AA 274: Principles of Robotic Autonomy | Winter 2018

Lecture 3: Open-Loop and Closed-Loop Motion Control

Introduction

This lecture discusses the main techniques in optimal control and trajectory optimization.

First, we will review constrained optimization by formulating the problem using the Lagrangian. Then we will explore two different methods in solving an optimal control problem: we will thoroughly explore some equations behind Indirect Methods, enabling its computation along with examples, and give an overview on another approach: Direct Methods. Then we will explore differentially flat problems, and how these enable the use of algebraic equations in non-linear problem solving. Next we will explore closed-loop control, and some implementations. Finally, we will combine open-loop and closed-loop approaches by introducing trajectory tracking.

3.1 Constrained Optimization

Before presenting the optimal control problem it is first useful to briefly review a standard constrained optimization problem given by

$$\begin{aligned} &\min_{u} \ f(x) \\ &\text{subject to } h_i(x) = 0, \quad i = 1, \dots, m \end{aligned} \tag{3.1}$$

Problem 3.1 is solved by introducing the ***Lagrangian*** and then generating the necessary conditions for optimality (NOC). The Lagrangian is given by

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i h_i(x)$$

and the necessary conditions for optimality are

$$\begin{aligned} \nabla_x L(x^*, \lambda^*) &= 0 \\ \nabla_\lambda L(x^*, \lambda^*) &= 0 \end{aligned} \tag{3.2}$$

One way to think about the necessary conditions (Eq. 3.2) for optimality is as a filter that takes in all points x and outputs a subset of potential optimal points x^*. If several potential points exist that satisfy the NOC, these could be local optima.

3.2 Optimal Control Problem

$$\begin{aligned} \min_u \; & h(x(t_f), t_f) + \int_{t_0}^{t_f} g(x(t), u(t), t) dt \\ \text{subject to } & \dot{x}(t) = a(x(t), u(t), t) \\ & x(t) \in \mathcal{X}, \quad u(t) \in \mathcal{U} \end{aligned} \tag{3.3}$$

In general, $x(t) \in R^n$, $u(t) \in \mathbb{R}^m$, and the initial condition $x(0) = x_0$ is given. In the context of this course the constraint $x(t) \in \mathcal{X}$ is typically relaxed to just be $x(t) \in \mathbb{R}^n$. In other words there are no state constraints.

The problem given by Eq. 3.3 contains many details. In the objective function, the term $h(x(t_f), t_f)$ represents a cost associated with the end condition. For example if you wanted to minimize time you could have $h = t_f$. Also in the objective function, the integral produces a path cost, or in other words produces penalties along the way from t_0 to t_f. This would be useful if you wanted to minimize fuel use: for example you could set $g = u^2$.

The first constraint in the problem defines the differential constraints that represent the system dynamics. These in general are nonlinear which is why they are represented as $a(x(t), u(t), t)$.

Problem 3.3 is typically solved to yield an open-loop trajectory and control, which are denoted as $x^*(t)$ and $u^*(t)$. Note that these are entire trajectories through time, not just individual points as we saw in Eq. 3.1. Since the control is open loop we also typically write

$$u^*(t) = f(x(t_0), t)$$

to indicate that the control is a function of time only, and does not utilize any state feedback.

It is important to realize the difference between solving these optimal control problems and solving general optimization problems. As mentioned, in optimal control we are solving for *trajectories*, and not just points. For any given set of boundary conditions there are an infinite number of possible trajectories, and with the dynamics constraints also involved this becomes a formidable task.

The tools used to solve this Problem 3.3 are generally classified as ***Indirect Methods*** and ***Direct Methods***.

3.2.1 Indirect Methods

Given the control inputs and thus the control function (either an open loop or closed loop scenario) we look to optimize over, one can proceed with either a direct or indirect method for solving. A good thought process to follow for an indirect method is to ***"first optimize, then discretize"***. The indirect method works by imposing all NCOs, looking at the optimal functions of time we generated, and then selecting the one that is the best. Take the unconstrained finite dimensional case of a simple function f(x) that we look to optimize over $x \in \mathbb{R}^n$:

$$\min_x \; \nabla_f(x) = 0$$

Taking the gradient of my function and looking at those candidates that survive, one can determine which values would optimize their function (looking to minimize for our case). Our parallel infinite dimensional

problem takes the solutions from NCOs as differential equations which should then be discretized to solve accordingly.

After a series of complicated derivations to establish an augmented cost function and referring to section **3.2** on how to generally handle constrained optimization, one can arrive at what is known as the Hamiltonian to solve our functions:

$$H := g(x(t), u(t), t) + p^{\mathbf{T}}(t)[a(x(t), u(t), t] \tag{3.4}$$

where the first paranthetical term denotes the running cost, $p^{\mathbf{T}}(t)$ denotes the vector of Lagrange Mutlipliers (one for each individual constraint),and the final term denoting the RHS of the differential equation. From here we can declare the NCO's that our system requires for optimality:

$$\begin{aligned} \dot{x}^*(t) &= \frac{\partial H}{\partial p} = (x^*(t), u^*(t), p^*(t), t) \\ \dot{p}^*(t) &= -\frac{\partial H}{\partial x} = (x^*(t), u^*(t), p^*(t), t) \\ 0 &= \frac{\partial H}{\partial u} = (x^*(t), u^*(t), p^*(t), t) \end{aligned} \tag{3.5}$$

The first equation represents the derivative of the Hamiltonian with respect to your individual Lagrange Multipliers which interestingly enough gives back the kinematic state equations of the robot. The second equation is called the costate of your function - looking to optimize the function over continuous infinite time, the lagrange multipliers become functions of time. The final equation is the optimality condition which is simply the Hamiltonian differiniated with respect to the constraint producing an algebraic equation. This is **under the assumption that we have no state constraints** which is sufficient for our focus of the class. The tedious part of optimal control is now solving these equations: we have produced 2N optimization variables in $x^*(t)$ and $p^*(t)$ with one M varibale $u^*(t)$ totaling 2N + M variables alongside our 2N differinitable equations and M algebraic equation.

To solve we must impose boundary conditions on our function with respect to the robot's final position and time such that:

$$[\frac{\partial h}{\partial x}(x^*(t_f), t_f - p^*(t_f))]^T \delta x_f \; + [H(x^*(t_f), u^*(t_f), p^*(t_f), t_f + \frac{\partial h}{\partial t}(x^*(t_f), t_f)]\delta x_f = 0 \tag{3.6}$$

which produces four possible solutions to satisfy the equation:

t_f = fixed and $x(t_f)$ = fixed ... $\delta t_f = 0$ and $\delta x(t_f) = 0$

$$\begin{aligned} \mathbf{BC} : x^*(t_0) &= x_0 \\ x^*(t_f) &= x_f \end{aligned}$$

t_f = fixed and $x(t_f)$ = free ... $\delta t_f = 0$ and $\delta x(t_f)$ = arbitrary

$$\begin{aligned} \mathbf{BC} : x^*(t_0) &= x_0 \\ [\frac{\partial h}{\partial x}(x^*(t_f), -p^*(t_f))] &= 0 \end{aligned}$$

t_f = free and $x(t_f)$ = fixed ... δt_f = arbitrary and $\delta x(t_f) = 0$

$$\mathbf{BC} : x^*(t_0) = x_0$$
$$x^*(t_f) = x_f$$
$$[H(x^*(t_f), u^*(t_f), p^*(t_f), t_f + \frac{\partial h}{\partial t}(x^*(t_f), t_f)]\delta x_f = 0$$

t_f = free and $x(t_f)$ = free ... δt_f = arbitrary and $\delta x(t_f)$ = arbitrary

$$\mathbf{BC} : x^*(t_0) = x_0$$
$$[\frac{\partial h}{\partial x}(x^*(t_f), -p^*(t_f))] = 0$$
$$[H(x^*(t_f), u^*(t_f), p^*(t_f), t_f + \frac{\partial h}{\partial t}(x^*(t_f), t_f)]\delta x_f = 0$$

The last scenario is what we call a free final time state and introduces an unkown variable to our system of equations making it a set of 2N + M + 1. Think of this as an optimal control problem where we look to minimize the cost of our function at any given final time. We look at a thorough example for clarification.

Let's consider an example of free final time. Consider a point that moves in one dimension. Control input of the system is acceleration of the point. Boundary conditions for positions and velocities at initial and final times are given, but the final time is not fixed.

$$\ddot{x} = u, x(0) = 10, \dot{x} = 0, x(t_f) = 0, \dot{x}(t_f) = 0 \tag{3.7}$$
$$J = \frac{1}{2}\alpha t_f^2 + \frac{1}{2}\int_{t_0}^{t_f} bu^2(t)\mathrm{d}t \tag{3.8}$$

Cost function shown in (3.8) is typical in control problems. The final term in cost function tries to minimize total time to get back to origin, while the stagewise cost attempts to keep acceleration from being too high, which requires a lot of control effort. Analytical solution (3.9) of the final time is pretty reasonable. When b increases, penalty on control effort will be more serious, which causes the acceleration to be relatively low and t_f to be longer. When α increases, the final cost term will become dominant, so t_f will become shorter.

$$t_f = (1800b/\alpha)^{1/5} \tag{3.9}$$

In order to solve the control problem, firstly we need to define state $\mathbf{z}$ and Hamiltonian. From the first equation of (3.7) we have two state constraints (in which $x_1 = x$). Thus we have two Lagrange multipliers p_1 and p_2 correspondingly.

$$\begin{aligned} \dot{x_1} &= x_2 \\ \dot{x_2} &= u \end{aligned} \tag{3.10}$$

According to the trick for free final time, we have a dummy state r that is equal to t_f with dynamics $\dot{r} = 0$. So state $\mathbf{z}$ has five elements.

$$\mathbf{z} = [z_1, z_2, z_3, z_4, z_5]^T = [x_1^*, x_2^*, p_1^*, p_2^*, r^*]^T \tag{3.11}$$

Then we form Hamiltonian H based on (3.8) and (3.10).

$$H = \frac{1}{2}bu^2 + p_1 x_2 + p_2 u \tag{3.12}$$

We can get Hamiltonian equations from H.

$$\begin{aligned}
\dot{x_1^*} &= \frac{\partial H}{\partial p_1}(\mathbf{x}^*(t), u^*(t), \mathbf{p}^*(t), t) = x_2^* \\
\dot{x_2^*} &= \frac{\partial H}{\partial p_2}(\mathbf{x}^*(t), u^*(t), \mathbf{p}^*(t), t) = u^* \\
\dot{p_1^*} &= -\frac{\partial H}{\partial x_1}(\mathbf{x}^*(t), u^*(t), \mathbf{p}^*(t), t) = 0 \\
\dot{p_2^*} &= -\frac{\partial H}{\partial x_2}(\mathbf{x}^*(t), u^*(t), \mathbf{p}^*(t), t) = -p_1^* \\
0 &= \frac{\partial H}{\partial u}(\mathbf{x}^*(t), u^*(t), \mathbf{p}^*(t), t) = bu^* + p_2^*
\end{aligned} \tag{3.13}$$

Boundary conditions include the original ones and substitution for free t_f and fixed $\mathbf{x}(t_f)$ problem.

$$x_1^*(0) = 10, x_2^*(0) = 0, x_1^*(t_f) = 0, x_2^*(t_f) = 0 \tag{3.14}$$

$$\begin{aligned}
H(\mathbf{x}^*(t_f), u^*(t_f), \mathbf{p}^*(t_f), t_f) + \frac{\partial h}{\partial t}(\mathbf{x}^*(t_f), t_f) = 0 \\
\frac{1}{2}bu^*(t_f)^2 + p_1^*(t_f)x_2^*(t_f) + p_2^*(t_f)u^*(t_f) + \alpha t_f = 0
\end{aligned} \tag{3.15}$$

When you put $u^* = -\frac{1}{b}p_2^*$ and $x_2^*(t_f) = 0$ into (3.15), you can get a more clear version of the last boundary equation.

$$-\frac{1}{2b}p_2^*(t_f)^2 + \alpha t_f = 0 \tag{3.16}$$

After substitution of $u^* = -\frac{1}{b}p_2^*$ into other Hamiltonian equations in (3.13), we unify the notation of states to $\mathbf{z}$ and rescale time with $\tau = t/t_f$. BVP becomes

$$\frac{\mathrm{d}\mathbf{z}}{\mathrm{d}\tau} = t_f \frac{\mathrm{d}\mathbf{z}}{\mathrm{d}t} = z_5 \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{b} & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{z} \tag{3.17}$$

After the same operation, BC become

$$\begin{aligned} z_1(0) = 10, z_2(0) = 0, z_1(1) = 0, z_2(1) = 0, \\ -\frac{1}{2b} z_4(1)^2 + \alpha z_5(1) = 0 \end{aligned} \tag{3.18}$$

Then we can input these BVP and BC into Python to solve the problem.

References

[1] Aicardi, Casalino, Bicchi, Balestrino. *Closed loop steering of unicycle like vehicles via Lyapunov techniques*. IEEE, Genoa, Italy, 1995.

[2] Wang, Zhanshan, Liu, Zhenwei, Zheng, Chengde *Qualitative Analysis and Control of Complex Neural Networks with Delays*. Springer-Verlag Berlin Heidelberg, 2016.

[3] Richard M Murray, *Optimization-based control*. California Institute of Technology CA, 2009.

[4] D. K. Kirk. *Optimal Control Theory: An introduction*. Dover Publications, 2004.

[5] Ascher, U., & Russell, R. D. *Reformulation of boundary value problems into "standard" form*. SIAM review, 23(2), 238-254, 1981.

[6] Notes from ee363 at Stanford University *https://stanford.edu/class/ee363/lectures/lyap.pdf*

Contributors

Winter 2019: [Your Names Here]
Winter 2018: Joseph Lorenzetti, Kenneth Hoffmann, Oriana Peltzer, Zhe Huang, William Mangram