FL. 2018 ESE 447.02 Robotics Lab

Lab Journal 11

10/14/2018

1. Hamilton theory
   Hamiltonian mechanics is a theory developed as a reformulation of classical mechanics and predicts the same outcomes as non-Hamiltonian classical mechanics. It uses a different mathematical formalism, providing a more abstract understanding of the theory. In practice of mechanical engineers, conventional Newton's mechanics is predominantly used. It was investigated as the approach that uses the property of passivity of the robot. Such approach can modify the natural energy of the robot so that it can satisfy the desired objectives (position or tracking control). In previous labs, we have used the Lagrangian to simulate the 2-link robot system with 6 given parameters for the system, but the parameters were not as realistic as the real physical system. Therefore, we are using the Hamiltonian to process the data collected from the real system to get the 6 parameters for the physical system.

The Hamiltonian function is defined by

$$H = \sum_{i=1}^{n} p_i \dot{q}_i - L$$

Where generalized momentums are defined by

$$p_j = \frac{\partial L}{\partial \dot{q}_j}, j = 1,21 \dots, n.$$

The Hamilton's equation can be written as

$$\dot{q_J} = \frac{\partial H}{\partial p_j}, \dot{p_j} = F_j - \frac{\partial H}{\partial q_j}$$

These equations can be rewritten in the matrix form

$$\dot{\boldsymbol{q}} = \left(\frac{\partial H}{\partial \boldsymbol{p}}\right)^T, \dot{p} = \boldsymbol{F} - \left(\frac{\partial H}{\partial \boldsymbol{q}}\right)^T$$

The Hamiltonian is total energy: the sum of kinetic and potential energies. The Lagrangian L depends on positions and velocities, but the Hamiltonian depends on positions and generalized momentums, so we can write

$$H(\boldsymbol{q}, \boldsymbol{p}) = K(\boldsymbol{q}, \boldsymbol{p}) + V(\boldsymbol{q})$$

The kinetic energy in coordinates p and q has the following form

$$K(\boldsymbol{q}, \boldsymbol{p}) = \frac{1}{2} \boldsymbol{p}^T \boldsymbol{M}^{-1}(\boldsymbol{q}) \boldsymbol{p}$$

The Hamiltonian equations for the real system are given as

$$H\big(q(t), q(\dot{t})\big) = \sum_{i=1}^{n} h_i\,(q(t), q(\dot{t}))\theta_i' + m_i g z_{ci}$$

Where

$$h_1(q(t), q(\dot{t})) = \frac{1}{2}(\dot{q}_1^2)$$

$$h_2\big(q(t), q(\dot{t})\big) = \frac{1}{2} sin^2(q_2)(\dot{q}_1^2) + \frac{1}{2}\dot{q}_2{}^2$$

$$h_3(q(t), q(\dot{t})) = \cos(\dot{q}_2)\dot{q}_1\dot{q}_2$$

$$h_4(q(t), q(\dot{t})) = g\cos(q_2)$$

Based on conservation of energy,

$$\int_{t_0}^{t} r(s) q_{(s)}^{\cdot}\, ds = \bar{H}(t, t_0)\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\\ \theta_4\end{bmatrix} + \bar{F}(t, t_0)\begin{bmatrix}\theta_5\\ \theta_6\end{bmatrix}$$

Where $\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6$ are the system parameters we need to get.

2. Finding thetas for a given system.

   We wrote a Matlab function that took in the real system states and calculate the parameters by using the equations we have derived.

   The state parameters we can get from the physical system are:

$$q_1 - the\ angle\ by\ which\ the\ first\ link\ rotates$$
$$\dot{q}_1 - the\ angular\ velocity\ of\ the\ first\ link$$
$$q_2 - the\ angle\ by\ which\ the\ second\ link\ rotates$$
$$\dot{q}_2 - the\ angular\ velocity\ of\ the\ second\ link$$
$$v - the\ input\ voltage\ to\ the\ motor\ for\ the\ first\ link$$
$$t - the\ time\ line\ used\ to\ record\ the\ system\ state$$

   Therefore the function takes in the 6 parameters above as the function input arguments, the output argument was a matrix that contained the 6 θ values.

   The Matlab code used to get the input parameters is:

```
function [ theta ] = ham( rbd,V )
t = rbd(:,1);
v = V(:,2);
q1 = rbd(:,2);
q1_dot = rbd(:,3);
q2 = rbd(:,4);
q2_dot = rbd(:,5);
```

   Where rbd is the name of the array that was recorded by the scope in the Simulink model.

   By observing the rbd format, we determined which columns of the input array correspond to the input parameters respectively and gave them to our Matlab variables t, v q1, q2, q1_dot and q2_dot.

   The Matlab code for the integration and matrix calculation is:

```
length =size(rbd)
F1 = zeros(length(1),1);
F2 = zeros(length(1),1);
H = zeros(length(1),4);
F = zeros(length(1),2);
D1 = zeros(length(1),1);
d = zeros(length(1),1);

for i= (2:length(1)-1)
      H(i-1,1) = [h1(i)-h1(1)];
      H(i-1,2) = [h2(i)-h2(1)];
      H(i-1,3) = [h3(i)-h3(1)];
      H(i-1,4) = [h4(i)-h4(1)];

      F1(i-1) = q1_dot(i-1).^2;
      F2(i-1) = q2_dot(i-1).^2;

      F(i-1,1) = trapz(t(1:i),F1(1:i));
      F(i-1,2) = trapz(t(1:i),F2(1:i));

      D1(i-1) = v(i-1).*q1_dot(i-1);
      d(i-1) = trapz(t(1:i),D1(1:i));
end

A=[H F];
A1= pinv(A);
theta = A1*d

end
```

where the “zeros” was used to initialize the matrices space, the “trapz” was used to integrate the equations with respect to time for different time intervals. The “pinv” calculated the Moore-Penrose Pseudoinverse of matrix A.

The coding of this part was based on the equations we derived, here we initialized the variables F1, F2, H, F, D1 and d as zero matrices. Length is simply the size of our input array and we used it to keep the dimensions of our matrices consistent with each other. The “for loop” was used to calculate the difference and integrals between different time points. The final output of this function was theta, which is the matrix that contained the theta values.