

Nonlinear Model Predictive Control (NMPC) with CasADi

=====

Overview

This document explains the implementation of a Nonlinear Model Predictive Control (NMPC) scheme for a differential-drive robot, based on the provided Python code that uses CasADi.

Key Concepts

1. **Differential Drive Robot Model**

- State: $[x, y, \theta]$
- Control: $[v, \omega]$
- Kinematics:
 - $\dot{x} = v \cdot \cos(\theta)$
 - $\dot{y} = v \cdot \sin(\theta)$
 - $\dot{\theta} = \omega$

2. **Prediction Horizon**

- The controller predicts the robot's motion over $N = 20$ future timesteps.
- Each timestep is of duration $dt = 0.1$ s.

3. **Cost Function**

- Penalizes deviation from reference trajectory (Q matrix).
- Penalizes control effort (R matrix).
- $Q = \text{diag}([1.0, 1.0, 0.05])$
- $R = \text{diag}([0.08, 0.09])$

4. **Constraints**

- Velocity bounds: $v \in [-0.2, 0.2]$ m/s
- Angular velocity bounds: $\omega \in [-\pi/4, \pi/4]$ rad/s
- Dynamics constraints ensure that predicted states follow robot motion equations.

5. **Optimization Problem**

- Variables: Future states X ($3 \times (N+1)$) and controls U ($2 \times N$).
- Objective: Minimize cost subject to constraints.
- Solver: IPOPT (via CasADi `nlp`).

Implementation Details

- `DiffDriveNMPC.__init__`: Initializes parameters, cost weights, and bounds.
- `__define_dynamics`: Defines robot kinematics in CasADi symbolic form.
- `__setup_solver`: Builds the nonlinear optimization problem with constraints and cost.
- `__solve`:
 1. Pads reference trajectory to horizon length.
 2. Creates optimization variables (X, U).
 3. Solves NLP with IPOPT.
 4. Extracts the first control input $[v, \omega]$.

Automation Workflow

1. Current robot state $x_0 = [x, y, \theta]$ is given.
2. Desired trajectory ($x_{ref}, y_{ref}, \theta_{ref}$) is provided.
3. NMPC builds a prediction and optimizes controls.
4. Solver returns optimal v and ω for the current step.

5. Apply control → repeat at next timestep.

Benefits of NMPC

- Handles nonlinear robot dynamics.
- Optimizes performance while respecting constraints.
- Flexible: easy to adjust horizon length, cost weights, and constraints.

Conclusion

The code demonstrates a compact and practical NMPC setup for differential-drive robots. CasADi enables efficient symbolic modeling and nonlinear optimization, making this approach suitable for real-time autonomous navigation tasks.