Underactuated Robots

Optimization methods for Planning and Control: Introduction

Tommaso Belvedere

DIAG Robotics Lab

Introduction

- controlling robotic systems with traditional techniques can become impractical when we have to deal with
    - complex models
    - constraints
    - complex objectives
- on the other hand, optimization can be a powerful tool to find a solution to such problems
- this methods can be applied underactuated systems with little effort (but intrinsic limits are still present!)

The general problem

- many problems can be framed as the minimization of some function of the state/input trajectory while satisfying proper constraints

$$\begin{cases} \displaystyle\min_{\boldsymbol{u}(t)} \int_0^T \ell(\boldsymbol{x}(t), \boldsymbol{u}(t)) dt + \ell_f(\boldsymbol{x}(T)) \\ \text{subject to:} \\ - \text{ initial condition } \boldsymbol{x}(0), \\ - \text{ the dynamics } \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \\ - \text{ constraints } \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t)) \leq \boldsymbol{0} \end{cases}$$

- this formulation can be adapted and extended to many problems: tracking, planning, minimum time, minimum energy, infinite time...

Example: planning

- starting from an initial state $\boldsymbol{x}_0$ we want to find a feasible trajectory that reaches the goal $\boldsymbol{x}_g$ in a finite time T

$$\begin{cases} \min\limits_{\boldsymbol{u}(t)} \int_0^T \|\boldsymbol{x}(t) - \boldsymbol{x}_g\|^2 + \rho \|\boldsymbol{u}(t)\|^2 dt \\ \text{subject to:} \\ \boldsymbol{x}(0) = \boldsymbol{x}_0 \\ \boldsymbol{x}(T) = \boldsymbol{x}_g \\ \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) \leq \boldsymbol{0} \qquad \text{e.g., input/velocity limits, obstacle avoidance...} \end{cases}$$

- we have a state penalty in the cost function to get to the goal faster, if possible

Example: trajectory tracking

- given a desired trajectory $\boldsymbol{x}_d(t)$ and possibly a feedforward input $\boldsymbol{u}_d(t)$, the objective is to make $\boldsymbol{x} \to \boldsymbol{x}_d$ by minimizing the deviation from the trajectory

$$\begin{cases} \min\limits_{\boldsymbol{u}(t)} \int_0^T \|\boldsymbol{x}(t) - \boldsymbol{x}_d(t)\|^2 + \rho\|\boldsymbol{u}(t) - \boldsymbol{u}_d(t)\|^2 dt \\ \text{subject to:} \\ \boldsymbol{x}(0) = \boldsymbol{x}_0 \\ \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) \leq \boldsymbol{0} \qquad \text{e.g., input/velocity limits, obstacle avoidance...} \end{cases}$$

- we can use constraints to impose a convergence time, bound the error

Example: trajectory tracking (continued)

$$\begin{cases} \min\limits_{\boldsymbol{u}(t)} \displaystyle\int_0^T \|\boldsymbol{x}(t) - \boldsymbol{x}_d(t)\|^2 + \rho\|\boldsymbol{u}(t) - \boldsymbol{u}_d(t)\|^2 dt \\ \text{subject to:} \\ \boldsymbol{x}(0) = \boldsymbol{x}_0 \\ \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) \leq \boldsymbol{0} \qquad \text{e.g., input/velocity limits, obstacle avoidance...} \end{cases}$$

- in general, exact tracking can be achieved only if we know $\boldsymbol{u}_d(t)$
- we may have an output trajectory $\boldsymbol{y}_d$ with $\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x})$ instead of the full state (beware the zero dynamics!)

Local and Global methods

- the first distinction we can make is between global and local methods, which determines the kind of solution we seek
- global methods search for an optimal control policy $\boldsymbol{u} = \boldsymbol{k}(\boldsymbol{x})$ (feedback/closed-loop)
  - ideal solution, but often too complex to solve
  - **examples:** Dynamic Programming, Hamilton-Jacobi-Bellman
- local methods search for a pair of trajectories for input $\boldsymbol{u}(t)$ and state $\boldsymbol{x}(t)$ starting from some $\boldsymbol{x}(0)$ (feedforward/open-loop)
  - usually fall under the umbrella term of trajectory optimization
  - not complete: even if a solution exist, you have no guarantee of finding it
  - **examples:** Direct Methods, Differential Dynamic Programming, Pontryagin's Minimum Principle

Direct vs Indirect methods

- depending on how the problem is tackled, we can distinguish between direct and indirect methods
- direct methods first transcribe the problem into a general Nonlinear Program (NLP), to be solved with appropriate numerical optimization techniques
- indirect methods are based on optimal control principles like DP,HJB,PMP which provide necessary and/or sufficient conditions for optimality