Learning By Doing

ROBO Track Solution

João Bravo

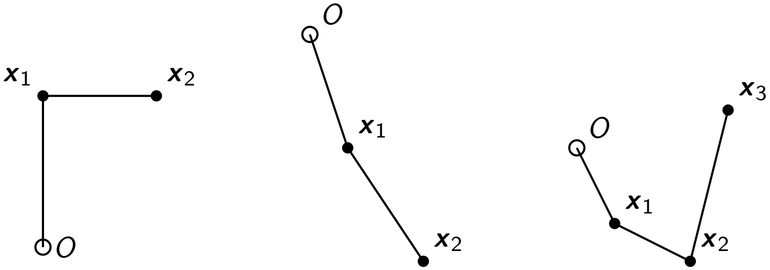
Feedzai

December 8, 2021

Introduction

Goal is to track an end effector trajectory for 3 types of robots

- **Bumblebee:** A prismatic robot arm
- **Beetle:** A two joint rotational robot arm
- **Butterfly:** A three joint rotational robot arm



Introduction

Goal is to track an end effector trajectory for 3 types of robots

Given a set of training trajectories comprising:

- Joint positions and velocities
- Control variables

My solution is a straightforward control approach comprising the following steps:

1. **Modelling (and System Identification)** Model each of the robot types using first principles and estimate their parameters
2. **Control** Design an effective control strategy for each type of robot

Kinematics

- Outputs are the d joint positions $(\mathbf{x}_1, \ldots, \mathbf{x}_d) \in \mathbb{R}^{2d}$ and their velocities
- Configuration space is a submanifold of $\mathbb{R}^{2d}$

Kinematics

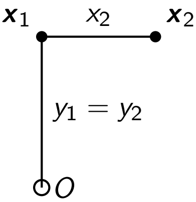
- Outputs are the d joint positions $(\mathbf{x}_1, \dots, \mathbf{x}_d) \in \mathbb{R}^{2d}$ and their velocities
- Configuration space is a submanifold of $\mathbb{R}^{2d}$

Example

For the bumblebee robot it is $\mathbb{R}^2$:

$$x_1 = 0, \quad y_2 = y_1$$

We can think of the configuration space as the position of the tip of the robot, $\mathbf{x}_2$



Kinematics

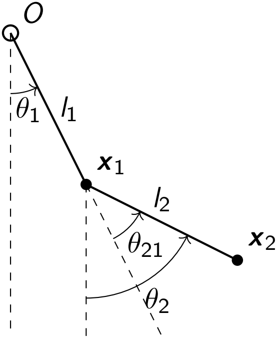
- Outputs are the d joint positions $(\mathbf{x}_1, \ldots, \mathbf{x}_d) \in \mathbb{R}^{2d}$ and their velocities
- Configuration space is a submanifold of $\mathbb{R}^{2d}$

Example

For the beetle robot it is $\mathbb{T}^2$:

$$\|\mathbf{x}_1\| = l_1, \quad \|\mathbf{x}_2 - \mathbf{x}_1\| = l_2$$

The phase space can be though of the space of joint angles and their angular velocities: $(\theta, \dot{\theta}) \in \mathbb{T}^2 \times \mathbb{R}^2$



Kinematics

- Outputs are the d joint positions $(\mathbf{x}_1, \dots, \mathbf{x}_d) \in \mathbb{R}^{2d}$ and their velocities
- Configuration space is a submanifold of $\mathbb{R}^{2d}$
- Kinematics map between the robot's phase space and the output positions and velocities:

$$\mathbf{x}_i = h_i(\mathbf{q}), \quad \dot{\mathbf{x}}_i = J_i(\mathbf{q})\dot{\mathbf{q}}$$

- Only parameters are geometric (such as link lengths)
- We model the dynamics in the phase space and not in the output space

Dynamics

The dynamics of a mechanical system can typically be written as:

$$M(\mathbf{q})\ddot{\mathbf{q}} + c(\mathbf{q}, \dot{\mathbf{q}}) + D\dot{\mathbf{q}} + \nabla V(\mathbf{q}) = B\mathbf{u}$$

Dynamics

The dynamics of a mechanical system can typically be written as:

$$M(\mathbf{q})\ddot{\mathbf{q}} + c(\mathbf{q}, \dot{\mathbf{q}}) + D\dot{\mathbf{q}} + \nabla V(\mathbf{q}) = B\mathbf{u}$$

Example

For the bumblebee robot with $\mathbf{q} = \mathbf{x}_2$:

$$M(\mathbf{q}) = \begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix}$$

$$c(\mathbf{q}, \dot{\mathbf{q}}) = 0, \quad \nabla V(\mathbf{q}) = \begin{bmatrix} 0 \\ m_2 g \end{bmatrix}$$

Dynamics

The dynamics of a mechanical system can typically be written as:

$$M(\mathbf{q})\ddot{\mathbf{q}} + c(\mathbf{q}, \dot{\mathbf{q}}) + D\dot{\mathbf{q}} + \nabla V(\mathbf{q}) = B\mathbf{u}$$

Example

For the beetle robot with $\mathbf{q} = (\theta_1, \theta_2)$:

$$M(\mathbf{q}) = \begin{bmatrix} J_1 + m_2 l_1^2 & m_2 l_1 l_{c2} \cos\theta_{21}, \\ m_2 l_1 l_{c2} \cos\theta_{21} & J_2 \end{bmatrix}$$

$$c(\mathbf{q}, \dot{\mathbf{q}}) = \begin{bmatrix} -m_2 l_1 l_{c2} \sin(\theta_{21})\dot{\theta}_2^2 \\ m_2 l_1 l_{c2} \sin(\theta_{21})\dot{\theta}_1^2 \end{bmatrix}, \quad \nabla V(\mathbf{q}) = \begin{bmatrix} (m_1 + m_2) g l_{c1} \sin\theta_1 \\ m_2 g l_{c2} \sin\theta_2 \end{bmatrix}$$

Estimating the Parameters

Not all parameters are identifiable but we can "estimate" the remaining unknown parameters by a suitable parameterization

Estimating the Parameters

Not all parameters are identifiable but we can "estimate" the remaining unknown parameters by a suitable parameterization

Example

For the two link rotational robot we can divide the equations of motion by $m_2 l_1 l_{c2}$:

$$\cos(\theta_{21})\ddot{\theta}_2 - \sin(\theta_{21})\dot{\theta}_2^2 + j_1\ddot{\theta}_1 + \mu_1 g \sin\theta_1 + D_{1.}\dot{\boldsymbol{\theta}} = B_{1.}\mathbf{u}$$

$$l_1(\cos(\theta_{21})\ddot{\theta}_1 + \sin(\theta_{21})\dot{\theta}_1^2) + g\sin\theta_2 + j_2\ddot{\theta}_2 + D_{2.}\dot{\boldsymbol{\theta}} = B_{2.}\mathbf{u}$$

Estimating the Parameters

Not all parameters are identifiable but we can "estimate" the remaining unknown parameters by a suitable parameterization

- Compute $\mathbf{q}$ and $\dot{\mathbf{q}}$ by inverting the kinematics
- Approximate $\ddot{\mathbf{q}}$ using finite differences (the sampling period in the training data is small)
- Estimate the identifiable parameters by solving a least squares problem
- Since there is very little noise in the training data this approach works well

Input Matrix

- For some robots the input has a number of dimensions higher than the number of degrees of freedom, d, of the robot
- However, in these cases, only a linear subspace of dimension d is used in the training data
- We can't estimate the full matrix B but we can find an orthogonal basis, V, for this subspace:

$$B = \bar{B} V^T + B_0 V_\perp^T$$

- In controlling the system we will only make use of this subspace:

$$\mathbf{u} = V\bar{\mathbf{u}} \implies B\mathbf{u} = \bar{B}\bar{\mathbf{u}}$$

Bumblebee Control

- The dynamical system model is an affine system of the form:

$$\dot{\mathbf{x}} = A\mathbf{x} + \bar{B}\bar{\mathbf{u}} + \mathbf{c}, \quad \mathbf{x} = (\mathbf{x}_2, \mathbf{v}_2)$$

- We use an infinite time horizon LQR controller for stabilization, minimizing:

$$J = \int_0^\infty \|\mathbf{x}_2(t) - \mathbf{x}_{ref}\|^2 + \rho \|\bar{\mathbf{u}}(t)\|^2 dt$$

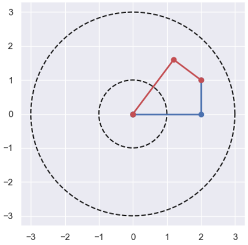
- This yields the controller:

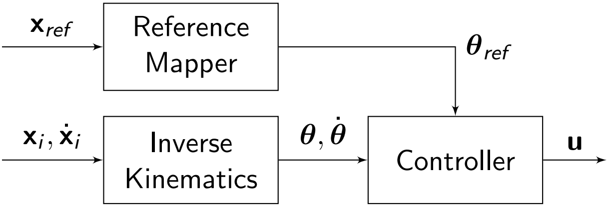
$$\bar{\mathbf{u}} = -K_p(\mathbf{x}_2 - \mathbf{x}_{ref}) - K_d \mathbf{v}_2 - \mathbf{k}$$

  which we then use as a tracking controller
  - Not ideal but we don't know the trajectory to track beforehand
  - It's simpler to have a controller that doesn't depend explicitly on time

Beetle Control

- Except in extreme cases, there are two possible configuration that lead to the same end effector position
- We can map an end effector reference into a configuration reference
  - Use previous reference to resolve ambiguity
- Allows us to control robot in joint space





Beetle Control

We adopted the following simple PD control law (with gravity compensation):

$$\bar{\mathbf{u}} = \bar{B}^{-1}\left[\nabla V(\boldsymbol{\theta}) - k_p(\boldsymbol{\theta} - \boldsymbol{\theta}_{ref}) - k_d\dot{\boldsymbol{\theta}}\right]$$

This modifies the dynamics so that the closed loop behaves as:

$$M(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + c(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + D\dot{\boldsymbol{\theta}} + \nabla V(\boldsymbol{\theta}) = \bar{B}\bar{\mathbf{u}}$$

Beetle Control

We adopted the following simple PD control law (with gravity compensation):

$$\bar{\mathbf{u}} = \bar{B}^{-1}\left[\nabla V(\boldsymbol{\theta}) - k_p(\boldsymbol{\theta} - \boldsymbol{\theta}_{ref}) - k_d\dot{\boldsymbol{\theta}}\right]$$

This modifies the dynamics so that the closed loop behaves as:

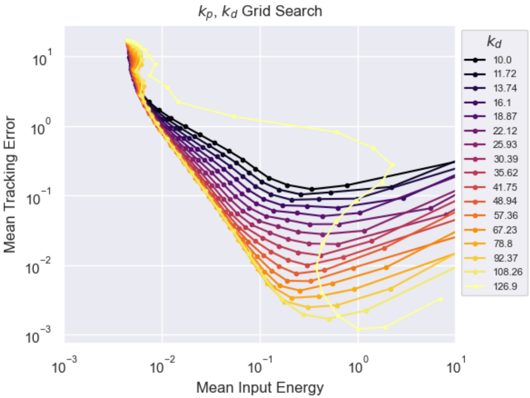
$$M(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + c(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + (D + k_d)\dot{\boldsymbol{\theta}} + k_p(\boldsymbol{\theta} - \boldsymbol{\theta}_{ref}) = 0$$

- Adding more damping
- Replacing the gravitational potential with a different potential, $k_p\|\theta - \theta_{ref}\|^2$, with a minimum at the desired configuration

Beetle Control

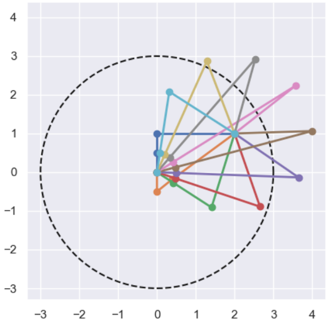
To tune the controller gains we perform a grid search

- Using the training trajectories as a reference to track
- Compute both tracking error and input energy



Butterfly Control

- Here, an end effector reference only specifies 2 out of 3 DOF
- Could use some heuristic for setting the missing degree of freedom
- But a naive choice could do poorly for some trajectories
- Opted for a transpose Jacobian control law that uses the errors in output space directly



Butterfly Control

The adopted transpose Jacobian PD control law (with gravity compensation) is given by:

$$\bar{\mathbf{u}} = \bar{B}^{-1}\left\{\nabla V(\boldsymbol{\theta}) - J^T(\boldsymbol{\theta})\left[k_p(\boldsymbol{x}_3 - \boldsymbol{x}_{ref}) + k_d\dot{\boldsymbol{x}}_3\right]\right\}$$

This modifies the dynamics so that the closed loop behaves as:

$$M(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + c(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + D\dot{\boldsymbol{\theta}} + \nabla V(\boldsymbol{\theta}) = \bar{B}\bar{\mathbf{u}}$$

Butterfly Control

The adopted transpose Jacobian PD control law (with gravity compensation) is given by:

$$\bar{\mathbf{u}} = \bar{B}^{-1}\left\{\nabla V(\boldsymbol{\theta}) - J^T(\boldsymbol{\theta})\left[k_p(\boldsymbol{x}_3 - \boldsymbol{x}_{ref}) + k_d\dot{\boldsymbol{x}}_3\right]\right\}$$

This modifies the dynamics so that the closed loop behaves as:

$$\begin{aligned} M(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + c(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + (D + k_d J^T(\boldsymbol{\theta})J(\boldsymbol{\theta}))\dot{\boldsymbol{\theta}} \\ + k_p J^T(\boldsymbol{\theta})(h_3(\boldsymbol{\theta}) - \boldsymbol{x}_{ref}) = 0 \end{aligned}$$

- Tune k_p and k_d with a grid search

Conclusion

- A typical control theory approach to the problem seemed to work well
- Even though I was able to find a good model for the robot it should still be possible to design effective controllers without detailed knowledge of the system