École Centrale de Nantes

Humanoid Robots

Laboratory Report 2

Isabella-Sole Bisio

December 26, 2020

Contents

1 Introduction **3**

2 Simulation of the passive gait **4**
2.1 Simulation of one single support phase 4
2.2 Simulation of several steps 4

3 Periodic Motion **6**
3.1 Poincaré return map 6
3.2 Periodic passive motion on a slope 6

4 Stability check **7**

5 Conclusions **7**
5.1 Periodic motion 1 7
5.2 Periodic motion 2 8

List of Figures

1 Compass-walking robot 3
2 Compass-walking robot 5
3 Periodic motion 1, 30 steps 7
4 Periodic motion 1, 50 steps 8
5 Periodic motion 2, 30,50 and 200 steps 8

1 Introduction

In this second laboratory we were asked to perform the three following tasks on the compass-walking robot 1 :

1. Simulation of the passive gait
2. Periodic motion
3. Stability analysis

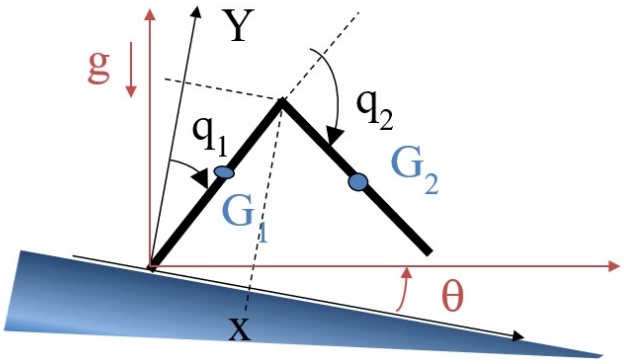

Figure 1: Compass-walking robot

As it is possible to see from the image, the robot is composed by two identical links of length l and mass m, having an inertia I and a distance between the centre of mass G and the hip of s. Furthermore, the robot is walking along a slope, having θ angle; it is possible in fact to observe two frames. The black one is the reference frame R_0, which is attached to the ground with slope. In order to check the correctness of the three required tasks, is necessary to run the **main.m**, **main_periodic1.m**, **main_periodic2.m** files; the resulting motions will be displayed along-with the Displacement-Velocity correlation graphs. The known quantities that have been used to test the code are the following:

- l = 0.8 m;
- m = 2 Kg;
- g = 9.8 Kg s^{-2};
- $\theta = 2\pi/180$;

The values of I and s will vary in the two different cases that we have to analyse. In fact in the first case:

- I = 0.1 Kg m^2
- s = 0.5 m

whereas in the second case:

- I = 0.05 Kg m^2
- s = 0.45 m

2 Simulation of the passive gait

To check the results of this section, run **main.m** file. In this section, the simulation of the gait of the compass walking robot has been performed. There are two main phases: support on the leg1 and impact of the leg2 on the ground. After that a cycling motion can be performed with a restarting of the first phase, this time with leg2 and so on. No torques are applied to create the motion, but it is just present a slope to create passively the robot movement. In order to extract these passive values of velocities and accelerations, I will use the functions developed in the first laboratory assignment **function_dyn** and **function_impact** and, to simulate the robot walking, I will use (as suggested) the *ode45* function, that is able to compute for each time instant, the integration of velocities and accelerations to find the current value of the state z.

2.1 Simulation of one single support phase

In order to use the *ode45* function, two new functions have been built: **PEvent** and **SS_passif**. Starting with the first one, **PEvent** function is used to tell the system when one leg impacts with the ground. *VALUE*, *ISTERMINAL* and *DIRECTION* are the three outputs of this function. *ISTERMINAL* and *DIRECTION* can just be assigned to 0 or 1 whereas *VALUE* has a different value each time since it is equal to the y value; in this way when the leg touches the ground, y will be equal to 0, and same for *VALUE*, and this fact will be taken into account during performance of *ode45* function and an additional check will be performed in order to avoid that the step will stop in the (0,0) configuration. The **SS_passif** function is in charge of evaluate the derivative of the state as function of the state, defined in this way:

$$z = \begin{bmatrix} q_1 \\ q_2 \\ \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \tag{1}$$

$$\dot{z} = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} \tag{2}$$

I can use the formula:

$$A(q)\ddot{q} + H(q,\dot{q}) + Q(q,\theta) = 0 \tag{3}$$

to find the acceleration since I can use the **function_dyn** function, developed in the last lab, to derive the A, H and Q matrices. At the end of this section, thanks to the above functions, I am able to create a simulation of the first step of the walking robot, obtaining also the time instant and the corresponding state.

2.2 Simulation of several steps

Completed the first step, now I had to continue the walking using the relabeling equation; this allows to relabel the state vector in order to switch the support and the swing leg that will perform the next steps. To write the new model I have kept into account the figure 2:

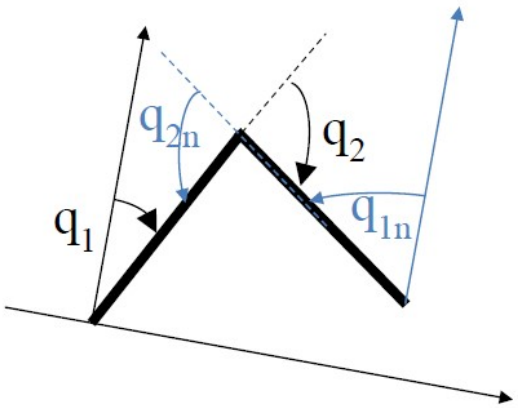

Figure 2: Compass-walking robot

From this figure I can retrieve the new state values in this way:

$$\begin{aligned} q_{1new} &= q_1 + q_2 - \pi \pm 2\pi \\ q_{2new} &= -q_2 \\ \dot{q}_{1new} &= \dot{q}_1 + \dot{q}_2 \\ \dot{q}_{2new} &= -\dot{q}_2 \end{aligned} \tag{4}$$

These equations have been then transformed into an equation mapping the new angles as a function of the old ones:

$$z_{new} = E z_{old} - \begin{bmatrix} \pi \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5}$$

with:

$$E = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & -1 \end{bmatrix} \tag{6}$$

Now using the **function_impact** build in the first assignment, I can deduce the Impact Model, knowing the velocity just before the impact. The following model creates a relation between the moment before and after the impact:

$$\begin{bmatrix} A_1 & -J_{R2}{}^T \\ J_{R2} & 0_{2x2} \end{bmatrix} \begin{bmatrix} \dot{x}^+ \\ \dot{y}^+ \\ \dot{q_1}^+ \\ \dot{q_2}^+ \\ I_{R_2x} \\ I_{R_2y} \end{bmatrix} = \begin{bmatrix} A_1 \\ 0_{2x4} \end{bmatrix} \begin{bmatrix} \dot{x}^- \\ \dot{y}^- \\ \dot{q_1}^- \\ \dot{q_2}^- \end{bmatrix} \tag{7}$$

To compute the new velocities of the new state I can just apply the inverse on the previous equation. At this point is possible to write the following algorithm to complete the two-steps motion, and then cycle to repeat it again and again:

1. For the initial state simulate single support until the impact
2. Compute state value before the impact
3. Compute the new velocities after the impact
4. Apply relabeling equation
5. Cycle back to step 1

3 Periodic Motion

In this section, I will briefly describe how I used the Poincaré return map and optimisation techniques to obtain a cyclic passive motion.

3.1 Poincaré return map

To obtain a cyclic motion I should have a fixed point on the Poincaré return map. At this point, I can retrieve the following definition to describe the relationship between q1 and q2 when the robot is in double support:

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

from which I can derive:

$$q_2 = \pi - 2q_1 \tag{9}$$

and I also can define the state vector:

$$X = \begin{bmatrix} q_1 \\ \dot{q_1} \\ \dot{q_2} \end{bmatrix} \tag{10}$$

It is now possible to define the function representing the Pointcaré return map: basically, most of the steps performed by this function were already used in the previous functions so I will skip this explanation. In conclusion, the new state value is extracted and become exactly the output of this function:

$$X_{k+1} = minimise(X) \tag{11}$$

3.2 Periodic passive motion on a slope

When the value of the new state corresponds exactly to the value of the initial state a motion can be considered cyclic; at each iteration, it returns in the same point on the Poincaré return map. Knowing that what I needed to do is to find and fix a point to make this requirement doable. I have implemented an optimization problem to find this point using the function *fsolve*.

4 Stability check

The last section is the stability check: as it was said during the lessons, a periodic motion is considered stable if, when starting from a certain state, it converges to a periodic state. Following this definition, to check the stability of the motion, an analysis of the Jacobian of the Poicaré map is required. After having found the Jacobian matrix, its eigenvalues are computed; if the norm of each eigenvalue is less then 1, the motion is stable, otherwise in unstable.

5 Conclusions

In this lab we were asked to check the evaluations made with two different pair of data:

- CASE 1: I = 0.1 kgm^2 and s = 0.5 m
- CASE 2: I = 0.08 kgm^2 and s = 0.45 m

To check the results graphically I have tried the simulation with several quantities of steps: 30, 50 and 200. In order to check the results that I am going to explain, run the **main_periodic1.m** and **main_periodic2.m** files.

5.1 Periodic motion 1

For this first case, it is possible to see from the walking animation, that the robot follows down: this is possible to be observed using the version with 50 steps. Also from the trend of the Phase map (Displacement-Velocity graph) shows clearly that the motion diverges drastically, and never recover, from the periodic state X, visualised ad a red star. Moreover, also the stability check code categorises this motion as **UNSTABLE**.

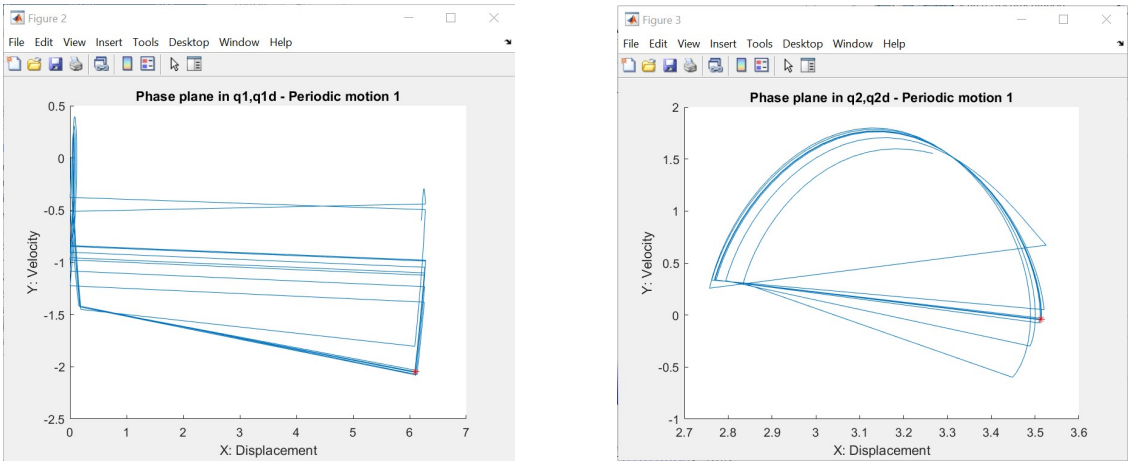

Figure 3: Periodic motion 1, 30 steps

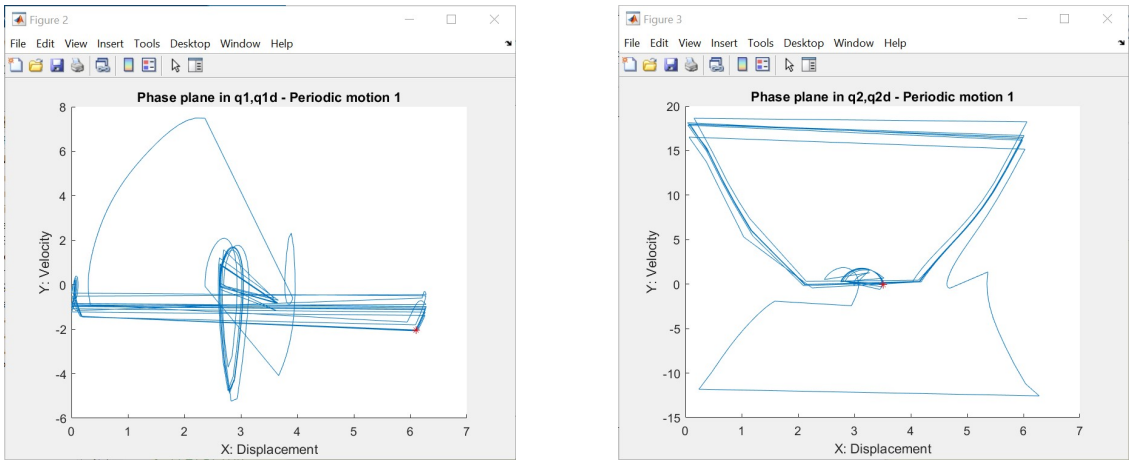

Figure 4: Periodic motion 1, 50 steps

5.2 Periodic motion 2

For this second case, it is possible to see from the walking animation, that the robot never follows down, neither in the 30, 50 or 200 steps version of the code. This leads me to think that the motion is **STABLE**. In favour of this thesis, also the trend of the Phase map (Displacement-Velocity graph) shows clearly that the motion converges, at each iteration, to the periodic state X, visualised ad a red star. Nevertheless, the stability check code categorises this motion as **UNSTABLE**. So, there exist two possibilities to interpret this second result:

- Solution 1: the motion is **STABLE** so I have reliable walking animation and phase map, whereas the stability check code is unreliable and should be written in a more detailed manner
- Solution 2: the motion is **UNSTABLE** so I have reliable stability check code, whereas I can not trust walking animation and phase map, maybe because the robot will follow down after 200 steps (the longest path that I have tried)

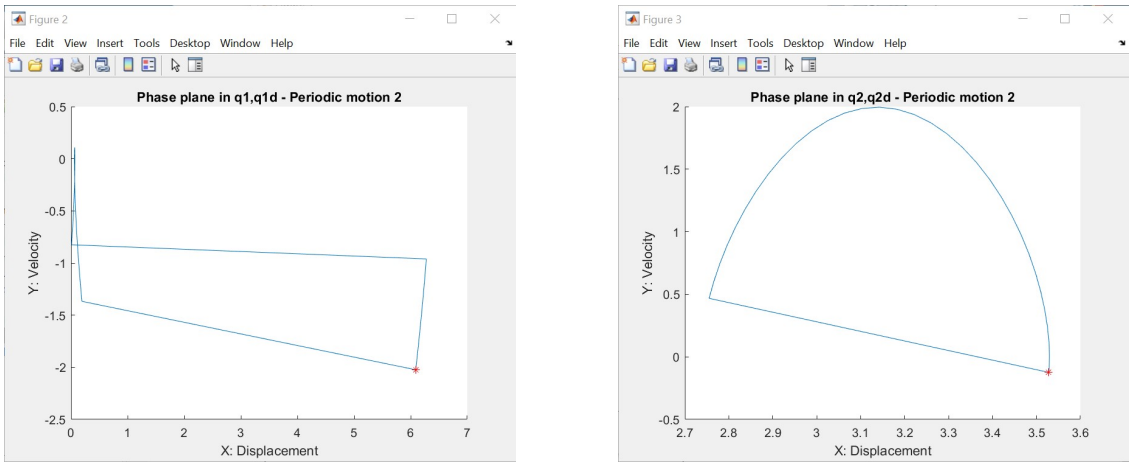

Figure 5: Periodic motion 2, 30,50 and 200 steps