Report Assignment 1

Submitted by: Prasham Patel

Student ID: 871563809

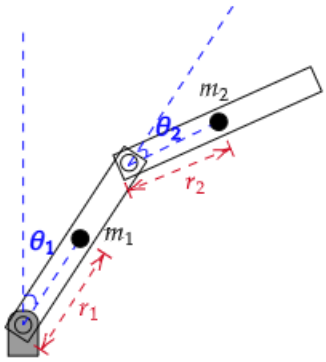

Fig 1. 2-link system

Generalize co-ordinates

$$q = \begin{bmatrix}\theta_1 \\ \theta_2\end{bmatrix} \qquad u = \begin{bmatrix}u_1 \\ u_2\end{bmatrix}$$

The angles are in radians

$$x_1 = r_1 sin\theta_1 \qquad y_1 = r_1 cos\theta_1$$
$$\dot{x}_1 = r_1\dot{\theta}_1 cos\theta_1 \qquad \dot{y}_1 = -r_1\dot{\theta}_1 sin\theta_1 \qquad \omega_1 = \dot{\theta}_1$$

$$x_2 = l_1 sin\theta_1 + r_2\sin(\theta_1 + \theta_2) \qquad y_2 = l_1 cos\theta_1 + r_2\cos(\theta_1 + \theta_2)$$
$$\dot{x}_2 = l_1\dot{\theta}_1 cos\theta_1 + r_2(\dot{\theta}_1 + \dot{\theta}_2)\cos(\theta_1 + \theta_2)$$
$$\dot{y}_2 = -l_1\dot{\theta}_1 sin\theta_1 - r_2(\dot{\theta}_1 + \dot{\theta}_2)\sin(\theta_1 + \theta_2)$$
$$\omega_2 = \dot{\theta}_1 + \dot{\theta}_2$$

Kinetic Energy

K.E for link1 = $0.5(m1(r1\cdot\dot{\theta}1')^2 + I1(\dot{\theta}1')^2)$

K.E for link2 = $0.5\cdot m2[(L1\cdot\dot{\theta}1)^2 + (r2(\dot{\theta}1 + \dot{\theta}2))^2 + 2\cdot\dot{\theta}1\cdot L1\cdot r2(\dot{\theta}1 + \dot{\theta}2)\cdot\cos(\theta2)] + 0.5\cdot I2(\dot{\theta}1 + \dot{\theta}2)^2$

K.E = $0.5(m1(r1\cdot\dot{\theta}1)^2 + I1(\dot{\theta}1)^2) + 0.5\cdot m2[(L1\cdot\dot{\theta}1)^2 + 0.5\cdot m2[(L1\cdot\dot{\theta}1)^2 + (r2(\dot{\theta}1 + \dot{\theta}2))^2 + 2\cdot\dot{\theta}1\cdot L1\cdot r2(\dot{\theta}1 + \dot{\theta}2)\cdot\cos(\theta2)] + I2(\dot{\theta}1 + \dot{\theta}2)^2$

Potential Energy

P.E for link 1 = $m1\cdot g\cdot r1\cdot\cos(\theta1)$

P.E for link2 = $m2\cdot g(L1\cdot\cos(\theta1) + r2\cdot\cos(\theta1 + \theta2))$

P.E = $m1\cdot g\cdot r1\cdot\cos(\theta1) - m2\cdot g(L1\cdot\cos(\theta1) + r2\cdot\cos(\theta1 + \theta2))$

Langrangian function

L = K.E - P.E

L = $0.5(m1(r1\cdot\dot{\theta}1)^2 + I1(\dot{\theta}1)^2) + 0.5\cdot m2[(L1\cdot\dot{\theta}1)^2 + 0.5\cdot m2[(L1\cdot\dot{\theta}1)^2 + (r2(\dot{\theta}1 + \dot{\theta}2))^2 + 2\cdot\dot{\theta}1\cdot L1\cdot r2(\dot{\theta}1 + \dot{\theta}2)\cdot\cos(\theta2)] + I2(\dot{\theta}1 + \dot{\theta}2)^2 - m1\cdot g\cdot r1\cdot\cos(\theta1) - m2\cdot g(L1\cdot\cos(\theta1) + r2\cdot\cos(\theta1 + \theta2))$

Partial differentiations

Equations are calculated with help of following command in MATLAB commands

```
>>  dl_dtheta1 = jacobian(L, theta1);
```

$\frac{\partial L}{\partial \theta 1}$ = g*m2*(r2*sin(theta1 + theta2) + l1*sin(theta1)) + g*m1*r1*sin(theta1)

```
>>  dl_dtheta2 = jacobian(L, theta2);
```

$\frac{\partial L}{\partial \theta 2}$ = g*m2*r2*sin(theta1 + theta2) - l1*m2*r2*theta_dot1*sin(theta2)*(theta_dot1 + theta_dot2)

```
>>  dl_dtheta_dot1 = jacobian(L, theta_dot1);
```

$\frac{\partial L}{\partial \dot{\theta 1}}$ = theta_dot1*l1^2 + m1*theta_dot1*r1^2 + (l2*(2*theta_dot1 + 2*theta_dot2))/2 + (m2*(r2^2*(2*theta_dot1 + 2*theta_dot2) + 2*l1^2*theta_dot1 + 2*l1*r2*cos(theta2) * (theta_dot1 + theta_dot2) + 2*l1*r2*theta_dot1*cos(theta2)))/2

```
>>  dl_dtheta_dot2 = jacobian(L, theta_dot2);
```

$\frac{\partial L}{\partial \dot{\theta 2}}$ = (l2*(2*theta_dot1 + 2*theta_dot2))/2 + (m2*((2*theta_dot1 + 2*theta_dot2)*r2^2 + 2*l1*theta_dot1*cos(theta2)*r2))/2

```
>>  ddl_dtheta_dot1_dt = jacobian(dl_dtheta_dot1, [theta1; theta_dot1])*[theta_dot1; theta_ddot1] + jacobian(dl_dtheta_dot1, [theta2; theta_dot2])*[theta_dot2; theta_ddot2];
```

$\frac{d\frac{\partial L}{\partial \dot{\theta 1}}}{dt}$ = theta_ddot1*(l1^2 + m1*r1^2 + l2 + (m2*(2*l1^2 + 4*cos(theta2)*l1*r2 + 2*r2^2))/2) + theta_ddot2*(l2 + (m2*(2*r2^2 + 2*l1*cos(theta2)*r2))/2) - (m2*theta_dot2*(2*l1*r2*sin(theta2) *(theta_dot1 + theta_dot2) + 2*l1*r2*theta_dot1*sin(theta2)))/2

```
>>  ddl_dtheta_dot2_dt = jacobian(dl_dtheta_dot2, [theta1; theta_dot1])*[theta_dot1; theta_ddot1] + jacobian(dl_dtheta_dot2, [theta2; theta_dot2])*[theta_dot2; theta_ddot2];
```

$\frac{d\frac{\partial L}{\partial \dot{\theta 1}}}{dt}$ = theta_ddot2*(m2*r2^2 + l2) + theta_ddot1*(l2 + (m2*(2*r2^2 + 2*l1*cos(theta2)*r2))/2) - l1*m2*r2*theta_dot1*theta_dot2*sin(theta2)

Euler langrange Equation

u1 = $\frac{d\frac{\partial L}{\partial \dot{\theta 1}}}{dt} - \frac{\partial L}{\partial \theta 1}$

u1 is calculated with help of following MATLAB command

```
>>  u1 = ddl_dtheta_dot1_dt - dl_dtheta1;
```

u1 = theta_ddot1*(l1^2 + m1*r1^2 + l2 + (m2*(2*l1^2 + 4*cos(theta2)*l1*r2 + 2*r2^2))/2) + theta_ddot2*(l2 + (m2*(2*r2^2 + 2*l1*cos(theta2)*r2))/2) - (m2*theta_dot2*(2*l1*r2*sin(theta2) *(theta_dot1 + theta_dot2) + 2*l1*r2*theta_dot1*sin(theta2)))/2 - g*m2*(r2*sin(theta1 + theta2) + l1*sin(theta1)) - g*m1*r1*sin(theta1)

u2 = $\frac{d\frac{\partial L}{\partial \dot{\theta 2}}}{dt} - \frac{\partial L}{\partial \theta 2}$

u2 is calculated with help of following MATLAB command

```
>>    u2 = ddl_dtheta_dot2_dt - dl_dtheta2;
```

u2 = theta_ddot2*(m2*r2^2 + I2) + theta_ddot1*(I2 + (m2*(2*r2^2 + 2*l1*cos(theta2)*r2))/2) - g*m2*r2*sin(theta1 + theta2) + l1*m2*r2*theta_dot1*sin(theta2)*(theta_dot1 + theta_dot2) - l1*m2*r2*theta_dot1*theta_dot2*sin(theta2)

State Space representation

u1 and u2 are solved using the following MATLAB Command

```
>>    sol = solve([u1==0, u2==0], [theta_ddot1, theta_ddot2]);
>>    theta_ddot1 = sol.theta_ddot1;
```

$\ddot{\theta}1$ = (I1*m2^2*r2^3*theta_dot1^2*sin(theta2) + I1*m2^2*r2^3*theta_dot2^2*sin(theta2) + g*I1*m2^2*r2^2*sin(theta1) + I2*g*I1*m2*sin(theta1) + I2*g*m1*r1*sin(theta1) + 2*I1*m2^2*r2^3*theta_dot1*theta_dot2*sin(theta2) + I1^2*m2^2*r2^2*theta_dot1^2*cos(theta2)*sin(theta2) - g*I1*m2^2*r2^2*sin(theta1 + theta2)*cos(theta2) + I2*I1*m2*r2*theta_dot1^2*sin(theta2) + I2*I1*m2*r2*theta_dot2^2*sin(theta2) + g*m1*m2*r1*r2^2*sin(theta1) + 2*I2*I1*m2*r2*theta_dot1*theta_dot2*sin(theta2))/(I1^2*I2 + I1^2*m2*r2^2 + I1^2*m2^2*r2^2 + I2*I1^2*m2 + I2*m1*r1^2 - I1^2*m2^2*r2^2*cos(theta2)^2 + m1*m2*r1^2*r2^2)

```
>>    theta_ddot2 = sol.theta_ddot2 ;
```

$\ddot{\theta}2$ = -(I1*m2^2*r2^3*theta_dot1^2*sin(theta2) - I1^2*g*m2*r2*sin(theta1 + theta2) + I1^3*m2^2*r2*theta_dot1^2*sin(theta2) + I1*m2^2*r2^3*theta_dot2^2*sin(theta2) - g*I1^2*m2^2*r2*sin(theta1 + theta2) + g*I1*m2^2*r2^2*sin(theta1) + I2*g*I1*m2*sin(theta1) + I2*g*m1*r1*sin(theta1) + I1^2*I1*m2*r2*theta_dot1^2*sin(theta2) + 2*I1*m2^2*r2^3*theta_dot1*theta_dot2*sin(theta2) + 2*I1^2*m2^2*r2^2*theta_dot1^2*cos(theta2)*sin(theta2) + I1^2*m2^2*r2^2*theta_dot2^2*cos(theta2)*sin(theta2) - g*I1*m2^2*r2^2*sin(theta1 + theta2)*cos(theta2) + g*I1^2*m2^2*r2*cos(theta2)*sin(theta1) - g*m1*m2*r1^2*r2*sin(theta1 + theta2) + I2*I1*m2*r2*theta_dot1^2*sin(theta2) + I2*I1*m2*r2*theta_dot2^2*sin(theta2) + g*m1*m2*r1*r2^2*sin(theta1) + 2*I1^2*m2^2*r2^2*theta_dot1*theta_dot2*cos(theta2)*sin(theta2) + I1*m1*m2*r1^2*r2*theta_dot1^2*sin(theta2) + 2*I2*I1*m2*r2*theta_dot1*theta_dot2*sin(theta2) + g*I1*m1*m2*r1*r2*cos(theta2)*sin(theta1))/(I1^2*I2 + I1^2*m2*r2^2 + I1^2*m2^2*r2^2 + I2*I1^2*m2 + I2*m1*r1^2 - I1^2*m2^2*r2^2*cos(theta2)^2 + m1*m2*r1^2*r2^2)

Plotting state trajectory

The ode function for getting the state space values and used state space representation derived above in the function. The command to simulate in ODE45 and plot is as below:

```
>>    [t, y] = ode45(@ode_2link, [0, 10], [pi/6, p1/4, 0, 0]);
>>    plot(t, y);
```

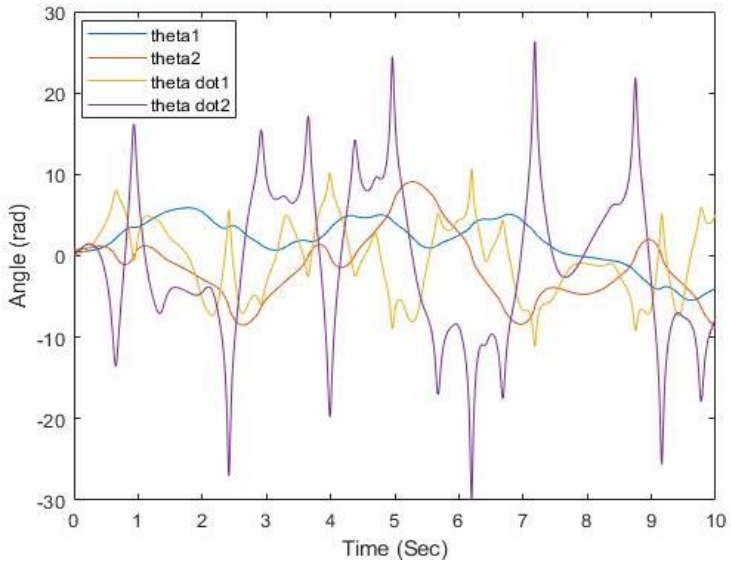

Fig 2. State space trajectory plot