# 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:

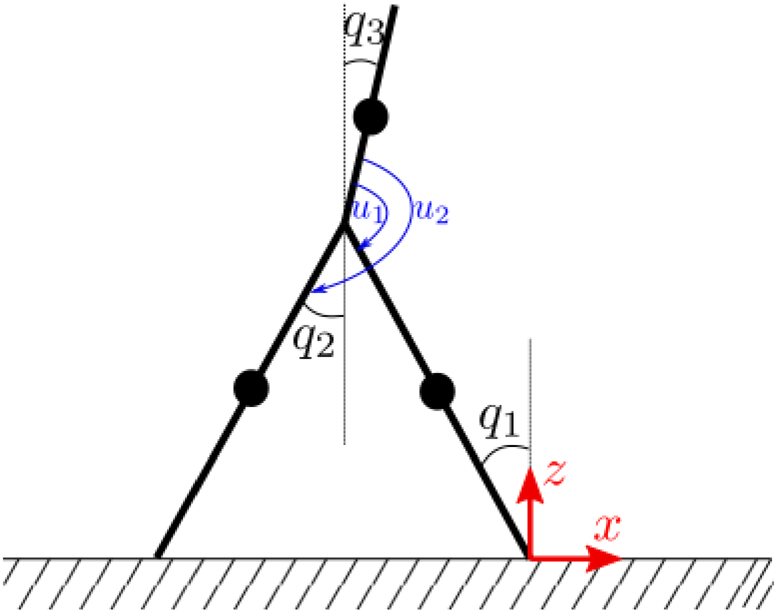
1. The controller is able to generate a family of walking gaits (different velocities).
2. The robot starts from zero velocity (i.e., $\dot{q}_0 = [0; 0; 0]$).
3. Maximum available torque of each actuator is 30 Nm. Also, the torque signals have to be continuous (i.e., no spikes)
4. With the controller the robot can tolerate external perturbations (more details on this later).
5. The controller should be able to generate energy-efficient gaits (there will be more instructions on this, recall the notion of Cost of Transport)
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.