CS 333: Safe and Interactive Robotics Fall 2017

Lecture 4: Trajectory Optimization & Introduction to Formal Methods

Scribes: Hesam H., Eli Wu, Kyle Julian, Chelsea Sidrane

4.1 Trajectory Optimization

In the previous lecture, we recapped how to compute gradients, Jacobian, and the Euler-Lagrange equation. We begin today's lecture by finishing the proof of the Euler-Largrange equation, and then we discuss different approaches for optimizing trajectories.

4.1.1 Euler-Lagrange Equation

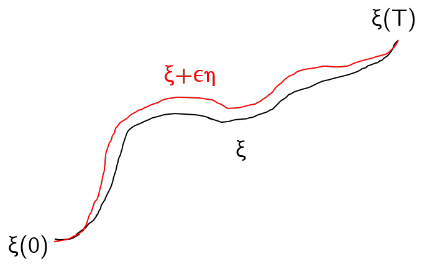
Let Ξ be a Hilbert space with Euclidean inner products, which defines our trajectory space. For a given trajectory $\xi \in \Xi$, let U be some utility function mapping trajectories to costs, so $U : \Xi \to \mathcal{R}^+$. Define U as

$$U(\xi) = \int_0^T F(t, \xi(t), \xi'(t))dt \tag{4.1}$$

We want to prove that

$$\nabla_\xi U = \frac{\partial F}{\partial \xi}(t) - \frac{d}{dt}\frac{\partial F}{\partial \xi'}(t) \tag{4.2}$$

s Let $\eta \in \Xi$ be a trajectory with $\eta(0) = \eta(T) = 0$. Using small $\eta \in \mathcal{R}$, we can perturb the trajectory ξ as $\xi + \epsilon\eta$, as shown in the figure below.



We begin by using the Taylor Expansion and neglecting higher order terms:

$$U[\xi + \epsilon\eta] \approx U[\xi] + \epsilon < \nabla_\xi U, \eta > \tag{4.3}$$

We can define the inner product above in two ways using the definitation of derivatives and inner products:

$$< \nabla_\xi U, \eta >= \lim_{\epsilon \to 0} \frac{U[\xi + \epsilon\eta] - U[\xi]}{\epsilon} \tag{4.4}$$

$$< \nabla_\xi U, \eta >= \int_0^T \nabla_\xi U^T \eta dt \tag{4.5}$$

For convenience, define $\phi(\epsilon) = U[\xi + \epsilon\eta]$. This allows us to write

$$< \nabla_\xi U, \eta > = \lim_{\epsilon\to 0} \frac{\phi(\epsilon) - \phi(0)}{\epsilon} = \frac{d\phi}{d\epsilon}\bigg|_{\epsilon=0} \tag{4.6}$$

$$= \frac{d}{d\epsilon}\int_0^T F[t, \xi(t) + \epsilon\eta(t), \xi'(t) + \epsilon\eta'(t)]dt\bigg|_{\epsilon=0} \tag{4.7}$$

$$= \int_0^T \frac{d}{d\epsilon} F[t, \xi(t) + \epsilon\eta(t), \xi'(t) + \epsilon\eta'(t)]dt\bigg|_{\epsilon=0} \tag{4.8}$$

$$\tag{4.9}$$

To simplify, let

$$x(\epsilon) = \xi(t) + \epsilon\eta(t) \tag{4.10}$$

$$y(\epsilon) = \xi'(t) + \epsilon\eta'(t) \tag{4.11}$$

We can expand the derivative with respect to ϵ into two parts, as shown below.

$$< \nabla_\xi U, \eta >= \int_0^T \left(\frac{\partial F[t, x(\epsilon), y(\epsilon)]}{\partial x}^T \frac{\partial x}{\partial \epsilon} + \frac{\partial F[t, x(\epsilon), y(\epsilon)]}{\partial y}^T \frac{\partial y}{\partial \epsilon} \right) dt\bigg|_{\epsilon=0} \tag{4.12}$$

From Equations (4.10) and (4.11), we know that

$$\frac{\partial x}{\partial \epsilon} = \eta(t), \qquad x(0) = \xi(t), \qquad \partial x = \partial\xi\bigg|_{\epsilon=0} \tag{4.13}$$

$$\frac{\partial y}{\partial \epsilon} = \eta'(t), \qquad y(0) = \xi'(t), \qquad \partial y = \partial\xi'\bigg|_{\epsilon=0} \tag{4.14}$$

Making these substitutions in Equation (4.12), we get

$$< \nabla_\xi U, \eta >= \int_0^T \left(\frac{\partial F}{\partial \xi}^T \eta(t) + \frac{\partial F}{\partial \xi'}^T \eta'(t) \right) dt \tag{4.15}$$

The next step uses integration by parts. The goal is to substitute the $\eta'(t)$ term for $\eta(t)$, which will allow us to simplify our equation further. Recall that integration by parts follows the rough form

$$\int A \times B' = A \times B - \int A' \times B \tag{4.16}$$

To apply integration by parts, we can split the integral in Equation (4.15) into two parts and focus on just the term with $\eta'(t)$. Applying integration by parts, we get

$$\int_0^T \frac{\partial F}{\partial \xi'}^T \eta'(t)dt = \left(\frac{\partial F}{\partial \xi'}\right)^T \eta(t)\bigg|_0^T - \int_0^T \frac{d}{dt}\left(\frac{\partial F}{\partial \xi'}\right)^T \eta(t)dt \tag{4.17}$$

By definition, $\eta(0) = \eta(T) = 0$, so the first term in Equation (4.17) goes to 0, leaving us with

$$\int_0^T \frac{\partial F}{\partial \xi'}^T \eta'(t)dt = -\int_0^T \frac{d}{dt}\left(\frac{\partial F}{\partial \xi'}\right)^T \eta(t)dt \tag{4.18}$$

Substituting this result back into Equation (4.15), we can factor out $\eta(t)$, leaving us with

$$< \nabla_\xi U, \eta >= \int_0^T \left(\frac{\partial F}{\partial \xi} - \frac{d}{dt}\frac{\partial F}{\partial \xi'} \right)^T \eta(t)dt \tag{4.19}$$

Recall Equation (4.5), which is also equal to $< \nabla_\xi U, \eta >$. Setting the two equations equal yields

$$\int_0^T \nabla_\xi U^T \eta(t)dt = \int_0^T \left(\frac{\partial F}{\partial \xi} - \frac{d}{dt}\frac{\partial F}{\partial \xi'} \right)^T \eta(t)dt \tag{4.20}$$

This holds for all U, so we can compare terms inside the integral, which gives us our desired result.

$$\nabla_\xi U = \frac{\partial F}{\partial \xi} - \frac{d}{dt}\frac{\partial F}{\partial \xi'} \tag{4.21}$$

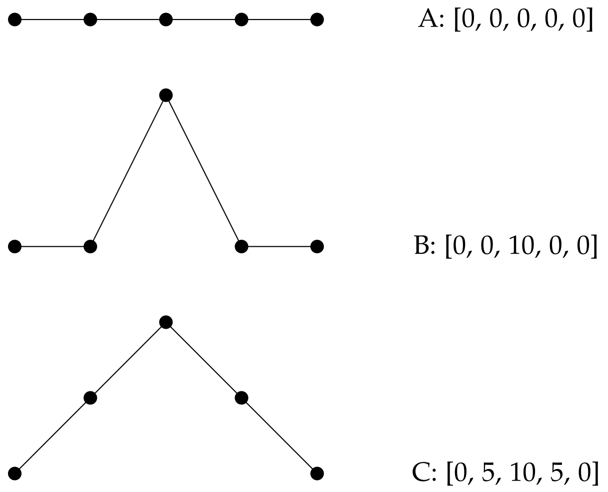
4.1.2 Approaches for Trajectory Optimization

How do we define the distance between two trajectories? One common approach is to take the norm of the distance between two trajectories:

$$\|\xi\| = \sqrt{< \xi, \xi >} \tag{4.22}$$

$$\text{Distance: } \|\xi_1 - \xi_2\| \tag{4.23}$$

Let's imagine this is discrete time, with five elements. Consider three trajectories shown below.



Is trajectory B or C closer to trajectory A? Computing the norm of the distance to A, we see that

$$\|a - b\|^2 = 100 \tag{4.24}$$

$$\|a - c\|^2 = 150 \tag{4.25}$$

By our definition, B is closer to A than C. However, C is a smoother trajectory, so it could be argued that C is really closer to A because it does not have as sharp of accelerations as B does.

Let's define a different kind of norm. In continuous time, let

$$\mathcal{U}_{smooth} = \frac{1}{2}\int_0^T \|\xi'(t)\|^2 dt \tag{4.26}$$

This utility function means we penalize velocity, which leads to trajectories that reach the goal with minimal energy expended. In discrete time, the integral changes to a sum.

$$\mathcal{U}_{smooth} = \frac{1}{2}\sum_{i=0}^{N} \|q_{i+1} - q_i\|^2 \tag{4.27}$$

The i^{th} component of our discrete gradient of $\mathcal{U}_{smooth}$ is

$$\frac{\partial \mathcal{U}_{smooth}}{\partial q_i} = 2q_i - q_{i+1} - q_{i-1} \tag{4.28}$$

We can write this equation in the form of a matrix, which we'll denote as A, times the vector q, as shown below.

$$\frac{\partial \mathcal{U}_{smooth}}{\partial q_i} = \begin{bmatrix} 2 & -1 & 0 & 0 & \dots \\ -1 & 2 & -1 & 0 & \dots \\ 0 & -1 & 2 & -1 & \dots \\ \vdots & \vdots & \vdots & \ddots & \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ \\ q_N \end{bmatrix}$$

With this approach, we get different results for the norm values

$$\|a - b\|_A^2 = b^T A b = 200 \tag{4.29}$$

$$\|a - c\|_A^2 = c^T A c = 100 \tag{4.30}$$

With this norm, C is defined as closer to A than B. We can use this norm in the derivation of a trajectory optimization method using gradient descent. Using a Taylor expansion:

$$\mathcal{U}[\xi] = \mathcal{U}[\xi_i] + \nabla_{\xi_i}\mathcal{U}^T(\xi - \xi_i) + \frac{1}{2}\alpha\|\xi - \xi_i\|_A^2 \tag{4.31}$$

We want to choose the next ξ, ξ_{i+1}, such that $mathcalU[\xi_{i+1}]$ is minimized. To find the minimizing ξ, we differentiate Equation (4.31) with respect to ξ and set the result equal to 0.

$$0 = 0 + \nabla_{\xi_i}\mathcal{U} + \alpha A(\xi - \xi_i) \tag{4.32}$$

$$\xi = \xi_{i+1} = \xi_i - \frac{1}{\alpha}A^{-1}\nabla_{\xi_i}\mathcal{U} \tag{4.33}$$

This defines our update rule for our trajectory optimization using our definition of norm. Note that α determines the step size to take during the optimization. The direction ξ changes during optimization is determined by choice of norm.

4.2 Formal Methods

4.2.1 Motivation

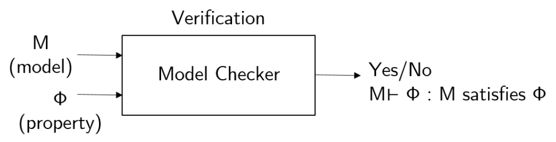
Formal methods is an area of computer science; namely, it started from computer science theory. Its main idea was to give proofs for algorithms, electronic circuits, etc.

One of the most basic ideas in formal methods is the concept of satisfiability, or more specifically, boolean satisfiability, an NP-complete problem. (Cook 1971) Boolean satisfiability is the problem of finding inputs to a boolean expression that satisfy the expression. For example:

$$(X+Y)\cdot(\neg X+\neg Y)\overset{?}{=}1 \tag{4.34}$$

This expression is satisfied when $X=0$ and $Y=1$, or $X=1$ and $Y=0$.

The idea gained traction in the 1960's when the problem of verifying electronic circuits (with AND, OR, or other gates) became increasingly difficult.

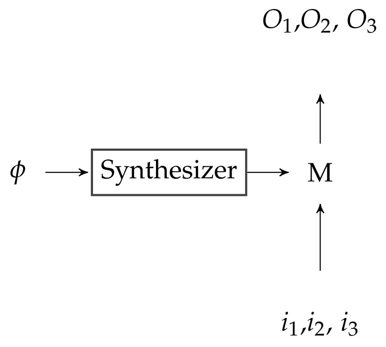


Nowadays, we have several sat-solvers, such as SMV (Berezin 1998), SPIN (Holzmann 1991-2017), SLAM (Ball and Rajamani 2002).

4.2.2 Reactive Synthesis from Linear Temporal Logic

Reactive System

1. At every step, the system receives an input sequence and outputs a response sequence synchronously.
2. Non-terminating - does not terminate on any input.



Realizability

1. Does there exist a finite state system that can satisfy ϕ ? In other words, does there exist a finite state system that *realizes* the specification?
2. Can we construct it?

4.2.3 Historical Overview

- 1957: Alonso Church details the idea of circuit synthesis in his paper ”Application of Recursive Arithmetic to the Problem of Circuit Synthesis” (Church 1957)
- 1969: Buchi & Landweber show that the problem of realizability is decidable, and that it is finite if it exists. (Buchi and Landweber 1969)
- 1989: Pnuelli & Rosner publish a paper detailing Linear Temporal Logic Synthesis (A. Pnueli and Rosner 1989)
- 2006: Piterman et al publish their paper ”Synthesis of Reactive (1) Designs”, detailing GR(1) Synthesis (Piterman, Amir Pnueli, and Sa’ar 2006)

4.2.3.1 Note: Game View of Synthesis

You can think of synthesis as a two player game between the environment and the controller.

4.2.4 Overview of Linear Temporal Logic

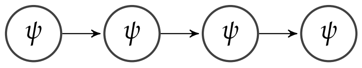
LTL answers the question: How do I even write ϕ?

Four Temporal Operators:

- G Globally
- F Future/Eventually
- X Next
- U Until

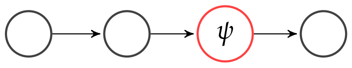
4.2.4.1 Examples

1. $\phi_1 = G\psi$

   This means that ψ holds at every timestep in the sequence.

2. $\phi_2 = F\psi$

   This means that at some point in the future, ψ will hold.

3. $\phi_3 = X\psi$

$$\begin{tikzcd} \bigcirc \arrow[r] & \psi \arrow[r] & \bigcirc \arrow[r] & \bigcirc \end{tikzcd}$$

This means that ψ will hold at the next timestep.

4. $\phi_4 = \psi_1 U \psi_2$

$$\begin{tikzcd} \psi_1 \arrow[r] & \psi_2 \arrow[r] & \psi_2 \arrow[r] & \psi_2 \end{tikzcd}$$

Note the following sequence satisfies ϕ_4 :.

$$\begin{tikzcd} \psi_2 \arrow[r] & \psi_2 \arrow[r] & \psi_2 \arrow[r] & \psi_2 \end{tikzcd}$$

This means that ψ_1 will hold until ψ_2 holds.

5. $FG\psi$

Eventually, ψ will hold globally. Note: Smells like a stability property.

6. $GF\psi$

Wherever I am, ψ will be true in the future. This can be thought of as a recurrent property, or a fairness property.

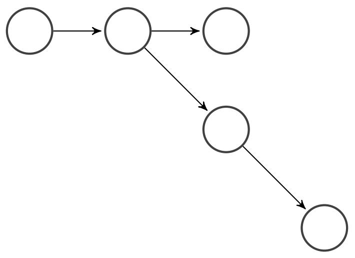
7. G(request $\rightarrow$ F grant)

At all times, if I get a request, at some point in the future, I will grant it.

4.2.4.2 Note on Other Logics

There are other kinds of temporal logics that exist in addition to Linear Temporal Logic (LTL). The "Linear" of LTL refers to linear time.

For example, there is also Computation Tree Logic (CTL) which permits branching. You can ask questions like: Does a property hold for one of these paths?



References

Ball, Thomas and Sriram K. Rajamani (2002). "The SLAM Project: Debugging System Software via Static Analysis". In: *SIGPLAN Not.* 37.1, pp. 1–3. ISSN: 0362-1340. DOI: `10.1145/565816.503274`. URL: `http://doi.acm.org/10.1145/565816.503274`.

Berezin, Sergey (1998). *SMV*. `http://www.cs.cmu.edu/~modelcheck/smv.html`.

Buchi, J. Richard and Lawrence H. Landweber (1969). "Definability in the Monadic Second-Order Theory of Successor". In: *J. Symbolic Logic* 34.2, pp. 166–170. URL: `https://projecteuclid.org:443/euclid.jsl/1183736763`.

Church, Alonzo (1957). "Application of recursive arithmetic to the problem of circuit synthesis". In:

Cook, Stephen A. (1971). "The Complexity of Theorem-proving Procedures". In: *Proceedings of the Third Annual ACM Symposium on Theory of Computing*. STOC '71. Shaker Heights, Ohio, USA: ACM, pp. 151–158. DOI: `10.1145/800157.805047`. URL: `http://doi.acm.org/10.1145/800157.805047`.

Holzmann, Gerard J. (1991-2017). *SPIN Model Checker*. `http://spinroot.com/spin/whatispin.html`.

Piterman, Nir, Amir Pnueli, and Yaniv Sa'ar (2006). "Synthesis of Reactive(1) Designs". In: *Verification, Model Checking, and Abstract Interpretation: 7th International Conference, VMCAI 2006, Charleston, SC, USA, January 8-10, 2006. Proceedings*. Ed. by E. Allen Emerson and Kedar S. Namjoshi. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 364–380. ISBN: 978-3-540-31622-0. DOI: `10.1007/11609773_24`. URL: `https://doi.org/10.1007/11609773_24`.

Pnueli, A. and R. Rosner (1989). "On the Synthesis of a Reactive Module". In: *Proceedings of the 16th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. POPL '89. Austin, Texas, USA: ACM, pp. 179–190. ISBN: 0-89791-294-2. DOI: `10.1145/75277.75293`. URL: `http://doi.acm.org/10.1145/75277.75293`.