Principles of Robot Autonomy I

Advanced methods for trajectory optimization
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
Optimal control problem

The problem:

\[
\min_{\mathbf{u}} \quad h(x(t_f), t_f) + \int_{t_0}^{t_f} g(x(t), u(t), t) \, dt
\]

subject to
\[
\dot{x}(t) = a(x(t), u(t), t) \\
x(t) \in \mathcal{X}, \quad u(t) \in \mathcal{U}
\]

where \( x(t) \in \mathbb{R}^n, u(t) \in \mathbb{R}^m, \) and \( x(t_0) = x_0 \)

• In trajectory optimization, we typically consider the case
\[
\mathcal{X} = \mathbb{R}^n
\]
Open-loop control

• We want to find

\[ u^*(t) = f(x(t_0), t) \]

• In general, two broad classes of methods:

1. **Indirect methods**: attempt to find a minimum point “indirectly,” by solving the necessary conditions of optimality ⇒ “First optimize, then discretize”

2. **Direct methods**: transcribe infinite problem into finite dimensional, nonlinear programming (NLP) problem, and solve NLP ⇒ “First discretize, then optimize”
Preliminaries: constrained optimization

$$\min \ f(x)$$
subject to $$h_i(x) = 0, \quad i = 1, \ldots, m$$

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

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

• If $$x^*a$$ is a local minimum which is regular, the NOC conditions are

$$\nabla_x L(x^*, \lambda^*) = 0$$
$$\nabla_\lambda L(x^*, \lambda^*) = 0$$

• First order condition represents a system of $$n + m$$ equations with $$n + m$$ unknowns
Indirect methods: NOC

Assume no state/control constraints

• Form Hamiltonian \( H := g(x(t), u(t), t) + p^T(t)[a(x(t), u(t), t)] \)

• Hamiltonian equations

\[
\begin{align*}
\dot{x}^*(t) &= \frac{\partial H}{\partial p}(x^*(t), u^*(t), p^*(t), t) \\
\dot{p}^*(t) &= -\frac{\partial H}{\partial x}(x^*(t), u^*(t), p^*(t), t) \\
0 &= \frac{\partial H}{\partial u}(x^*(t), u^*(t), p^*(t), t)
\end{align*}
\]

• Boundary conditions: \( x^*(t_0) = x_0 \), and

\[
\left[ \frac{\partial h}{\partial x}(x^*(t_f), t_f) - p^*(t_f) \right]^T \delta x_f + \left[ H(x^*(t_f), u^*(t_f), p^*(t_f), t_f) + \frac{\partial h}{\partial t}(x^*(t_f), t_f) \right] \delta t_f = 0
\]
Indirect methods: NOC

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

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

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

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

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

- Boundary conditions: \(x^*(t_0) = x_0\), and

\[
\left[\frac{\partial h}{\partial x}(x^*(t_f), t_f) - p^*(t_f)\right]^T \delta x_f + \left[H(x^*(t_f), u^*(t_f), p^*(t_f), t_f) + \frac{\partial h}{\partial t}(x^*(t_f), t_f)\right] \delta t_f = 0
\]
Substitutions for boundary conditions

<table>
<thead>
<tr>
<th>Problem</th>
<th>Substitution</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_f ) fixed</td>
<td>( \delta t_f = 0 )</td>
</tr>
<tr>
<td>( \mathbf{x}(t_f) ) fixed</td>
<td>( \delta \mathbf{x}_f = 0 )</td>
</tr>
<tr>
<td>BC ( \mathbf{x}^*(t_0) = \mathbf{x}_0 )</td>
<td>( \mathbf{x}^*(t_f) = \mathbf{x}_f )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Problem</th>
<th>Substitution</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_f ) free</td>
<td>( \delta t_f ) arbitrary</td>
</tr>
<tr>
<td>( \mathbf{x}(t_f) ) fixed</td>
<td>( \delta \mathbf{x}_f = 0 )</td>
</tr>
<tr>
<td>BC ( \mathbf{x}^*(t_0) = \mathbf{x}_0 )</td>
<td>( \mathbf{x}^*(t_f) = \mathbf{x}_f )</td>
</tr>
<tr>
<td>( H(\mathbf{x}^<em>(t_f), \mathbf{u}^</em>(t_f), \mathbf{p}^<em>(t_f), t_f) + \frac{\partial h}{\partial t}(\mathbf{x}^</em>(t_f), t_f) = 0 )</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Problem</th>
<th>Substitution</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_f ) fixed</td>
<td>( \delta t_f = 0 )</td>
</tr>
<tr>
<td>( \mathbf{x}(t_f) ) free</td>
<td>( \delta \mathbf{x}_f ) arbitrary</td>
</tr>
<tr>
<td>BC ( \mathbf{x}^*(t_0) = \mathbf{x}_0 )</td>
<td>( \frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}^<em>(t_f)) - \mathbf{p}^</em>(t_f) = 0 )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Problem</th>
<th>Substitution</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_f ) free</td>
<td>( \delta t_f ) arbitrary</td>
</tr>
<tr>
<td>( \mathbf{x}(t_f) ) free</td>
<td>( \delta \mathbf{x}_f ) arbitrary</td>
</tr>
<tr>
<td>BC ( \mathbf{x}^*(t_0) = \mathbf{x}_0 )</td>
<td>( \frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}^<em>(t_f), t_f) - \mathbf{p}^</em>(t_f) = 0 )</td>
</tr>
<tr>
<td>( H(\mathbf{x}^<em>(t_f), \mathbf{u}^</em>(t_f), \mathbf{p}^<em>(t_f), t_f) + \frac{\partial h}{\partial t}(\mathbf{x}^</em>(t_f), t_f) = 0 )</td>
<td></td>
</tr>
</tbody>
</table>
Indirect methods: practical aspects


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

\[ \dot{z} = g(z, t), \quad l(z(t_0), z(t_f)) = 0 \]

• Syntax: `solve(bvp_problem, solution_guess)`

• In Matlab: `bvp4c`
Example

\[
\begin{align*}
\dot{z}_1(t) &= z_2(t) \\
\dot{z}_2(t) &= -|z_1(t)| \\
z_1(0) &= 0 \\
z_1(4) &= -2
\end{align*}
\]
Extensions

• What about problems whose necessary conditions do not fit directly the “standard” form (e.g., free end time problems)?


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:

\[
t_f = \left(\frac{1800b}{\alpha}\right)^{1/5}
\]
Example (solution)

• Define state as $z = [x, p, r]$

• BC are:

\[
\begin{align*}
x_1(0) &= 10, \quad x_2(0) = 0, \quad x_1(t_f) = 0, \quad x_2(t_f) = 0, \\
&\quad -0.5b(-p_2(t_f)/b)^2 + \alpha t_f = 0
\end{align*}
\]

• BVP becomes

\[
\begin{align*}
dz/d\tau &= t_f \frac{dz}{dt} = z_5 \\
A &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}
\end{align*}
\]

• BC become

\[
\begin{align*}
z_1(0) &= 10, \quad z_2(0) = 0, \quad z_1(1) = 0, \quad z_2(1) = 0, \\
&\quad -0.5b(-z_4(1)/b)^2 + \alpha z_5(1) = 0
\end{align*}
\]
Direct methods - nonlinear programming transcription

\begin{equation}
\min_{x(t), u(t)} \int_{t_0}^{t_f} g(x(t), u(t), t) \, dt
\end{equation}

\begin{align*}
\dot{x}(t) &= a(x(t), u(t), t), \ t \in [t_0, t_f] \\
x(0) &= x_0, \ x(t_f) \in M_f \\
u(t) &\in U \subseteq \mathbb{R}^m, \ t \in [t_0, t_f]
\end{align*}

\textbf{(OCP)}

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 $x_i \sim x(t), \ u_i \sim u(t), \ t \in (t_i, t_{i+1})$ and $x_0 \sim x(0)$

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

\begin{equation}
\min_{x_i, u_i} \sum_{i=0}^{N-1} h_i g(x_i, u_i, t_i)
\end{equation}

\textbf{(NLOP)}

\begin{align*}
x_{i+1} &= x_i + h_i a(x_i, u_i, t_i), \quad i = 0, \ldots, N-1 \\
u_i &\in U, \ i = 0, \ldots, N-1, \quad F(x_N) = 0
\end{align*}
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 $h_i \to 0$

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 $i = 0, \ldots, N - 1$, define $x_i \sim x(t), u_i \sim u(t), t \in (t_i, t_{i+1}]$ and $x_0 \sim x(0)$

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

\[
\min_{\{x_i, u_i\}} \sum_{i=0}^{N-1} h_i g(x_i, u_i, t_i)
\]

*(NLOP)*

\[
x_{i+1} = x_i + h_i a(x_i, u_i, t_i), \quad i = 0, \ldots, N - 1
\]

\[
u_i \in U, i = 0, \ldots, N - 1, \quad F(x_N) = 0
\]
Consistency of time discretization

Simplified Formulation

\[
\min \int_0^{t_f} g(x(t), u(t)) \, dt
\]

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

(OCP)

\[
x(0) = x_0
\]

Pontryagin’s Minimum Principle (PMP)

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

- Co-state equation:

\[
\dot{p}(t) = -\frac{\partial a}{\partial x}(x(t), u(t))' p(t) - \frac{\partial g}{\partial x}(x(t), u(t))
\]

- Control equation:

\[
\frac{\partial a}{\partial u}(x(t), u(t))' p(t) + \frac{\partial g}{\partial u}(x(t), u(t)) = 0
\]
Consistency of time discretization

Simplified Formulation

Related non-linear program (NLOP)

After discretization in time:

\[
\min \int_0^{t_f} g(x(t), u(t)) \, dt
\]

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

\[
(x(0) = x_0)
\]

\[
\min_{(x_i, u_i)} \sum_{i=0}^{N-1} h_i g(x_i, u_i) \quad \text{(NLOP)}
\]

\[
x_i + h_i a(x_i, u_i) - x_{i+1} = 0, \quad i = 0, \ldots, N - 1
\]
Consistency of time discretization

KKT Related to (NLOP)

Denote the Lagrangian related to (NLOP) as

\[ \mathcal{L} = \sum_{i=0}^{N-1} h_i g(x_i, u_i) + \sum_{i=0}^{N-1} \lambda'_i (x_i + h_i a(x_i, u_i) - x_{i+1}) \]

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

- Derivative w.r.t. \( x_i \):
  \[ h_i \frac{\partial g}{\partial x_i} (x_i, u_i) + \lambda_i - \lambda_{i-1} + h_i \frac{\partial a}{\partial x_i} (x_i, u_i)' \lambda_i = 0 \]

- Derivative w.r.t. \( u_i \):
  \[ h_i \frac{\partial g}{\partial u_i} (x_i, u_i) + h_i \frac{\partial a}{\partial u_i} (x_i, u_i)' \lambda_i = 0 \]

Related non-linear program (NLOP)

After discretization in time:

\[ \min_{(x_i, u_i)} \sum_{i=0}^{N-1} h_i g(x_i, u_i) \]

\[ x_i + h_i a(x_i, u_i) - x_{i+1} = 0, \quad i = 0, ..., N - 1 \]
Consistency of time discretization

KKT Related to (NLOP)

Denote the Lagrangian related to (NLOP) as

\[ \mathcal{L} = \sum_{i=0}^{N-1} h_i g(x_i, u_i) + \sum_{i=0}^{N-1} \lambda'(x_i + h_i a(x_i, u_i) - x_{i+1}) \]

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

- Derivative w.r.t. \( x_i \):
  \[ h_i \frac{\partial g}{\partial x_i}(x_i, u_i) + \lambda_i - \lambda_{i-1} + h_i \frac{\partial a}{\partial x_i}(x_i, u_i)' \lambda_i = 0 \]

- Derivative w.r.t. \( u_i \):
  \[ h_i \frac{\partial g}{\partial u_i}(x_i, u_i) + h_i \frac{\partial a}{\partial u_i}(x_i, u_i)' \lambda_i = 0 \]

Consistency with the PMP

We finally obtain:

\[ \frac{\lambda_i - \lambda_{i-1}}{h_i} = -\frac{\partial a}{\partial x_i}(x_i, u_i)' \lambda_i - \frac{\partial g}{\partial x_i}(x_i, u_i) \]
\[ \frac{\partial a}{\partial u_i}(x_i, u_i)' \lambda_i + \frac{\partial g}{\partial u_i}(x_i, u_i) = 0 \]

Let \( p(t) = \lambda_i \) for \( t \in [t_i, t_{i+1}] \), \( i = 0, ..., N-1 \) and \( p(0) = \lambda_0 \). Then, the equations above are the discretized version of the necessary conditions for (OCP):

\[ \dot{p}(t) = -\frac{\partial a}{\partial x_i}(x(t), u(t))' p(t) - \frac{\partial g}{\partial x}(x(t), u(t)) \]
\[ \frac{\partial a}{\partial u_i}(x(t), u(t))' p(t) + \frac{\partial g}{\partial u_i}(x(t), u(t)) = 0 \]
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