Tutorial on Computational Design for Robotics

Introduction

Plan of the tutorial

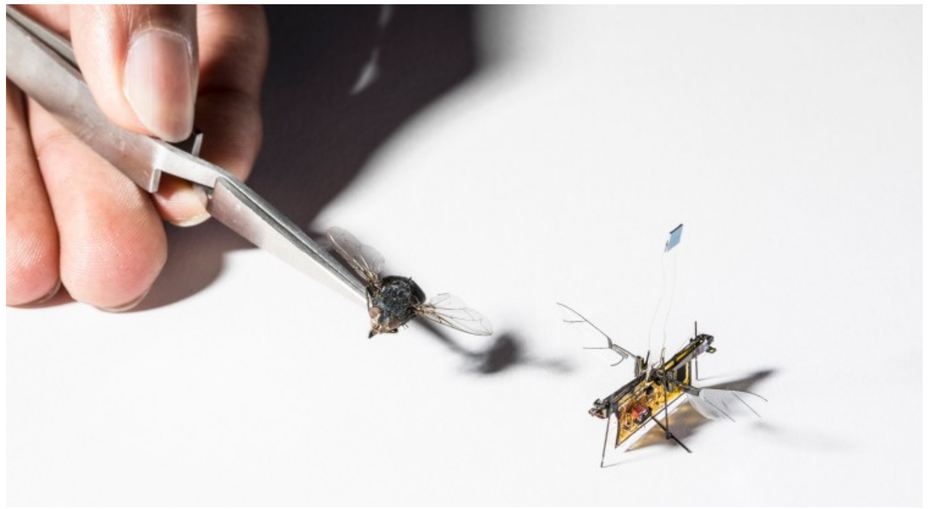
Time (EDT)	Speaker	Title
09:00-09:15	Gioele Zardini (ETH Zürich)	*Intro and plan of the tutorial*
09:15-10:15	Andrew Spielberg (Harvard University)	*Computational Design*
10:15-11:00	Coffee break	
11:00-12:30	Andrew Spielberg (Harvard University)	*Computational Design and hands-on activity*
12:30-14:00	Lunch beak	
14:00-14:45	Gioele Zardini (ETH Zürich)	*A monotone theory of co-design*
14:45-onwards	Andrea Censi (ETH Zürich)	*Co-design hands-on*

Designing today's engineering systems is as crucial as complex

- **Autonomous systems** as a proxy for **complex systems**, which might have **positive societal impact**

Autonomy for safer and efficient mobility (Motional)

Autonomous robots for space exploration (Pavone et al.)



Roboflies to monitor environments (Fuller et al.)

UAVs for search and rescue tasks (Scaramuzza et al.)

Need new tools to model and solve complex systems design optimization problems

- **Societal impact** of new technologies depends on their **joint design** with **existing systems**:

Intermodal mobility networks (NASA UAM)

Networks of tankers (Signal Ocean)

Example - Autonomy: **Heaven** or **hell**?

***30%** of the cars would be enough*

First**- and **last-mile** mobility could make **public transit** more **convenient** and **attractive

*More **affordable**, **sustainable***

Your Uber Car Creates Congestion. Should You Pay a Fee to Ride?(New York Times)

Single components are slowly well understood, but we still lack a (*formal* and *practical*) theory for the **task-driven co-design** of **complex systems**

The vision of automated co-design

minimize
(resources usage)
subject to
(functionality constraints)

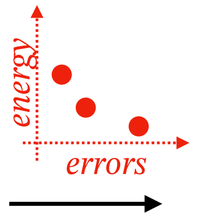
task specification
multi-domain knowledge
design options

“automated designer”

optimal design(s)

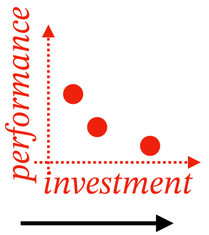
Autonomy co-design

task
robot autonomy, physics
components, algorithms



Mobility co-design

demand
networks, operations, infrastructure
mobility services, policies



Autonomy as the frontier of complexity for the co-design of complex systems

A fleet of autonomous vehicles

=

software **behavior** **coordination**
hardware
actuation **localization** **planning** **invasivity**
sensing
control **interaction** **learning** **liability**
computation
perception **mapping** **regulations**
energetics **communication** **infrastructure**

So many **components** (hardware, software, ...), and **choices** to make!

Nobody understands the **whole** thing!

We forget why we made **choices**, and we are afraid to make **changes** (high failure cost).

We need **faster** design cycles, **nimbler** execution.

anthropomorphization of 21st century engineering malaise

"My dear, it's simple: you lack a theory of **co-design**!"

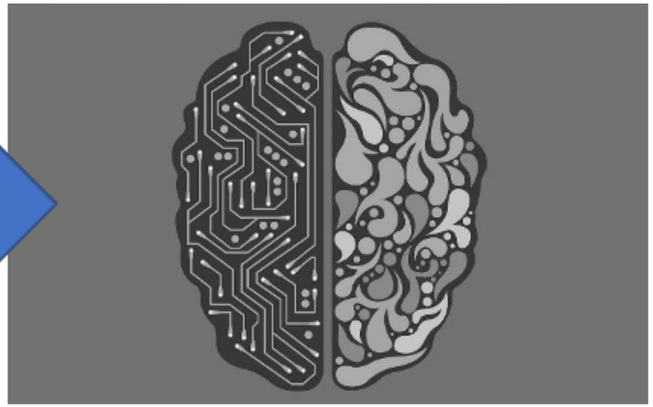
Formal
Quantitative
Intellectually tractable

Robot design is largely manual

- **Slow**, spiraling, linear workflow

RobotAcademy.com

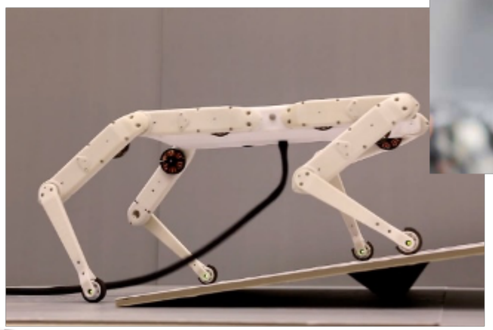
Boston Dynamics



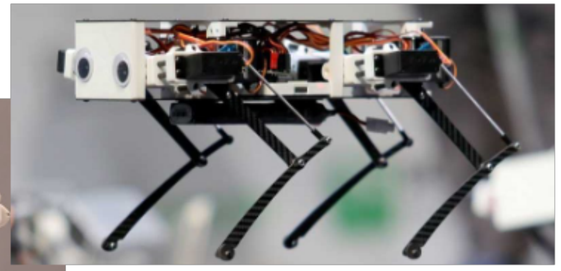
Robot design is largely manual

- Mistakes are **expensive**, stick to what exists

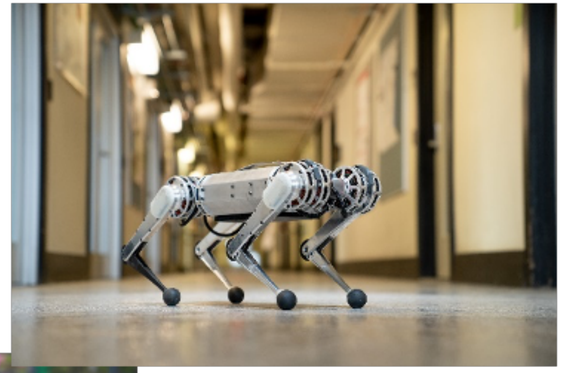
Boston Dynamics



Open Dynamic Robot Initiative



Stanford Pupper

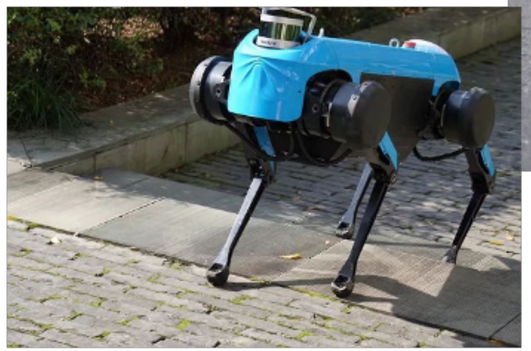


MIT Cheetah

ANYmal

Unitree

Xiaomi



Jueying

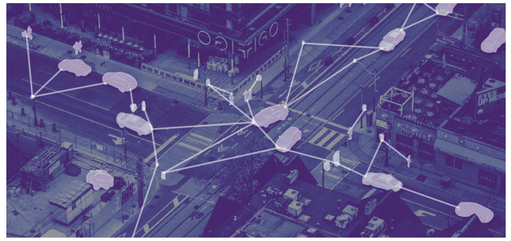
HyQ

Your system is just a component in another person's system

Infrastructure level

Optimal infrastructure choices

Service level



Optimal deployment

Platform level

Choice of components

Subsystem level

Single component design

Complex systems typically feature multi-stakeholders interactions

Mobility providers (Jamey Tesler, CEO MassDOT)

Liability (Christian Mumenthaler, CEO SwissRe)

Policy makers (Michelle Wu, Mayor Boston)

Academia Tech Developers (Sally Kornbluth, MIT president)

Industry (Laura Major, CTO Motional)

Desiderata for the automation of complex systems co-design

- **Formal**
- **Computationally tractable**
  - Need to compute solutions efficiently
- **Compositional, hierarchical**
  - My system is a component of somebody else's system
- **Collaborative**
  - Pooling knowledge from experts across fields.
- **Intellectually tractable**
  - Not exclusively accessible to system architects
- **Continuous**
  - Design is not static: it should be reactive to changes in goals and contexts

Quick instructions for the afternoon: Two ways to run the code

- **Option 1: Using the website editor.zuper.ai**
  - requires a Github account to link to the Github app
  - a convenient visual editor
  - much better error reporting
  - fewer options than the command line interface

- **Option 2: Using Docker on the command line**
  - requires Docker on your computer
  - command line interaction
  - better for longer queries

Using the Docker version

- Requires basic knowledge of Docker.

- Pull the following image:

```
docker pull zupermind/mcdp:2023
```

Plan of the tutorial

Time (EDT)	Speaker	Title
09:00-09:15	Gioele Zardini (ETH Zürich)	*Intro and plan of the tutorial*
09:15-10:15	**Andrew Spielberg (Harvard University)**	***Computational Design***
10:15-11:00	Coffee break	
11:00-12:30	Andrew Spielberg (Harvard University)	*Computational Design and hands-on*
12:30-14:00	Lunch beak	
14:00-14:45	Gioele Zardini (ETH Zürich)	*A monotone theory of co-design*
14:45-onwards	Andrea Censi (ETH Zürich)	*Co-design hands-on*