$$\begin{bmatrix} A_1 & J_R^T \\ J_R^T & A_1 \end{bmatrix} \begin{bmatrix} \dot{x}^+ \\ \dot{y}^+ \\ \dot{q}_1^+ \\ \dot{q}_2^+ \\ I_{R_{2x}} \\ I_{R_{2y}} \end{bmatrix} = \begin{bmatrix} A_1 \\ 0_{2x4} \end{bmatrix} \begin{bmatrix} \dot{x}^- \\ \dot{y}^- \\ \dot{q}_1^- \\ \dot{q}_2^- \end{bmatrix} \tag{5}$$

We already have the states before impact, and we ca easily compute $\dot{x}$ and $\dot{y}$. we can then find the after impact vector by:

$$\begin{bmatrix} \dot{x}^+ \\ \dot{y}^+ \\ \dot{q}_1^+ \\ \dot{q}_2^+ \\ I_{R_{2x}} \\ I_{R_{2y}} \end{bmatrix} = \begin{bmatrix} A_1 & J_R^T \\ J_R^T & A_1 \end{bmatrix}^{-1} \begin{bmatrix} A_1 \\ 0_{2x4} \end{bmatrix} \begin{bmatrix} \dot{x}^- \\ \dot{y}^- \\ \dot{q}_1^- \\ \dot{q}_2^- \end{bmatrix} \tag{6}$$

From here, after impact velocities can be extracted, while the displacement after the swing is given by **ze**: we can define all the required vectors to ultimately obtain:

$$z_{after} = \begin{bmatrix} q_1^+ \\ q_2^+ \\ \dot{q}_1^+ \\ \dot{q}_2^+ \end{bmatrix} \tag{7}$$

and by implementing 4, on 7, we can obtain a new set of starting condition for *Ode45*, namely **z_new**. This corresponds to the starting point of the next swing, which is then going to be animated again.

3 Periodic Motion

To perform a periodic motion of the robot, we first need to study its Poincaré return map, detailed in Figure 2.

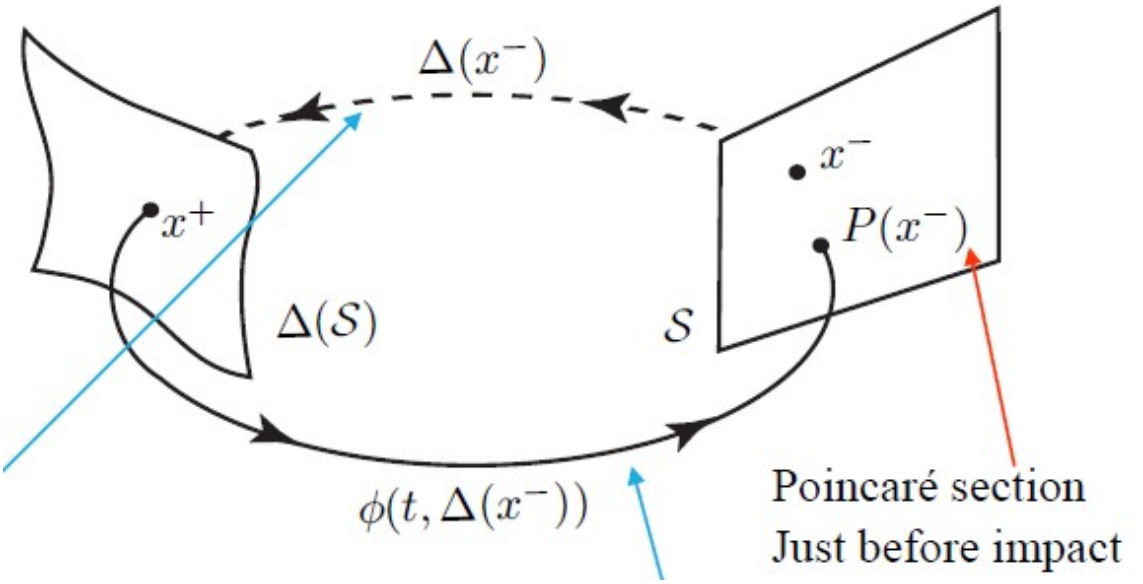

Figure 2: Poincaré Return Maps

First we define the condition of the robot in double support. This means defining a relation between q_1 and q_2, when both legs are touching the ground. We have two cases, one when both legs are still at the same position and one when the swinging leg is close to the ground, or almost on impact. Geometrically, the relation is then defined by:

$$\pi = 2q_1 + q_2 \tag{8}$$

With equation, we can reduce our state vector X of the Poincaré map to:

$$X = \begin{bmatrix} q1 \\ \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \tag{9}$$

This way, it is possible to define a Poincaré return map. This function contains many elements from the first map, and it includes both the *Dynamic Model in SS* and the *Impact Model*. Like before, the cycle is created by implementing the algorithm described in the assignment. From a starting point, complete one swing and find the state before impact. With the impact model, find the state after impact, perform a joint relabelling (as now the **swing leg** becomes the **support leg** and vice-versa, and lastly extract the new state variables from integration as defined. Then, a fixed point by optimization technique has been fixed, namely $X^* = Periodic(X^*)$ and a step simulation over a slope has been created. The functions **optimset** and **fsolve** have been used to solve the function and create a passive gait in 2 cases of different initial conditions, with different steps. The function to minimize, found in "*periodic_function*" is:

$$f = X_P eriodic(X) \tag{10}$$

It is possible to observe the result of the passive motion in Figure 3.

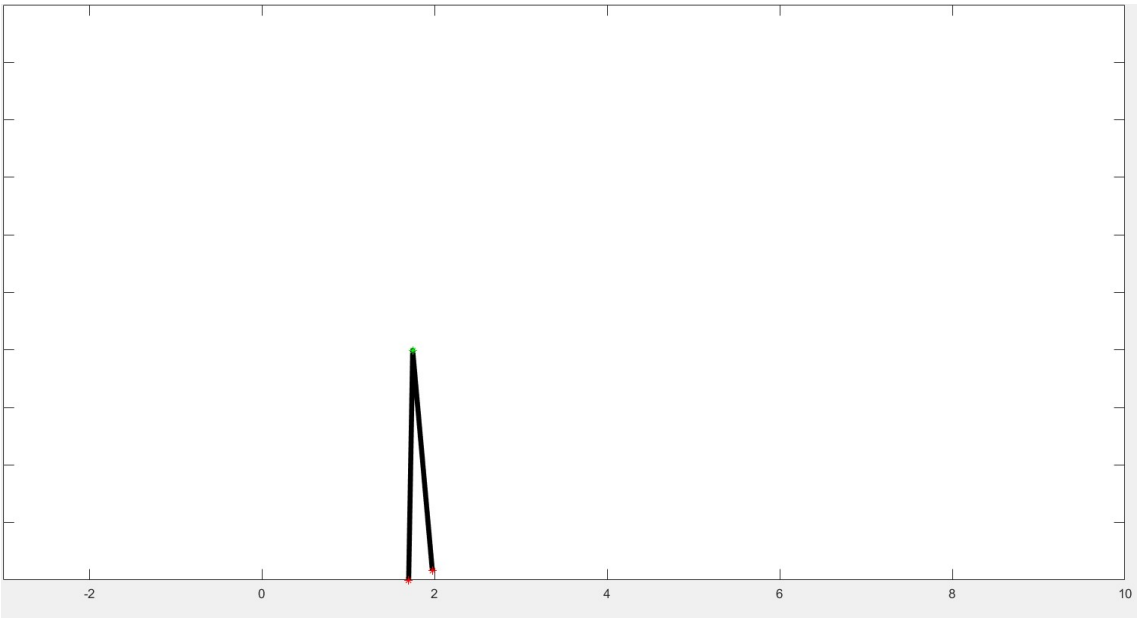

Figure 3: Periodic Walking

Finally, phase maps (Displacement against Velocity) of the 2 cases have been computed, for a different number of steps. Figures 4 to 7 show the phase maps of the 2 different gaits.

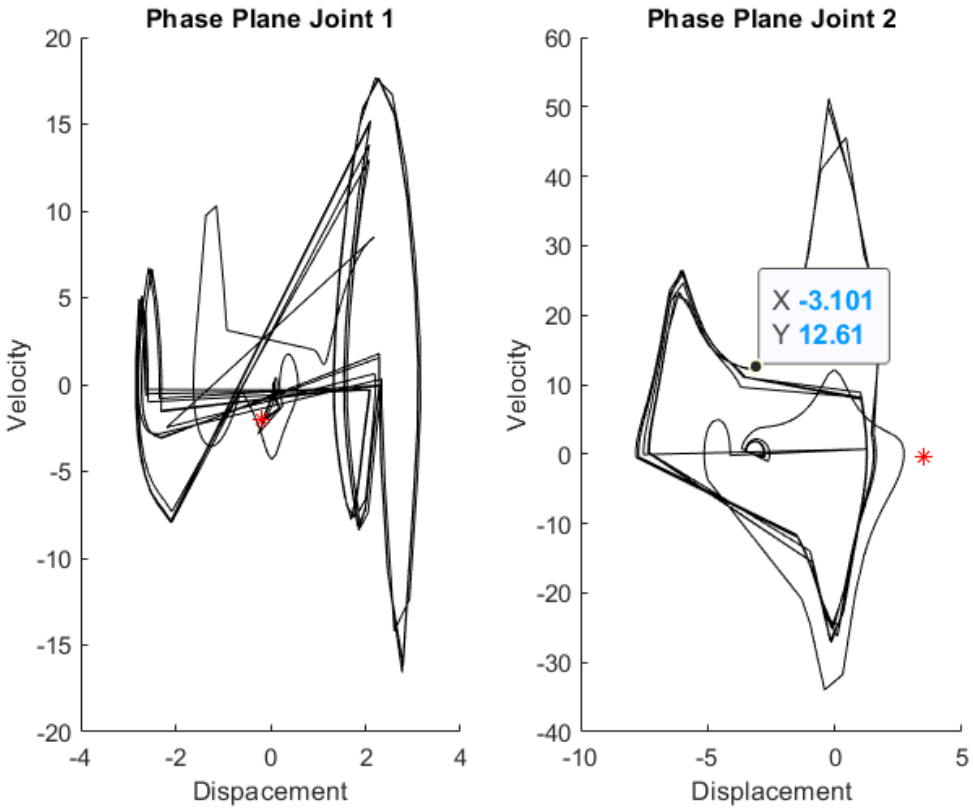

Figure 4: Motion 1, 30 Iterations

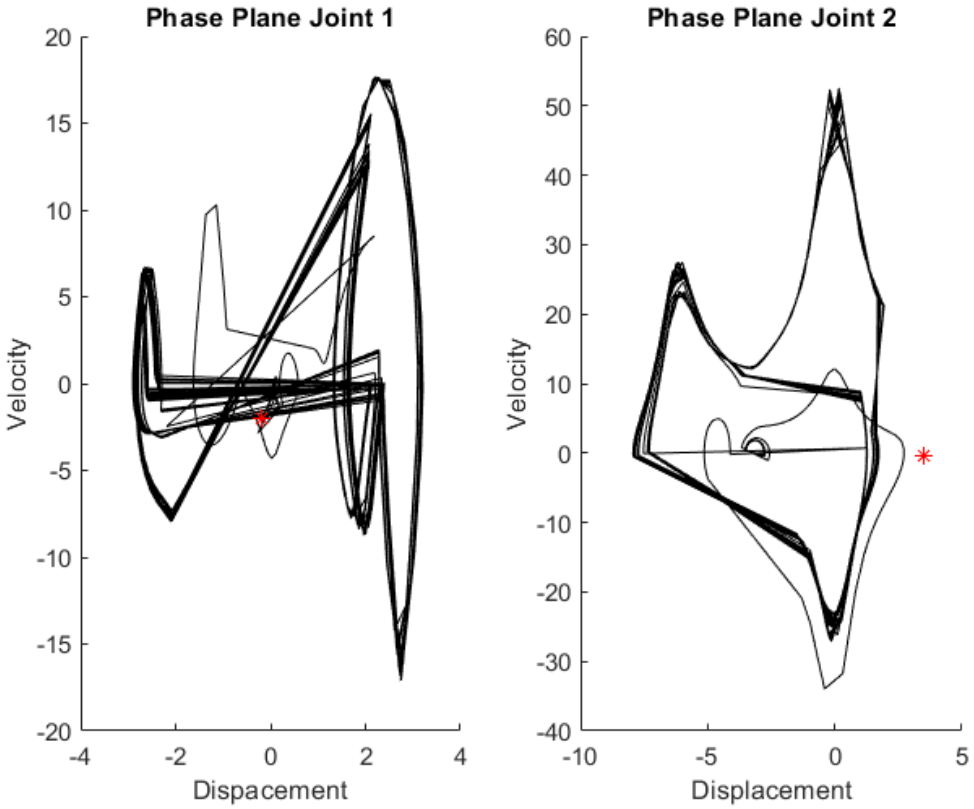

Figure 5: Motion 1, 100 Iterations

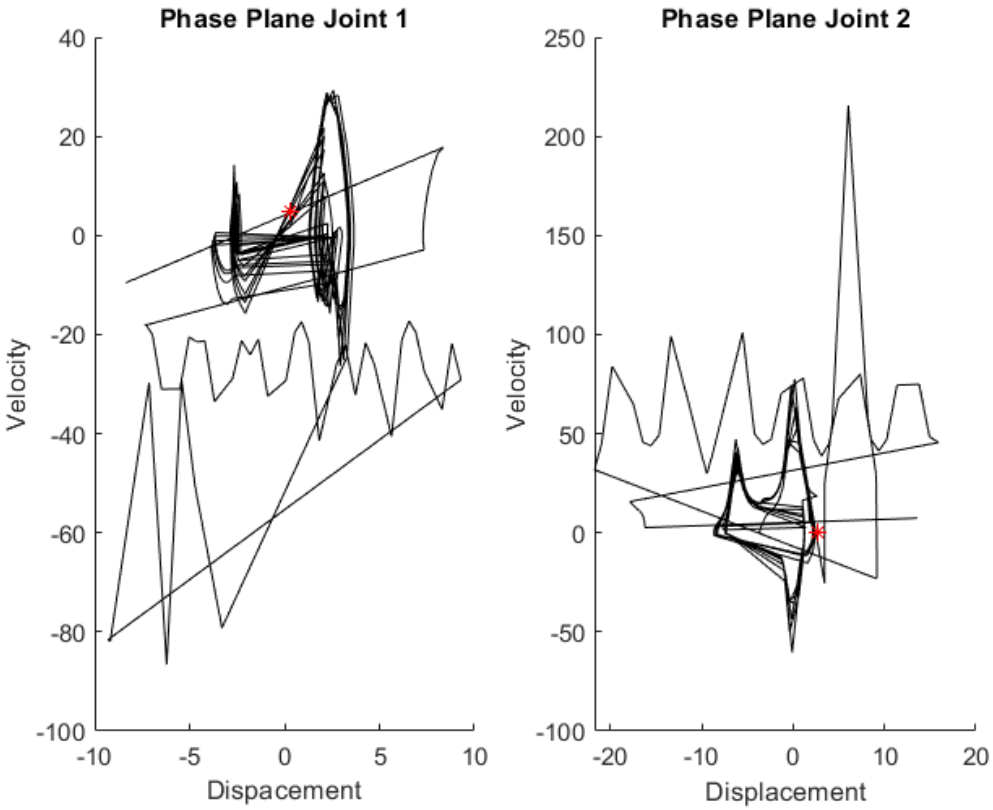

Figure 6: Motion 2, 30 Iterations

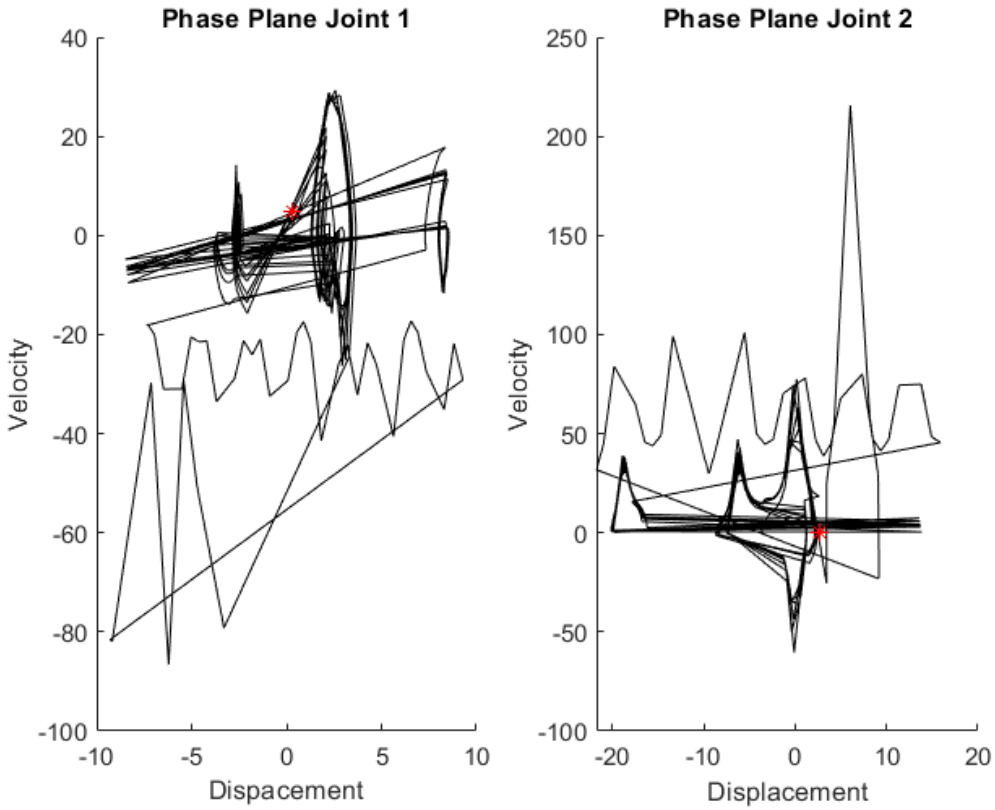

Figure 7: Motion 2, 100 Iterations

4 Stability

The third part of the laboratory deals with the stability of passive gait. This has been calculated by setting the errors on the position and velocity to $e_p = 0.5^{-4}$ and $e_v = 0.5^{-3}$, and calculating the Jacobian for each state varies according to the formula:

$$J_i = \frac{Periodic(X^* + \delta x_i e_i) - Periodic(X^* - \delta x_i e_i)}{2\delta x_i} \tag{11}$$

The stability is checked by calling the "*check_ stability.m*" function, where the Jacobian of each state variable is found, stacked, and then the norm of its eigenvalues (found via the function *eig()*) is found. If <1 the gait is STABLE, otherwise the gait is unstable. As we can see from the code and the phase planes, both gaits are unstable.

Appendix

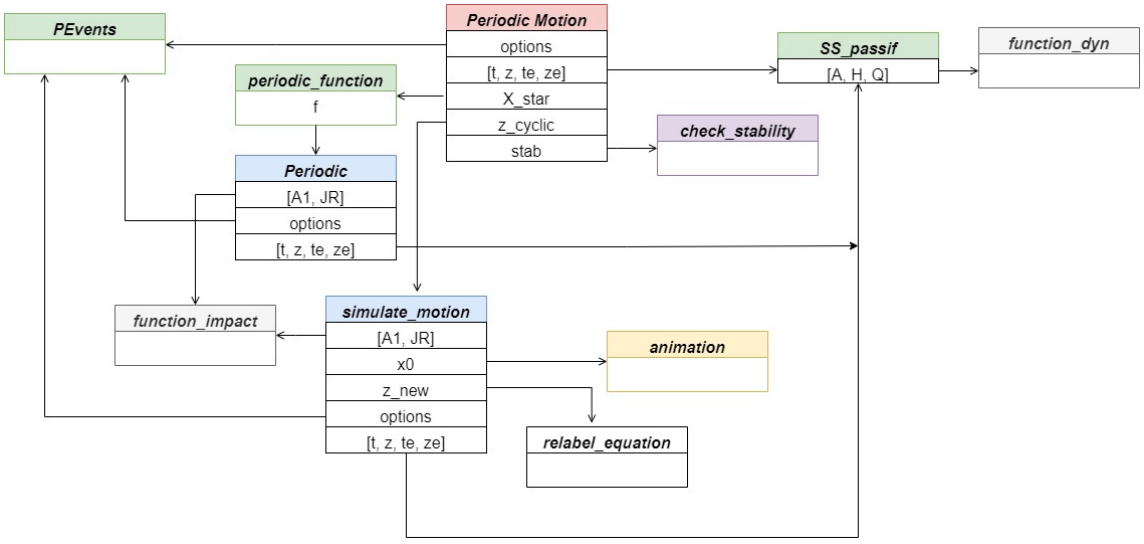

Figure 8: MATLAB Code Structure