Dynamics and Control

Homework 2

Jinaykumar Patel
1001937580

1. System Identification

Given that the system dynamics is of form

$$\tau = I\ddot{\Theta} + B\dot{\Theta} + G(\Theta) \tag{1}$$

Inertia of the arm (I), the viscous friction in the joint (B) and by gravity (G).
It is given that the function *PD_control()* receives all relevant information from the robot system and is called at a rate of 500 Hz, i.e. **0.002 sec**.

In order to identity the system parameters (I, B, G), first the gravity and viscous friction term should be identified. Few experiments are done to predict the gravity term by returning random numbers from the function *PD_Control()*. Also, the form gravity term could be easily determine from the force analysis. The gravity force always act downwards. Assuming the mass of robot manipulator to be m and length l, taking the components of this force we can seen from fig. (1) that the component $mg\cos\theta$ is responsible for the torque and it is perpendicular to the position vector. The other component $mg\sin\theta$ acts along the direction of position vector (or the length) hence, it doesn't produce any torque. Hence, the torque due to gravity is therefore $\mathbf{l} \times \mathbf{mg}\cos\theta$ or $\mathbf{mgl}\cos\theta$. Following is the code for the gravity term identification, where the constant number in the term is identified to be equal to **2.205195**.

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#ifndef M_PI
#define M_PI 3.1415927
#endif

int UTA_ID = 1001937580;


double PD_control(theta, theta_dot, theta_ref, theta_dot_ref)
double theta, theta_dot, theta_ref, theta_dot_ref;
{

  printf("%f %f\n", theta, theta_dot);
  return (2.205195*cos(theta));
}
```

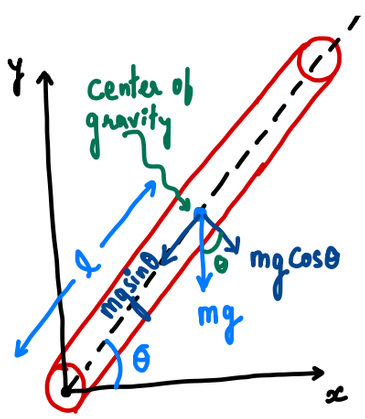
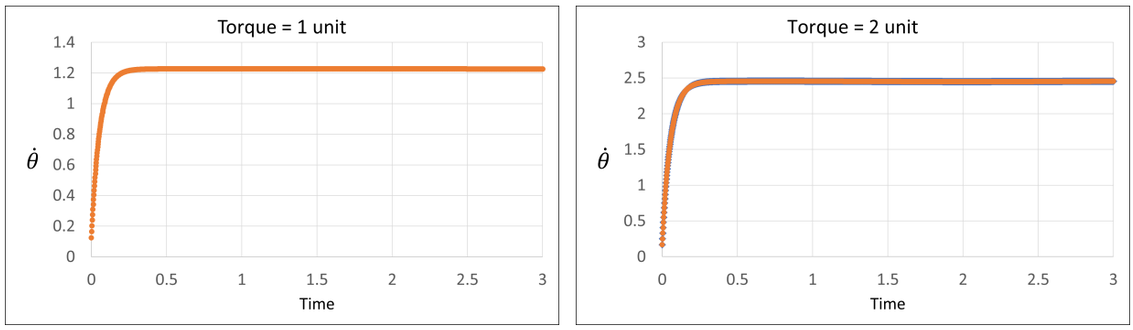


Figure 1: Gravity force acting through center of gravity



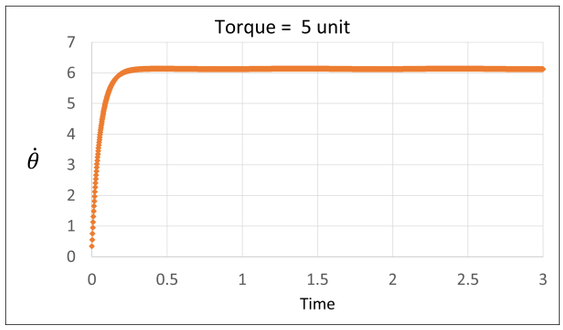


Figure 2: $\dot{\theta}$ for different values of constant torques

Now that we have $G(\Theta)$, we can compensate for it and look for a way to get $\ddot{\theta} = 0$. Need to achieve constant velocity, hence apply a constant torque and gravity compensation, i.e.

$$\tau + G(\theta) = I\ddot{\theta} + B\dot{\theta} + G(\theta) \tag{2}$$

System will eventually reach constant velocity, i.e. $\ddot{\theta} = 0$

$$\tau = B\dot{\theta} \quad \Rightarrow \quad B = \frac{\tau}{\dot{\theta}} \tag{3}$$

Three experiments are performed while applying different constant torques of $(1,2,5)$ units. Fig. (2) shows these different experiments which shows that it reach a constant value of $\dot{\theta}$. From eq. (3) we can arrive with the value of B. Following is the code snippet for this case, where B turns out to be equal to **0.815009**.

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#ifndef M_PI
#define M_PI 3.1415927
#endif

int UTA_ID = 1001937580;


double PD_control(theta, theta_dot, theta_ref, theta_dot_ref)
double theta, theta_dot, theta_ref, theta_dot_ref;
{

  double tau = 1; // tau = 2; // tau = 5;
  //double B = 0.815009;
  //double B = tau/theta_dot;
  //printf("%f\n", theta_dot); //B = 0.815009;

  printf("%f %f\n", theta, theta_dot);
  return (tau + 2.205195*cos(theta));
}
```

Now, knowing that B and $G(\Theta)$, compensate for it and get $\ddot{\theta} = 0$. Applying constant torque, gravity compensation and friction to achieve constant acceleration.

$$\tau + G(\theta) + B\dot{\theta} = I\ddot{\theta} + B\dot{\theta} + G(\theta) \tag{4}$$

System will eventually reach constant acceleration,

$$\tau = I\,\ddot{\dot{\theta}} \quad \Rightarrow \quad I = \frac{\tau}{\ddot{\theta}} \tag{5}$$

Three experiments are performed while applying different constant torques of $(1,2,5)$ units. For each constant torque, $\dot{\theta}$ values is collected and imported to excel sheet (which are attached in the zip file), where $\ddot{\theta}$ is calculated using the 500 Hz frequency given, i.e.

$$\ddot{\theta} = \frac{\Delta\dot{\theta}}{\Delta t} \tag{6}$$

Following is the code snippet for this case, where I turns out to be equal to $\boxed{\mathbf{0.046205}}$, which is taken as the average of three values from Table 1.

```c
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#ifndef M_PI
#define M_PI 3.1415927
#endif

int UTA_ID = 1001937580;


double PD_control(theta, theta_dot, theta_ref, theta_dot_ref)
double theta, theta_dot, theta_ref, theta_dot_ref;
{

  double tau = 1; // tau = 2; // tau = 5;
  //double B = 0.815009;
  //double B = tau/theta_dot;
  //printf("%f\n", theta_dot); //B = 0.815009;
  //double I = 0.046205
  printf("%f %f\n", theta, theta_dot);
  return (tau + b*theta_dot + 2.205195*cos(theta));
}
```

Torque value	**Inertia value, I**
1 unit	0.045891
2 unit	0.046107
5 unit	0.046617

Table 1: Average of $\ddot{\theta}$ over a time period of 2 *secs*

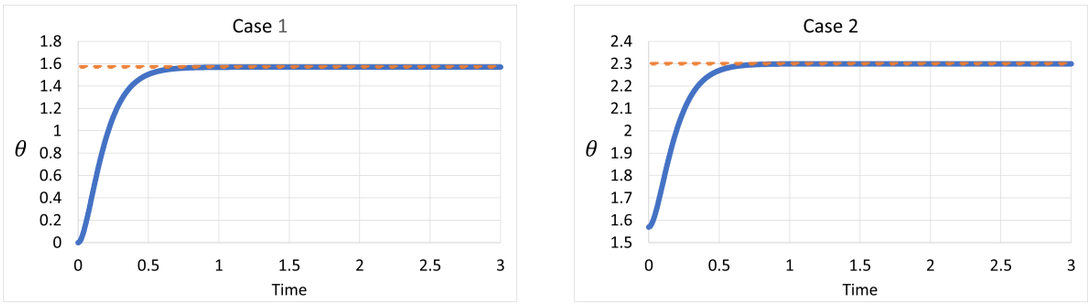
2. PD Controller Implementation

With the system parameters identified in Section 1, PD control is implement in this section with gain K_p andd K_d. The control force will be equal to,

$$F_c = I\left[(K_p(\theta_{ref} - \theta) + K_d(\dot{\theta}_{ref} - \dot{\theta})\right] + B\dot{\theta} + G(\theta) \tag{7}$$

Fig. (3) shows the actual implemenation of PD control for two cases.

- Case (a): Initial $\theta_0 = 0$ and the reference is set as $\theta_{ref} = 1.57$. It could be seen that the PD controller successfully takes the robot manipulator to the set reference value.
- Case (b): Here, the initial $\theta_0 = 1.57$ and the reference is set to $\theta_{ref} = 2.3$. We can see that it reaches the reference value by using the PD control.



(a) (b)

Figure 3: θ for different intial conditions

The overall code that implements the PD control is shown below.

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#ifndef M_PI
#define M_PI 3.1415927
#endif

int UTA_ID = 1001937580;


double PD_control(theta, theta_dot, theta_ref, theta_dot_ref)
double theta, theta_dot, theta_ref, theta_dot_ref;
{

  double b = 0.815009;
  double I = 0.046205;
  //double b = tau/theta_dot;

  printf("%f %f\n", theta, theta_dot);
  double K_p = 100.0;
  double K_d = 2*sqrt(K_p);

  double control_force = I*(K_p*(theta_ref-theta) + K_d*(theta_dot_ref-...
    theta_dot)) + b*theta_dot + 2.205195*cos(theta);
  return control_force;
}
```