# Robotics II
# Day 11: Non-Linear Control


Chua Tzong Lin
B7TB1703
chuatl@mems.mech.tohoku.ac.jp


Background Knowledge

The equation of motion of a 2 Link Manipulator is given as,

$$\begin{pmatrix}\tau_1\\ \tau_2\end{pmatrix} = \begin{pmatrix} I_1 + m_1 {l_{g_1}}^2 + m_2({l_1}^2 + {l_2}^2 + 2 l_1 l_2 cos(\theta_2)) & m_2({l_{g_2}}^2 + l_1 l_2 cos(\theta_2)) \\ m_2({l_{g_2}}^2 + l_1 l_2 cos(\theta_2)) & I_2 + m_2 {l_{g_2}}^2 \end{pmatrix}\begin{pmatrix}\ddot{\theta}_1\\ \ddot{\theta}_2\end{pmatrix}$$
$$+ m_2 l_1 l_{g_2} sin(\theta_2)\begin{pmatrix} -\dot{\theta}_2{}^2 \\ \dot{\theta}_1{}^2 \end{pmatrix} + 2 m_2 l_1 l_{g_2}\begin{pmatrix} -\dot{\theta}_1\dot{\theta}_2 \\ 0\end{pmatrix}$$
$$- g\begin{pmatrix} m_1 l_{g_1} cos(\theta_1) + m_2 l_1 cos(\theta_1) + m_2 l_{g_2} cos(\theta_1+\theta_2) \\ m_2 l_{g_2} cos(\theta_1+\theta_2)\end{pmatrix}$$

Enveloping all terms in a function,

$$\vec{\tau} = M(\vec{\theta})\ddot{\vec{\theta}} + H\left(\vec{\theta}, \dot{\vec{\theta}}\right) + G(\vec{\theta})$$

Since the equation of motion is a non-linear function, a non-linear control approach is used when selecting a controller. A non-linear control has the following properties, the output of the controller (which will be the input of the system), is selected using the following criteria which will linearize or more accurately cancel out the non-linear components of the system. The controller of a non-linear system is given as,

$$\vec{\tau} = \alpha\vec{\tau}' + \beta$$

Where, α will compensate for the non-linear coefficients of the linear components and β will compensate for the non-linear components. As a result, $\vec{\tau}'$ will be our linear controller from control theory.

In our system, α and β are chosen by comparing the equation of motion and non-linear controller input. By comparison,

$$\alpha = M(\vec{\theta})$$
$$\beta = H\left(\vec{\theta}, \dot{\vec{\theta}}\right) + G(\vec{\theta})$$

Hence, the remaining task will be selecting a suitable linear controller, $\vec{\tau}'$, which will modulate the system through the following equation.

$$\vec{\tau}' = \ddot{\vec{\theta}}$$

From the context of control theory, a feedback controller can be designed to suit this application. The feedback control of the system can be of any variable. However, in the given system, a direct approach would be the kinematic feedback of the system.
There exist 2 types of kinematics feedback regarding 2 link manipulator system, the first being a feedback from the joint angle or joint space; the second being the position of the end joint in Cartesian coordinates also call the feedback from the task space.

From the control theory, a feedback proportional-derivative controller based on the kinematic feedback of the system is designed.

From the control theory,

$$\vec{\tau}' - \ddot{\vec{\theta}} = \ddot{\mathrm{E}}(\text{Current State}) + \mathrm{K_v}\dot{\mathrm{E}}(\text{Current State}) + \mathrm{K_p}\mathrm{E}(\text{Current State})$$

Where, E (Current State) represents the error between the targeted state and current state.

Computed Torque Control

Since the equation of motion is a function of joint angles, the controller proposed above is valid in the joint space of the robot. Hence, the complete representation is given by the following equation.

$$\mathrm{E}(\vec{\theta}) = \vec{\theta}_{target} - \vec{\theta}$$
$$\vec{\tau}' - \ddot{\vec{\theta}} = \ddot{\mathrm{E}}(\vec{\theta}) + \mathrm{K_v}\dot{\mathrm{E}}(\vec{\theta}) + \mathrm{K_p}\mathrm{E}(\vec{\theta})$$

Simplifying by expanding E ($\vec{\theta}$) and its derivatives with time,

$$\vec{\tau}' = \ddot{\vec{\theta}}_{\boldsymbol{target}} + \mathrm{K_v}(\dot{\vec{\theta}}_{target} - \dot{\vec{\theta}}) + \mathrm{K_p}(\vec{\theta}_{target} - \vec{\theta})$$

Therefore, combining our linear components and non-linear components the final output of the controller (input to the system) to the system is given as,

$$\vec{\tau} = M(\vec{\theta})[\ddot{\vec{\theta}}_{target} + \mathrm{K_v}(\dot{\vec{\theta}}_{target} - \dot{\vec{\theta}}) + \mathrm{K_p}(\vec{\theta}_{target} - \vec{\theta})] + H\left(\vec{\theta}, \dot{\vec{\theta}}\right) + G(\vec{\theta})$$

The result is shown in figure 1.

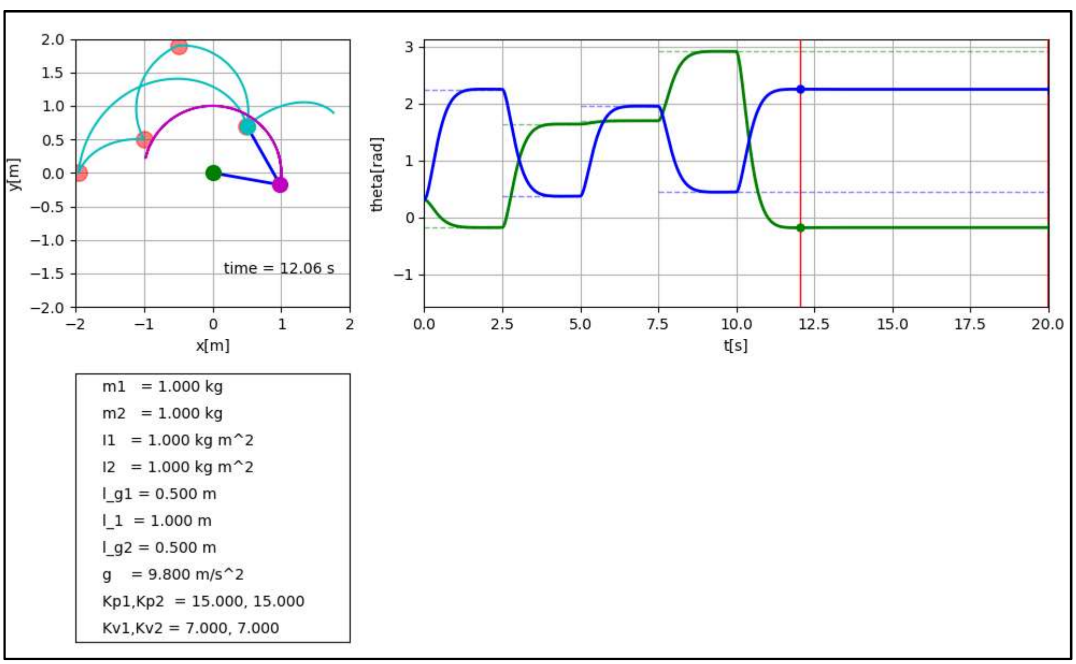


Figure 1 PD Controller using joint space feedbacks

Resolved Acceleration Control

However, if the desired controller is based on the feedback from the task space, certain transformations are required to be performed. Similar to the joint space controller, from the control theory,

$$\vec{\tau}' - \ddot{\vec{\theta}} = \ddot{\mathrm{E}}(\text{Current State}) + \mathrm{K_v}\dot{\mathrm{E}}(\text{Current State}) + \mathrm{K_p}\mathrm{E}(\text{Current State})$$

However, in this case the current state is the position vector of the end joint. Since the input of our system is in the joint space, the obtained controller has to be mapped into the joint space from the task space.

The mapping for position to joint angle is given by the inverse kinematics function,

$$\vec{\theta} = Inverse\ Kinematics(\vec{X})$$

The mapping for trajectory velocity to joint angle rotational velocity is given by the tensor transformation or inverse Jacobian transformation matrix.

$$\dot{\vec{\theta}} = J^{-1}(\vec{\theta})\dot{\vec{X}}$$

The result of the 2 transformations is given by,

$$\mathrm{E}(\vec{X}) = \vec{X}_{target} - \vec{X}$$
$$\vec{\tau}' - \ddot{\vec{\theta}} = J^{-1}(\vec{\theta})\{\ddot{\mathrm{E}}(\vec{X}) + \mathrm{K_v}\dot{\mathrm{E}}(\vec{X}) + \mathrm{K_p}\mathrm{E}(\vec{X})\}$$

However, in the case of the trajectory acceleration, the Jacobian Transformation cannot be used directly as a derivative of a tensor is not a tensor, hence during the transformation, the Christoffel symbols are required as a correction factor. The result given in the following equation,

$$\ddot{\vec{\theta}} = J^{-1}(\vec{\theta})\{\ddot{\vec{X}} - \dot{J}(\vec{\theta})\dot{\vec{X}}\}$$

The resulting controller is given by,

$$\vec{\tau}' - J^{-1}(\vec{\theta})\{\ddot{\vec{X}} - \dot{J}(\vec{\theta})\dot{\vec{X}}\} = J^{-1}(\vec{\theta})\{\ddot{\mathrm{E}}(\vec{X}) + \mathrm{K_v}\dot{\mathrm{E}}(\vec{X}) + \mathrm{K_p}\mathrm{E}(\vec{X})\}$$

Rearranging and expanding E(X),

$$\vec{\tau}' = J^{-1}(\vec{\theta})\{\ddot{\mathrm{E}}(\vec{X}) + \mathrm{K_v}\dot{\mathrm{E}}(\vec{X}) + \mathrm{K_p}\mathrm{E}(\vec{X}) + \ddot{\vec{X}} - \dot{J}(\vec{\theta})\dot{\vec{X}}\}$$
$$\vec{\tau}' = J^{-1}(\vec{\theta})\{\ddot{\vec{X}}_{target} + \mathrm{K_v}(\dot{\vec{X}}_{target} - \dot{\vec{X}}) + \mathrm{K_p}(\vec{X}_{target} - \vec{X}) - \dot{J}(\vec{\theta})\dot{\vec{X}}\}$$

Combining our linear components and non-linear components the final output of the controller (input to the system) to the system is given as,

$$\vec{\tau} = M(\vec{\theta})[J^{-1}(\vec{\theta})\{\ddot{\vec{X}}_{target} + \mathrm{K_v}(\dot{\vec{X}}_{target} - \dot{\vec{X}}) + \mathrm{K_p}(\vec{X}_{target} - \vec{X}) - \dot{J}(\vec{\theta})\dot{\vec{X}}\}] + H\left(\vec{\theta},\dot{\vec{\theta}}\right) + G(\vec{\theta})$$

The results are shown in figure 2.

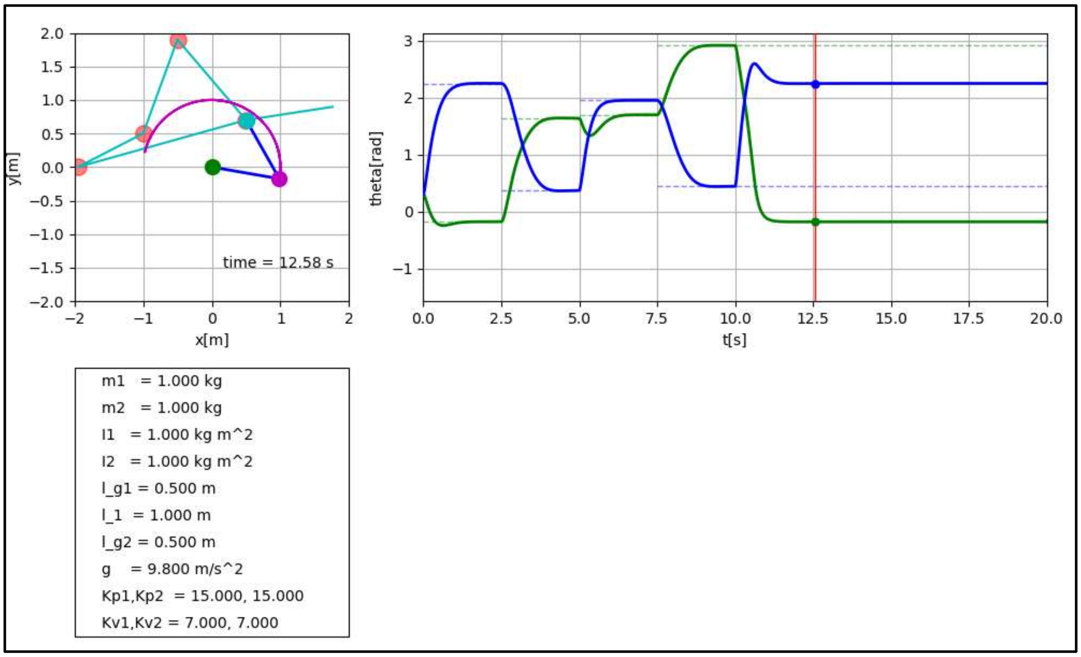

Figure 2 PD Controller using task space feedbacks

Discussion

From the results obtained, it can be observed that the end joint controlled using computed torque control moves in a curve. This is due to the optimized parameters being the joint angle movement. Therefore, the chosen trajectory will give the minimum joint rotation required to reach the target. This can also be interpreted as the torque or effort exerted by the joint being minimized.

In the case of resolved acceleration control, the optimized parameter is the position of the end joint. As a result, the chosen trajectory is the shortest path between the starting and ending point, the resulting trajectory of the end joint is a straight line. In contrast to the computed torque control, the resolved acceleration control minimizes the acceleration of the end joint by moving in the shortest path.

Python Script

The python script is included under the filename ManipulatorControl.py.

During execution of the python script, the program will prompt the user to select the desired feedback controller as shown below.

```
Select feedback signal
 1: Joint Space
 2: Task Space
```

To select a control method, input the number selection as shown (in this case task space feedback),

```
Select feedback signal
 1: Joint Space
 2: Task Space
2
```

The selection 1 will yield the result in figure 1, while 2 will yield the result in figure 2.