ECH 267 Nonlinear Control Theory
Project Abstract

Jonathan Dorsey: Department of Mechanical & Aerospace Engineering

February 8, 2021

Overview

The objective of this project is to implement an optimal Path Planner using Model Predictive Control (MPC). This methodology will be applied to a simplified robotic arm/manipulator. The responsibility of the MPC will be to generate the 'optimal' path, from a specified starting waypoint to a specified ending waypoint, as well as facilitate the control of the arm from the current position to the next.

This formulation should be able to use MPC as online means of generating optimal paths, in real-time, as well as avoid obstacles, which might exist in the robots workspace.

Problem Statement

The formulation of this project can be viewed as a Nonlinear Programming (NLP) problem. Using a standard formulation, we can construct the appropriate Optimal Control problem for the MPC formulation, which solves the NLP problem for a receding prediction horizon.

The Optimal Control formulation for **Multiple Direct Shooting** can written as shown below...

$$\begin{array}{l}
\min_u J_N(\mathbf{x}_0, \mathbf{u}) = \sum_{k=0}^{N-1} \ell(\mathbf{x}_\mathrm{u}(k), \mathbf{u}(k)) \\
\text{ s.t.: } \mathbf{x}_\mathbf{u}(k+1) = \mathbf{f}(\mathbf{x}_\mathbf{u}(k), \mathbf{u}(k)) \\
\mathbf{x}_\mathbf{u}(0) = \overline{\mathbf{x}}_0, \\
\mathbf{u}(k) \in U, \forall k \in [0, N-1] \\
\mathbf{x}_\mathbf{u}(k) \in X, \forall k \in [0, N]
\end{array}$$

It should be noted that the multiple shooting variant of this formulation is more suited than the single shooting version, as it is more efficient and handles constraints better according to the nonlinear nature of the problem being solved.

Solution Methodology

This project intends to use the optimization framework **CasADi**, which is an open source package for symbolic mathematics with a particular emphasis on solving optimization problems. Using this framework it is relatively straightforward to cast the NLP formulation for this project into a simple coding syntax. Additionally, CasADi has a number of frontend APIs including Matlab, Python, and C++ enabling a variety of languages for implementation.

Since CasADi directly solves the NLP formulation of the optimization, we can now cast the multiple shooting MPC optimal control formulation into its general NLP form. In general terms, this is given by the following expression..

$$\mathbf{w} = \begin{bmatrix} \mathbf{u}_0 & \cdots & \mathbf{u}_{N-1}, \mathbf{x}_0 & \cdots & \mathbf{x}_N \end{bmatrix} \text{ Problem Decision variables}$$

$$\min_{\pi} \Phi(w)$$

$$\text{subject to :}$$

$$g_1(w) = \begin{bmatrix} g_1\left(x_0, \mathbf{u}_0\right) \\ \vdots \\ g_1\left(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}\right) \\ g_1\left(\mathbf{x}_N\right) \end{bmatrix} \leq 0, g_2(\mathbf{w}) = \begin{bmatrix} \overline{\mathbf{x}}_0 - \mathbf{x}_0 \\ \mathbf{f}\left(\mathbf{x}_0, \mathbf{u}_0^{\triangle}\right) - \mathbf{x}_1 \\ \vdots \\ \mathbf{f}\left(\mathbf{x}_{N-1}, \mathbf{u}_{N-1}\right) - \mathbf{x}_N \end{bmatrix} = 0$$

By selecting the correct "cost" function, and deriving the appropriate dynamics of the system, it should be a straightforward task to implement these within the CasADi framework.

Modeling

In order to create a simulatable system, this project requries the developement of mathematical model that describes the robot arm, including its dynamics and kinematics. Not only is this model required for the simulation, but the developement of a model is pivotal for the use of MPC, where the model itself is used to make predictions to find the optimal controller inputs.

Modeling Assumptions:

In order to keep the modeling of the robot simple this project plans to following the following simplications and assumptions for the mathematical modeling of the robot.

1. **Robot Construction Assumptions**

   Typically, simplified robot models are assumed to be construction of lower pairs including revolute or prismatic joints. Each member of the robots construction are assumed to be rigid bodies with no appreciable compliance. This formulation ensures that the robot is not an under actuated system, as well as keeping the dynamics simple and straightforward. These are reasonable assumptions for most robot manipulators of simple construction.

2. **Actuator Assumptions**

   With the exception of actuator saturation, this project will assume perfect actuators, such that modeling of the actuators is not required and that if 'X' Newton-meters of torque is commanded, the actuator will supply that amount granted that it does not exceed the saturation limits specified. This enables the controller to directly command model by controlling joint torques.

Modeling Dynamics

In general, dynamics of robot manipulators can be made by the application of Lagrange Equations, to the chain of revolute/prismatic joints given there geometry and physical properties. Once derived the system can then be implemented in state space form where, numerical integrators (Euler's Method or RK4) can be applied to determine the evolution of the states in time under the action of control inputs.

Modeling Forward & Inverse Kinematics

The forward kinematics of a system are invaluable in the simulation of a manipulator type robot since given the value and orientation of the joint variables, it is possible to determine the Cartesian coordinates of the end-effector of the system.

Inverse kinematics, on the other hand, are invaluable for determining the joint positions and orientations given the current or desired position of the end-effector.

Denavit-Hartenberg Parameters:

One of the unique convetions in robotics is that of the Denavit-Hartenberg Parameters (DH Parameters) which enable a minimum set of four parameters per link fully define the kinematics of the robot. These are not unique, but merely a convention that is nearly universal in robot design and also has a clear intuitive meaning for each parameter.

Testing

References