Report of Legged Robots Project

Team Member: CUI Mingbo, ZHOU Xiao, LI Weipeng

Winter Semester 2018

1 Quick Summary

In the project, our group tried a lot of models and methods to make the robots walk at desired speed and stable gaits. Both nonlinear control (hard way) and linear control (easy way) in virtual constraints are tried and corresponding parameters are tuned. Finally, we choose the linear control as our may model implemented in our simulation since it achieves the best dynamical performance among all models we have tried.

Our model can make the robot walk stable in lower speed and conform to all the requirements. When we wish to reach a higher speed, for example, v=1.5m/s, we turn to another model that can fulfill the desired velocity. However, the torque of controller u_1 will go beyond the limit of 30N·m while other requirements can be achieved with acceptable dynamical performance.

2 Introduction

This report has shown the process of designing a controller for a simple biped waling robot. By adopting the thoughts of virtual constraints method, we have designed a virtual constraints based controller and hand-tuned pretty good hyperparameters which enable the biped robot walking in a stable gait with different velocities.

3 Designing Controller and Desired Behavior

3.1 Virtual Constraints: Linear control (easy way)

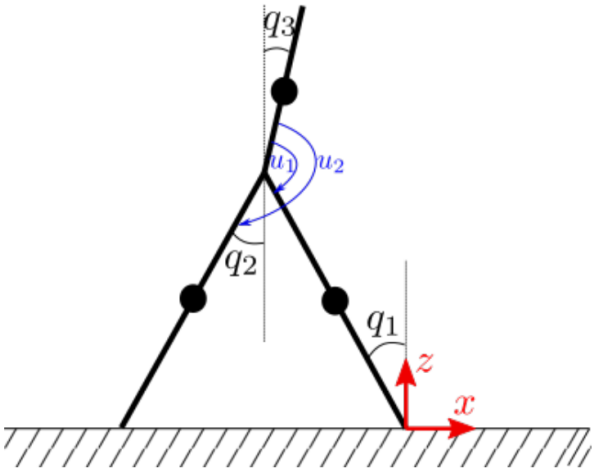

Figure 1: The description of the model

By using simple PD control to design the controller, we could derive the control as following:

$$u = -K_d(\dot{\theta_2} - \dot{\theta_1}) - K_p(\theta_2 - \theta_1) \tag{1}$$

We look for two torques: u_1, u_2.

u_2 is used for ensuring the symmetry of the two legs by the torso. It reflects the angle between the link2 and link3. The larger its value, the larger the angle between link2 and link3. Our desired behavior is "q_2 follows a constant $q_{2desired}$"
Thus we have the following equation:

$$u_2 = -K_{d,swing} \times (dq_2 + dq_1) - K_{p,swing} \times (q_2 + q_1) \tag{2}$$

Note: $q_{2desired}$ is constant so $\dot{q}_{2,desired} = 0$

u_1 is used for keeping the torso upright. It reflects the angle between the link1 and link3. The larger its value, the larger the angle between link1 and link3. Our desired behavior is "q_3 follows a constant $q_{3,desired}$"
Therefore, we have the following equation:

$$u_1 = K_{d,torso} \times dq_3 + K_{p,torso} \times (q_3 - q_{3,desired}) \tag{3}$$

Note1: $q_{3,desired}$ is constant so $\dot{q}_{3,desired} = 0$

Note2: That in the above equation, there is no negative term.
Because this is related to our choice of control and the choice of angle. In the case of lecture 5 and u_2, the bigger the q_i, the bigger the angle we want to control, the bigger the torque.
But in the case of u_1 is exactly the opposite. In this case the bigger the q_i, the smaller the angle we want to control, the smaller the torque.
So the sign of the formula coefficient at this time is just the opposite of the previous one. As a result, these two negative sign compensate so that there is no negative sign in this equation.

To conclude, we utilize the easy way to control the robotic. Our controller can use virtual constraints. We select q2 and q3 as output, and select q1 as "free variable". Although the easy way abandons two parameters about P and Q (relative to hard way), the control can be linear which is more easy and direct. We just need to adjust the parameters in the above equations to control the walking process of the robotic and enhance its stability.

3.2 Optimization for Faster Speed

The model above can help our robot achieve a stable walking gait at low speed (velocity within 0.4 m/s in our simulation) . However, if we desire that our robot could walk faster than 1.0m/s, it's difficult to let the robot reach that velocity with the same posture in the model. Therefore, we will have to change the model a bit for a faster speed.

Intuitively, when we try to walk faster or even attempt to run, we will not let our torso stay upright to achieve slow locomotion any more, instead a process of acceleration is required until the desired speed is fulfilled. Therefore, in order to get more speed in our simulation, we can't simply control the robot's torso to stay upright, but to tilt the body forward at a certain angle (just like when we run, the body will lean forward). As the velocity of robot increases,

the angle at which the body leans forward (q_3) will also increase. When the real velocity gets around our desired velocity , we wish our torso to stay in that angle so that the speed would not change a lot any more and we can achieve a faster dynamical stability.

In the acceleration process we let the robots torso to learn forward at a angle related to the angle of stance leg. Since when the robot accelerates, (q_3) does not try to track a constant angle, but change by following the change of the swing amplitude of the supporting foot(q_1). Thus we change the equation of u_1 as follows.

$$u_1 = K_{d,torso} \times (dq_3 - k \times dq_1) + K_{p,torso} \times ((q_3 - q_{3,desired}) + k \times (q_1 - q_{1,desired})) \quad (4)$$

When the velocity of our robot get around out desired velocity and the angle of q_3 reach a specific value (in our code the value corresponds to the paramter threshold), our code stops the process of acceleration and let the robot keep the angle the torso leans forward so that the robot could stay around the desired velocity. In this process, the will try to let the torso stay still and not change any more. At the same time u_1 will control the q_3 to be as close to 0 as possible. It turns out the switch of u_1 can help stabilize the torso orientation and velocity.

$$u_1 = K_{d,torso} \times dq_3 + K_{p,torso} \times (q_3 - 0) \quad (5)$$

In addition, when the robot attempts to walk stable in a high speed after acceleration, the control of u_2 should also be changed. After several simulations, we find that q_2 is supposed to track a desired value instead of making q_1+q_2=0. This will help stabilize the torso dynamics in high speed. Thus the control of u_2 should be rewrited as follows.

$$u_2 = K_{d,swing} \times (dq_2) + K_{swing} \times (q_2 - q_{2,desired})) \quad (6)$$

Our model can achieve good dynamical performance of the robot, and the robot can even walk faster than 1.6m/s. However, when the robot is walking at that speed, the value of q_3 will become quite large after the acceleration process, requiring a huge torque from u_1 to control the torso orientation. Therefore, although we can achieve a high velocity of robot, the torque produced by u_1 is beyond the 30 N·m in our model, which remains to be improved.

3.3 Other Candidate Control Method

To achieve a stable gait, we have tried to use different control methods. At the beginning, we thought that complex control like the non-linear control would give us better performance, but it turned out that it became so hard to comprise with hyperparameters and the intrinsic features of the model. Then we turned to linear model, which was easier to tune hyperparameters. However, it requires big torque to generate the target gait. The reason why this torque is so big is that we use a constant angle to serve as our desired output. To reduce the torque, we have tried to define a target curve generated by sigmoid function, which could reduce the torque a great deal. However, it turns out that we could hardly improve our velocity if we let q_2 follow the smooth curve. Finally, we give up

3.4 Parameters for Different Velocities

par	$K_{p,torso}$	$K_{d,torso}$	$K_{p,sw}$	$K_{d,sw}$	$q_{1,desired}$	$q_{2,desired}$	$q_{3,desired}$	k	switch	threshold
v=0.4	1.5	600	30	2.9	$\pi/25$	/	$\pi/30$	0.0025	0	/
v=0.6	1	1500	100	2	$\pi/15$	$-\pi/35$	$\pi/20$	0.02	1	$\pi/70$
v=0.8	1	1500	100	2	$\pi/15$	$-\pi/35$	$\pi/20$	0.02	1	$\pi/25$
v=1.0	1	1500	100	2	$\pi/15$	$-\pi/35$	$\pi/20$	0.02	1	$\pi/12$
v=1.2	1	1500	150	2	$\pi/15$	$-\pi/35$	$\pi/20$	0.03	1	$\pi/8$
v=1.5	2	400	75	1	$\pi/15$	$-\pi/35$	$\pi/20$	0.03	1	$\pi/3$

Table 1: Parameter Tuning for Different Velocities

It should be noted that in our model, the torques of u_1 will go beyond $30N \cdot m$ if the desired velocity is set more than 0.6m/s while u_2 still stays in the range. For these velocities, we remove the limit $(30N \cdot m)$ of u_1.

4 Analysis

In the analysis part, we mainly utilize the model described in part 3.1 and we will take the plots of first 100 steps for analysis.

4.1 Robustness against Perturbation

To test the robustness of our model, we added perturbation of force to test the stability. By testing we could see that our model could afford maximum $51N$ at step 50.

4.2 Analysis of q_i

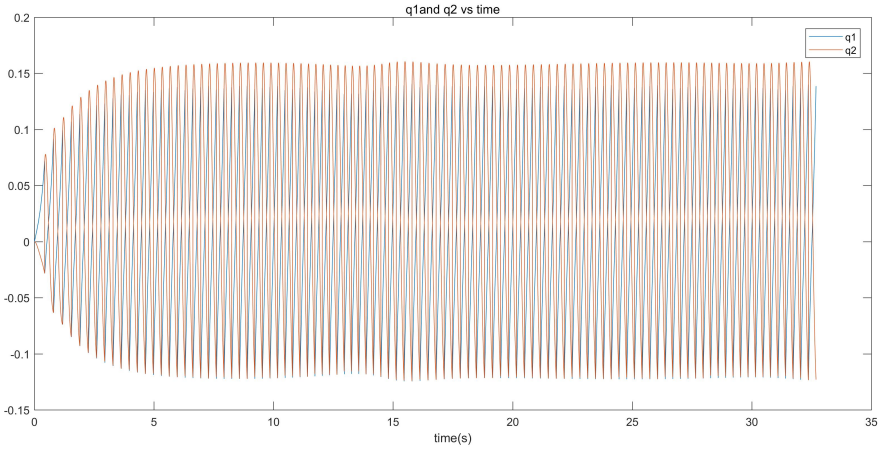

Figure 2: q1 and q2 vs time (v=0.4)

Through the Figure.2, we could find that over time the values of q_1 and q_2 show an approximately periodic variation around 0 within the set interval. Their phases always differ by an

angle value. This just reflects the fact that the biped robot constantly exchanges legs while walking.
And for q_3 we have the Figure.3

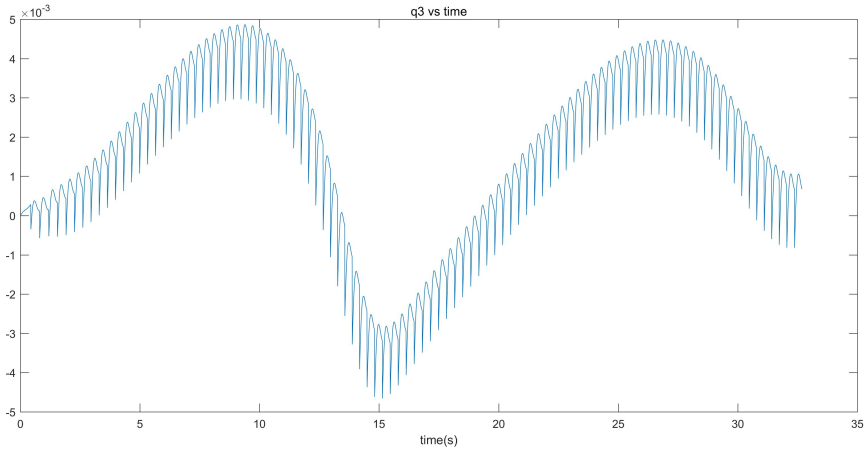

Figure 3: q3 vs time (v=0.4)

4.3 Analysis of $\dot{q_i}$

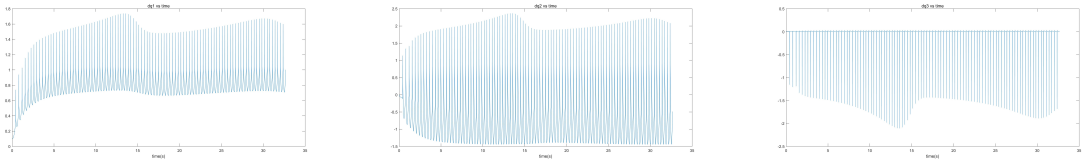

Figure 4: $\dot{q_1}(left), \dot{q_2}(middle), \dot{q_3}(right)$ vs time

Through the Figure.4 we could see the change of angular velocity during the simulation. Take the $\dot{q_i}$ for example, because we configure the q_1 as negative so q_1 will always increase in one single step(from negative to zero first and then). Therefor it is intuitively correct that all $\dot{q_1}$ stay always above the zero. Differ from $\dot{q_1}$, $\dot{q_2}$ was configured with positive, it will always decrease in the single step. Similar to what the figure has shown to us, $\dot{q_2}$ is stable at a negative value despite the spikes caused by the impact. Since the gait is slow(0.4m/s), we want to keep the torso upright to keep balance, the $\dot{q_3}$ remains as zero in the most of time if we just ignore the spikes caused by the impact.

4.4 Analysis of hip position

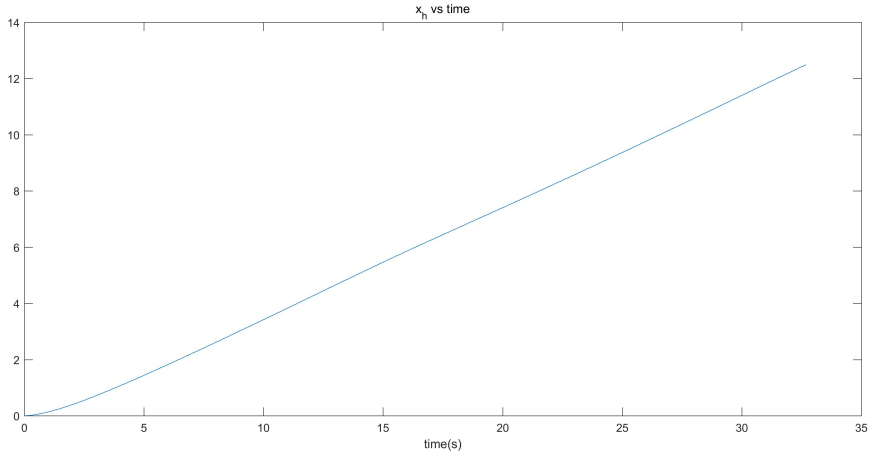

Figure 5: position of hip point vs time

The movement of hip could be seen as the average movement of our robot model, indicated by the Figure.5, it is easy to find the linear relationship between distance and time, which matches the fact that our robot is moving with constant velocity.

4.5 Analysis of hip velocity

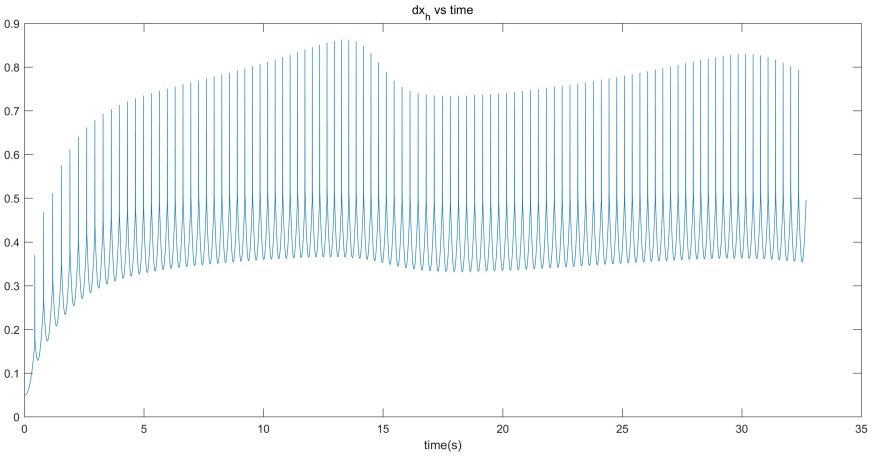

Figure 6: velocity of hip point vs time

We could use the velocity if hip serve as the velocity of our robots model. However, because the hip point will go up first and go down in each single step, so there are many oscillations around our target velocity. In addition, since there are obvious oscillations near each desired velocity, we get the value of the average in each step of our gaits.

$$v_{step} = \frac{\Delta x_h}{\Delta t_{step}} = \frac{x_{h,end} - x_{h,start}}{t_{step,end} - t_{step,start}} \tag{7}$$

According to observation in corresponding plots, after a period of acceleration, the average velocities in later steps are very close to each other around the desired value. We record the average velocities of last 10 steps for each gait.

desired velocity (m/s)	0.4	0.6	0.8	1.0	1.2	1.5
real velocity (m/s)	0.4066	0.6518	0.8372	1.0422	1.1952	1.4365

Table 2: Average Velocities in Last 10 Steps for Different Gaits

4.6 Analysis of step displacement

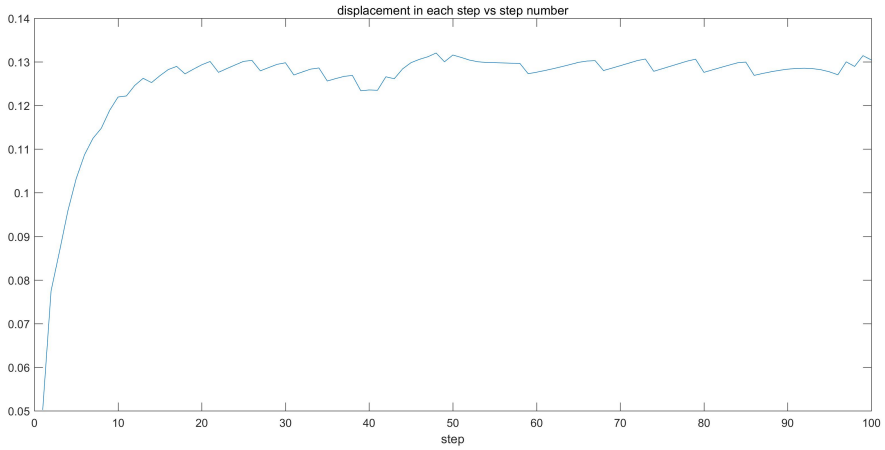

Figure 7: displacement in each step vs step number

From the Figure.7, the displacement in each step will converge to a constant, which intuitively matches to the fact that our velocity will be stable.

4.7 Analysis of step frequency

Through the Figure.8, we could find the step frequency will converges to about 3 steps/s with oscillations less than 1.

4.8 Analysis of control torques

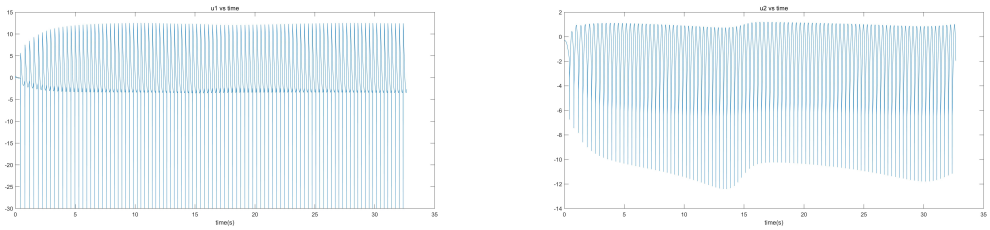

Figure 9: Analysis or control torque

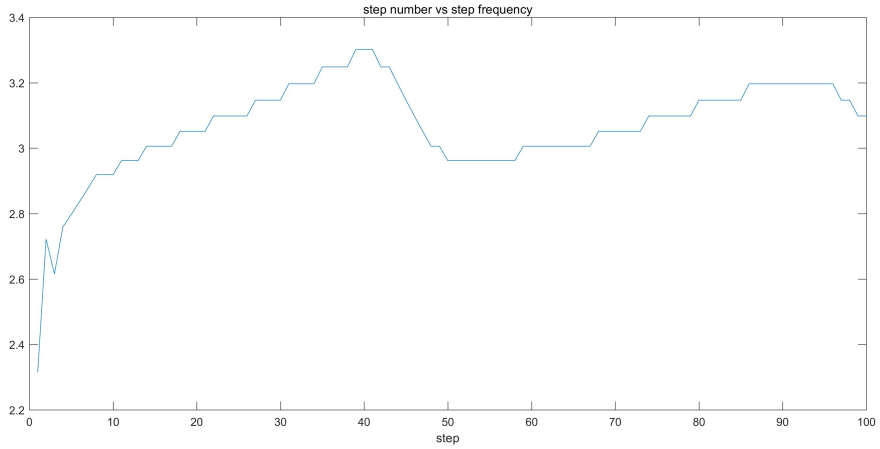

Figure 8: step frequency vs step number

Indicated by the Figure.9, there are many spikes happens when there is a step change. By saturating the torque under $30N \cdot m$, our model could fulfill the requirement of the motor.

4.9 Analysis of q_i and $\dot{q}_i$

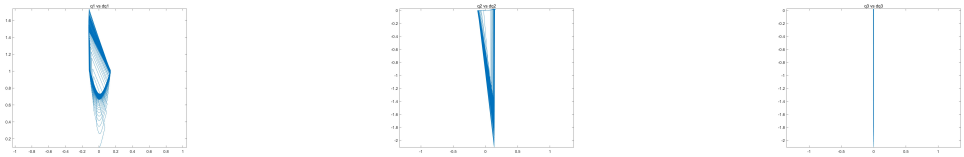

Figure 10: q_1 vs $\dot{q_1}$(left), q_2 vs $\dot{q_2}$(middle), q_3 vs $\dot{q_3}$(right)

Indicated by the Figure.10, our graph will plot the shape which is closed. It indicates that our control is stable.

4.10 Cost of Transport (COT)

Since preferred gait coincides with lowest energetic cost, so we calculate COT per step to verify that the gait we chose is working well. Let's start with the work done by torque u_1, u_2 in each step:

$$W_i = \int_0^T u_i \dot{\beta}_i dt \tag{8}$$

Where: β_1 is the angle between link3 and link1, β_2 is the angle between link3 and link2.(Figure.1) So for β we have:

$$\beta_1 = \pi + q_3 - q_1 \tag{9}$$

$$\beta_2 = \pi + q_3 - q_2 \tag{10}$$

Thus,

$$\dot{\beta}_1 = dq_3 - dq_1 \tag{11}$$

$$\dot{\beta}_2 = dq_3 - dq_2 \tag{12}$$

The energy consumption in each step is the **positive** work done by torque, so we have the formulas of COT_i about u_i

$$COT_i = \frac{\int_0^T \ max(0, u_i \dot{\beta}_i) dt}{\Delta x_h} \tag{13}$$

Since our data is discrete and cannot be integrated, we approximate this value by the summation method.

$$COT_i = \frac{\Sigma[\ max(0, u_i \dot{\beta}_i) * \Delta t)]}{\Delta x_h} \tag{14}$$

Where Δx_h is displacement of one step:

$$\Delta x_h = x_{h,endofstep} - x_{h,startofstep} \tag{15}$$

Consequently,

$$COT = COT_1 + COT_2 \tag{16}$$

So we get the Figure.11 of the COT.

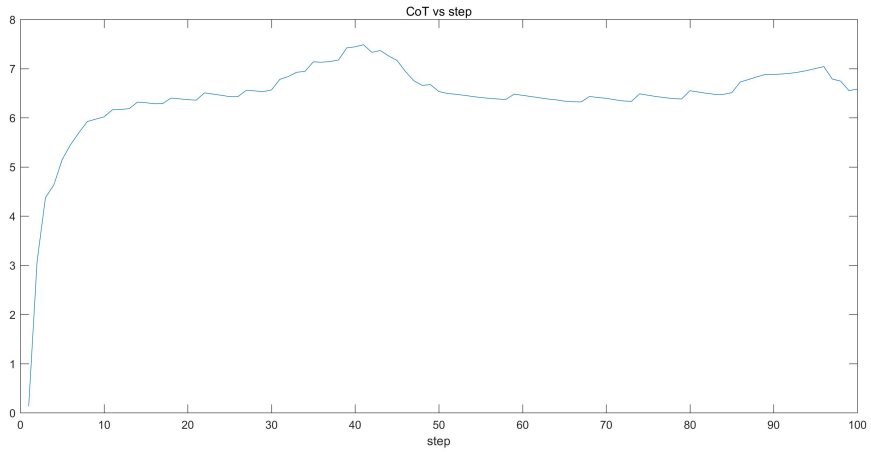

Figure 11: Cost of transport VS time (v = 0.4m/s)

You can see that in the Figure.11 the COT is small and tends to approach a fixed value as the number of steps increases. This means that the gait we chose is relatively good.

From Equation14 we can find that the value of COT is related to u_1, u_2, dq_1, dq_2 and Δx_h. If we want get a smaller COT, we should have smaller torque a smaller dq_i and a larger displacement of one step. But we can't do these three things at the same time. Because if we use less torque then we can only use smaller steps, which leads to a smaller displacement of one step. And if we use too small dq, the speed of the robot will be too small.Thus we have to adjust the parameters to find a balance

In the end we found that COT is the smallest when v=0.4m/s.

5 Model Analysis with faster speed

Our robot could achieve a set of velocity: 0.4m/s, 0.8m/s, 1.0m/s, 1.2m/s, but some of them will only be achieved by big torque of u_1. However, to make it walk in a faster speed, we have tuned many hyperparameters and changed control. Therefore we want to mention our work here. Inspired by the walking gait of human, we are likely to head our torso forward when we walk at a faster speed. Heading forward the torso could make the Center of Mass away from the center of two legs and it will be incline to the forward direction, which would give a inclination to the robot to make it move forward. The control model was introduced in section 3.2.

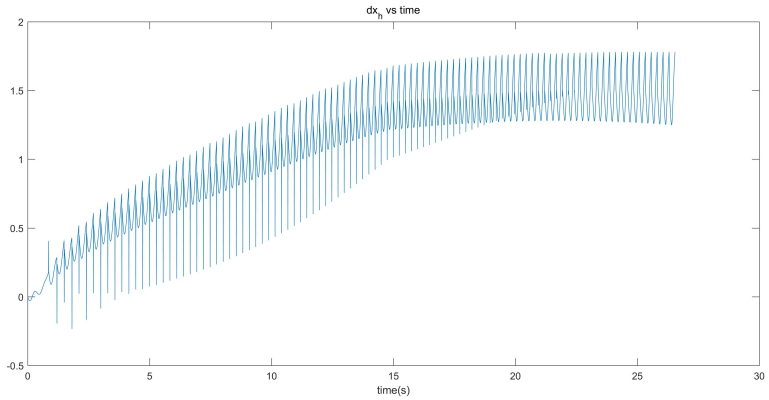

Figure 12: Velocity of model (v = 1.5m/s)

By heading forward the torso, we could achieve higher speed of 1.5m/s.

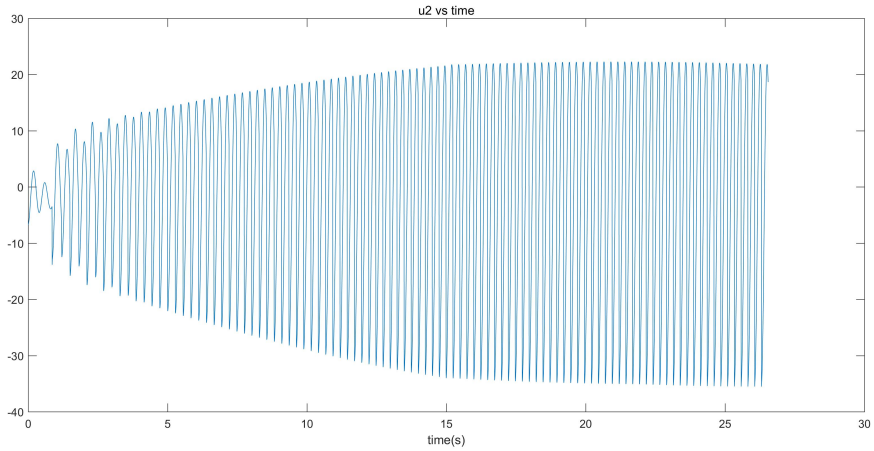

Figure 13: Torque of u_2 (v = 1.5m/s)

In the Figure.13 we could see that u_2 is pretty good because it is under 30 $N \cdot m$ without saturation.

Figure 14: Analysis of $\dot{q}$ vs q

We have pretty good stability indicated by the closed shape of Figure.14.

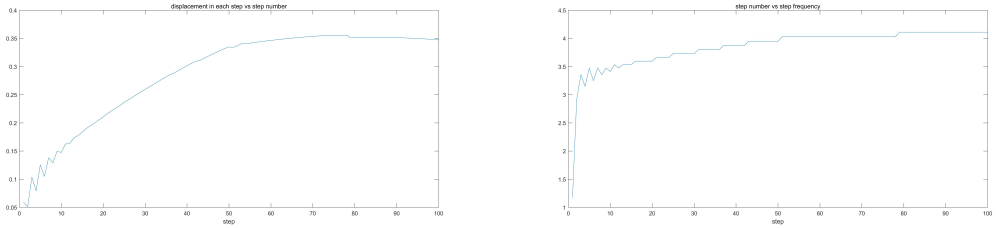

Figure 15: Analysis of Displacement and Frequency

Indicated by Figure.15 we could find the robot's step could converge to a stable phase even with a high velocity.

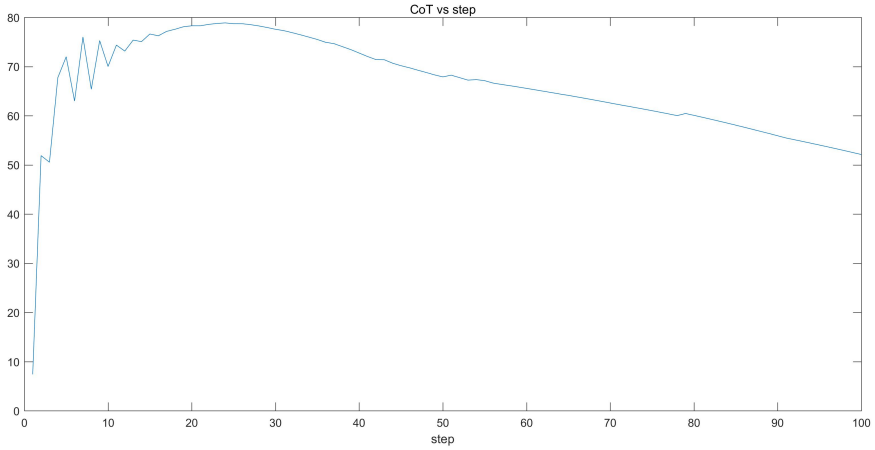

Figure 16: Cost of transport VS time (v = 1.5m/s)

When we try to achieve higher speeds (v=1.5m/s),we have the Figure.16. In this case, the COT will eventually approach 60. By comparing Figure.11 and Figure.16 we can find that the COT has greatly increased.

As we talked about in the lecture, the faster speed is more energy intensive. And in order to achieve this speed we use a larger torque (i.e. larger u), which also greatly increases the energy consumption, just like Equation14.