Principles of Robot Autonomy I

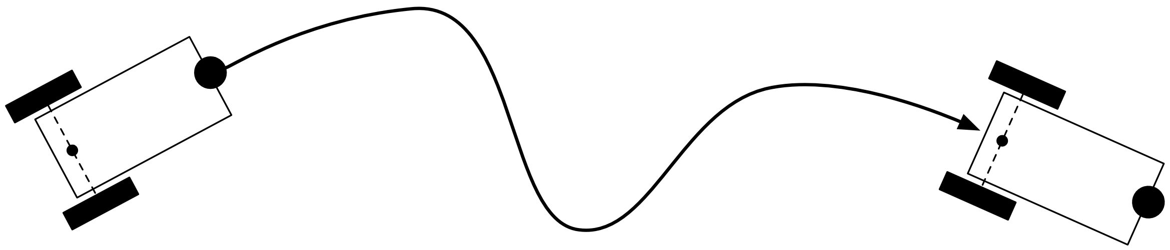
Advanced methods for trajectory optimization

Stanford University

Autonomous Systems Lab

Motion control

- Given a nonholonomic system, how to control its motion from an initial configuration to a final, desired configuration



- Aim
  - Revisit trajectory planning as optimal control problem
  - Learn key ideas underpinning indirect methods for optimal control
  - Establish link between direct and indirect methods
- Readings
  - D. K. Kirk. Optimal Control Theory: An introduction. 2004.

Optimal control problem

The problem:

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

where $\boldsymbol{x}(t) \in R^n, \boldsymbol{u}(t) \in R^m$, and $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$

- In trajectory optimization, we typically consider the case

$$\mathcal{X} = R^n$$

Open-loop control

- We want to find

$$\mathbf{u}^*(t) = \mathbf{f}(\mathbf{x}(t_0), t)$$

- In general, two broad classes of methods:

  1. Indirect methods: attempt to find a minimum point "indirectly," by solving the necessary conditions of optimality $\Rightarrow$ "First optimize, then discretize"

  2. Direct methods: transcribe infinite problem into finite dimensional, nonlinear programming (NLP) problem, and solve NLP $\Rightarrow$ "First discretize, then optimize"

Preliminaries: constrained optimization

$$\begin{array}{ll} \min & f(\mathbf{x}) \\ \text{subject to} & h_i(\mathbf{x}) = 0, \qquad i = 1, \ldots, m \end{array}$$

- Form Lagrangian function $L: R^{n+m} \rightarrow R$

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

- If $\boldsymbol{x}^*$a is a local minimum which is *regular*, the NOC conditions are

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*, \lambda^*) = 0$$
$$\nabla_{\lambda} L(\mathbf{x}^*, \lambda^*) = 0$$

- First order condition represents a system of $n + m$ equations with $n + m$ unknowns

Indirect methods: NOC

Assume no state/control constraints

- Form Hamiltonian $H := g(\mathbf{x}(t), \mathbf{u}(t), t) + \mathbf{p}^T(t)[\mathbf{a}(\mathbf{x}(t), \mathbf{u}(t), t)]$
- Hamiltonian equations

$$\dot{\mathbf{x}}^*(t) = \frac{\partial H}{\partial \mathbf{p}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t)$$

$$\dot{\mathbf{p}}^*(t) = -\frac{\partial H}{\partial \mathbf{x}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t)$$

$$\mathbf{0} = \frac{\partial H}{\partial \mathbf{u}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t)$$

- Boundary conditions: $\mathbf{x}^*(t_0) = \mathbf{x}_0$, and

$$\left[\frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}^*(t_f), t_f) - \mathbf{p}^*(t_f)\right]^T \delta \mathbf{x}_f + \left[H(\mathbf{x}^*(t_f), \mathbf{u}^*(t_f), \mathbf{p}^*(t_f), t_f) + \frac{\partial h}{\partial t}(\mathbf{x}^*(t_f), t_f)\right] \delta t_f = 0$$

Indirect methods: NOC

Assume control inequality constraints: e.g., $|u_i| \leq \bar{u}_i$ for all *i*

- Form Hamiltonian $H := g(\mathbf{x}(t), \mathbf{u}(t), t) + \mathbf{p}^T(t)[\mathbf{a}(\mathbf{x}(t), \mathbf{u}(t), t)]$
- Hamiltonian equations

$$\dot{\mathbf{x}}^*(t) = \frac{\partial H}{\partial \mathbf{p}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t)$$

$$\dot{\mathbf{p}}^*(t) = -\frac{\partial H}{\partial \mathbf{x}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t)$$

$$H(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t) \leq H(\mathbf{x}^*(t), \mathbf{u}(t), \mathbf{p}^*(t), t), \quad \forall \mathbf{u}(t) \in \mathcal{U}$$

- Boundary conditions: $\mathbf{x}^*(t_0) = \mathbf{x}_0$, and

$$\left[\frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}^*(t_f), t_f) - \mathbf{p}^*(t_f)\right]^T \delta \mathbf{x}_f + \left[H(\mathbf{x}^*(t_f), \mathbf{u}^*(t_f), \mathbf{p}^*(t_f), t_f) + \frac{\partial h}{\partial t}(\mathbf{x}^*(t_f), t_f)\right] \delta t_f = 0$$

Substitutions for boundary conditions

Problem		Substitution
t_f	fixed	$\delta t_f = 0$
$\mathbf{x}(t_f)$	fixed	$\delta \mathbf{x}_f = 0$

BC

$$\mathbf{x}^*(t_0) = \mathbf{x}_0$$
$$\mathbf{x}^*(t_f) = \mathbf{x}_f$$

Problem		Substitution
t_f	fixed	$\delta t_f = 0$
$\mathbf{x}(t_f)$	free	$\delta \mathbf{x}_f$ arbitrary

BC

$$\mathbf{x}^*(t_0) = \mathbf{x}_0$$
$$\frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}^*(t_f)) - \mathbf{p}^*(t_f) = \mathbf{0}$$

Problem		Substitution
t_f	free	δt_f arbitrary
$\mathbf{x}(t_f)$	fixed	$\delta \mathbf{x}_f = 0$

BC

$$\mathbf{x}^*(t_0) = \mathbf{x}_0$$
$$\mathbf{x}^*(t_f) = \mathbf{x}_f$$
$$H(\mathbf{x}^*(t_f), \mathbf{u}^*(t_f), \mathbf{p}^*(t_f), t_f) + \frac{\partial h}{\partial t}(\mathbf{x}^*(t_f), t_f) = 0$$

Problem		Substitution
t_f	free	δt_f arbitrary
$\mathbf{x}(t_f)$	free	$\delta \mathbf{x}_f$ arbitrary

BC

$$\mathbf{x}^*(t_0) = \mathbf{x}_0$$
$$\frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}^*(t_f), t_f) - \mathbf{p}^*(t_f) = \mathbf{0}$$
$$H(\mathbf{x}^*(t_f), \mathbf{u}^*(t_f), \mathbf{p}^*(t_f), t_f) + \frac{\partial h}{\partial t}(\mathbf{x}^*(t_f), t_f) = 0$$

Indirect methods: practical aspects

Reference for NOC: D. K. Kirk. Optimal Control Theory: An introduction. Dover Publications, 2004.

In practice: To obtain solution to the necessary conditions for optimality, one needs to solve two-point boundary value problems

- For example, in Python: https://pythonhosted.org/scikits.bvp_solver/
- Allows to solve problem of the form

$$\dot{\mathbf{z}} = \mathbf{g}(\mathbf{z}, t), \qquad \mathbf{l}(\mathbf{z}(t_0), \mathbf{z}(t_f)) = \mathbf{0}$$

- Syntax: `solve(bvp_problem, solution_guess)`
- In Matlab: `bvp4c`

Example

$$\dot{z}_1(t) = z_2(t)$$
$$\dot{z}_2(t) = -|z_1(t)|$$
$$z_1(0) = 0$$
$$z_1(4) = -2$$

Extensions

- What about problems whose necessary conditions do not fit directly the "standard" form (e.g., free end time problems)?
- Handy tricks exist to convert problems into standard form: Ascher, U., & Russell, R. D. (1981). Reformulation of boundary value problems into "standard" form. SIAM review, 23(2), 238-254.

Important case: free final time (Problem 4 in pset)

1. Rescale time so that $\tau = t/t_f$, then $\tau \in [0,1]$
2. Change derivatives $\frac{d}{d\tau} := t_f \frac{d}{dt}$
3. Introduce dummy state *r* that corresponds to t_f with dynamics $\dot{r} = 0$
4. Replace all instances of t_f with *r*

Example

- Dynamics:

$$\ddot{x} = u,\ x(0) = 10,\ \dot{x}(0) = 0,\ x(t_f) = 0,\ \dot{x}(t_f) = 0$$

- Cost:

$$J = \frac{1}{2}\alpha t_f^2 + \frac{1}{2}\int_{t_0}^{t_f} b\, u^2(t)\, dt$$

- Analytical solution gives:

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

Example (solution)

- Define state as $\boldsymbol{z} = [\boldsymbol{x}, \boldsymbol{p}, r]$
- BC are:

$$x_1(0) = 10,\ x_2(0) = 0,\ x_1(t_f) = 0,\ x_2(t_f) = 0,$$
$$-\,0.5b(-p_2(t_f)/b)^2 + \alpha t_f = 0$$

- BVP becomes

$$\frac{d\mathbf{z}}{d\tau} = t_f \frac{d\mathbf{z}}{dt} = z_5 \begin{bmatrix} A & -B\,[0\ 1]/b & 0 \\ 0 & -A' & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{z}$$

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

- BC become

$$z_1(0) = 10,\ z_2(0) = 0,\ z_1(1) = 0,\ z_2(1) = 0,$$
$$-\,0.5b(-z_4(1)/b)^2 + \alpha z_5(1) = 0$$

Direct methods - nonlinear programming transcription

$$\min \int_{t_0}^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), t)\ dt$$

(OCP)

$$\dot{\mathbf{x}}(t) = \mathbf{a}(\mathbf{x}(t), \mathbf{u}(t), t),\ \ t \in [t_0, t_f]$$

$$\mathbf{x}(0) = \mathbf{x}_0,\ \ \mathbf{x}(t_f) \in M_f$$

$$\mathbf{u}(t) \in U \subseteq \mathbb{R}^m,\ \ t \in [t_0, t_f]$$

Forward Euler time discretization

1. Select a discretization $0 = t_0 < t_1 < \cdots < t_N = t_f$ for the interval $[t_0, t_f]$ and, for every $i = 0, \ldots, N-1$, define $\mathbf{x}_i \sim \mathbf{x}(t)$, $\mathbf{u}_i \sim \mathbf{u}(t)$, $t \in (t_i, t_{i+1}]$ and $\mathbf{x}_0 \sim \mathbf{x}(0)$

2. By denoting $h_i = t_{i+1} - t_i$, (**OCP**) is transcribed into the following nonlinear, constrained optimization problem

$$\min_{(\mathbf{x}_i, \mathbf{u}_i)} \sum_{i=0}^{N-1} h_i g(\mathbf{x}_i, \mathbf{u}_i, t_i)$$

(NLOP)

$$\mathbf{x}_{i+1} = \mathbf{x}_i + h_i \mathbf{a}(\mathbf{x}_i, \mathbf{u}_i, t_i), \qquad i = 0, \ldots, N-1$$

$$\mathbf{u}_i \in U\ , i = 0, \ldots, N-1\ , \qquad F(\mathbf{x}_N) = 0$$

Direct methods - nonlinear programming transcription

Consistency of Time Discretization

Is this approximation consistent with the original formulation?

Yes!

Indeed, the KKT conditions for **(NLOP)** converge to the necessary optimality conditions for **(OCP)**, that are given by the Pontryagin's Minimum Principle, when $h_i \longrightarrow 0$

Forward Euler time discretization

1. Select a discretization $0 = t_0 < t_1 < \cdots < t_N = t_f$ for the interval $[t_0, t_f]$ and, for every $i = 0, \dots, N-1$, define $\mathbf{x}_i \sim \mathbf{x}(t)$, $\mathbf{u}_i \sim \mathbf{u}(t)$, $t \in (t_i, t_{i+1}]$ and $\mathbf{x}_0 \sim \mathbf{x}(0)$
2. By denoting $h_i = t_{i+1} - t_i$, (**OCP**) is transcribed into the following nonlinear, constrained optimization problem

$$\min_{(\mathbf{x}_i, \mathbf{u}_i)} \sum_{i=0}^{N-1} h_i g(\mathbf{x}_i, \mathbf{u}_i, t_i)$$

(NLOP)

$$\mathbf{x}_{i+1} = \mathbf{x}_i + h_i \mathbf{a}(\mathbf{x}_i, \mathbf{u}_i, t_i), \qquad i = 0, \dots, N-1$$

$$\mathbf{u}_i \in U \,, i = 0, \dots, N-1 \,, \qquad F(\mathbf{x}_N) = 0$$

Consistency of time discretization

Simplified Formulation

$$\min \int_0^{t_f} g\big(\mathbf{x}(t), \mathbf{u}(t)\big)\ dt$$

(**OCP**)

$$\dot{\mathbf{x}}(t) = \mathbf{a}\big(\mathbf{x}(t), \mathbf{u}(t)\big),\ \ t \in [0, t_f]$$

$$\mathbf{x}(0) = \mathbf{x}_0$$

Pontryagin's Minimum Principle (PMP)

Recall that the necessary optimality conditions for (OCP) are given by the following expressions

- Co-state equation:

$$\dot{\mathbf{p}}(t) = -\frac{\partial \mathbf{a}}{\partial \mathbf{x}}\big(\mathbf{x}(t), \mathbf{u}(t)\big)'\mathbf{p}(t) - \frac{\partial g}{\partial \mathbf{x}}(\mathbf{x}(t), \mathbf{u}(t))$$

- Control equation:

$$\frac{\partial \mathbf{a}}{\partial \mathbf{u}}\big(\mathbf{x}(t), \mathbf{u}(t)\big)'\mathbf{p}(t) + \frac{\partial g}{\partial \mathbf{u}}\big(\mathbf{x}(t), \mathbf{u}(t)\big) = \mathbf{0}$$

Consistency of time discretization

Simplified Formulation

$$\min \int_0^{t_f} g\big(\mathbf{x}(t), \mathbf{u}(t)\big)\ dt$$

(OCP)

$$\dot{\mathbf{x}}(t) = \mathbf{a}\big(\mathbf{x}(t), \mathbf{u}(t)\big),\ t \in [0, t_f]$$

$$\mathbf{x}(0) = \mathbf{x}_0$$

Related non-linear program (NLOP)

After discretization in time:

$$\min_{(\mathbf{x}_i, \mathbf{u}_i)} \sum_{i=0}^{N-1} h_i g(\mathbf{x}_i, \mathbf{u}_i) \qquad \textbf{(NLOP)}$$

$$\mathbf{x}_i + h_i \mathbf{a}(\mathbf{x}_i, \mathbf{u}_i) - \mathbf{x}_{i+1} = \mathbf{0}, \qquad i = 0, \dots, N-1$$

Consistency of time discretization

KKT Related to (NLOP)

Denote the Lagrangian related to **(NLOP)** as

$$\mathcal{L} = \sum_{i=0}^{N-1} h_i g(\mathbf{x}_i, \mathbf{u}_i) + \sum_{i=0}^{N-1} \boldsymbol{\lambda}_i' (\mathbf{x}_i + h_i \mathbf{a}(\mathbf{x}_i, \mathbf{u}_i) - \mathbf{x}_{i+1})$$

Then, the KKT conditions related to **(NLOP)** read as:

- Derivative w.r.t. $\mathbf{x}_i$:

$$h_i \frac{\partial g}{\partial \mathbf{x}_i}(\mathbf{x}_i, \mathbf{u}_i) + \boldsymbol{\lambda}_i - \boldsymbol{\lambda}_{i-1} + h_i \frac{\partial \mathbf{a}}{\partial \mathbf{x}_i}(\mathbf{x}_i, \mathbf{u}_i)' \boldsymbol{\lambda}_i = \mathbf{0}$$

- Derivative w.r.t. $\mathbf{u}_i$:

$$h_i \frac{\partial g}{\partial \mathbf{u}_i}(\mathbf{x}_i, \mathbf{u}_i) + h_i \frac{\partial \mathbf{a}}{\partial \mathbf{u}_i}(\mathbf{x}_i, \mathbf{u}_i)' \boldsymbol{\lambda}_i = \mathbf{0}$$

Related non-linear program (NLOP)

After discretization in time:

$$\min_{(\mathbf{x}_i, \mathbf{u}_i)} \sum_{i=0}^{N-1} h_i g(\mathbf{x}_i, \mathbf{u}_i) \qquad \textbf{(NLOP)}$$

$$\mathbf{x}_i + h_i \mathbf{a}(\mathbf{x}_i, \mathbf{u}_i) - \mathbf{x}_{i+1} = \mathbf{0}, \qquad i = 0, \dots, N-1$$

Consistency of time discretization

KKT Related to (NLOP)

Denote the Lagrangian related to **(NLOP)** as

$$\mathcal{L} = \sum_{i=0}^{N-1} h_i g(\mathbf{x}_i, \mathbf{u}_i) + \sum_{i=0}^{N-1} \boldsymbol{\lambda}_i'(\mathbf{x}_i + h_i \mathbf{a}(\mathbf{x}_i, \mathbf{u}_i) - \mathbf{x}_{i+1})$$

Then, the KKT conditions related to **(NLOP)** read as:

- Derivative w.r.t. $\mathbf{x}_i$:

$$h_i \frac{\partial g}{\partial \mathbf{x}_i}(\mathbf{x}_i, \mathbf{u}_i) + \boldsymbol{\lambda}_i - \boldsymbol{\lambda}_{i-1} + h_i \frac{\partial \mathbf{a}}{\partial \mathbf{x}_i}(\mathbf{x}_i, \mathbf{u}_i)' \boldsymbol{\lambda}_i = \mathbf{0}$$

- Derivative w.r.t. $\mathbf{u}_i$:

$$h_i \frac{\partial g}{\partial \mathbf{u}_i}(\mathbf{x}_i, \mathbf{u}_i) + h_i \frac{\partial \mathbf{a}}{\partial \mathbf{u}_i}(\mathbf{x}_i, \mathbf{u}_i)' \boldsymbol{\lambda}_i = \mathbf{0}$$

Consistency with the PMP

We finally obtain:

$$\frac{\boldsymbol{\lambda}_i - \boldsymbol{\lambda}_{i-1}}{h_i} = -\frac{\partial \mathbf{a}}{\partial \mathbf{x}_i}(\mathbf{x}_i, \mathbf{u}_i)' \boldsymbol{\lambda}_i - \frac{\partial g}{\partial \mathbf{x}_i}(\mathbf{x}_i, \mathbf{u}_i)$$

$$\frac{\partial \mathbf{a}}{\partial \mathbf{u}_i}(\mathbf{x}_i, \mathbf{u}_i)' \boldsymbol{\lambda}_i + \frac{\partial g}{\partial \mathbf{u}_i}(\mathbf{x}_i, \mathbf{u}_i) = \mathbf{0}$$

Let $\mathbf{p}(t) = \boldsymbol{\lambda}_i$ for $t \in [t_i, t_{i+1}]$, $i = 0, \dots, N-1$ and $\mathbf{p}(0) = \boldsymbol{\lambda}_0$. Then, the equations above are the discretized version of the necessary conditions for **(OCP)**:

$$\dot{\mathbf{p}}(t) = -\frac{\partial \mathbf{a}}{\partial \mathbf{x}}\big(\mathbf{x}(t), \mathbf{u}(t)\big)' \mathbf{p}(t) - \frac{\partial g}{\partial \mathbf{x}}(\mathbf{x}(t), \mathbf{u}(t))$$

$$\frac{\partial \mathbf{a}}{\partial \mathbf{u}}\big(\mathbf{x}(t), \mathbf{u}(t)\big)' \mathbf{p}(t) + \frac{\partial g}{\partial \mathbf{u}}\big(\mathbf{x}(t), \mathbf{u}(t)\big) = \mathbf{0}$$

Solution approaches:

1. state and control parameterization methods

2. control parameterization methods

Direct methods – software packages

Some software packages:

- DIDO: http://www.elissarglobal.com/academic/products/
- PROPT: http://tomopt.com/tomlab/products/propt/
- GPOPS: http://www.gpops2.com/
- CasADi: https://github.com/casadi/casadi/wiki
- ACADO: http://acado.github.io/

For an in-depth study of direct and indirect methods, see AA203 “Optimal and Learning-based Control” (Spring 2020)

Next time: graph search methods for motion planning