Technische Universität Berlin

Robotics

Thursday Group 1

# Lab Assignment 1

*Author:*

Ke Zhou(415253)
Changxin He (406129)
Gábor Lajkó(415284)
Janesh Alladi (415349)

November 3, 2019

# 1 Calculations

## 1.1 Find the DH parameters for the 3-DOF RRR Puma

| $i$ | $\alpha_{i-1}$ | $ai-1$ | $d_i$ | $\theta_i$ |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | $q_1$ |
| 2 | 0 | $L_1$ | 0 | $-90^0 + q_2$ |
| 3 | 0 | $L_2$ | 0 | $q_3$ |
| 4(E) | 0 | $L_3$ | 0 | 0 |

## 1.2 Compute the gravity vector

For simplification, we make the following replacements:

$$c\theta_1 = cos(q_1) \tag{1}$$

$$s\theta_{12} = sin(q_1 + q_2) \tag{2}$$

$$s\theta_{123} = sin(q_1 + q_2 + q_3) \tag{3}$$

The individual gravity forces per link

$$\tau_{m_1} = \begin{bmatrix} -r_1 c\theta_1 g m_1 \\ 0 \\ 0 \end{bmatrix} \tag{4}$$

$$\tau_{m_2} = \begin{bmatrix} -(l_1 c\theta_1 + r_2 s\theta_{12}) g m_2 \\ -r_2 s\theta_{12} g m_2 \\ 0 \end{bmatrix} \tag{5}$$

$$\tau_{m_3} = \begin{bmatrix} -(l_1 c\theta_1 + l_2 s\theta_{12} + r_3 s\theta_{123}) g m_3 - r_1 c\theta_1 g m_1 \\ -(l_2 s\theta_{12} + r_3 s\theta_{123}) g m_3 \\ -r_3 s\theta_{123} g m_3 \end{bmatrix} \tag{6}$$

Combined gravity torque

$$\tau_g = \tau_{m1} + \tau_{m2} + \tau_{m3} \tag{7}$$

Finally, the total gravity torque

$$\tau_g = \begin{bmatrix} -(l_1 c\theta_1 + l_2 s\theta_{12} + r_3 s\theta_{123}) g m_3 - (l_1 c\theta_1 + r_2 s\theta_{12}) g m_2 - r_1 c\theta_1 g m_1 \\ -(l_2 s\theta_{12} + r_3 s\theta_{123}) g m_3 - r_2 s\theta_{12} g m_2 \\ -r_3 s\theta_{123} g m_3 \end{bmatrix} \tag{8}$$

The rotation around axis Z

$$R_z = \begin{bmatrix} cosq_1 & -sinq_1 & 0 \\ sin(-90 + q_2) & cos(-90 + q_2) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{9}$$

Complete homogeneous transformation matrix

$$\begin{bmatrix} R(q) & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

Translation does not happen when counteracting gravity, therefore the matrix is just an extension of $R(q)$ to another dimension.
Which equals:

$$\begin{bmatrix} cos(q_1) & -sin(q_1) & 0 & 0 \\ sin(-90^0 + q_2) & cos(-90^0 + q_2) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{11}$$

# 2 Implementation

## 2.1 njmoveControl

According to the equation

$$\tau = -k_p(x - x_d) \tag{12}$$

provided in the lecture, the code to implement P-controller is shown as following:

```
void njmoveControl(GlobalVariables& gv)
{gv.tau = gv.kp*(gv.qd - gv.q);}
```

## 2.2 Tune the controllers

**What kind of behaviors do you observe with different gains?**
In general, higher gains result in more oscillation, because if it overshoots farther, it also takes longer time to resettle. On the other hand, If the value of a gain is too small, the link might not be able to reach the desired position.
**Why are well tuned gains different for each joint?**
The last joint needs very small torque, even with that there is oscillation. The first and second joints are more proportional to each other, however if any of the three gains is "over-torqued", all three of them are affected.
Our observation is that the different behaviour of each link is correlated to the length of the given link, and its actual distance from the base.

## 2.3 Document Behaviour

The parameters $Kp2$=500,$Kp3$=180,$Kp5$=30 are implemented to produce the following figure.
According to the lab assignment instruction, the torques should satisfy all the following conditions, which are satisfied in the implementation

$$|\tau_0| < 156.4Nm \tag{13}$$

$$|\tau_1| < 89.4Nm \tag{14}$$

$$|\tau_2| < 20.1Nm \tag{15}$$

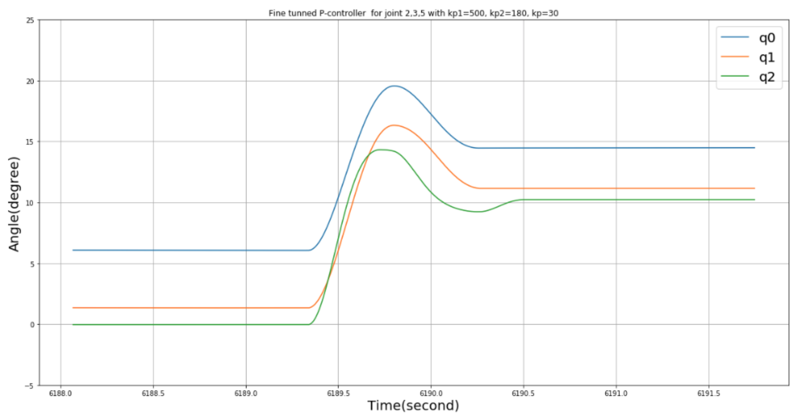


Figure 1: P-controller step response

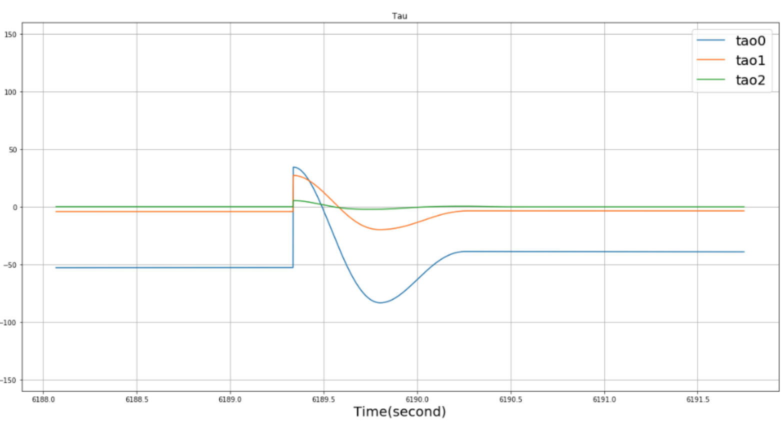


Figure 2: Torque in 3 joints during the response of P controller

## 2.4 Calculate the gravity vector

Using the gravity vector function we computed, the effect of gravity could be compensated. The vector is implemented in the method *PreprocessControl()*. The code is following.

```
float r1(R2);
float r2(0.189738);
float r3(R6);
float l1(L2);
float l2(L3);
float l3(L6);
float m1(M2);
float m2(M3+M4+M5);
float m3(M6);
float g(-9.81);
double tao1 (m1*r1*cos(q1)+m2*(l1*cos(q1)+r2*sin(q1+q2))
+m3*(l1*cos(q1)+l2*sin(q1+q2)+r3*sin(q1+q2+q3)));
double tao2 (m2*r2*sin(q1+q2)+m3*(l2*sin(q1+q2)));
double tao3 (m3*r3*sin(q1+q2+q3));
PrVector3 g123 = PrVector3(tao1*g,tao2*g,tao3*g);
```

## 2.5 floatControl()

Using the following code in the method *floatControl()*, the puma in the simulator can keep its current pose in the mode of 3-DOF and 6-DOF and only move when an external force is applied (when we drag it in the simulator).

```
void floatControl(GlobalVariables& gv)
{
gv.tau = gv.G;
PrintDebug(gv);
}
```

## 2.6 njgotoControl()

In order to implement a P-controller with gravity compensation, the gravity vector should be considered when we implement P-controller, the corresponding torque is following.

$$\tau = -k_p(x - x_d) + G(q) \tag{16}$$

The corresponding code to achieve the function is shown as following.

```
void njgotoControl(GlobalVariables& gv)
{
    gv.tau = gv.kp*(gv.qd - gv.q)+gv.G;
}
```

The parameters $Kp2$=500,$Kp3$=180,$Kp5$=30 are implemented to produce the following figure. Besides, the torques satisfy all the requirements.

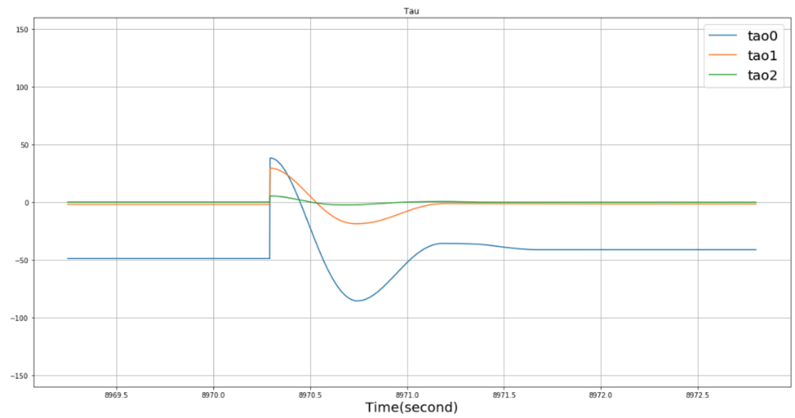


Figure 3: Torque in 3 joints during the response of P controller with gravity compensation

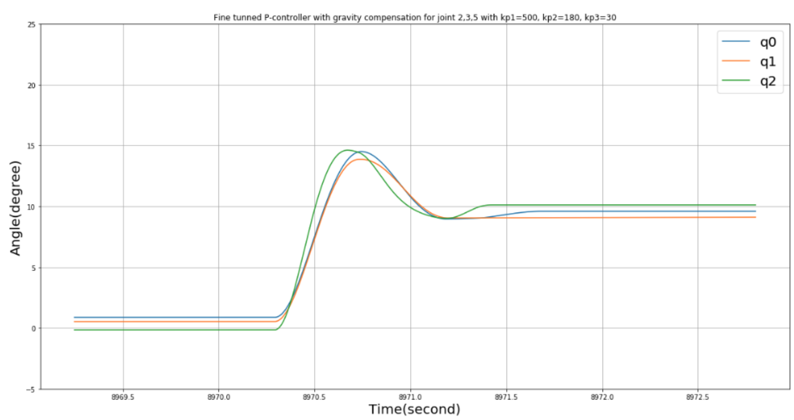


Figure 4: P-controller with gravity compensation step response

### 2.7 jgotoControl()

The following formular is used to implement the *jgotoControl* method

$$\tau = -k_p(x - x_d) - k_v\dot{q} + G(q) \quad (17)$$

The PD-controller with gravity compensation is achieved with the following code.

```
void njgotoControl(GlobalVariables& gv)
{
    gv.tau = gv.kp*(gv.qd - gv.q)-gv.G;
}
```

The parameters $Kp2$=1100,$Kp3$=480,$Kp5$=120, $Kv2$=130,$Kv3$=60,$Kv5$=12 are implemented to produce the following figure. Besides, the torques satisfy all the requirements.

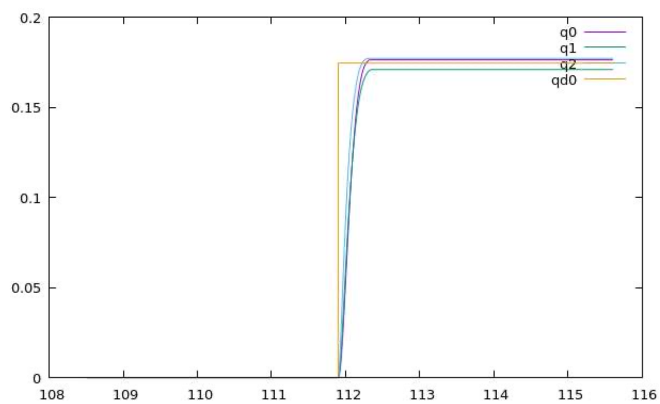


Figure 5: PD-controller step response

**Why can the gains *kp* now be higher compared to the P-controller** From the observation of the formula of P and PD controller, the implementation of $Kv$ apply the force opposing velocity, which leads to the dissipation or dampening, in this case, the force or torque provided by proportional control will be decreased in PD controller, so we can use higher gains of $Kp$.

## 3 Task distribution

| Student Name | A(1) | A(2) | B(1) | B(2) | B(3) | B(4) | B(5) | B(6) | B(7) |
|---|---|---|---|---|---|---|---|---|---|
| Ke Zhou | | | | X | | | | X | X |
| Janesh Alladi | X | | | X | | | | | X |
| Changxin He | | | X | | | X | X | X | X |
| Gábor Lajkó | | X | | X | X | | | | |

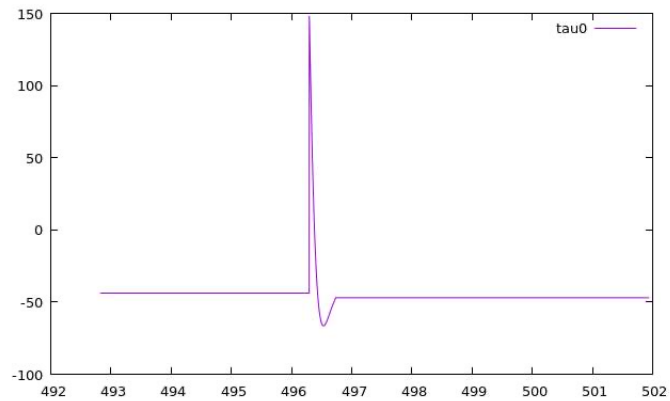


Figure 6: tau0 of PD-controller during step response

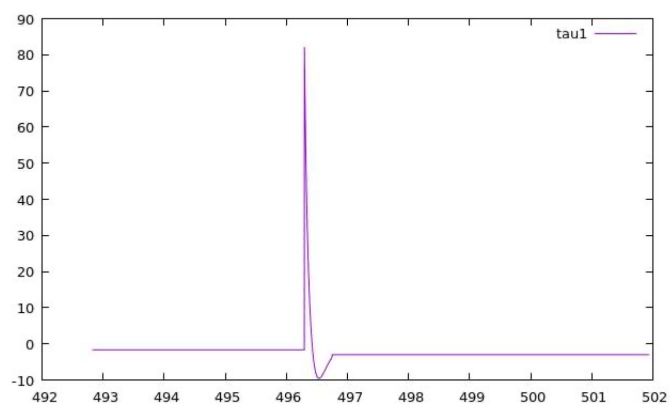


Figure 7: tau1 of PD-controller during step response

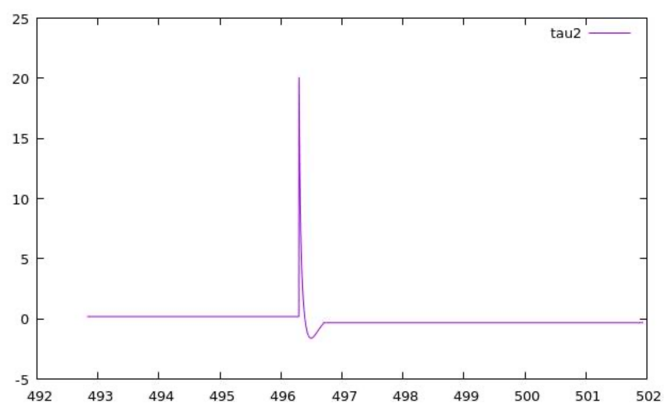


Figure 8: tau2 of PD-controller duringstep response