Designing a Walking Controller for the Three-link Biped (Assignment 4)

Submission deadline: December 17, 2018 at 23:59 (45% of the mini-project grade)

This is the final part of the mini-project. In the last two assignments, you developed the kinematics and dynamics model of the three-link biped and you built a simulator. In this assignment, you design a walking controller which enables the three-link biped to walk. To this end you need to modify the `control.m` script and add any other auxiliary scripts needed for your controller to run. Moreover, you need to write a script `analyze.m` to quickly analyze the resulting walking gaits.

Important note: For this assignment you need to submit a full report alongside with your codes.

1. Requirements

Your controller should satisfy the following requirements:

1. The controller is able to generate a family of walking gaits (different velocities). The range of velocities the controller should be able to achieve is from either from 0.7 m/s to 1.5 m/s or from $0.4\ m/s$ to $1.2\ m/s$. Ideally, the controller can get a desired velocity and make the robot follow that desired velocity. However, it is also fine if you can show that you can choose different parameters of your controller to achieve the velocities

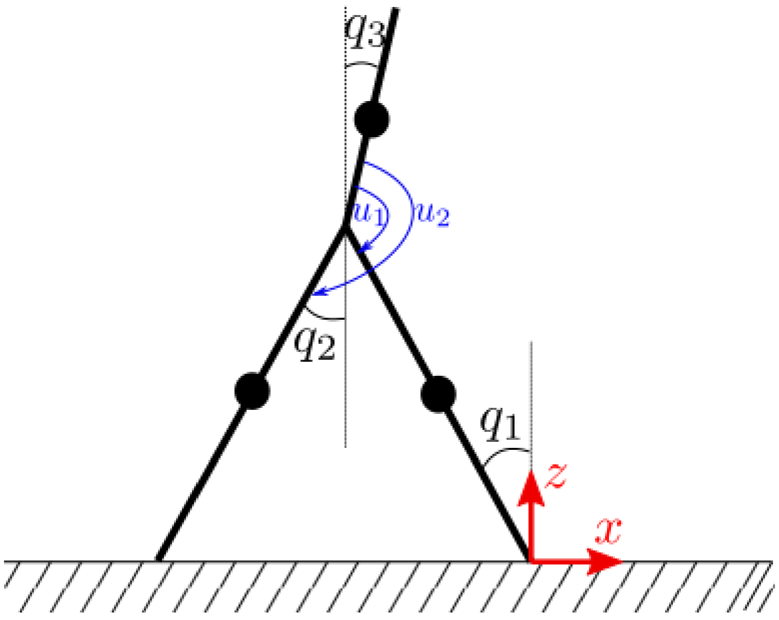
   Option 1: $0.7\ m/s, 0.9\ m/s, 1.1\ m/s, 1.3\ m/s, 1.5\ m/s$ within a maximum of 10% error.

   Option 2: $0.4\ m/s, 0.6\ m/s, 0.8\ m/s, 1\ m/s, 1.2\ m/s$ within a maximum of 10% error.
2. The robot starts from zero velocity (i.e., $\dot{q}_0 = [0; 0; 0]$). Ideally, the robot should start from zero pose as well (i.e., $q_0 = [0; 0; 0]$). If you could not achieve these two conditions at the same time, it is fine to give some initial momentum to the robot by assuming $\dot{q}_0 = [0.1; 0; 0]$ and $q_0 = [0; 0; 0]$.
3. Maximum available torque of each actuator is 30 Nm. Also, the torque signals have to be continuous (i.e., no spikes). It is fine to have spikes (less than 30 Nm) at the moment of switching the legs.
4. With the controller the robot can tolerate external perturbations (more details on this later). You are given a MATLAB script `perturbation.m`. Once you are happy with your controller, you can test its robustness to external pushes (applied during a full step) by calling this function inside `eqns.m` as you will be instructed in class. You can change the external force value. Check higher values, and report the highest force your controller was able to tolerate. The robot should at least tolerate an external force with a value of 50 N. You are welcome to update your controller as you wish to tolerate bigger external forces.
5. The controller should be able to generate energy-efficient gaits (recall the notion of Cost of Transport (CoT)). You should report the CoT of the gaits you achieved in requirement (1) above along with the control parameters you used for each gait. Which of these gaits had the lowest CoT? Can you argue why this happened?
6. If applicable, your controller can work with time-based and state-based phase variables.

2. Hints

You are free to design your own controller as long as it satisfies the conditions above. However, here are some general tips that may help you in designing your controller:

- Since this robot does not have feet, clearly ZMP-based approaches would fail here. In contrast, trajectory based approaches or the method of virtual constraints discussed in lecture 5 might be helpful.
- Note that you have only two actuators and three degrees of freedom. Hence, the robot has one degree of underactuation. It is important to decide which variables or combination of variables you would like to fully control. Think about q_1, q_2, q_3, which one of these seem more intuitive to directly control? Which one can be regarded as a "free variable"?
- It is recommended to use a function `control_hyper_parameters.m` which includes the controller parameters (e.g., desired step length, frequency if applicable)
- In case you decide to continue with virtual constraints or trajectory control, you can consider using **splines** or **Bezier** polynomials for designing your virtual constraints (or trajectories).
- It is recommended to first come up with a working version of your controller, then focus on the requirements mentioned above.
- Note that in the event function the condition of impact is slightly changed. This is to help the swing foot clear the ground (virtually).
- When designing your controller, be careful with the convention for the positive and negative signs; everything is based on a right-handed coordinate system. Look at the figure below.



3. Submission Instructions

As mentioned, you need to submit a full report alongside your codes. You need to include a README.pdf file which includes the instructions on how to run your controller. Other than `control.m` you need to include a script `analyze.m` which analyzes the resulting walking gait and plots the results, which include (but not limited to) plots of **the angles vs time, velocity of the robot vs time, displacement in each step vs step number, step frequency vs step number, torques vs time, cost of transport, and plots of $\dot{q}$ vs q for all three angles**.

You will receive more details about the submission instructions later.