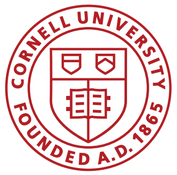


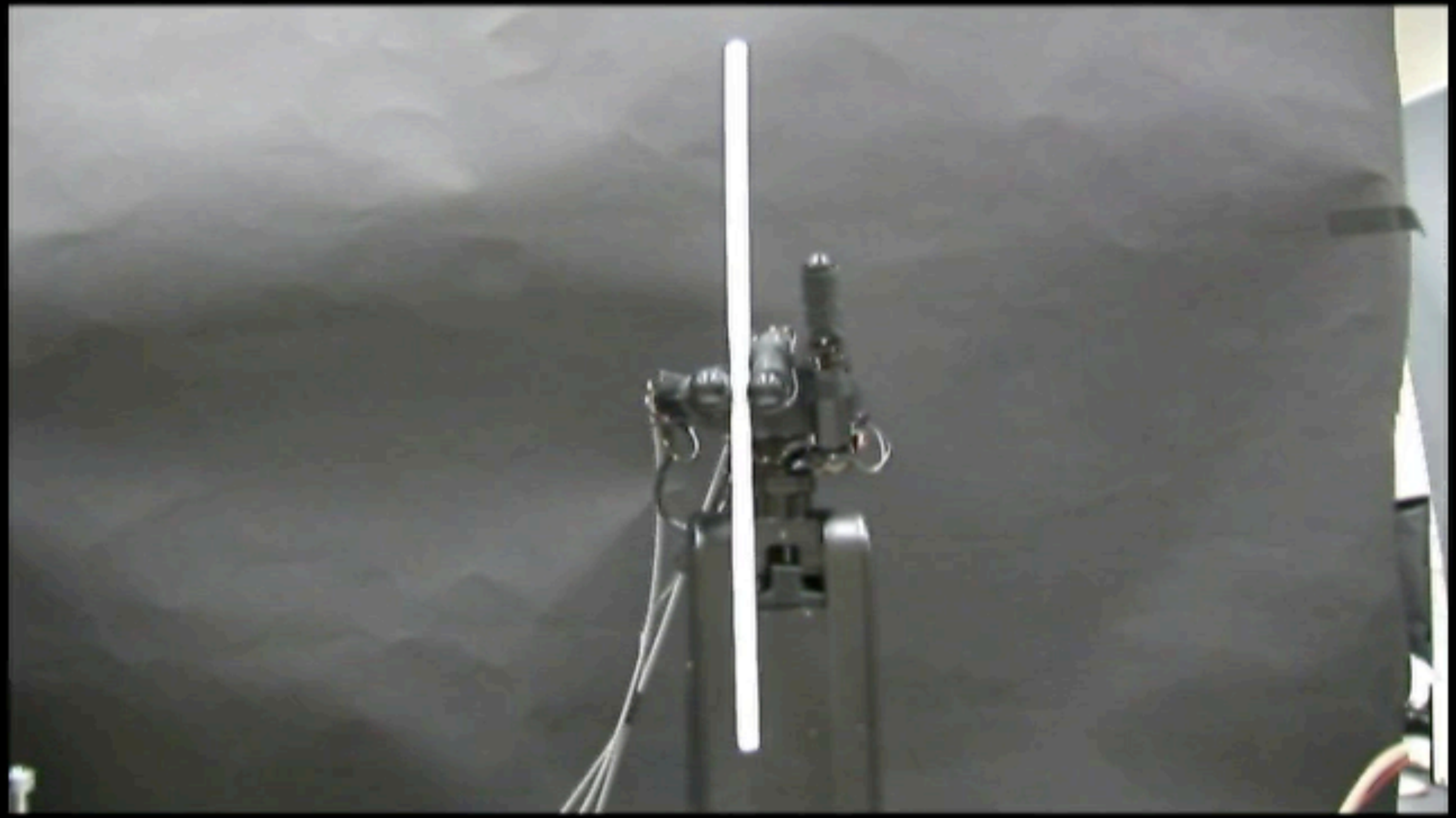
CornellBowers
Computer Science

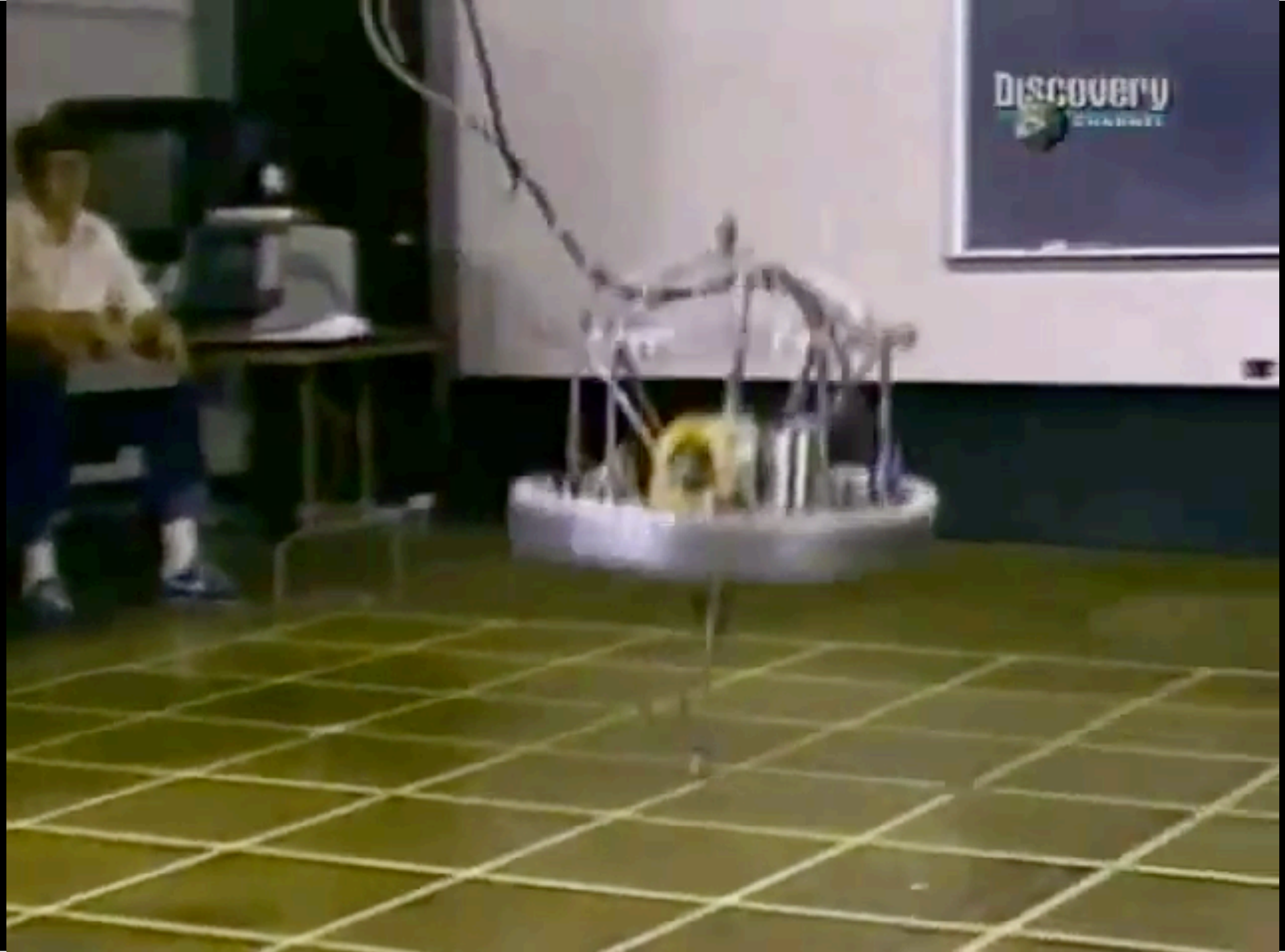
CS 5757: Optimization Methods for Robotics

Preston Culbertson

In the beginning… roboticists used hand-crafted controllers.

And they were (pretty) good!




Discovery

Why didn’t this solve robotics?

1. Onboarding new skills requires cleverness, a PhD.

Well, programming new robots looked like this:

1. Idea

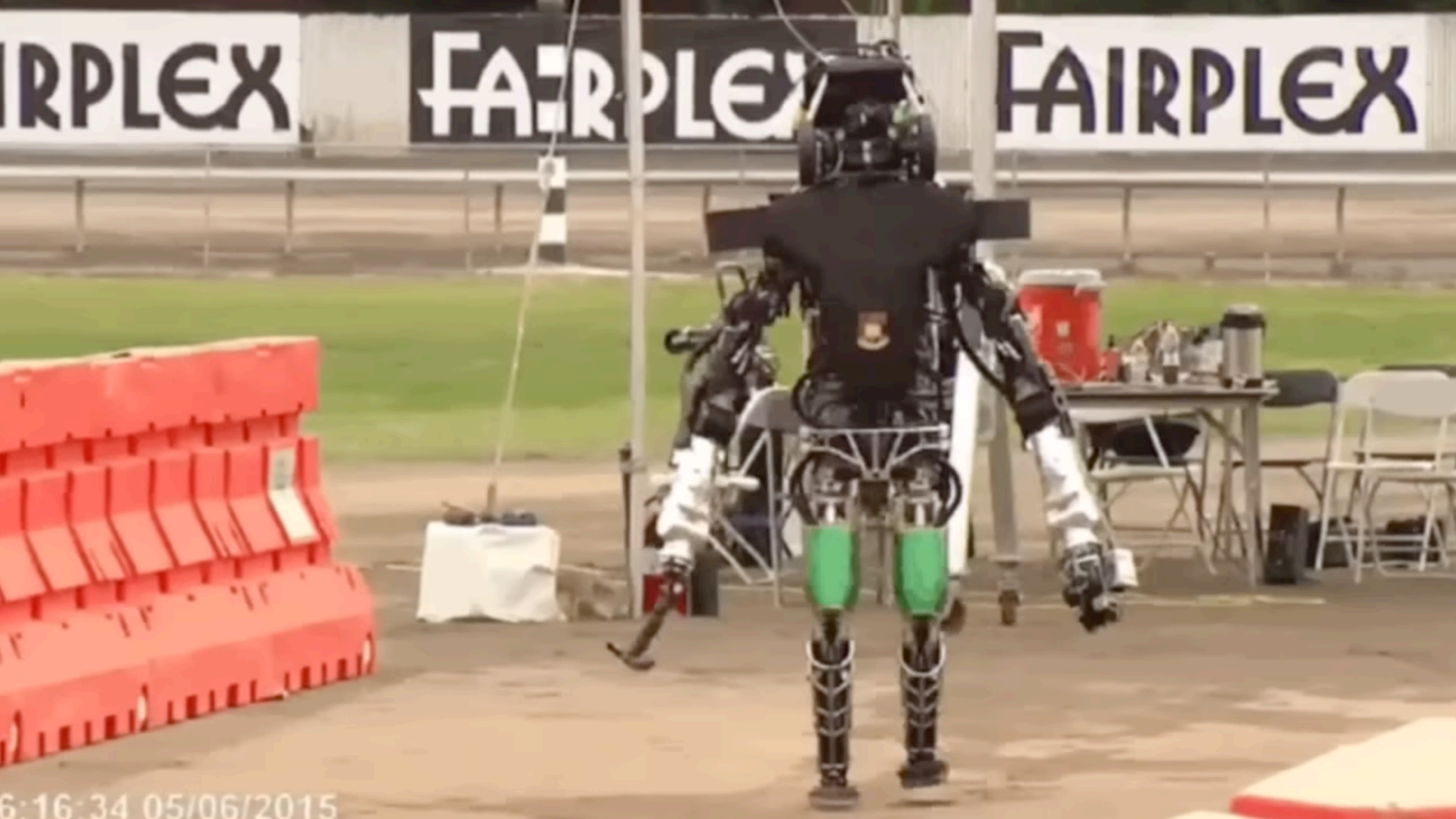
2. “Grad student descent”

3. Working robot

Why didn’t this solve robotics?

1. Onboarding new skills requires cleverness, a PhD.

2. Hand-designed controllers come with *assumptions* that fail.


FAIRPLEX
FAIRPLEX
FAIRPLEX
6:16:34 05/06/2015

In some sense, our field had to learn the *bitter lesson*.

The bitter lesson (robotics ed.)

The biggest lesson that can be read from 70 years of AI research is that **general methods** that **leverage computation** are ultimately the most effective, and by a large margin…

Seeking an improvement that makes a difference in the shorter term, researchers seek to leverage their human knowledge of the domain, but the only thing that matters in the long run is the leveraging of computation…

The two methods that seem to scale arbitrarily in this way are ***search*** and ***learning***.

Rich Sutton

"The Bitter Lesson," 2019.

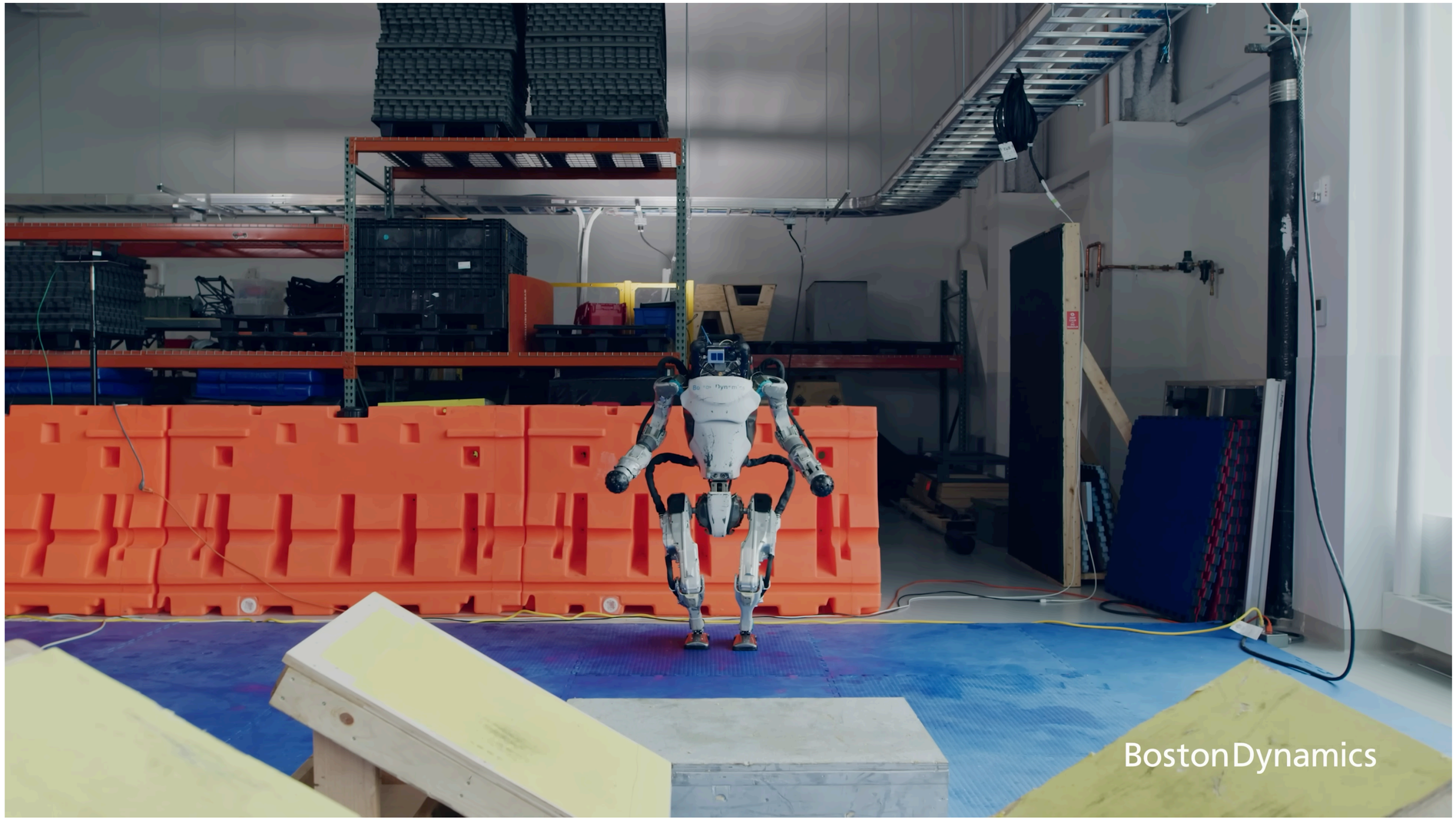
Success story #1: Robot learning

Success story #1: Robot learning

Success story #1: Robot learning

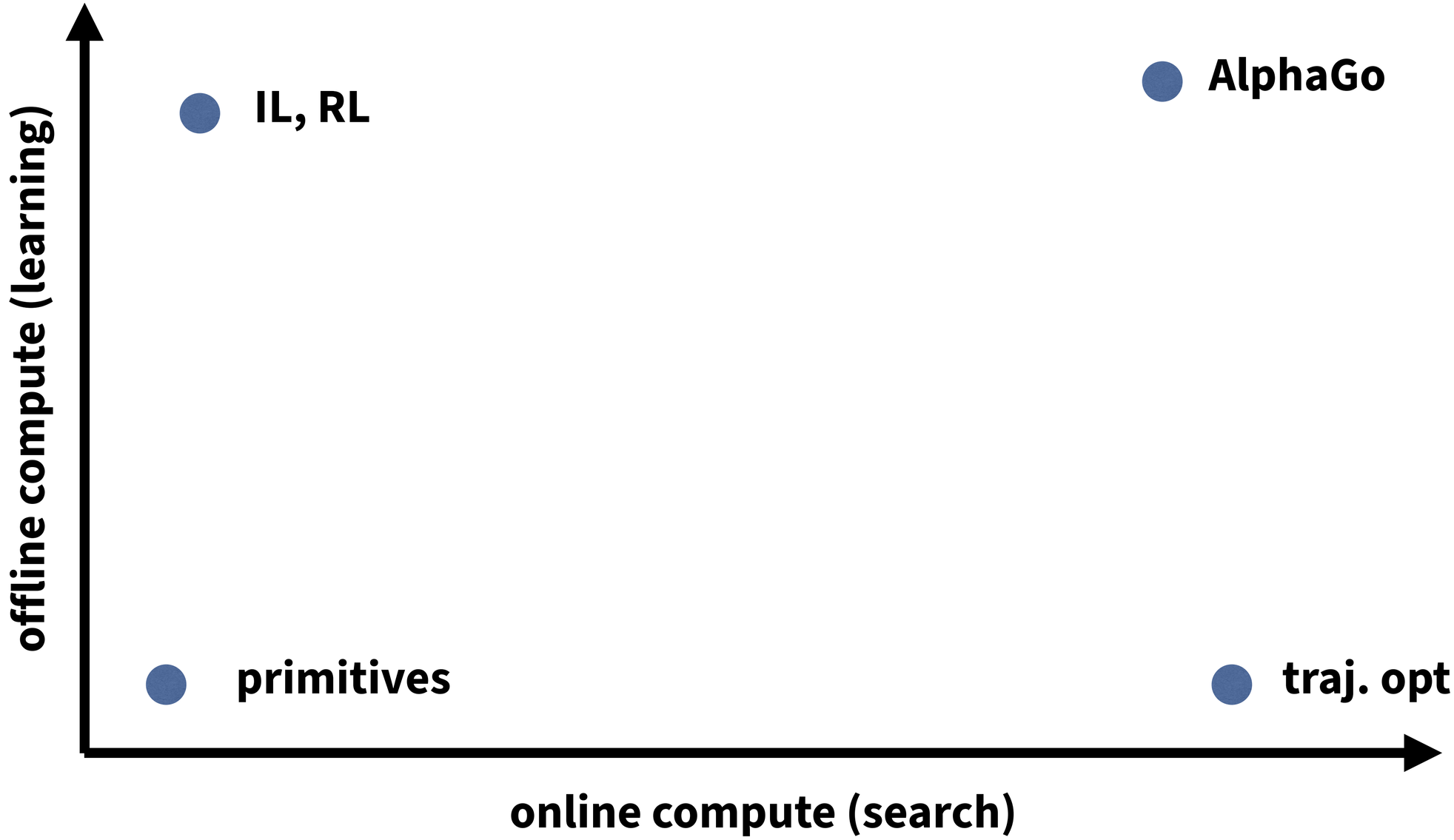
But - does learning solve everything?

Success story #2: Search (aka “optimization”)

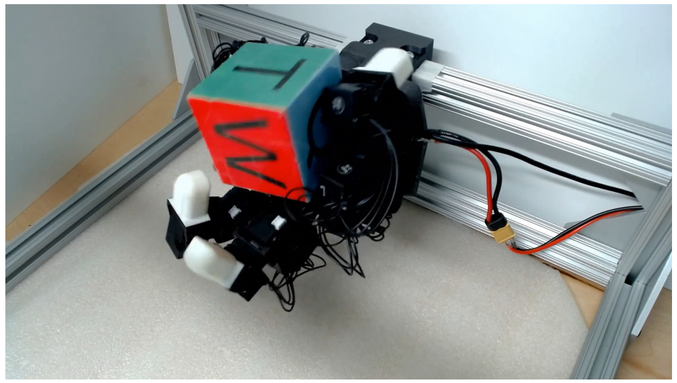


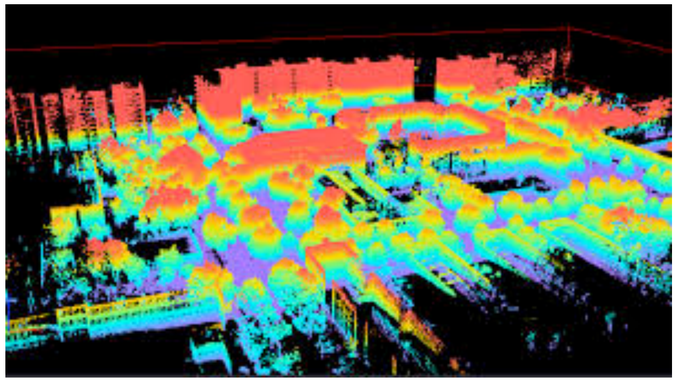
Success story #2: Search (aka “optimization”)

A spectrum of algorithms



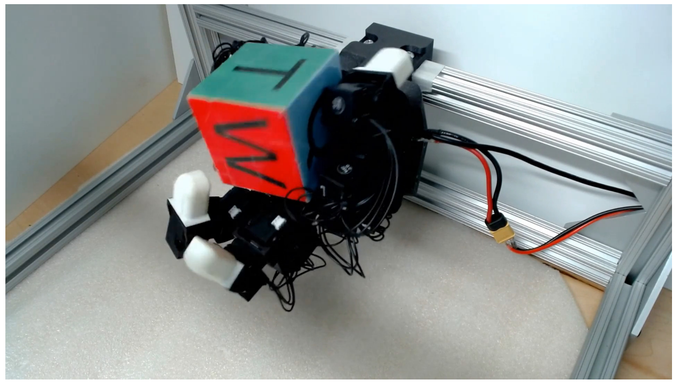
Why this class?

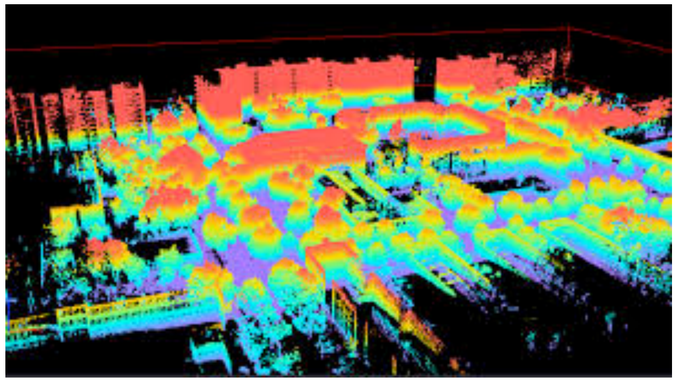




🛠️ **Learn reusable tools across many platforms/problems**

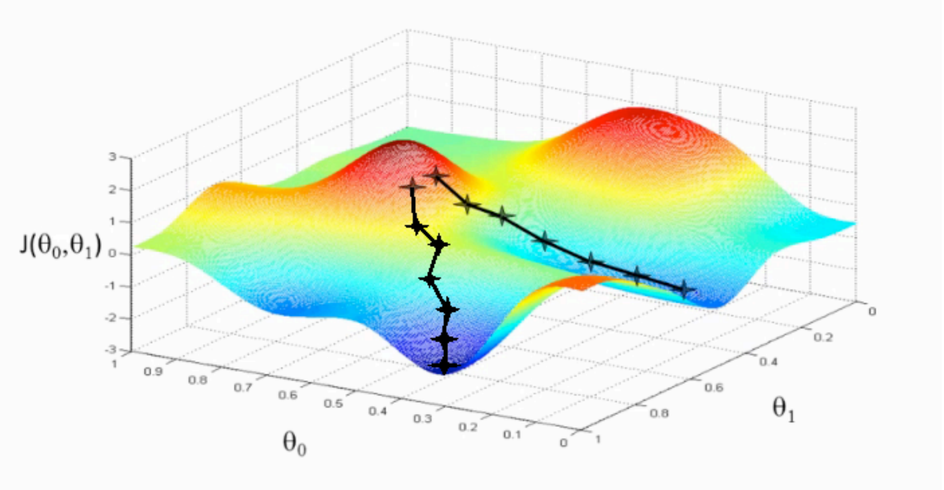
Why this class?





$$\min_{\boldsymbol{x}} \quad f(\boldsymbol{x})$$

$$\text{s.t.} \quad \boldsymbol{g}(\boldsymbol{x}) \leq 0$$



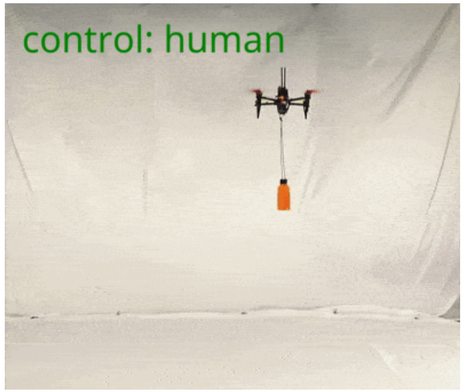
🛠️ **Learn reusable tools across many platforms/problems**

📝 **Build solid mathematical and programming foundations**

[introductions 👋]

Georgia Institute of Technology

Caltech



CORNELL UNIVERSITY
FOUNDED A.D. 1865

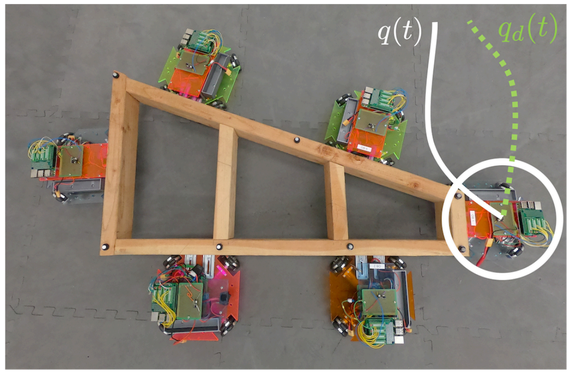
“Preston”

he / they

2016 **2022** **2024** **2025**

Non-robot stuff:

- **climbing**
- **baking**
- **volunteering**



Stanford University

rai Robotics and AI Institute

Teaching Assistant: Shengmiao ‘Samuel’ Jin

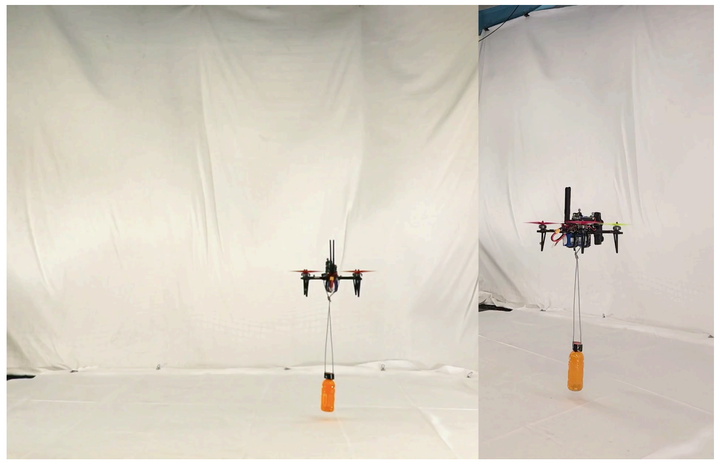
I am a first-year Robotics PhD student in the Computer Science department co-advised by Professor Preston Culbertson and Professor Tapomayukh Bhattacharjee. His research interest lies in the intersection of **Tactile Sensing, Dexterous and Contact-Rich Manipulation, and Uncertainty Quantification.**

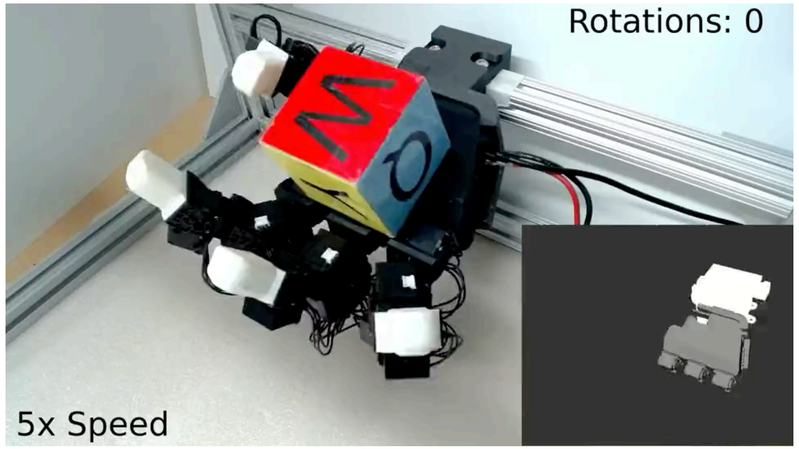
Prior to Cornell, I obtained a B.S. in Electrical Engineering with Highest Honors from UIUC

Office Hour: Thursday 5-7, Room TBD

We're the Praxis Lab

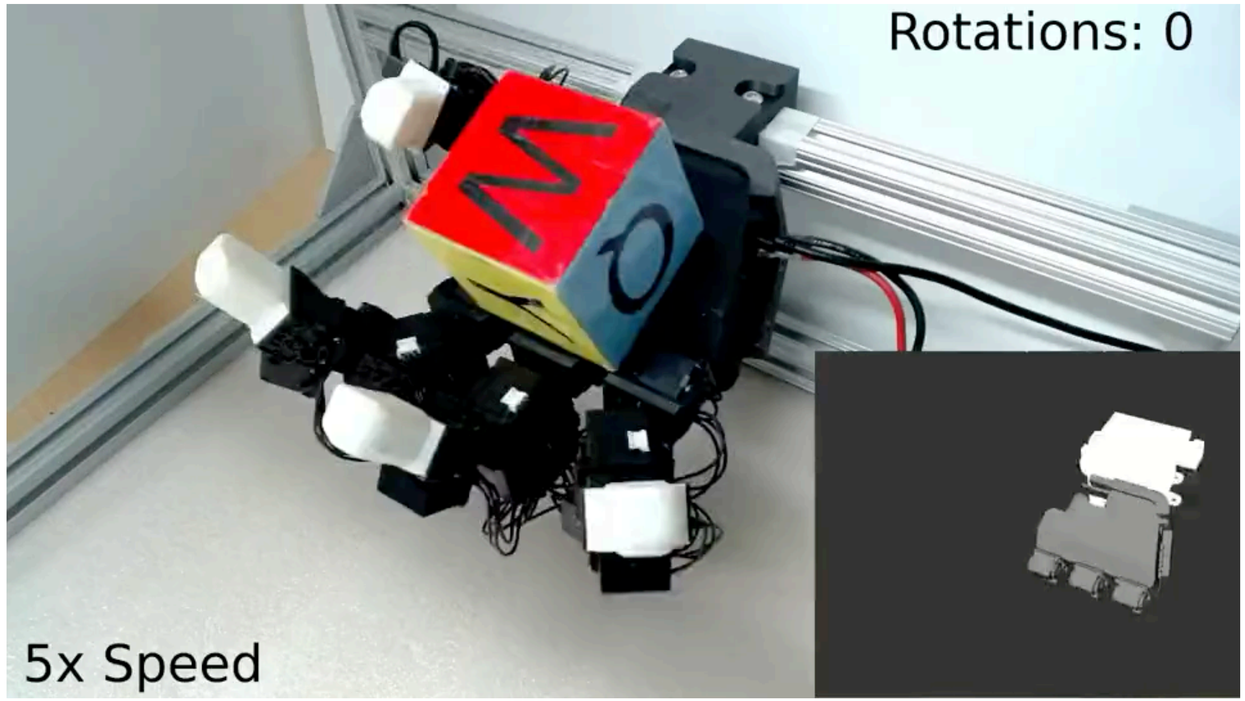
prax·is *(noun)* — theory in action

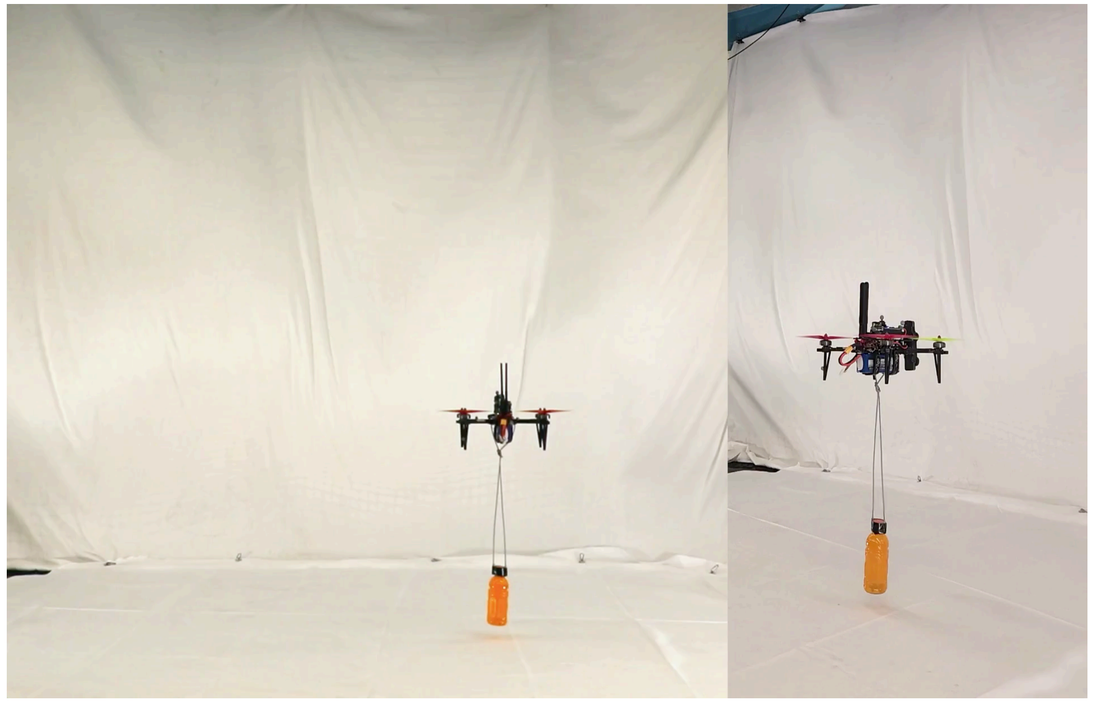




Goal: Enable adaptive + robust control needed for dynamic motion.

Some examples of how we use optimization



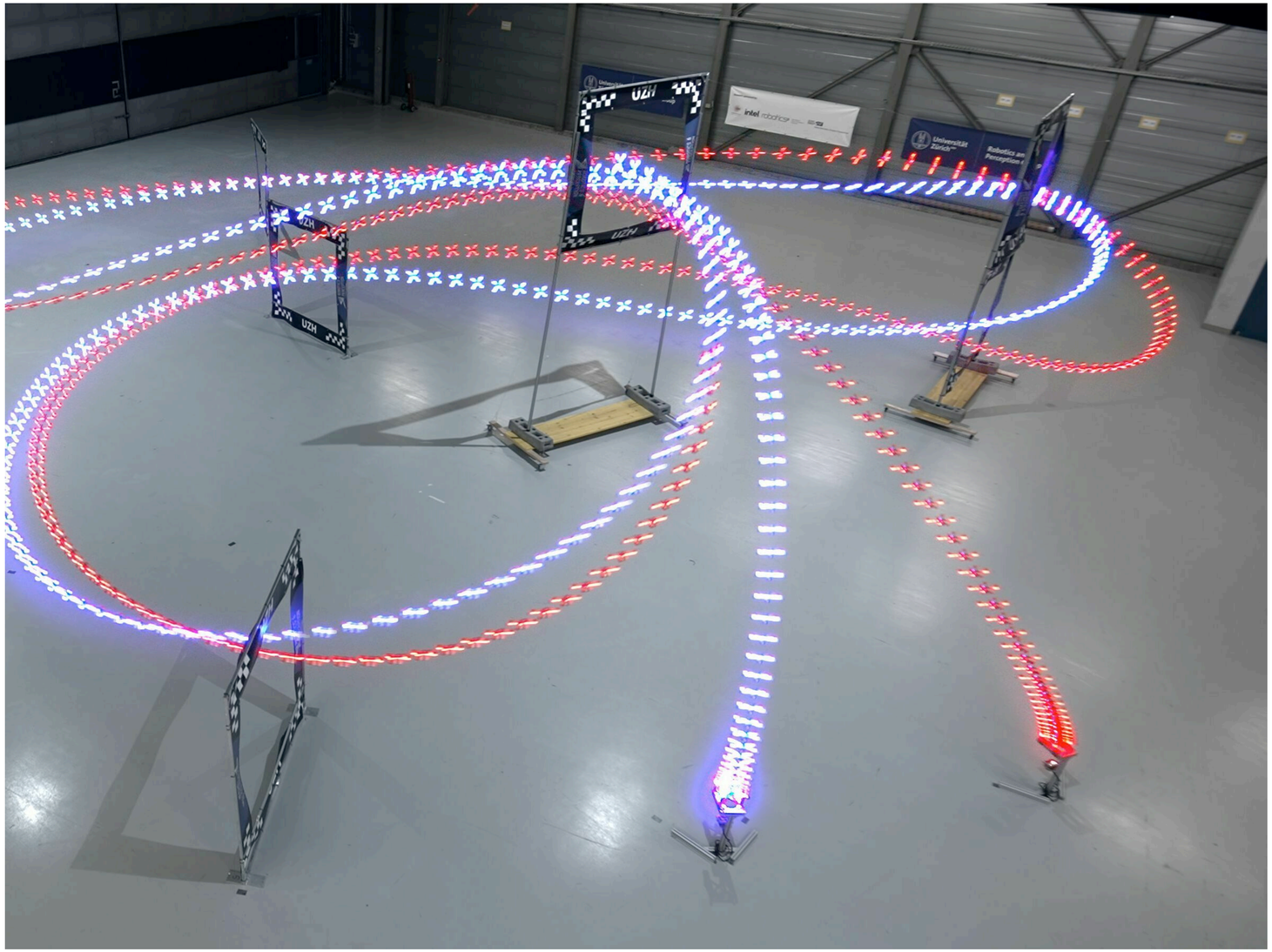


1. Robot planning

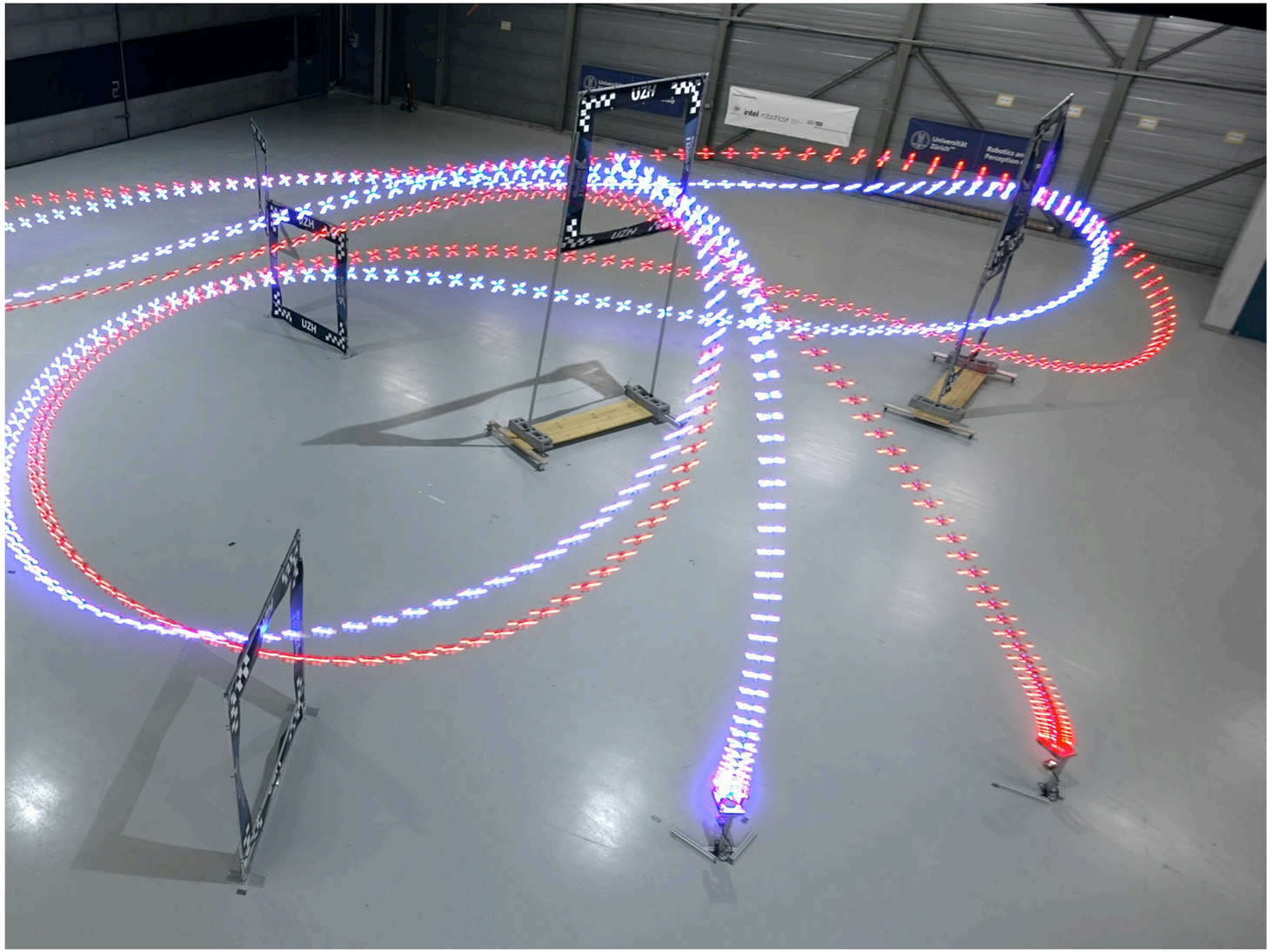
2. Robot safety

… and many more!

Brainstorm: What is “good” drone racing?



Brainstorm: What is “good” drone racing?



Want to minimize:

- Time to complete course
- Energy usage

Cannot:

- Crash (with gates/others)
- Violate physics
- Run out of battery

Optimization: A general method for decision-making

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \quad f(\boldsymbol{x})$$

$$\text{s.t.} \quad \boldsymbol{g}(\boldsymbol{x}) \leq 0$$

Optimization: A general method for decision-making

"decision variable"

$$\min_{x \in \mathbb{R}^n} \quad f(x)$$

$$\text{s.t.} \quad g(x) \leq 0$$

Optimization: A general method for decision-making

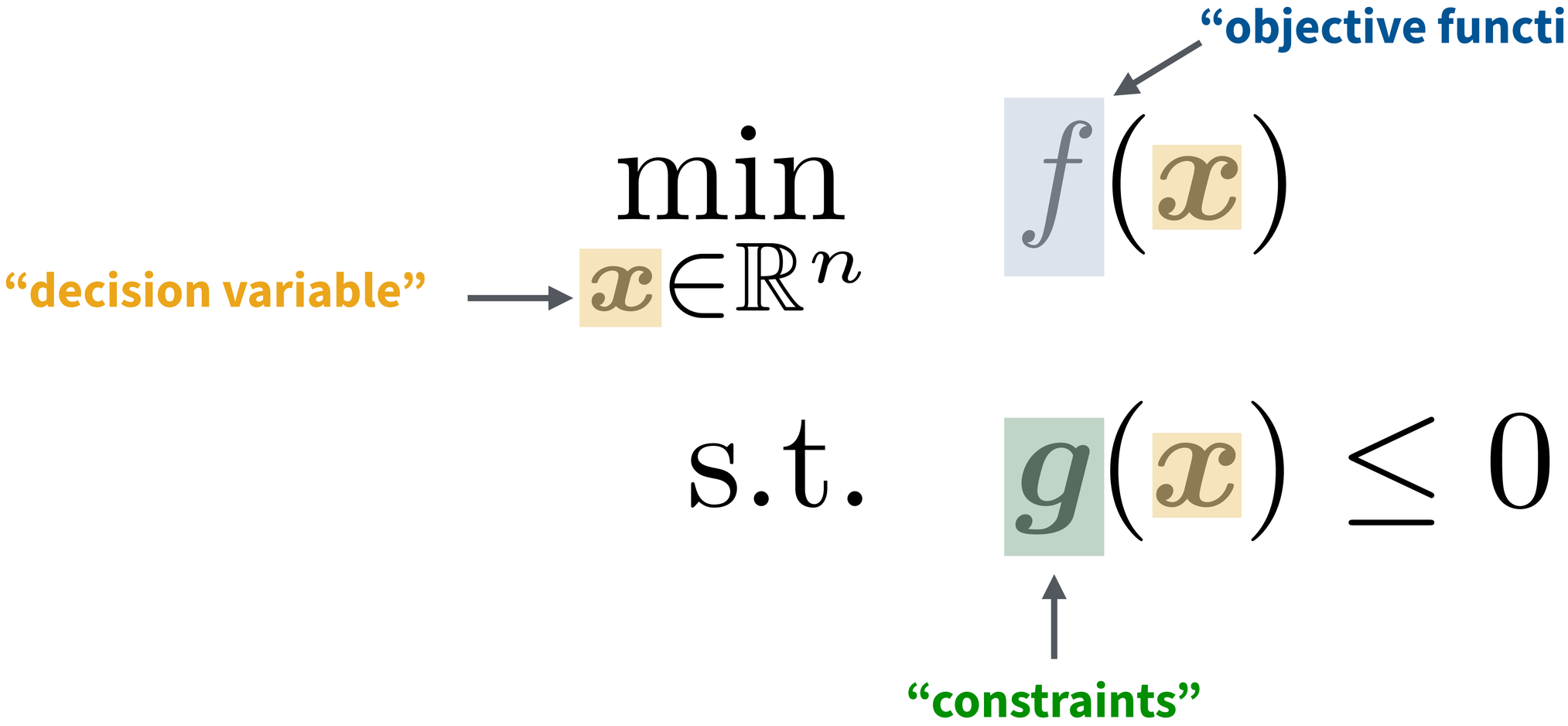
"objective function"

"decision variable"

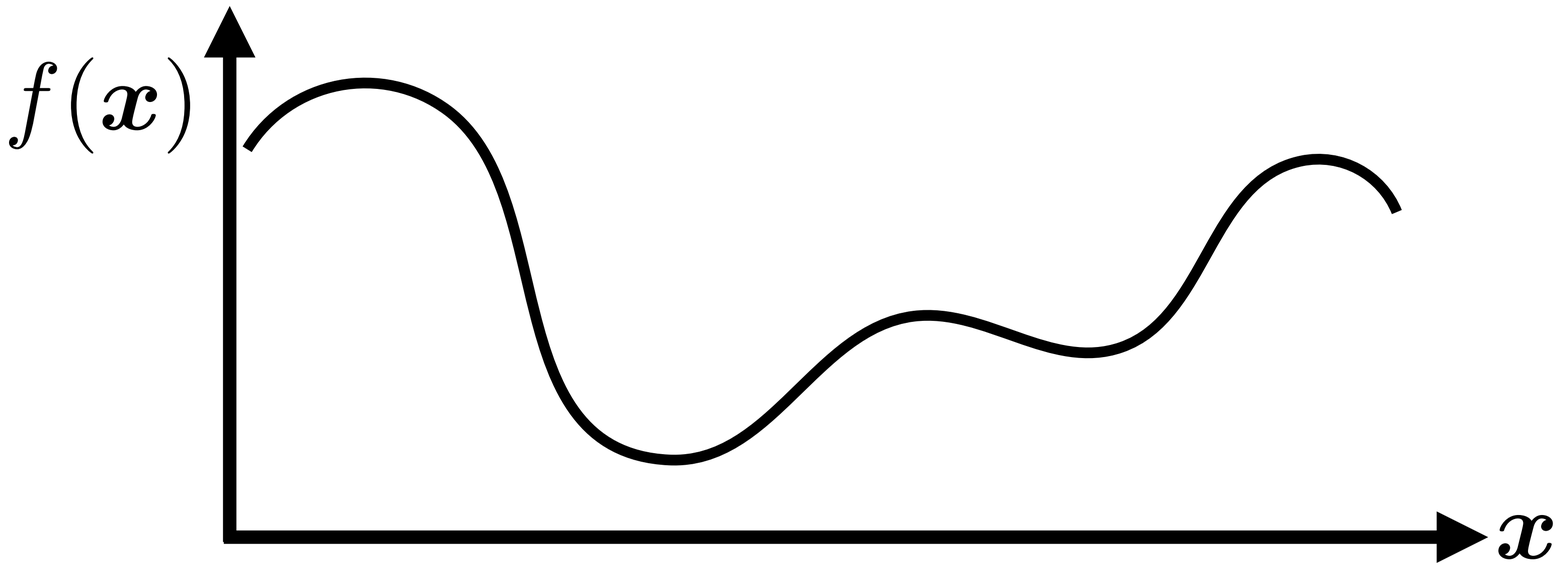
$$\min_{x \in \mathbb{R}^n} \quad f(x)$$

$$\text{s.t.} \quad g(x) \leq 0$$

Optimization: A general method for decision-making



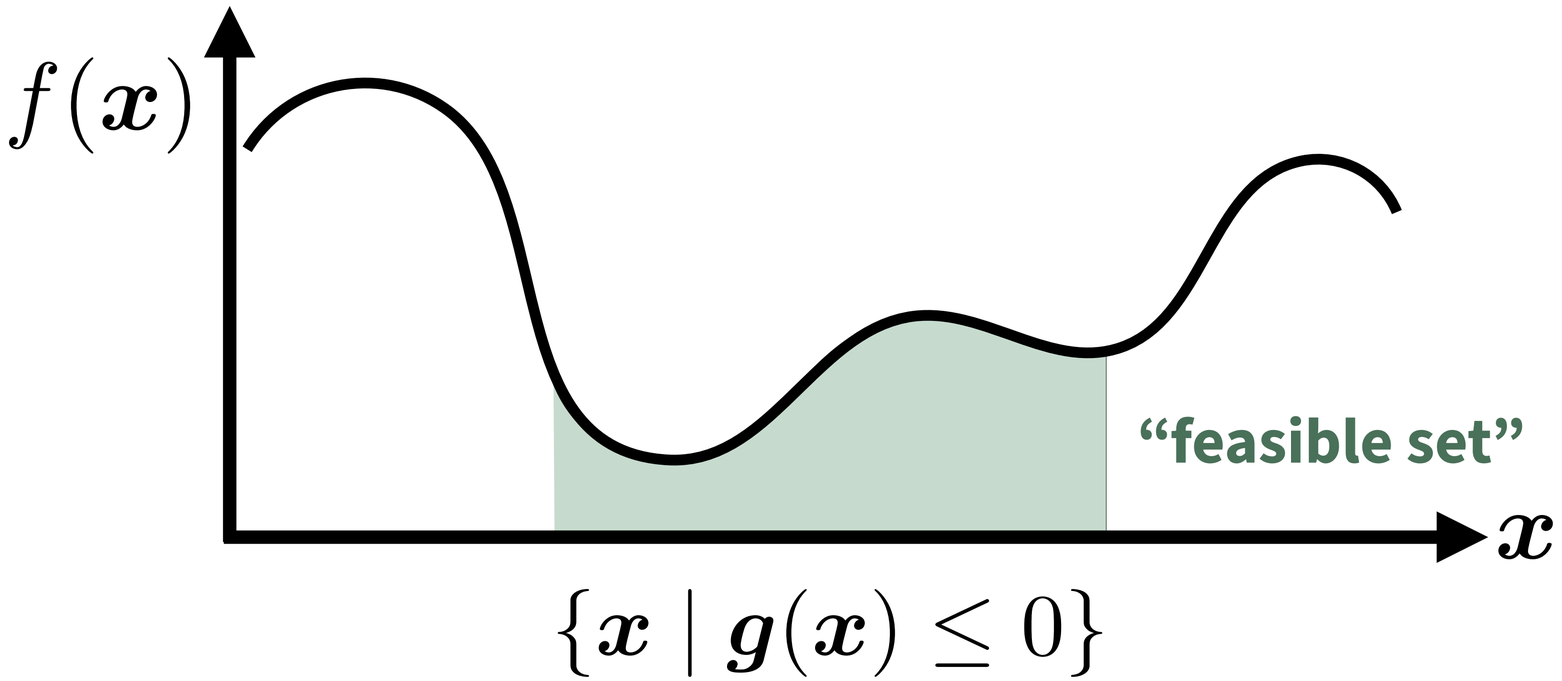
Some optimization terminology / notation



$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \quad f(\boldsymbol{x})$$

$$\text{s.t.} \quad \boldsymbol{g}(\boldsymbol{x}) \leq 0$$

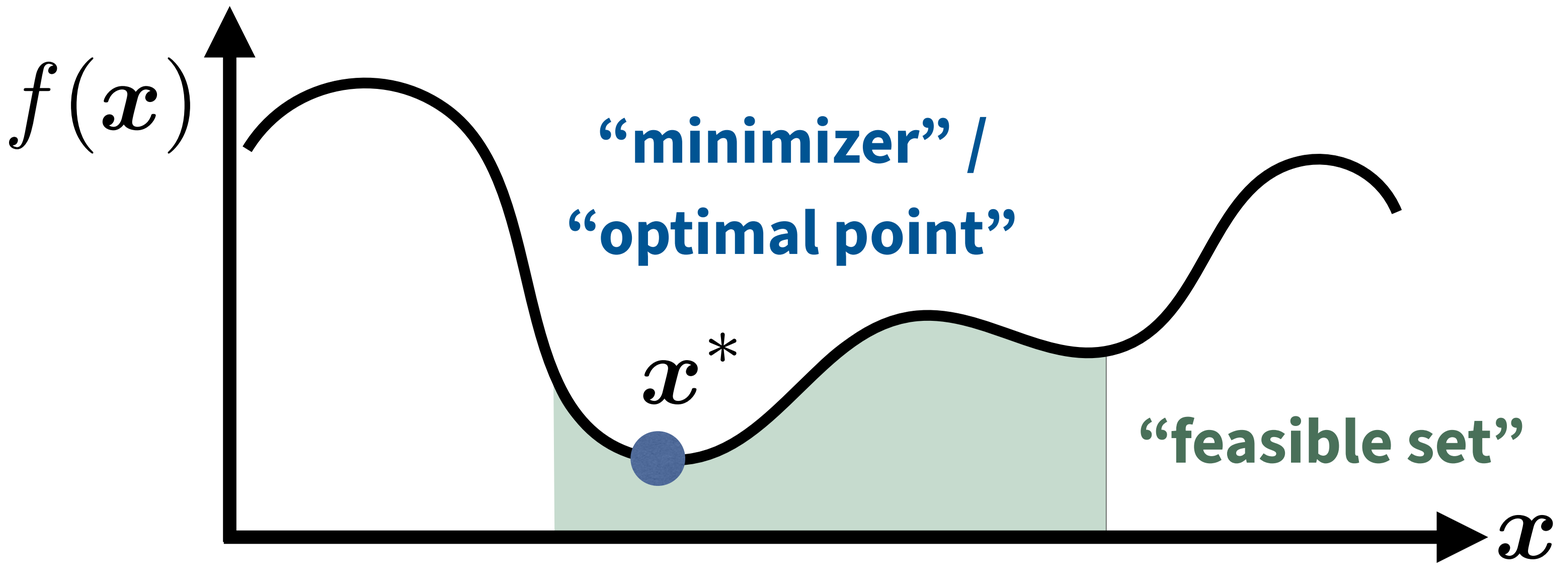
Some optimization terminology / notation



$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \quad f(\boldsymbol{x})$$

$$\text{s.t.} \quad \boldsymbol{g}(\boldsymbol{x}) \leq 0$$

Some optimization terminology / notation



$$\{\boldsymbol{x} \mid \boldsymbol{g}(\boldsymbol{x}) \leq 0\}$$

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \quad f(\boldsymbol{x}) \qquad \text{s.t.} \quad \boldsymbol{g}(\boldsymbol{x}) \leq 0$$

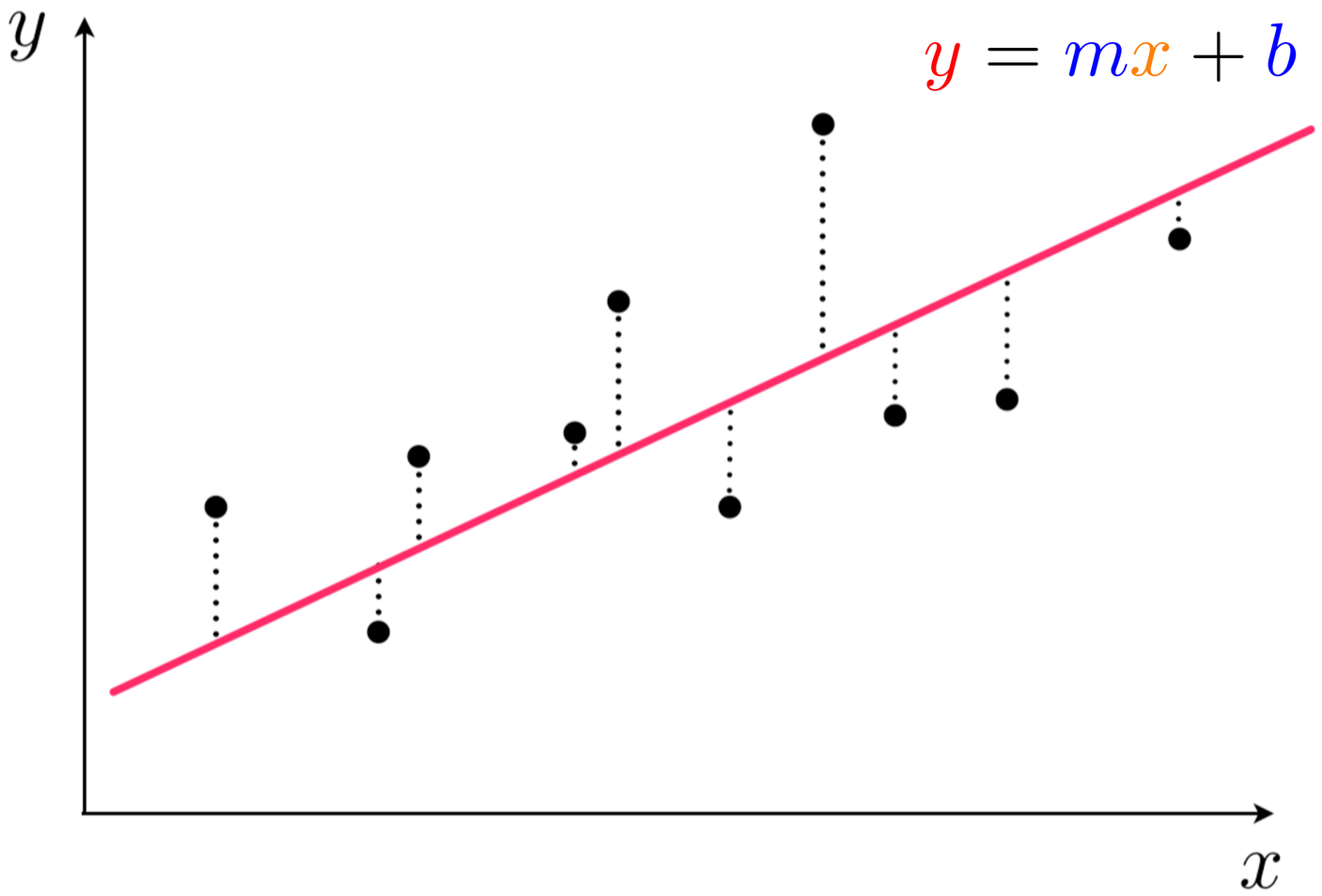
$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \quad f(\boldsymbol{x}) \qquad \text{s.t.} \quad \boldsymbol{g}(\boldsymbol{x}) \leq 0$$

Core philosophy:

- Instead of specifying a decision rule (controller, estimator, …) $\boldsymbol{x}(\theta)$
- Articulate what you want (costs, constraints, parameters, …) $f(\boldsymbol{x}), \boldsymbol{g}(\boldsymbol{x})$
- Let a (general) optimization algorithm find the best decision.

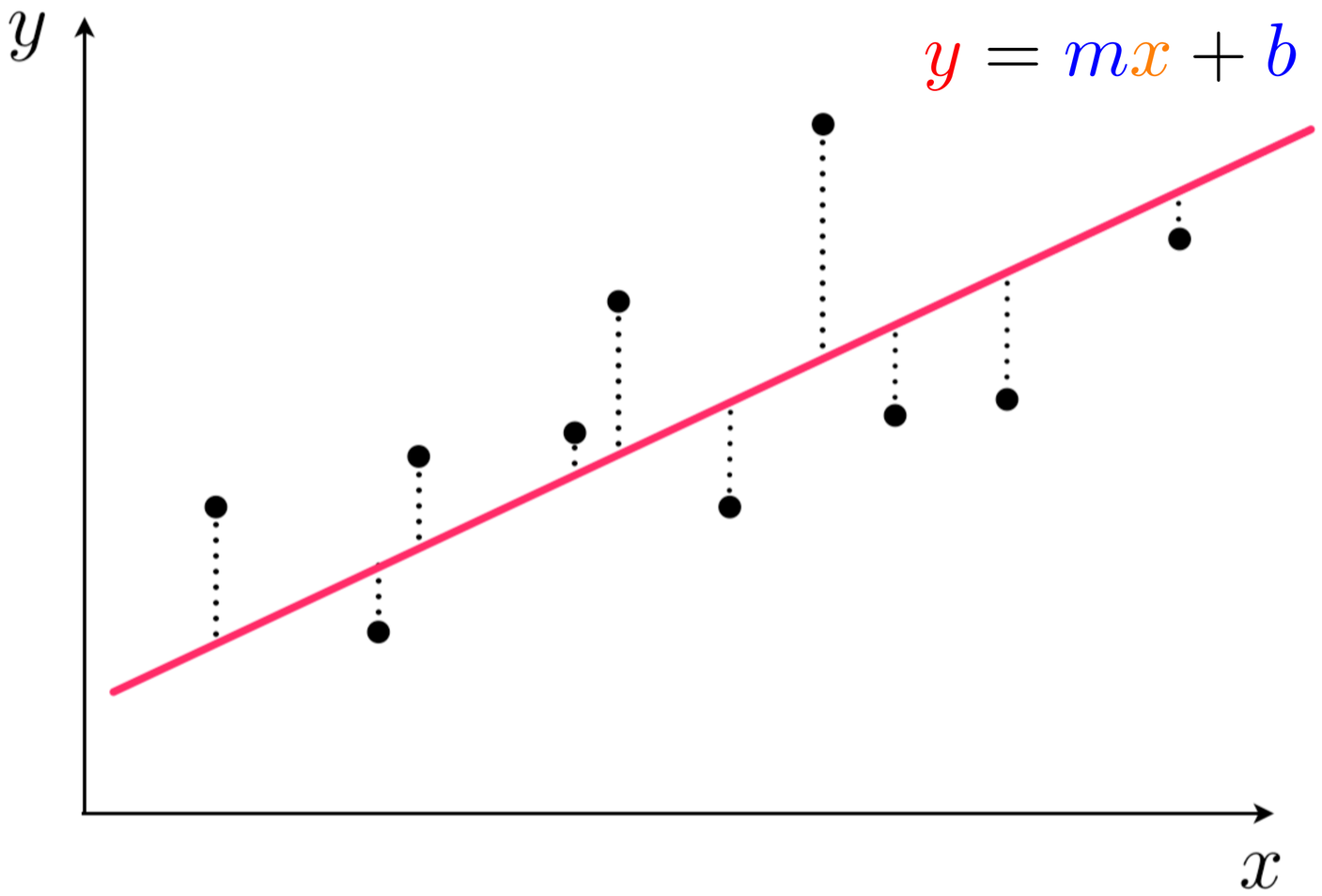
$$\boldsymbol{x}^* = \arg\min\ f(\boldsymbol{x})$$
$$\text{s.t. } g(\boldsymbol{x}) \leq 0$$

e.g., least-squares fitting



$$\min_{m,b} \sum_{i=1}^{n} (y_i - mx_i - b)^2$$

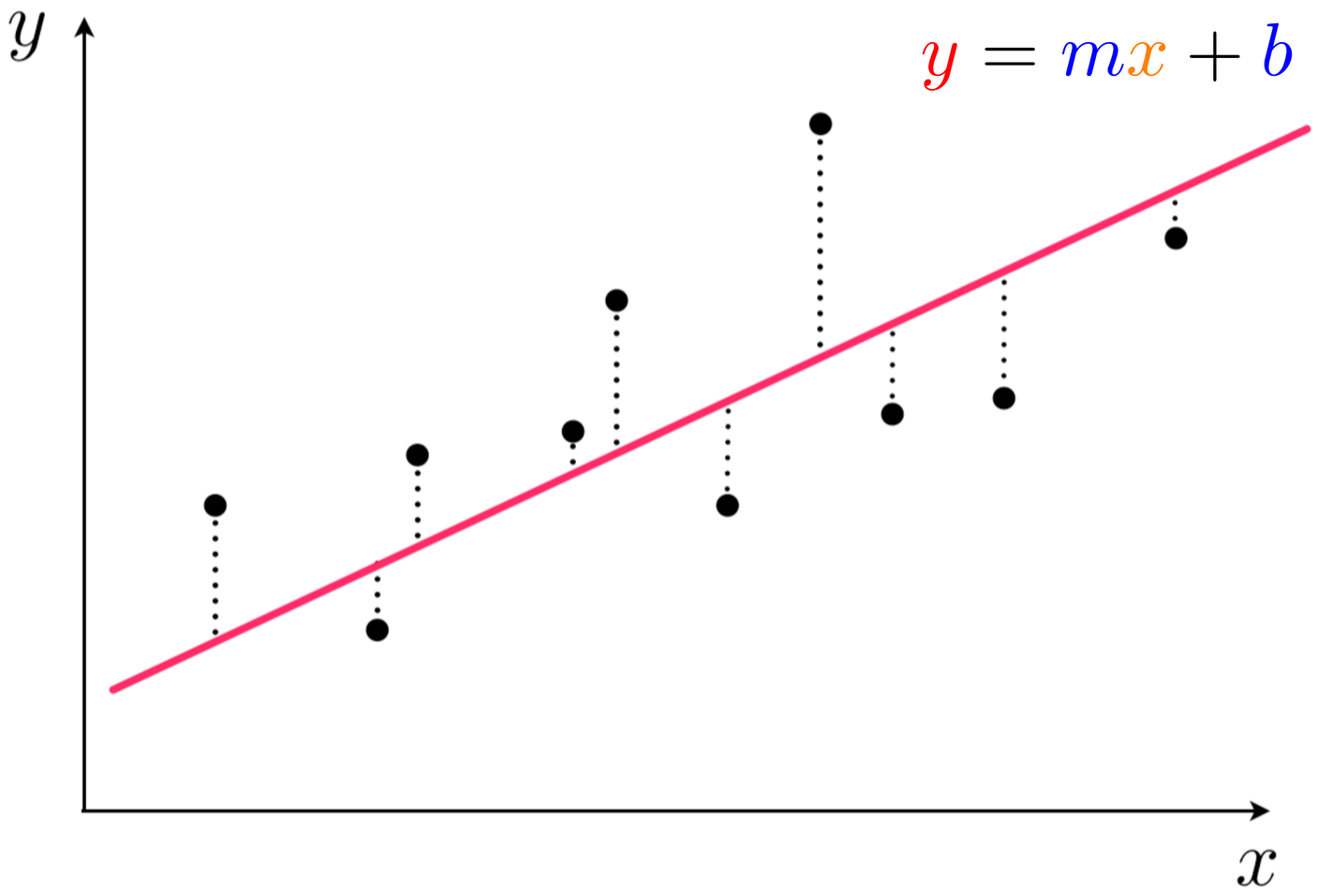
e.g., least-squares fitting



$$\min_{m,b} \sum_{i=1}^{n} (y_i - mx_i - b)^2$$

$$\min_{\boldsymbol{\theta}} \quad ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}||_2^2$$

e.g., least-squares fitting

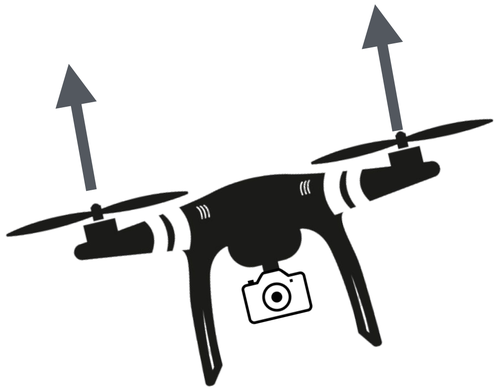


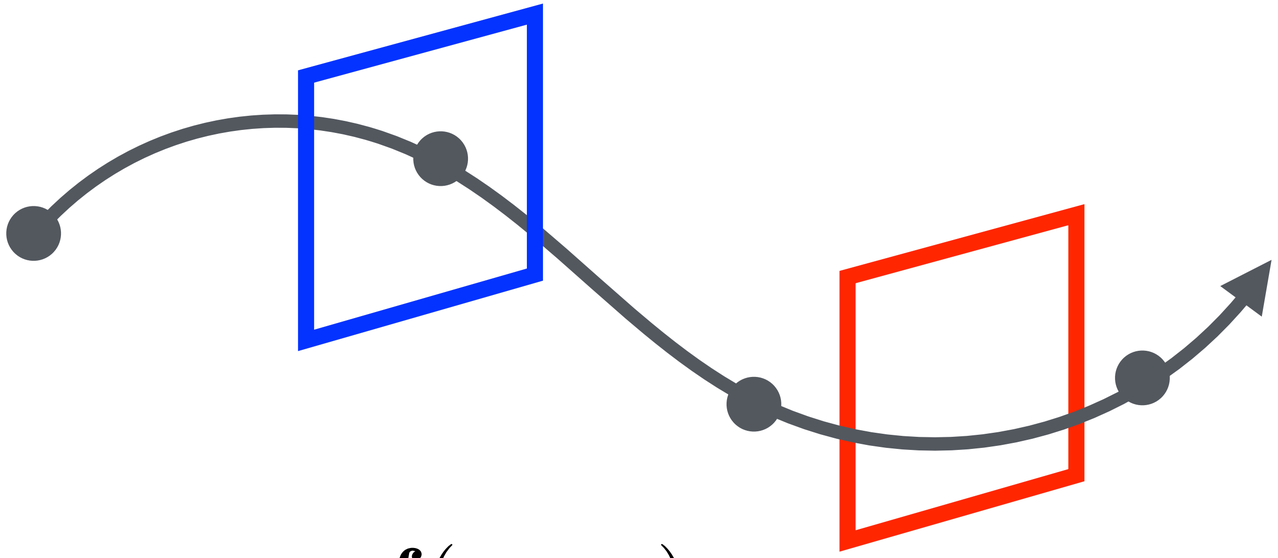
$$\min_{m,b} \sum_{i=1}^{n} (y_i - mx_i - b)^2$$

$$\min_{\boldsymbol{\theta}} \quad ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}||_2^2$$

$$\boldsymbol{\theta}^* = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

What's so special about robotics problems?





states: $\boldsymbol{x}_t$

controls: $\boldsymbol{u}_t$

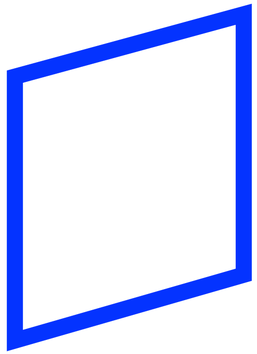
$$\boldsymbol{x}_{t+1} = \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t)$$

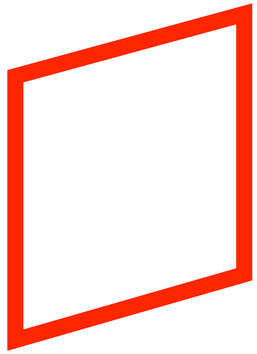
1. Temporal Structure ⏰ **2. Geometry** 📐 **3. Low compute** 👾

Problems we'll study in this class:

1. Optimal Control / Trajectory Opt.

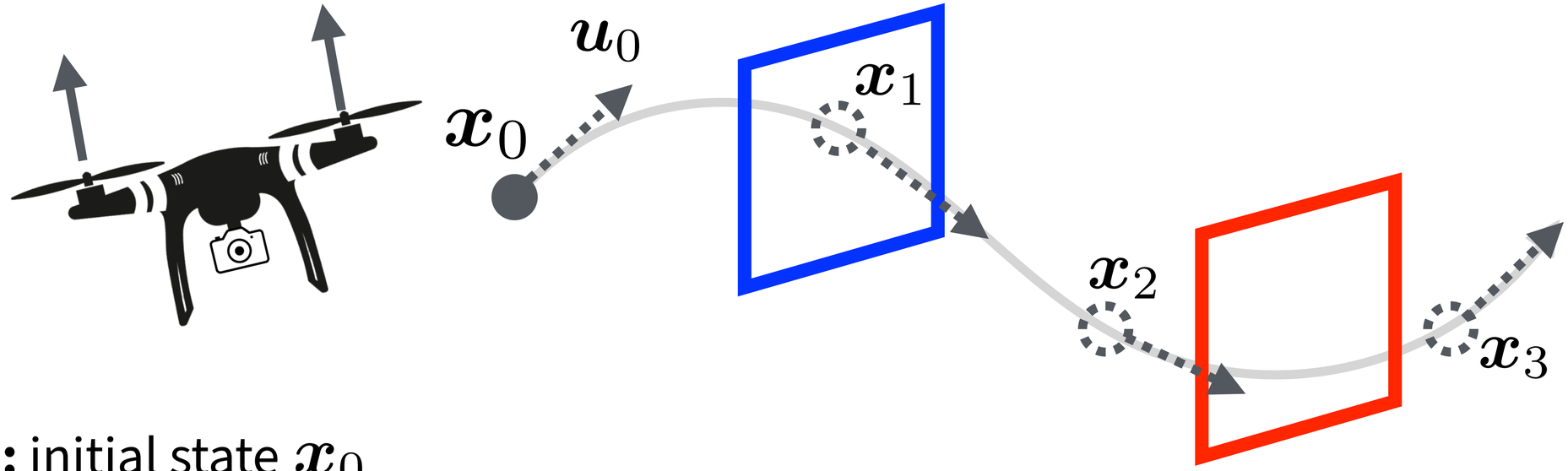
x_0





Known: initial state x_0

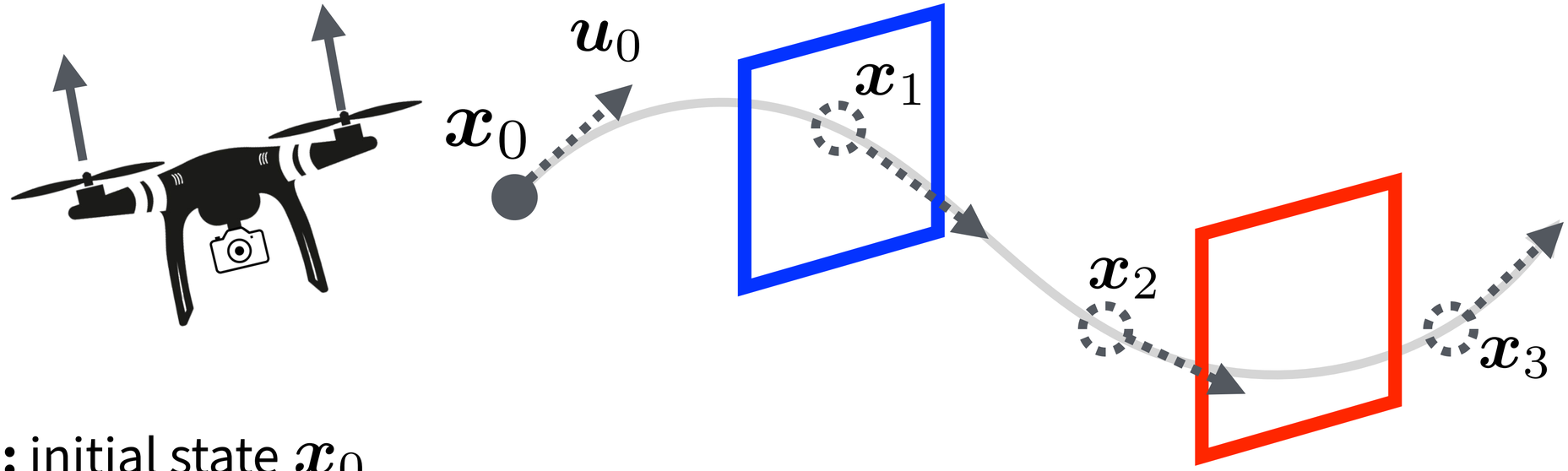
1. Optimal Control / Trajectory Opt.



Known: initial state x_0

Decisions: controls, future states $u_{0:t-1}, x_{1:t}$

1. Optimal Control / Trajectory Opt.



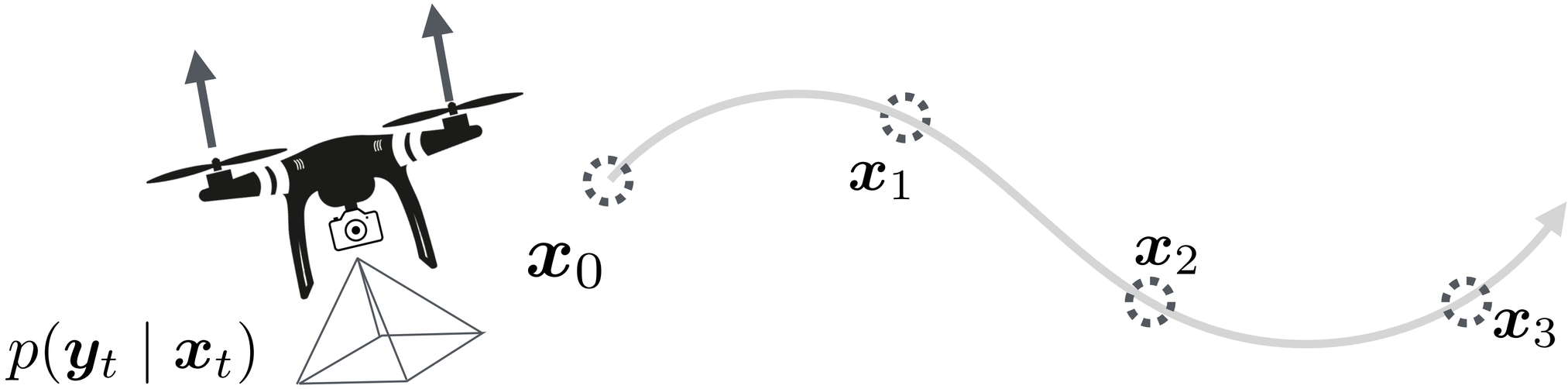
Known: initial state $\boldsymbol{x}_0$

Decisions: controls, future states $\boldsymbol{u}_{0:t-1}, \boldsymbol{x}_{1:t}$

Costs: energy, time $J(\boldsymbol{x}_{1:t}, \boldsymbol{u}_{0:t-1})$

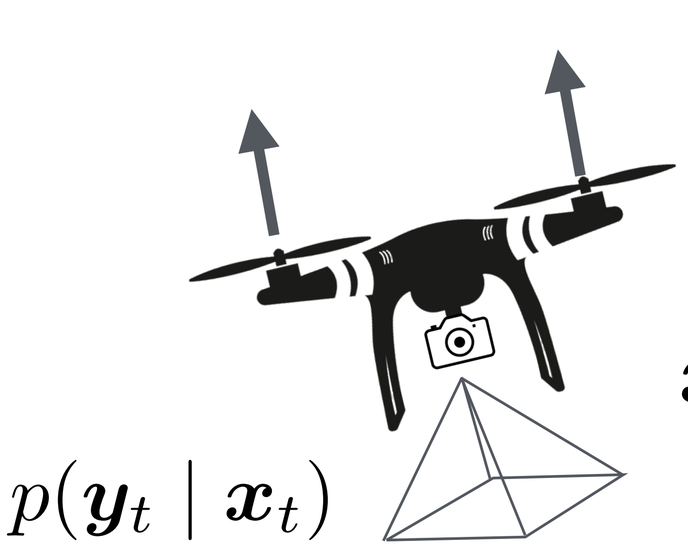
Constraints: collisions, physics

2. State Estimation

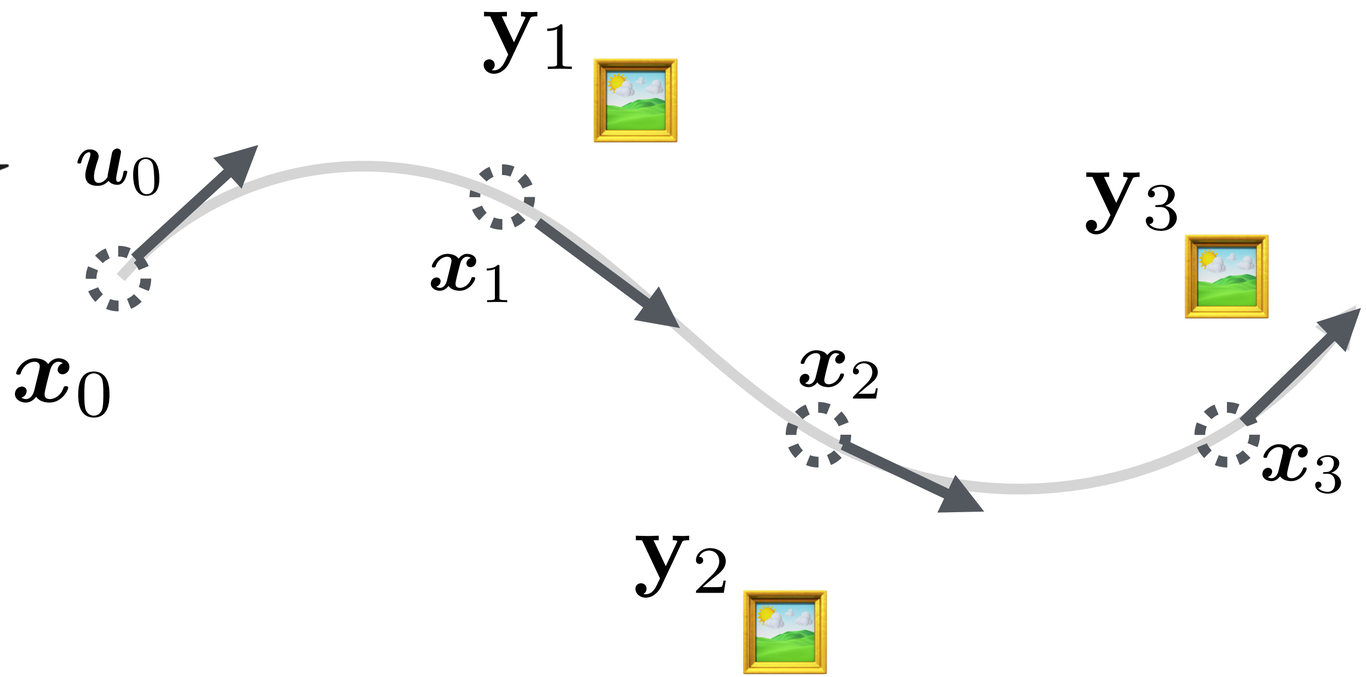


Measurement model

2. State Estimation

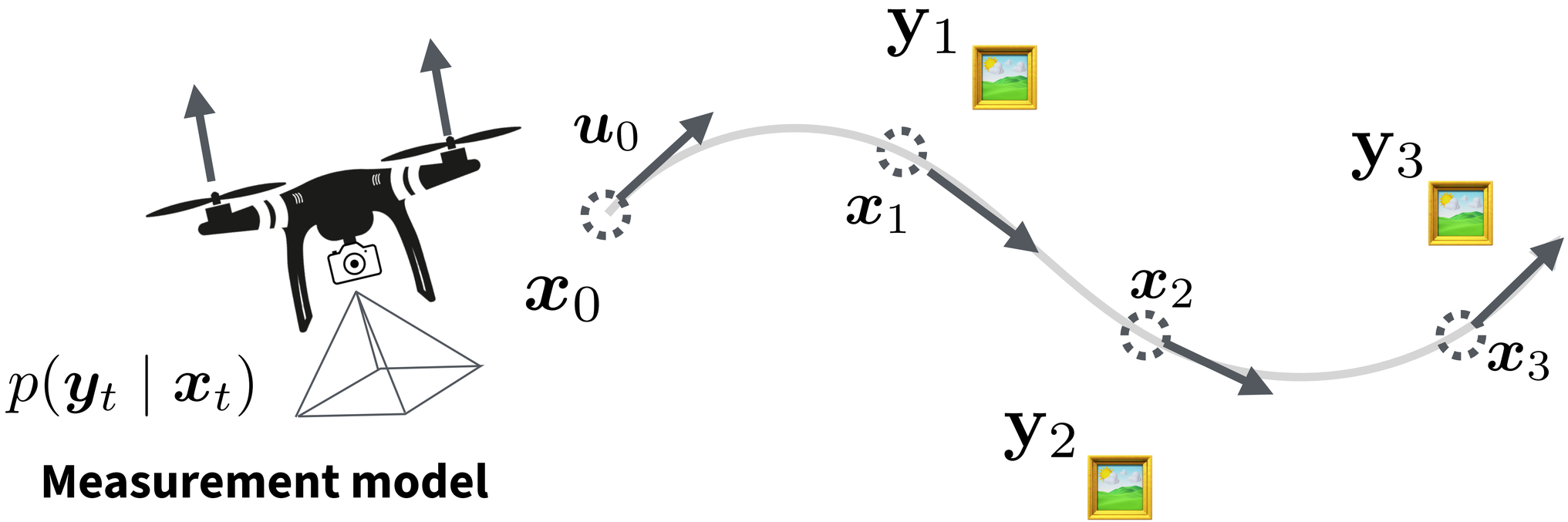


Measurement model



Known: controls, measurements $\mathbf{u}_{0:t}, \mathbf{y}_{0:t}$

2. State Estimation



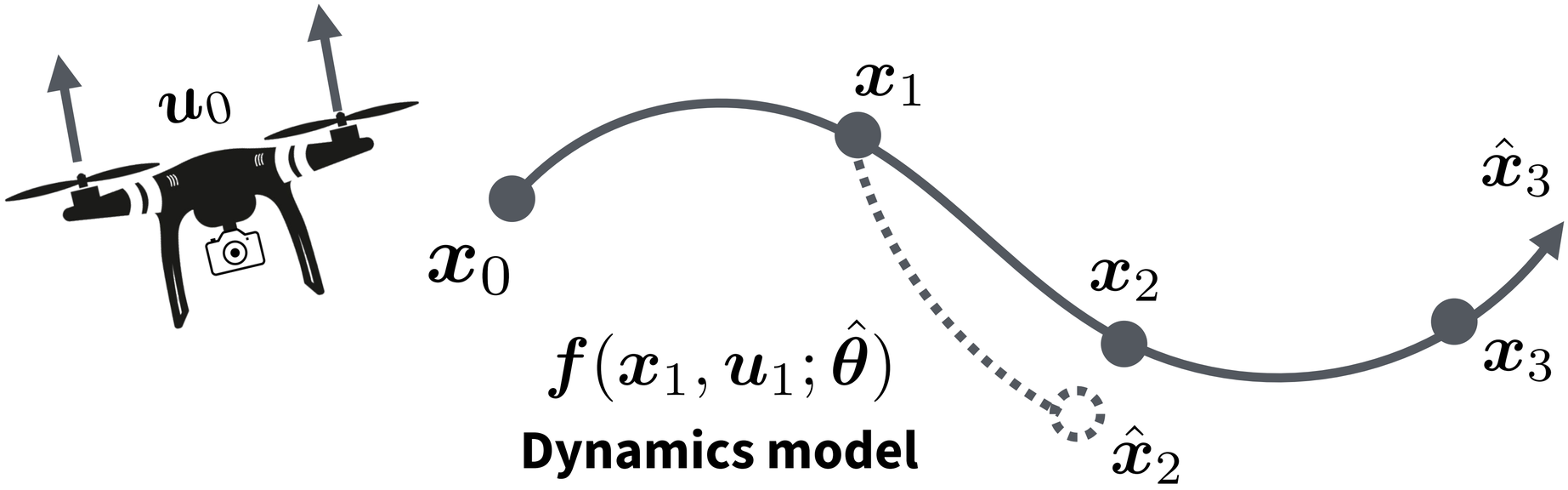
Measurement model

Known: controls, measurements $\mathbf{u}_{0:t}, \mathbf{y}_{0:t}$

Decisions: states $\mathbf{x}_{0:t}$

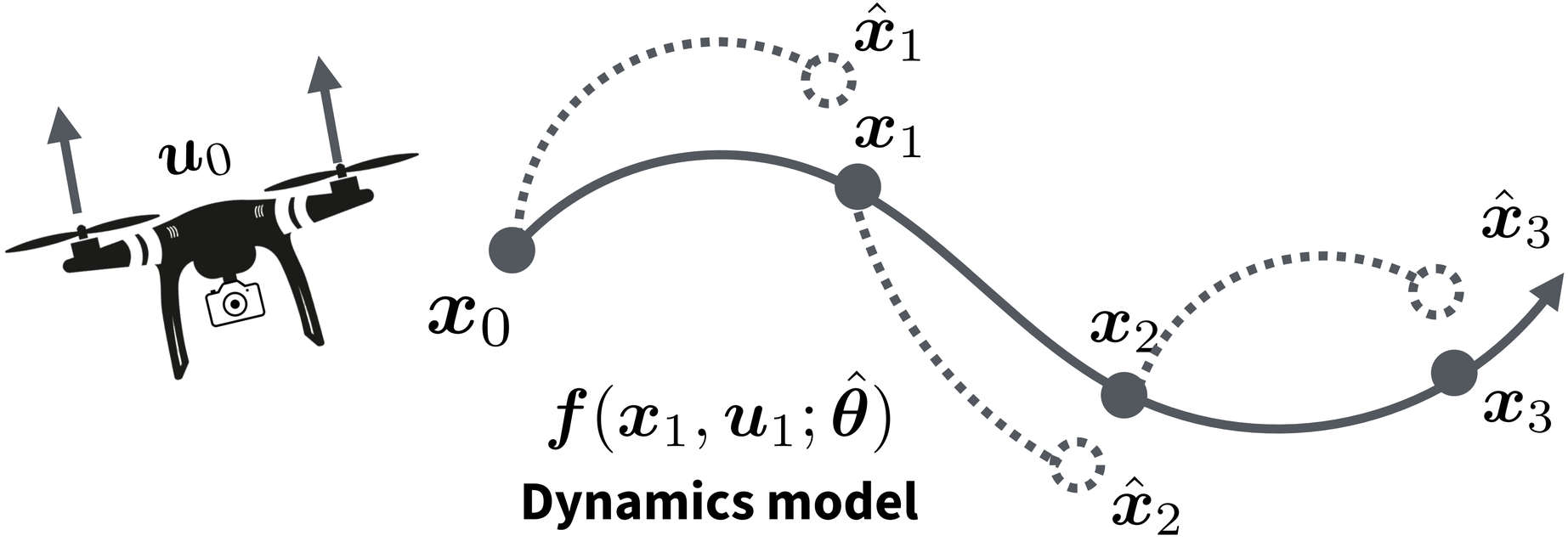
Costs: likelihood $p(\mathbf{x}_{0:t} \mid \mathbf{u}_{0:t}, \mathbf{y}_{0:t})$

3. System Identification



Known: states, controls $\mathbf{x}_{0:t}$, $\mathbf{u}_{0:t-1}$

3. System Identification



Known: states, controls $\mathbf{x}_{0:t}$, $\mathbf{u}_{0:t-1}$

Decisions: dynamics parameters $\boldsymbol{\theta}$

Costs: reconstruction error $\sum_{i=0}^{t-1} ||\boldsymbol{x}_{i+1} - \boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{u}_i; \boldsymbol{\theta})||$

Optimization provides a unified computational approach to all of these problems!

[logistics 🔧]

Logistics

Lectures: Interactive, emphasis on math/applications

Assignments: [4 assignments * 12.5% each = 50%]

Mixed written/Python, emphasis on applications

Project: [Proposal 5% + Milestone 10% + Report/Video 30% = 45%]

Open-ended, apply tools from class to an interesting problem. Groups of ≤ 3. Be creative!

Participation: [Intro 1% + Mid-semester 1% + In-class questions 3% = 5%]

Class surveys, lecture activities.

Bonus: [up to 5%]

0.5% per instructor-endorsed answer on Ed.

Resources

Course Website: https://cs.cornell.edu/courses/cs5757/2026sp

Ground truth - all resources posted here, including HW

Ed Discussion: https://edstem.org/us/courses/94233/discussion

Easiest place for Q/A - both on technical / logistical questions.

Gradescope: https://www.gradescope.com/courses/1218746

For homework submission and grading.

Canvas: https://canvas.cornell.edu/courses/86534

Homework solutions and final grade post.

Course structure

1. Robotics fundamentals

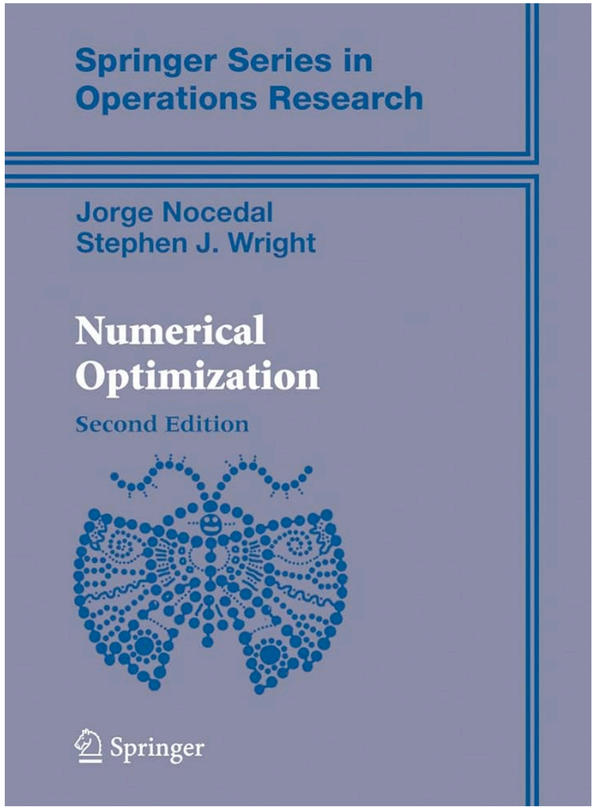
2. Numerical methods

3. Optimal control

4. State estimation

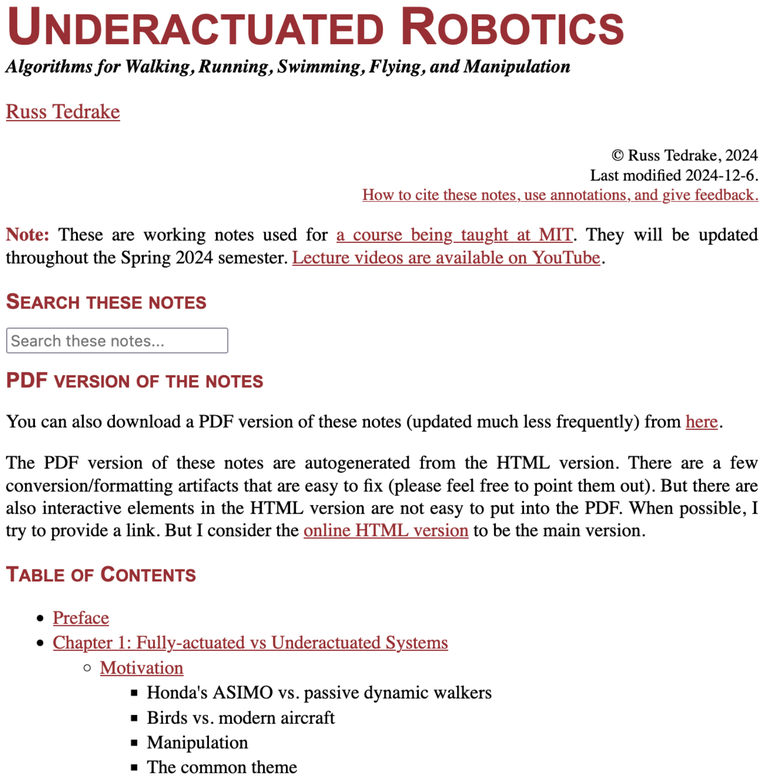
5. Robustness

Course materials



Numerical Optimization

Nocedal and Wright



UNDERACTUATED ROBOTICS

Algorithms for Walking, Running, Swimming, Flying, and Manipulation

Russ Tedrake

Last modified 2024-12-6.
How to cite these notes, use annotations, and give feedback.

Note: These are working notes used for a course being taught at MIT. They will be updated throughout the Spring 2024 semester. Lecture videos are available on YouTube.

SEARCH THESE NOTES

Search these notes...

PDF VERSION OF THE NOTES

You can also download a PDF version of these notes (updated much less frequently) from here.

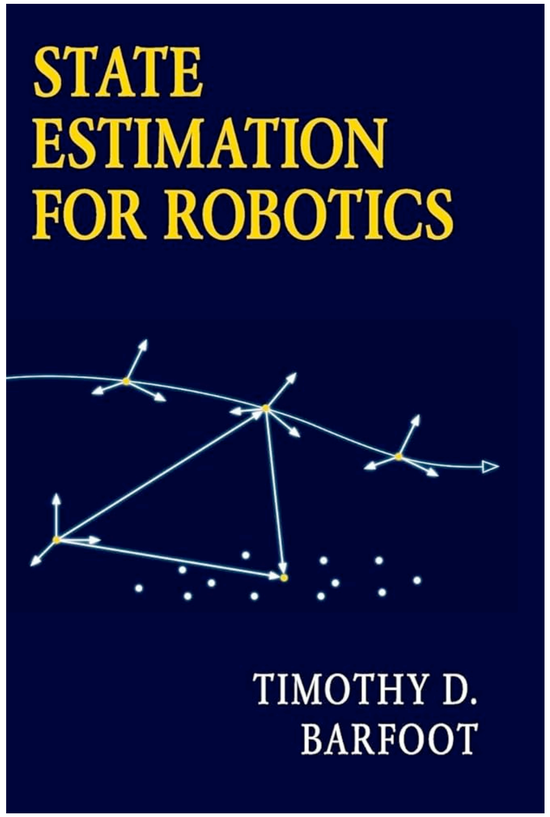
The PDF version of these notes are autogenerated from the HTML version. There are a few conversion/formatting artifacts that are easy to fix (please feel free to point them out). But there are also interactive elements in the HTML version are not easy to put into the PDF. When possible, I try to provide a link. But I consider the online HTML version to be the main version.

TABLE OF CONTENTS

- Preface
- Chapter 1: Fully-actuated vs Underactuated Systems
  - Motivation
    - Honda's ASIMO vs. passive dynamic walkers
    - Birds vs. modern aircraft
    - Manipulation
    - The common theme

Underactuated Robotics

Tedrake



State Est. for Robotics

Barfoot

Expectations

- This class will be ***fun*** but ***hard!*** A strong math background (especially in linear algebra, vector calculus) is needed to do well.
- Assignments are programming-heavy! Need comfort with Python and / or scientific computing (`numpy`, `Julia`, `MATLAB`, etc).
- We released an ungraded pre-test (HW0) on the course website.
  If you feel uncomfortable with any material here, you'll likely need to self-study to catch up.

Late days

- Assignments will be due on Gradescope at 11:59pm on their due date (Thursdays).
- Students have **six** late days they can use across assignments, with a max of **three** days for each assignment.
- Extensions beyond these late days will only be given in extenuating circumstances (e.g., documented illness).

Generative AI

“You may use generative AI tools (ChatGPT, Copilot, and so on) in this course but should **treat them as you would a human collaborator.** This means that their use should be **clearly disclosed** (assignments will require a statement on AI usage), and that **copying solutions exactly from them is not acceptable.**

We do not recommend the use of IDEs with integrated AI tools (e.g., Cursor) for this class, and will treat the use of agentic workflows (i.e., automatic file creation and editing) as a form of copying code.”

First assignment: Course survey

Quick intro survey! Getting to know you and your backgrounds.

Worth 1% of participation.

Questions?