

W

Autonomous Robotics

Winter 2024

Abhishek Gupta

TAs: Karthikeya Vemuri, Arnav Thareja

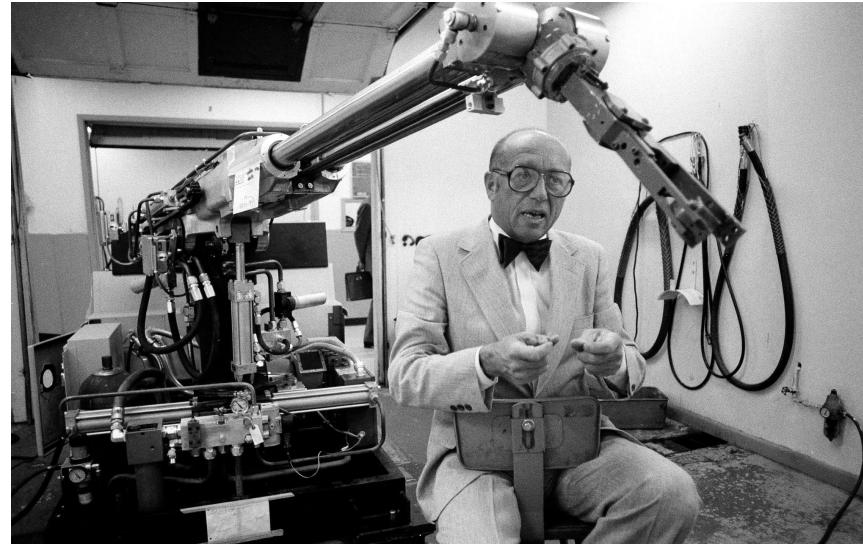
Marius Memmel, Yunchu Zhang



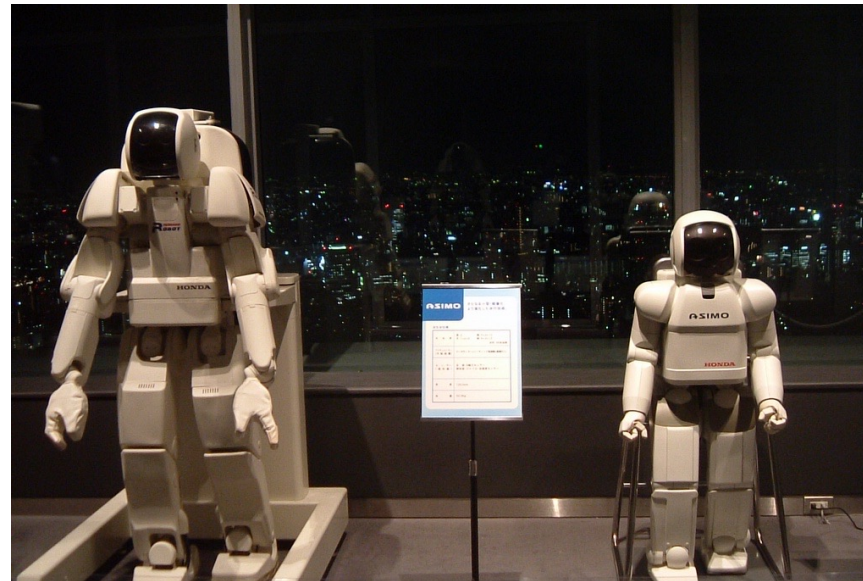
The first wave of robots



Shakey

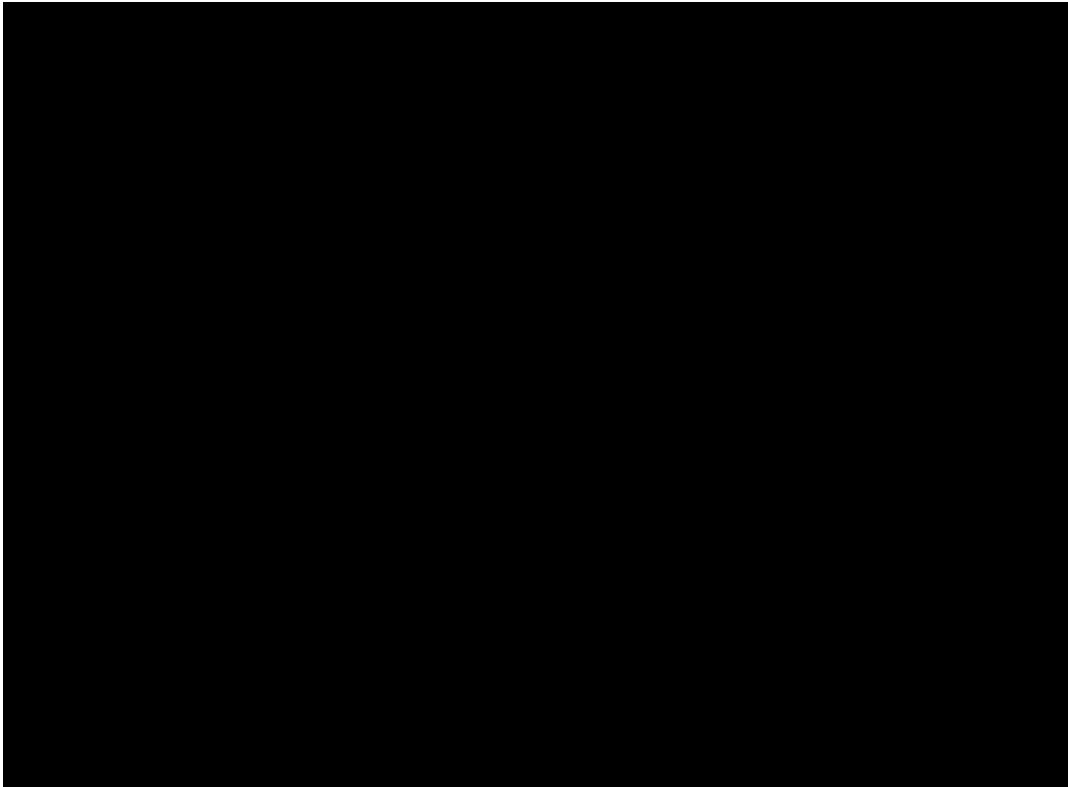


Engelberger
(Unimate ++)



Honda P series

The second wave of robots



DARPA Grand Challenge



PR1
Robot

Robots Today



Everyday Robotics - Google



Atlas – Boston Dynamics

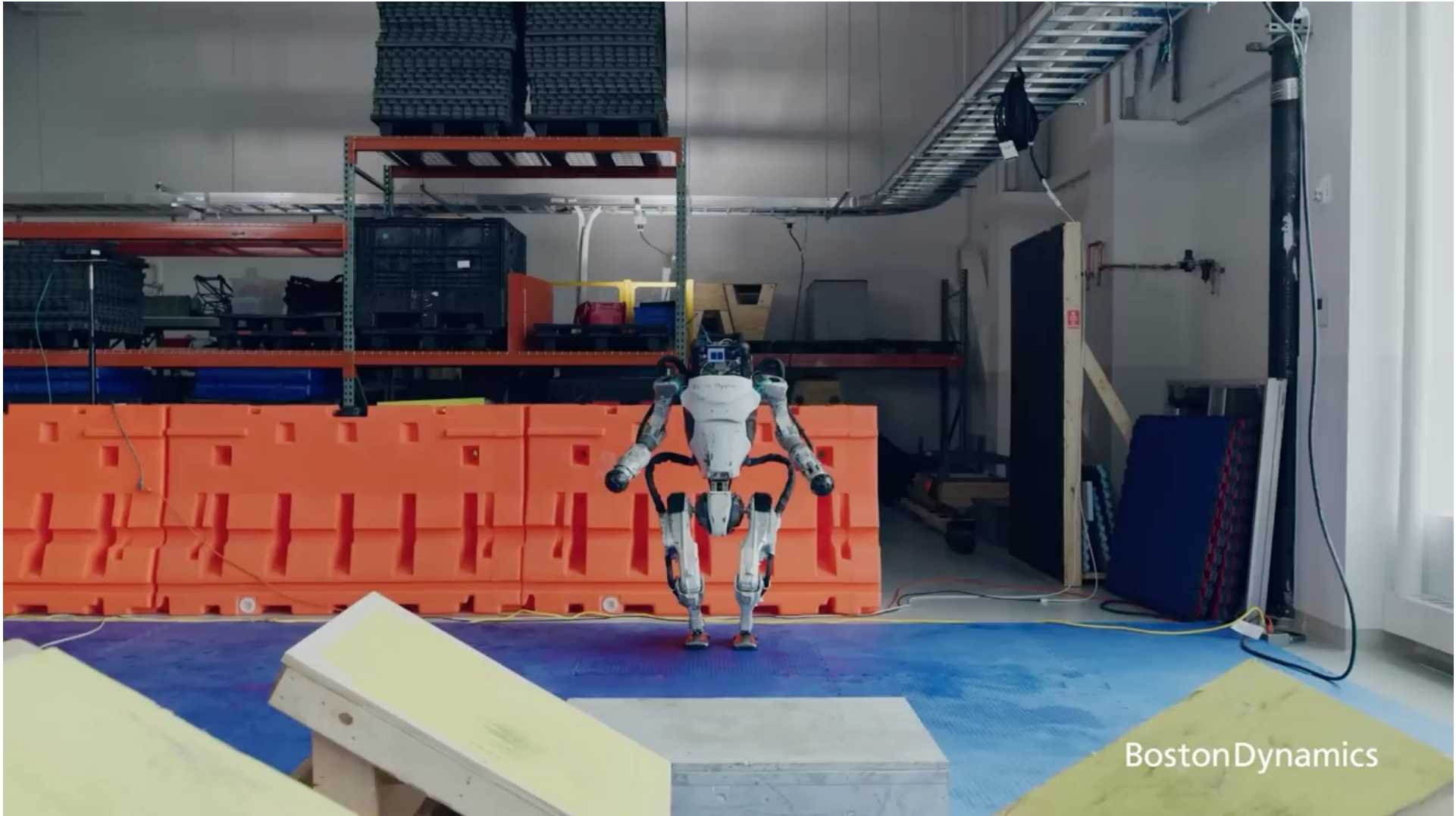
Industrial Robotics Today



Self-Driving Cars



Humanoid Parkour

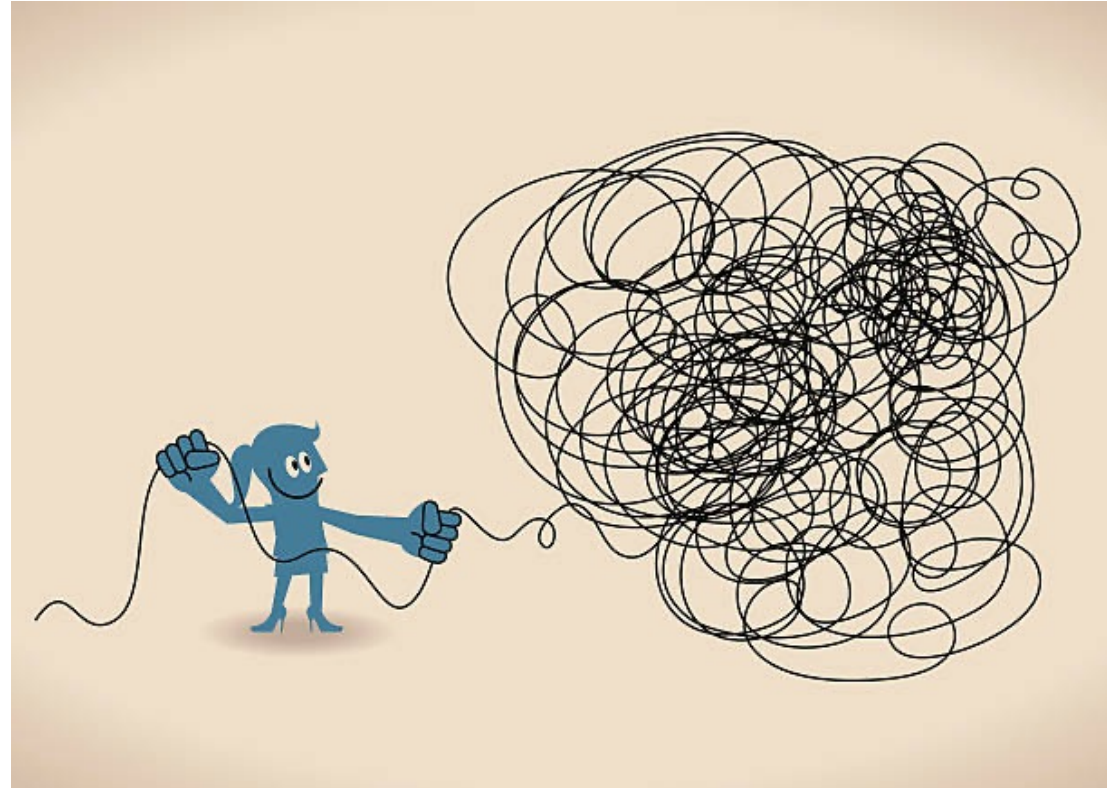


Dexterous Manipulation



Ok this is great – how do we build these robots?

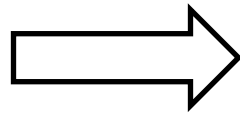
- Need a formal framework for problem definition and a set of tools to solve them



- Sense-plan-act framework with probabilistic inference!

Today's Objective: Work through a robotic design process

- Learn how to think about the design of mobile robot systems
- Work through the steps for a mobile helicopter
- Start thinking about how to do this for a car!



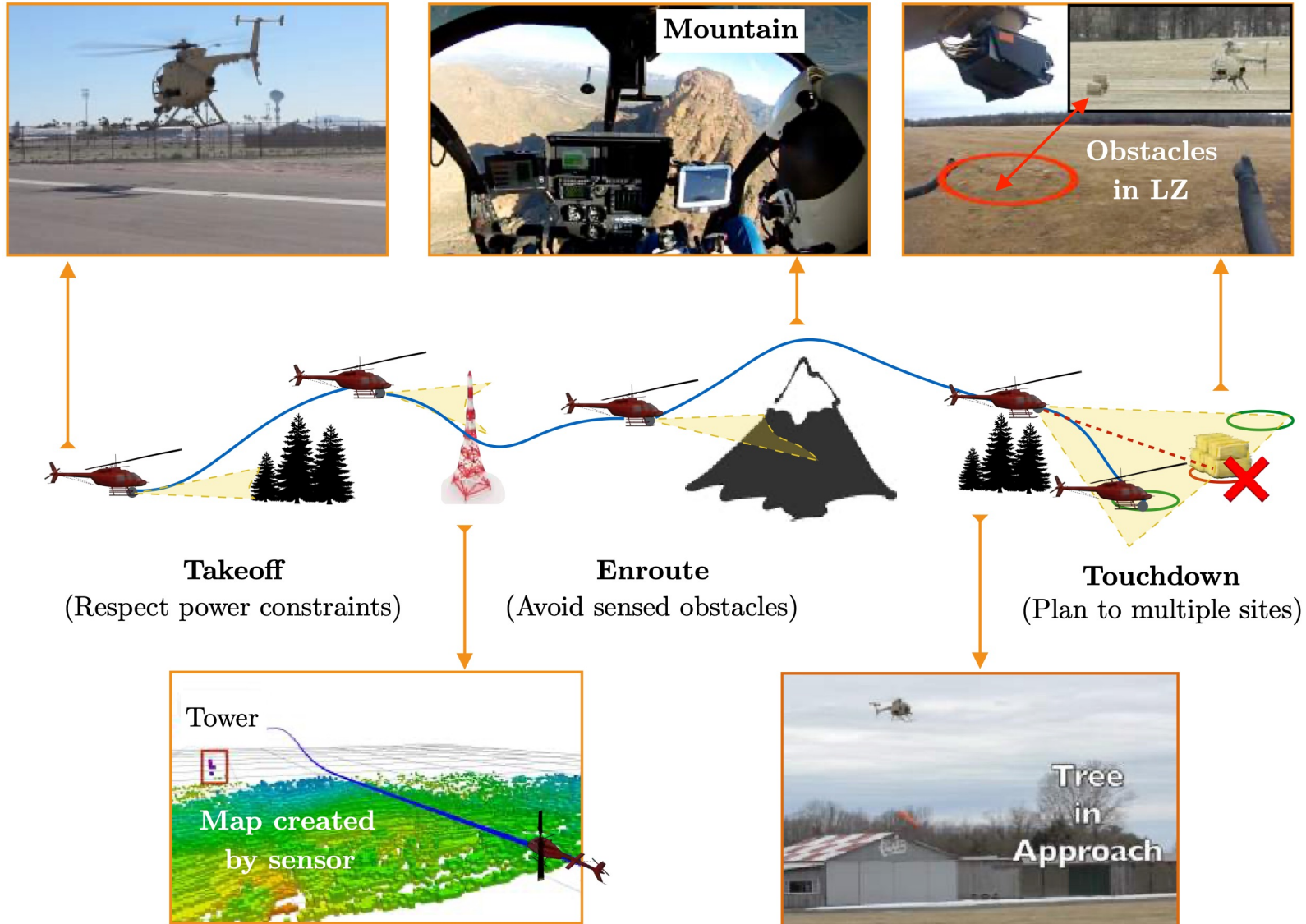
Anatomy of a flying vehicle



Mission: Takeoff to Landing



Mission



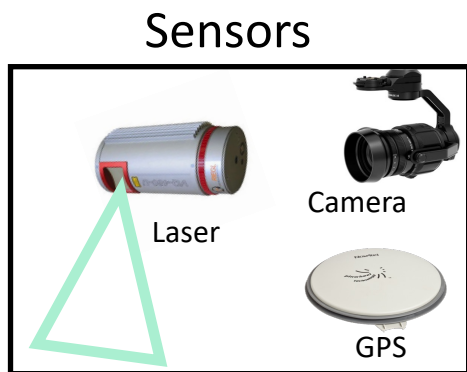
Task: Sets of conditions the robot needs to satisfy

- Given:
 - Start (latitude, longitude), Goal (latitude, longitude)
 - List of no-fly-zones (unsafe air space)
 - Coarse terrain map of continental USA
 - Sensors - GPS, Laser, etc
- Objective:
 - **Minimize time** it takes to complete mission
- Constraint:
 - Don't come close to obstacles / don't enter no-fly-zones
 - Don't exceed limits of the vehicle (flying upside down)
 - Don't run out of fuel!

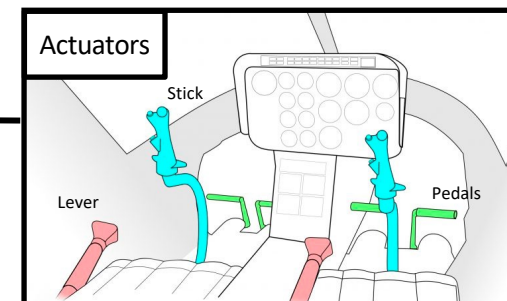


How do we **tractably**
solve the task?: The
model-based way

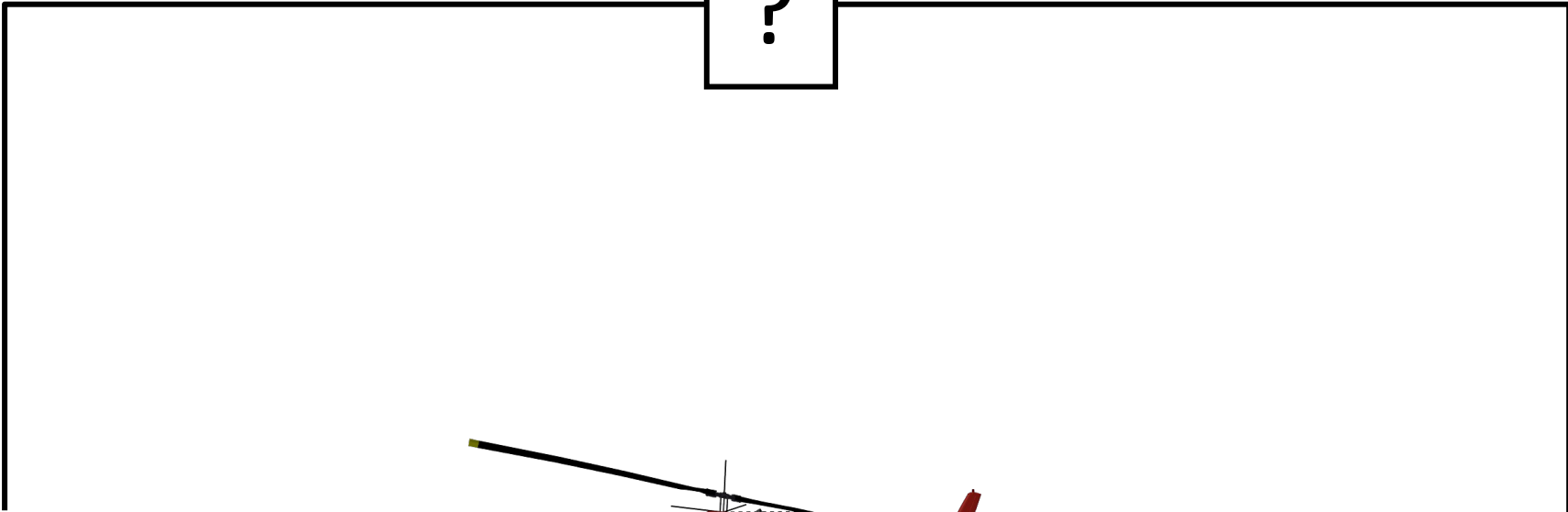
Begin with a blank slate



Robot interacts with environment

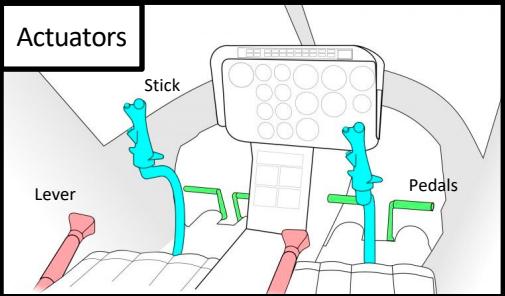
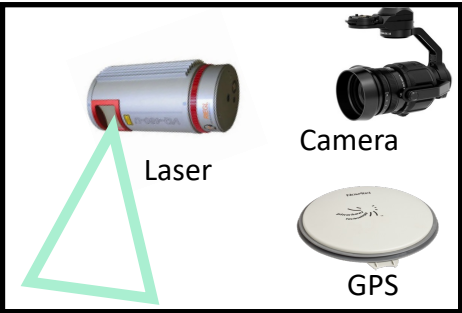


?

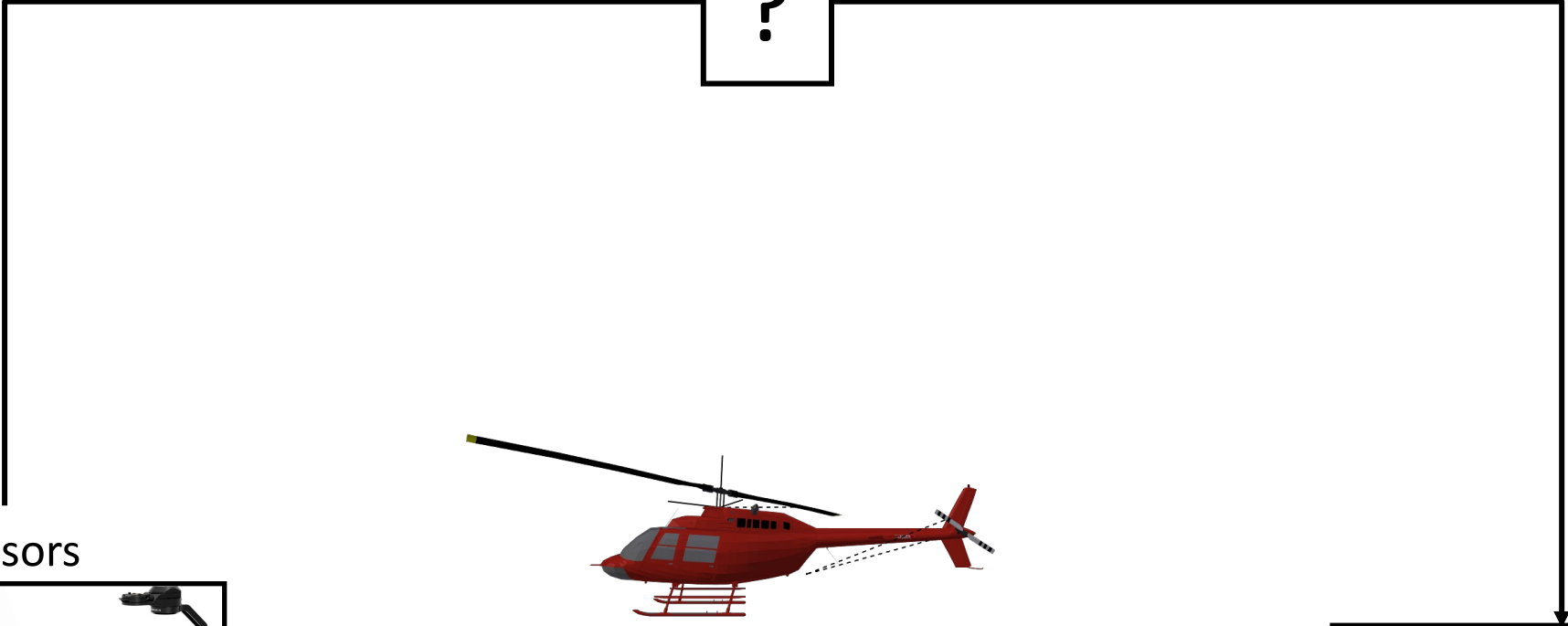
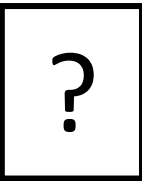
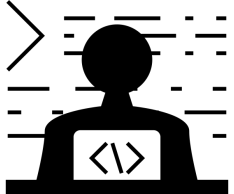


Robot interacts with environment

Sensors

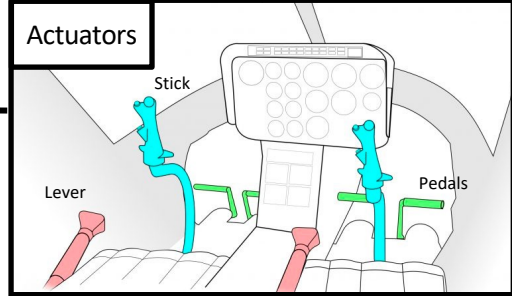
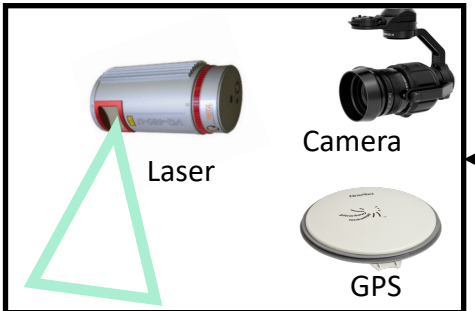


Let's use our knowledge about the world to solve this?

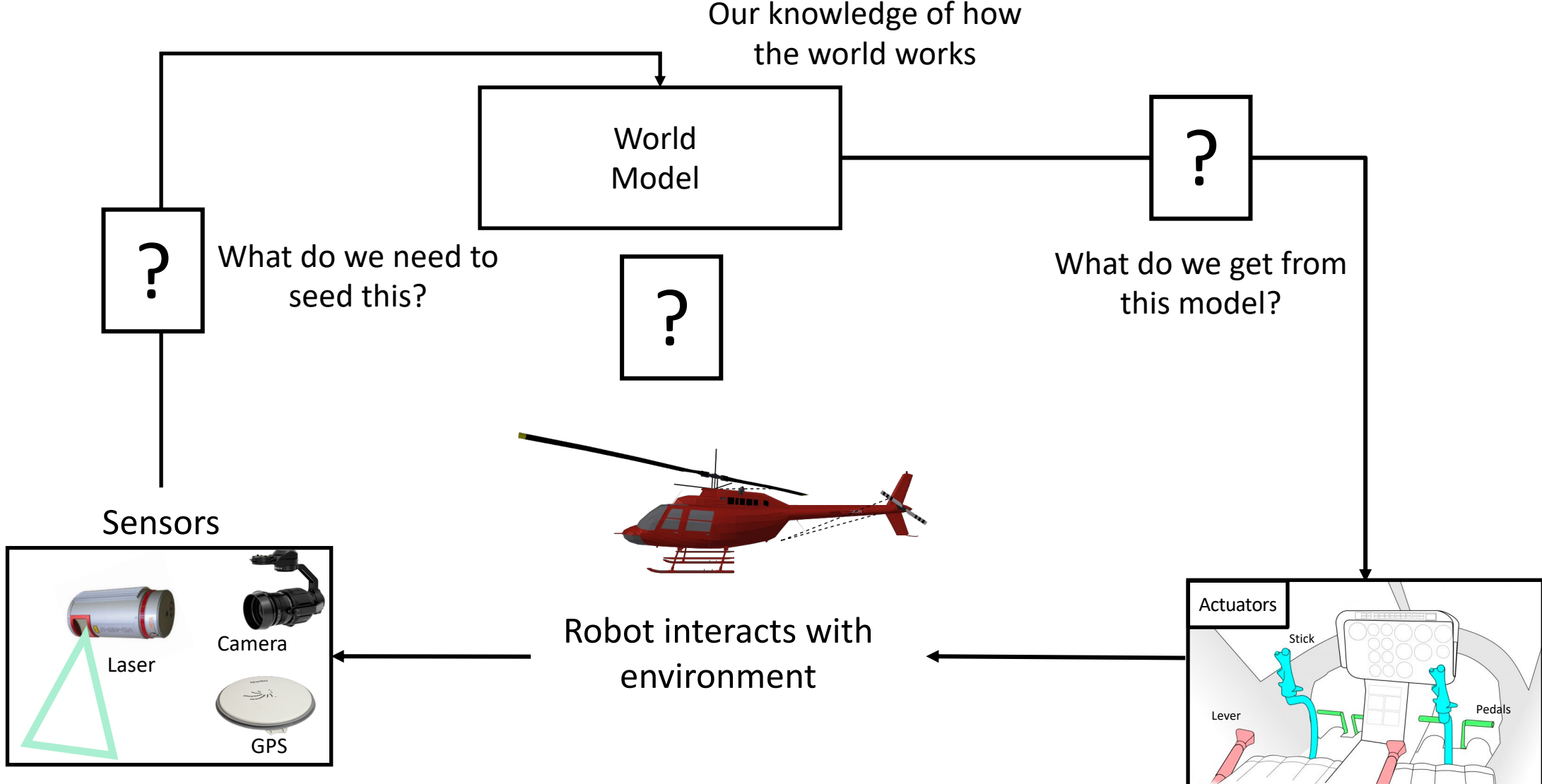


Robot interacts with environment

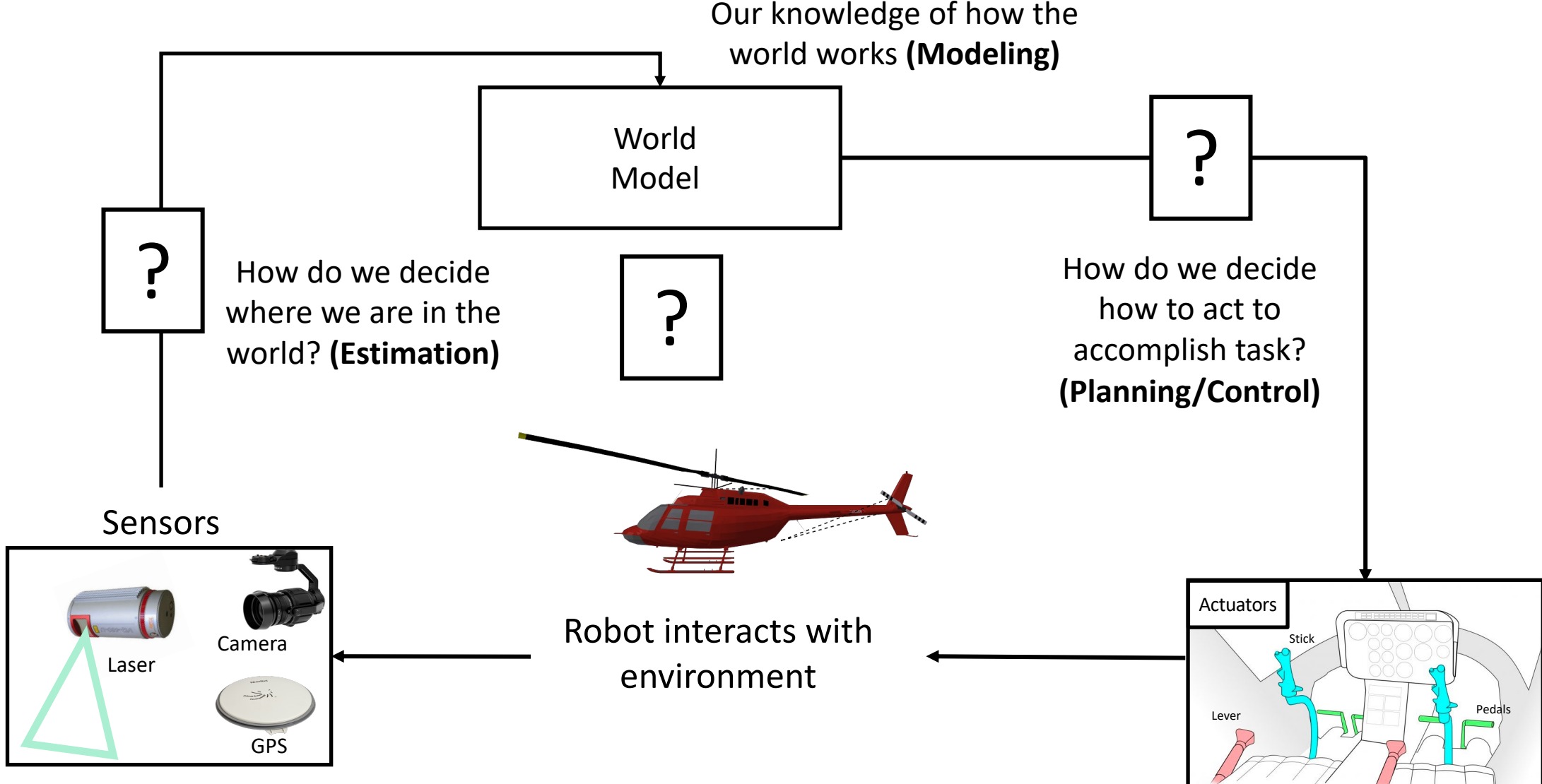
Sensors



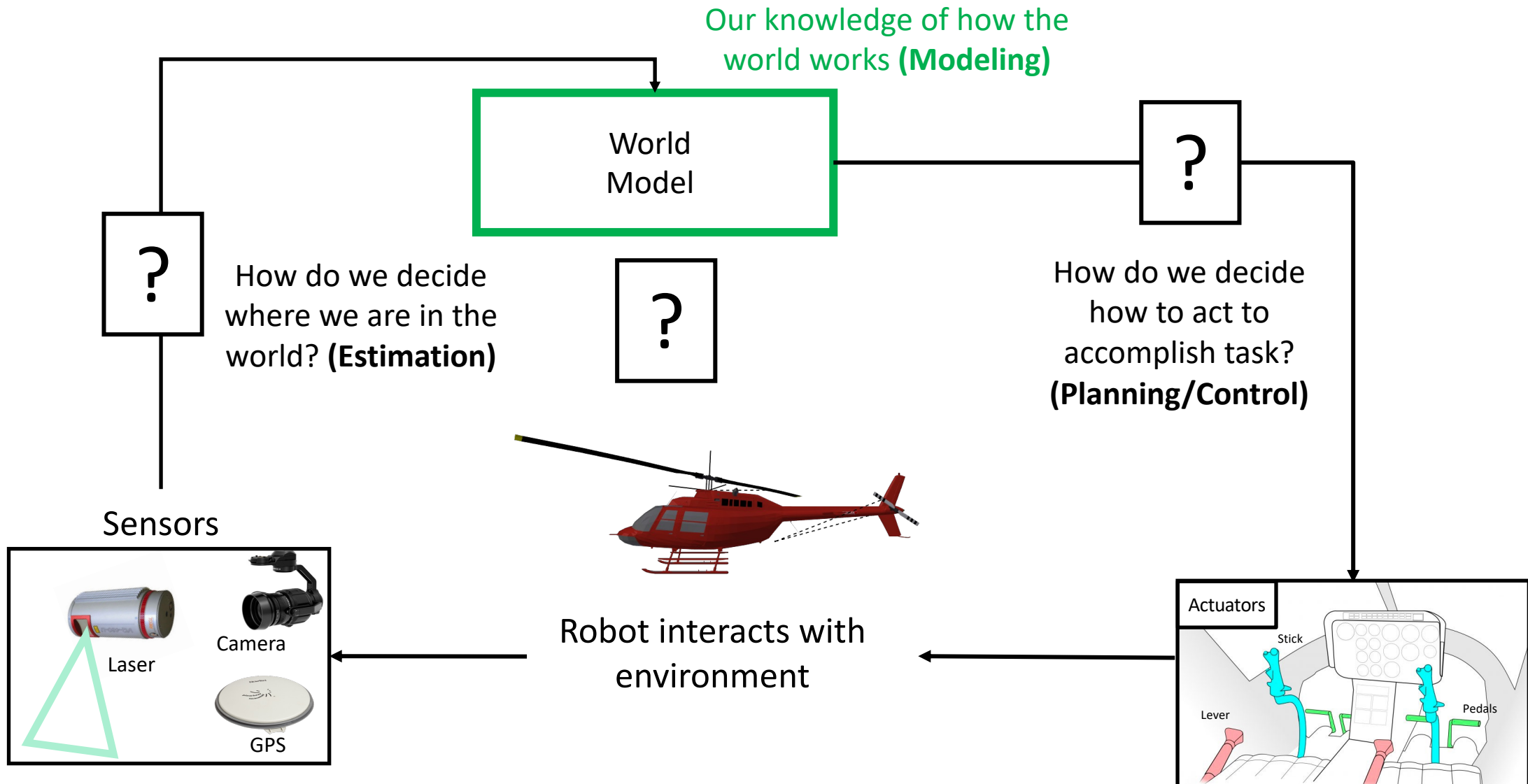
Let's use our knowledge about the world to solve this?



Let's use our knowledge about the world to solve this?

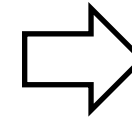
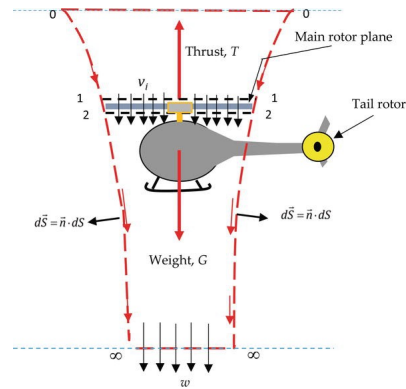
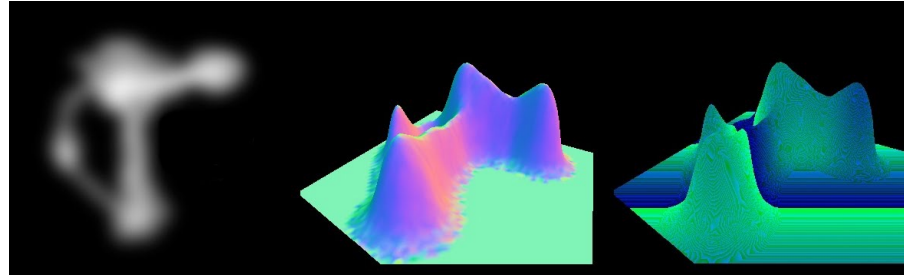
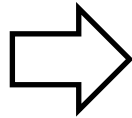


Let's start by thinking about how to model the world



How can we model how the world works?

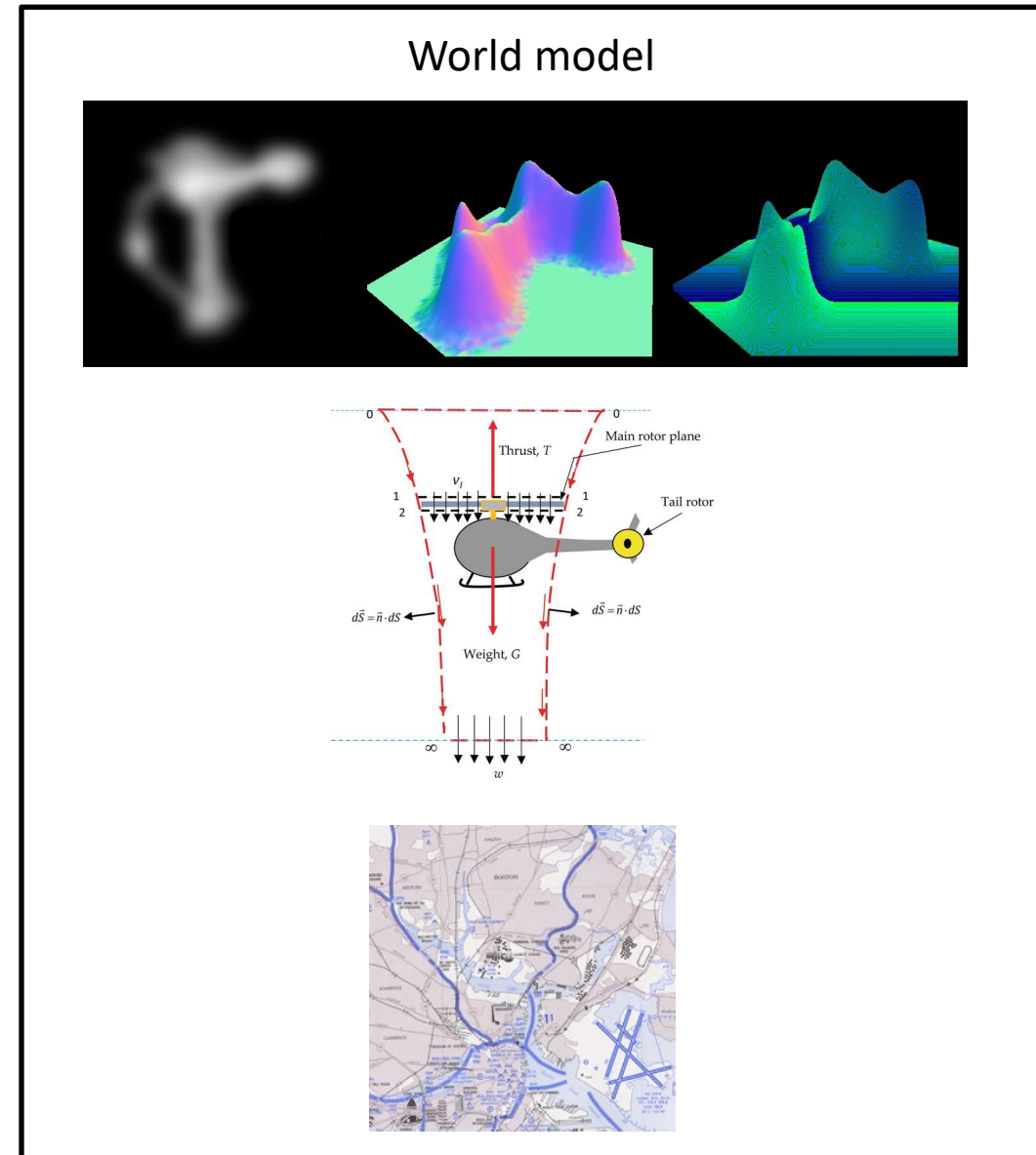
World model



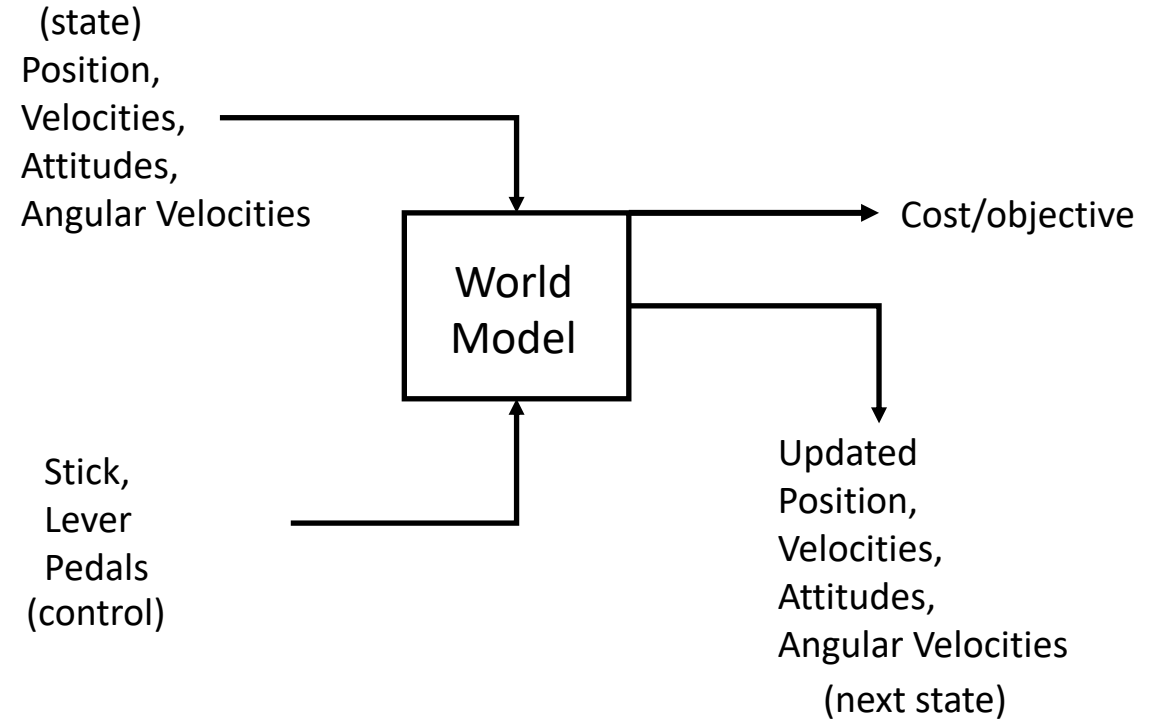
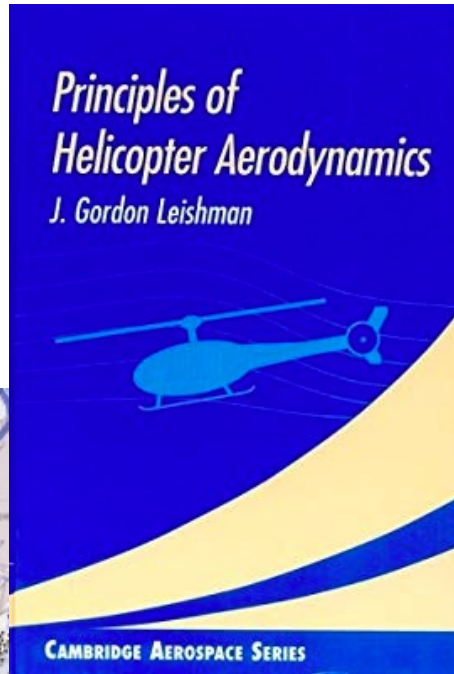
How can we model how the world works?

Model describes our understanding of how the world **works**:

- Variables that are modeled:
 - Position of helicopter
 - Speed of helicopter
 - Payload distribution
 - ...
- How they change:
 - Physics (including wind/temperature)
 - Obstacles/maps of the world
 - Other aircrafts
 - ...
- Cost / objective of the mission
 - No fly zones
 - Flight preferences
 - ...

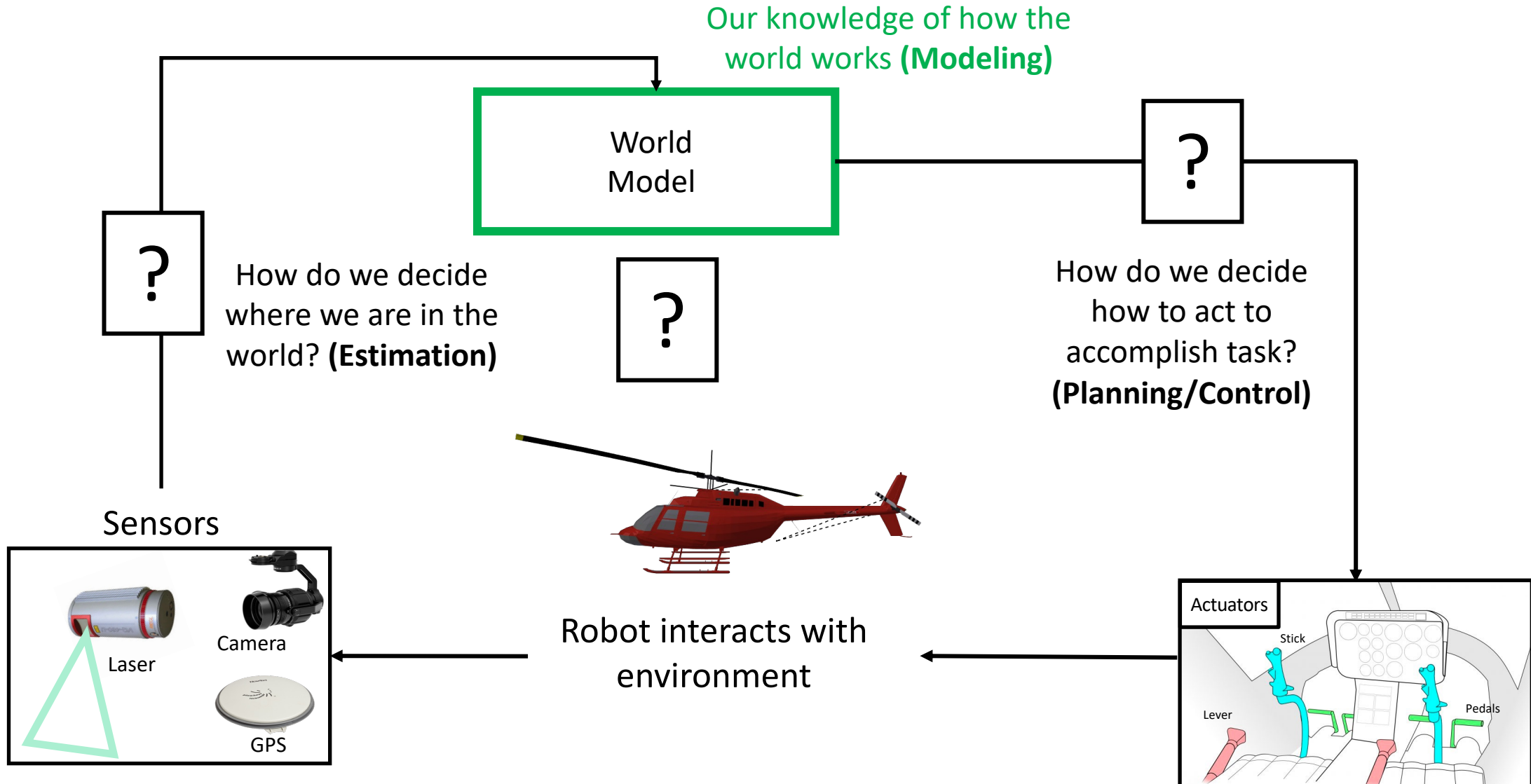


How can we instantiate a helicopter world model?

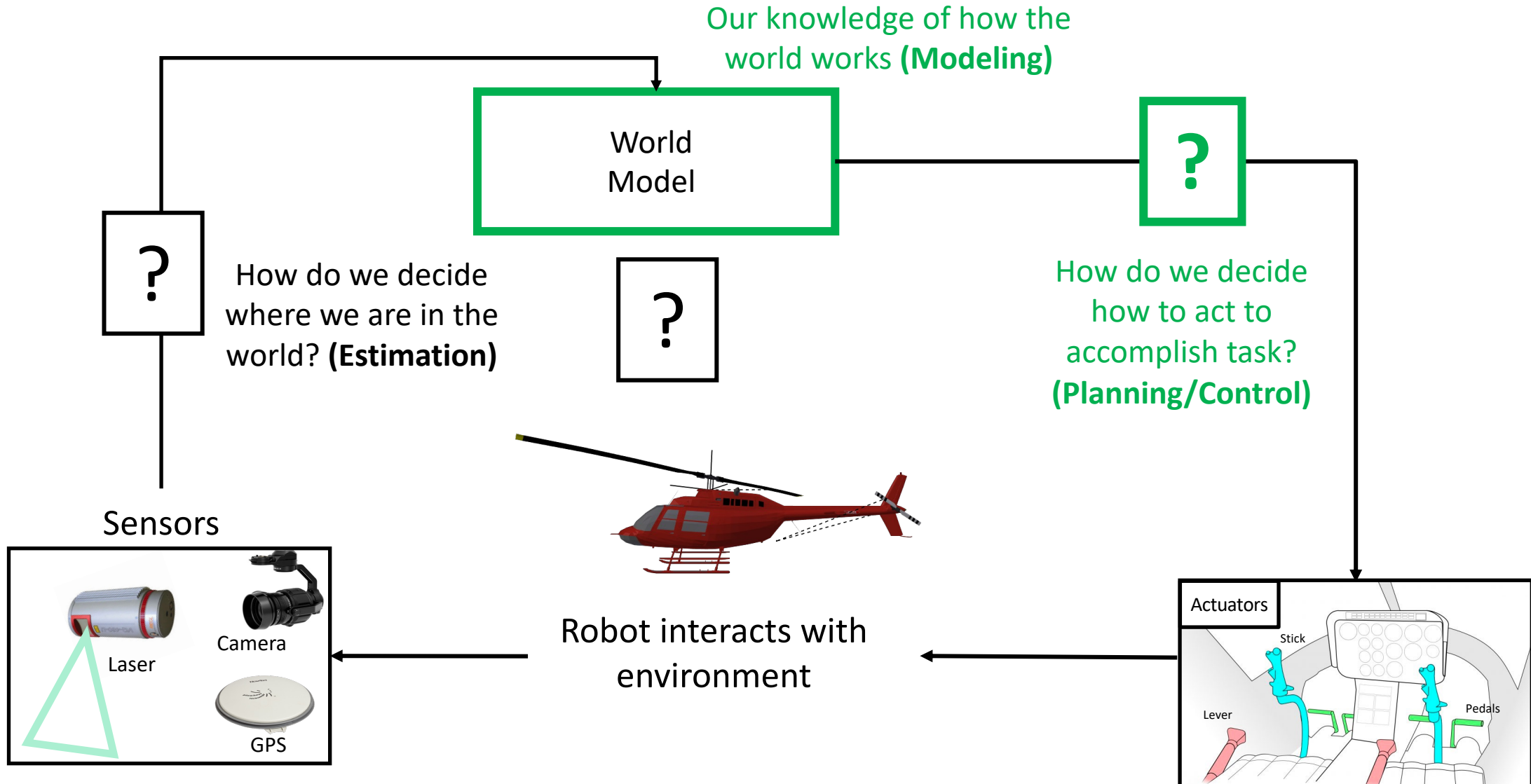


“All models are **wrong**, but some are **useful**” –
George Box

How do we actually use the model?

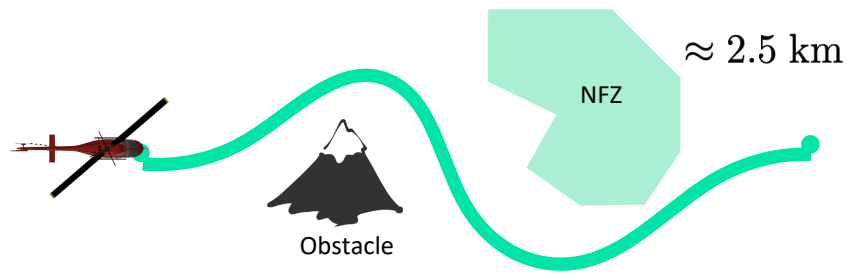


How do we actually use the model?



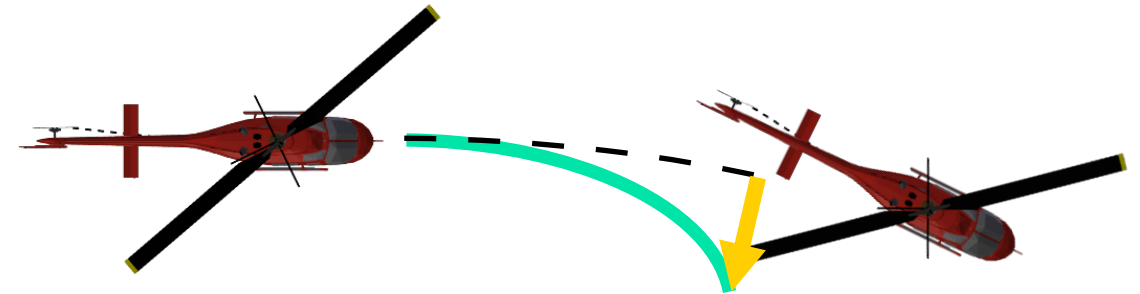
Using world models to decide behavior

Planning (coarse/global)



“Plan” at coarse (10 m) resolution,
follow the global route,
avoid all obstacles, produce
smooth
dynamically feasible paths

Control (fine/local)



“Control” at fine (10cm) resolution
to follow designated path,
rejecting disturbances and model
errors

What is planning?

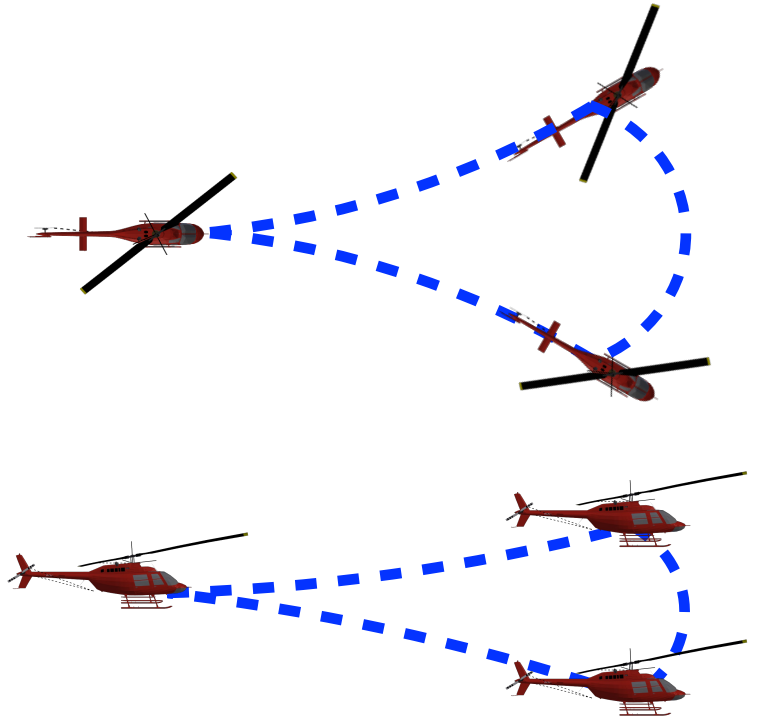
Planning is an optimization problem in which ...

we search over a sequence of actions...

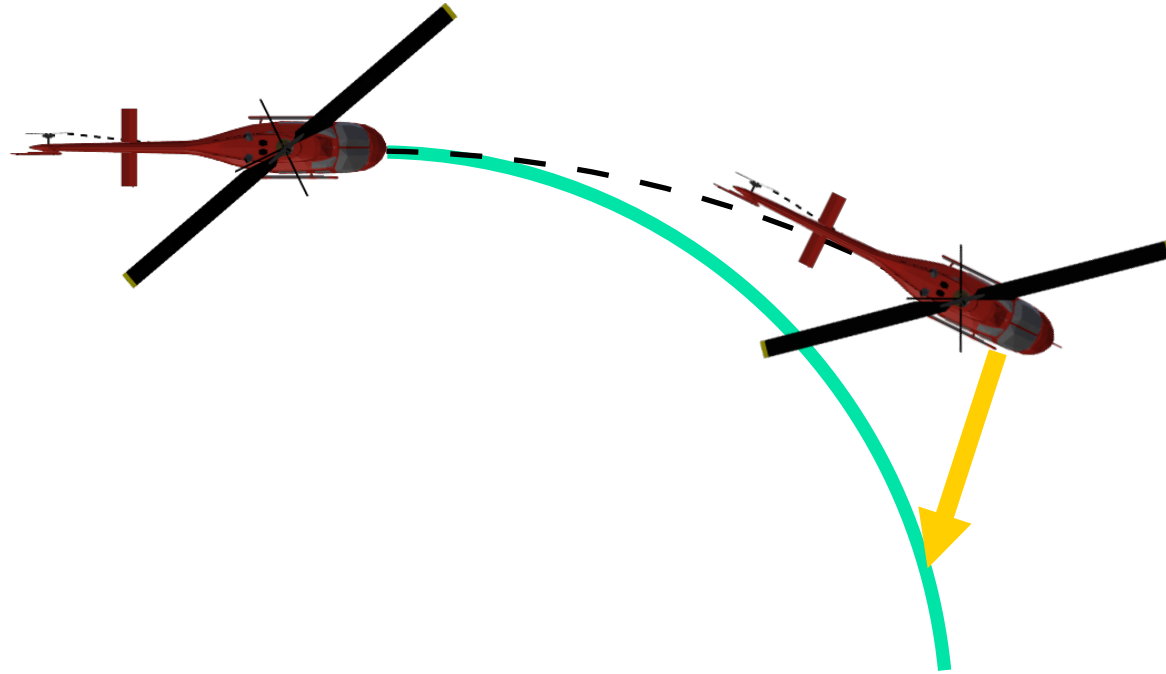
towards minimizing a cost function (e.g., time)...

using a model of the robot to predict where it will go...

while making sure we are not violating constraints (e.g. crash).



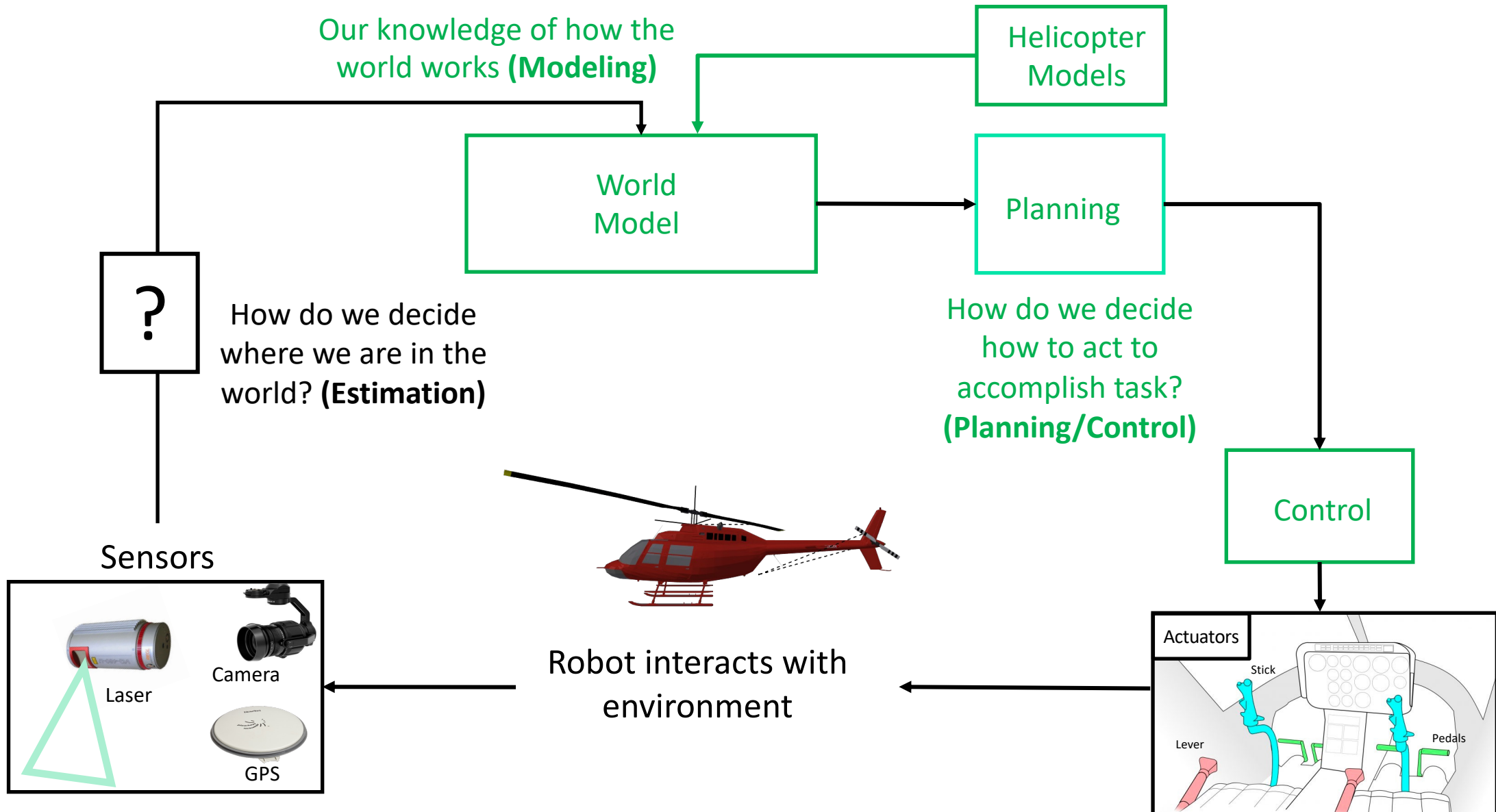
What is control?



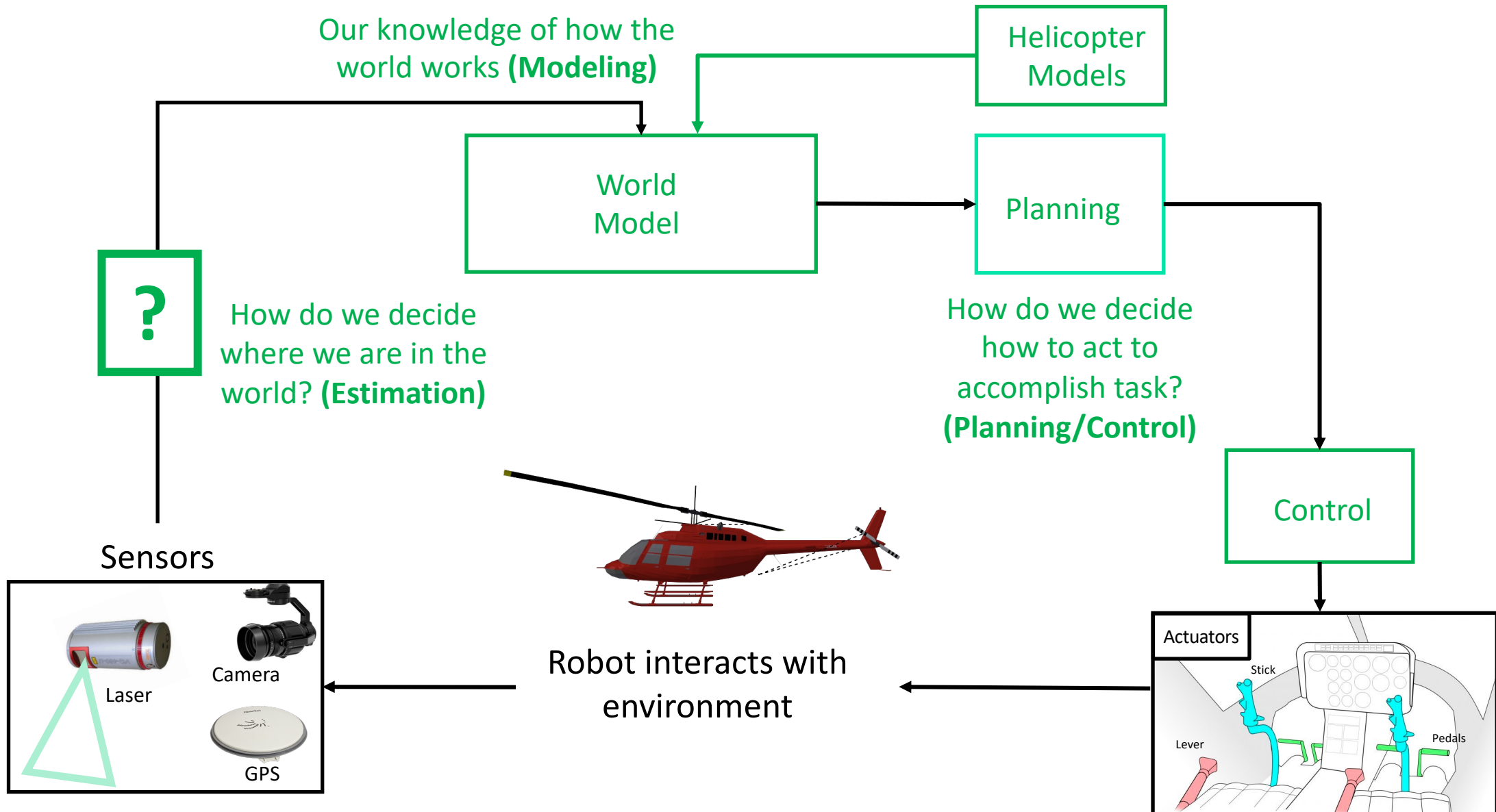
Robot will go “off” the plan for many reasons
(disturbance, model errors, actuation errors, ...)

A **controller** immediately corrects for any tracking error
and gets the robot back on the path

Let's zoom out



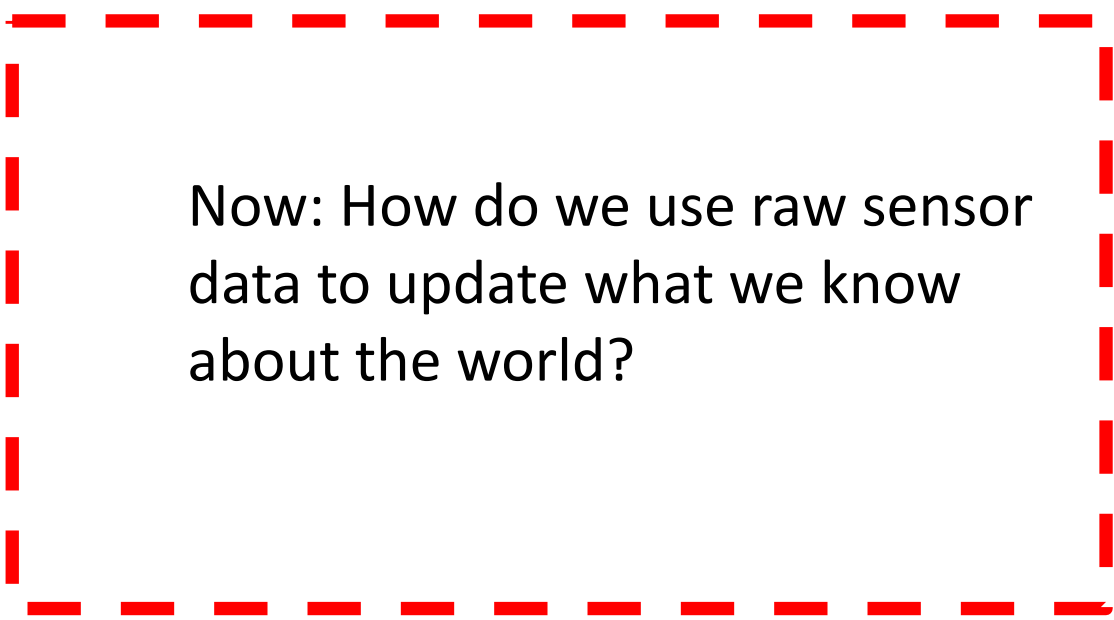
Let's zoom out



How do we estimate “state” for the world model?

So far: Assume we know everything about the world.

What commands should we send to the actuator?



Now: How do we use raw sensor data to update what we know about the world?

How can we use the sensor readings to update the world model?

World models need to be grounded in the “state” of the world

- Where is the robot in the world? What is it's state?
- What are the obstacles in the world?
- What type are the obstacles (radio towers, trees)?
- What are the no-fly-zones?
- Are there other aircrafts?
- What is the wind, temperature, etc?

GPS

Laser

Camera

Radio

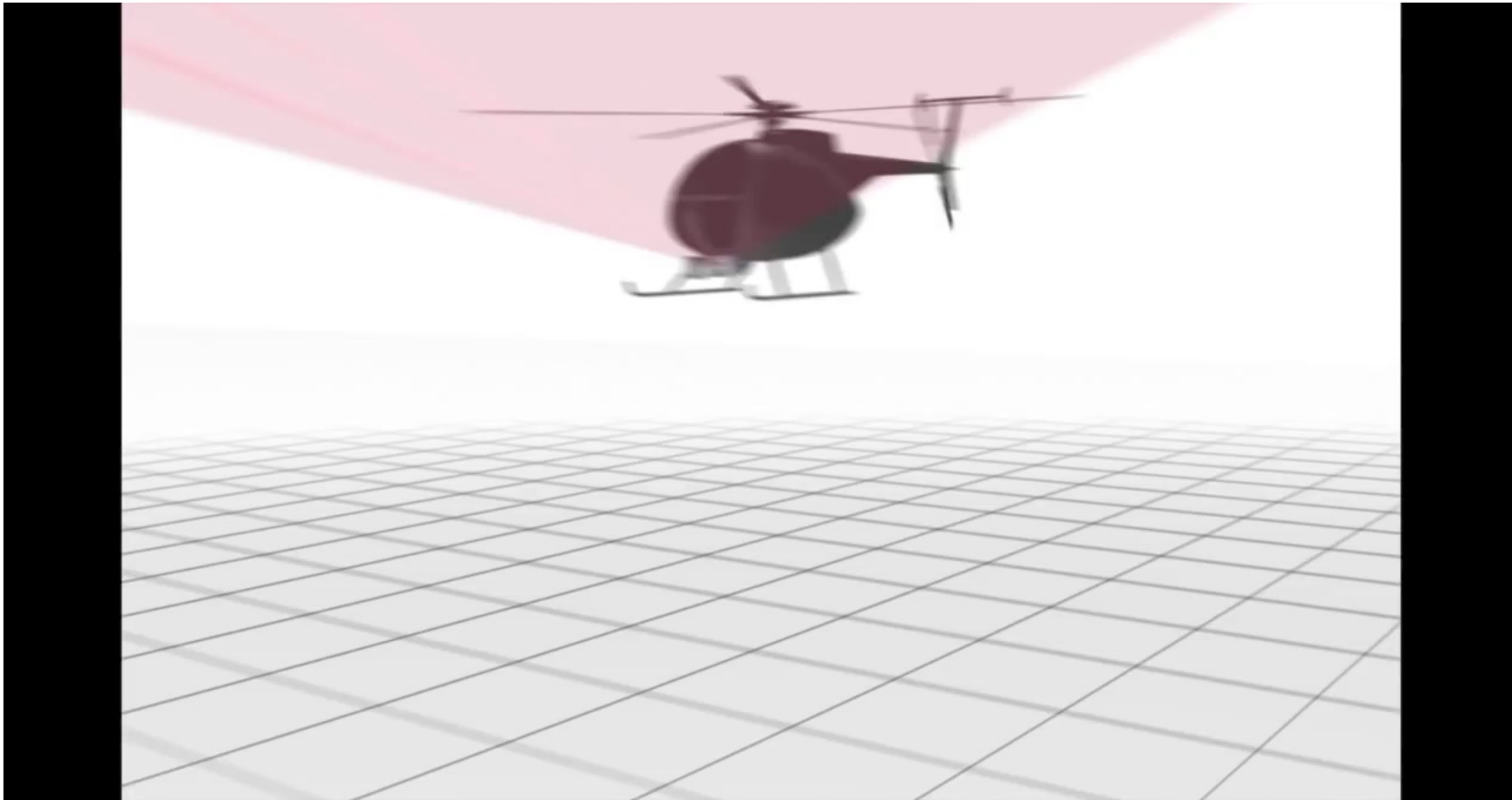
Radio

Pitot tube, barometer,

Why is this difficult?

Why is estimating the state of the world model difficult?

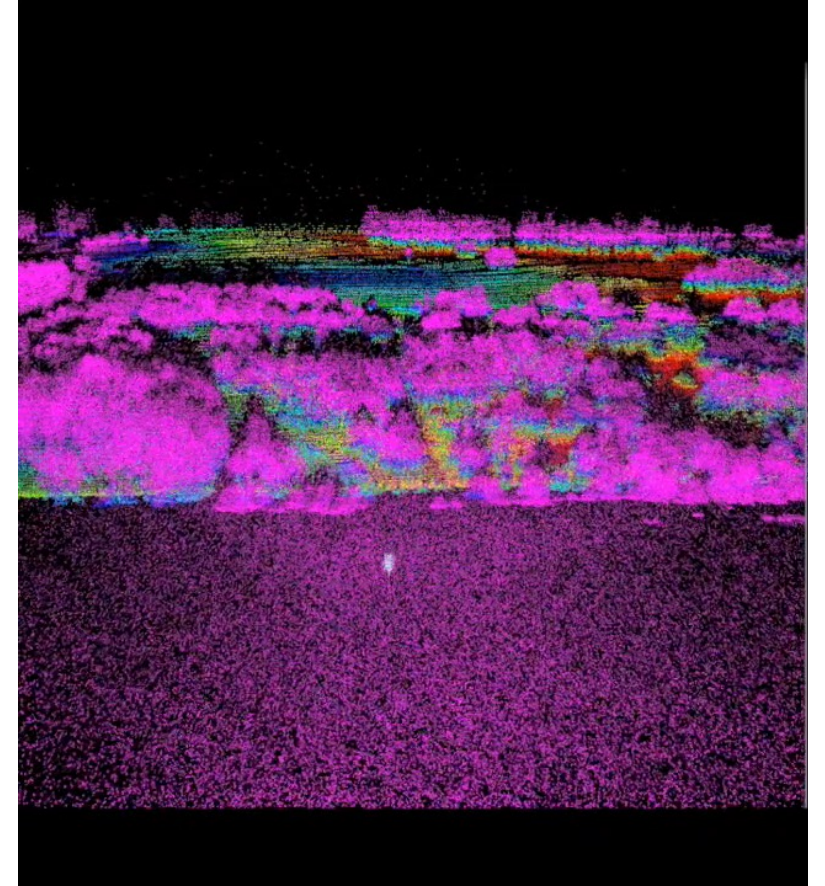
Large amounts of information



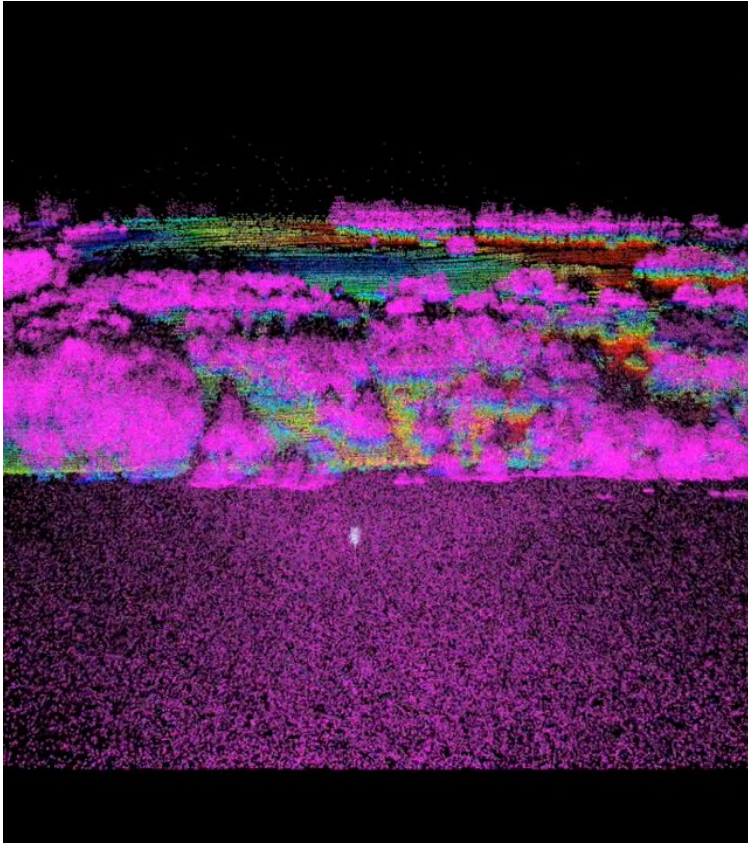
(courtesy Chamberlain et al.)

Why is estimating the state of the world model difficult?

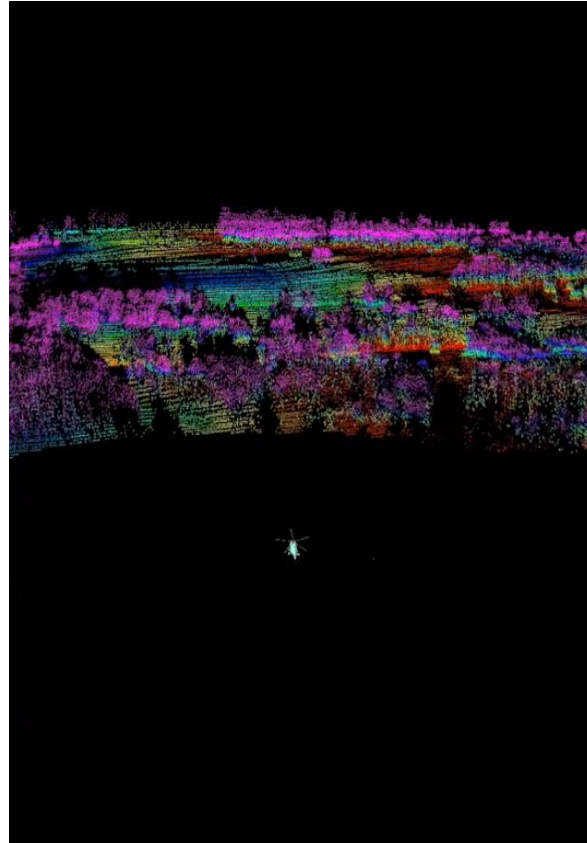
Noisy sensor readings – e.g., flying in a snowstorm



Estimate a “belief” over the state of the system in the world model



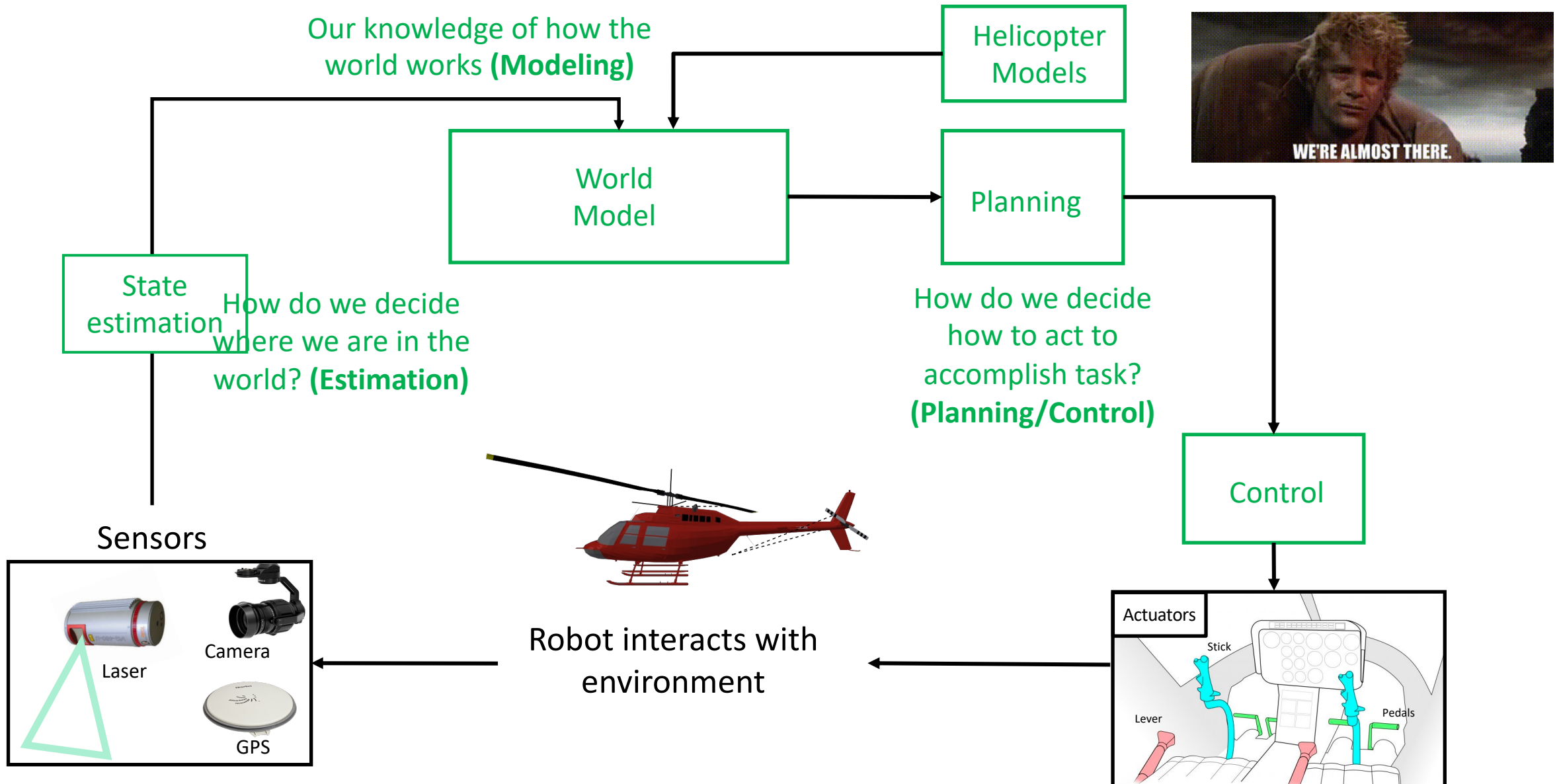
Laser reflected by snow!



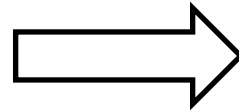
Correctly fused laser data using probabilistic models

- Obtain probabilistic estimates of state
 $P(\text{world model}|\text{data})$
- Use the probabilistic estimates of state for robust “planning” under uncertainty

Yay! We have most of the elements of a system



But we have no helicopters,
only cars! Do the same
lessons apply?



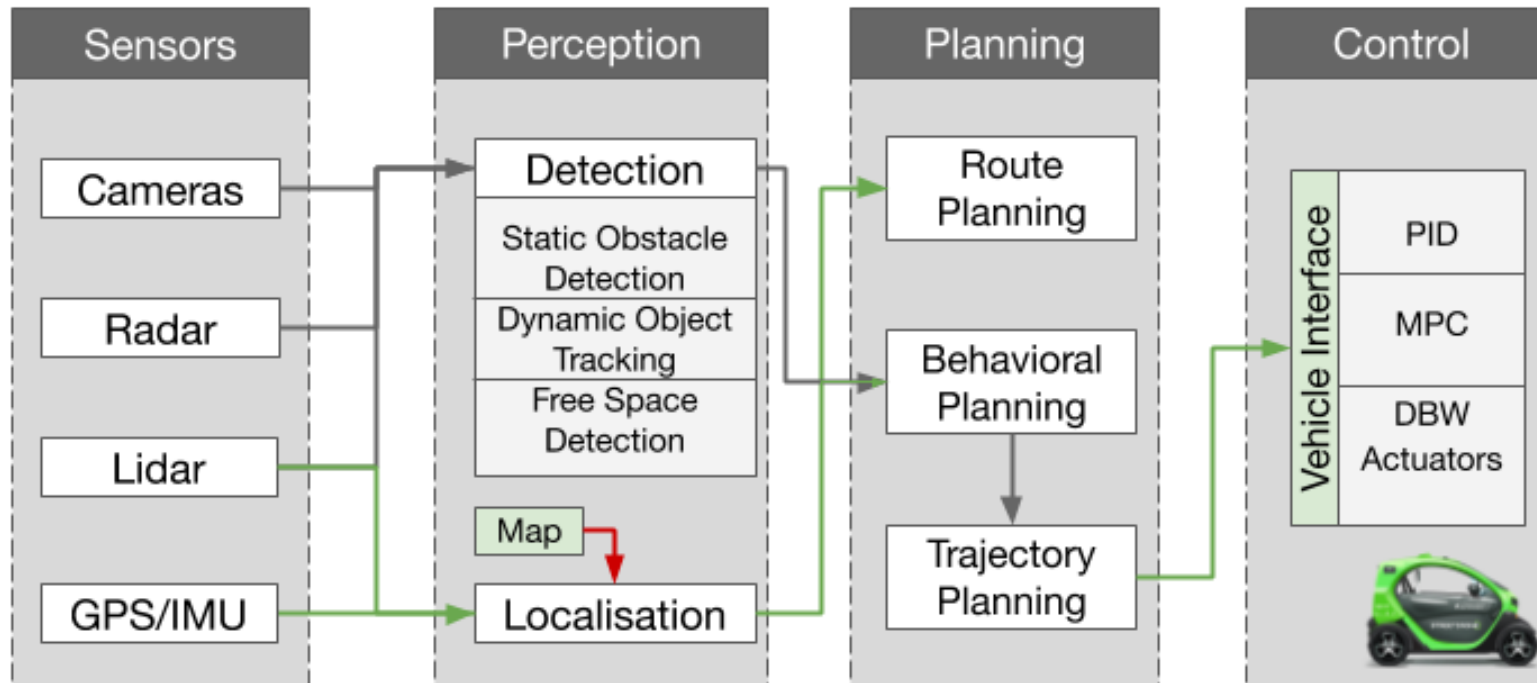
We'll figure that out this quarter!



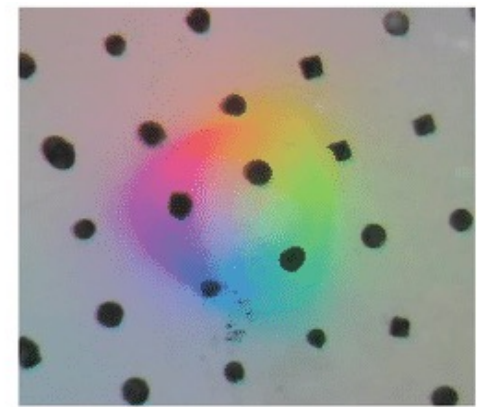
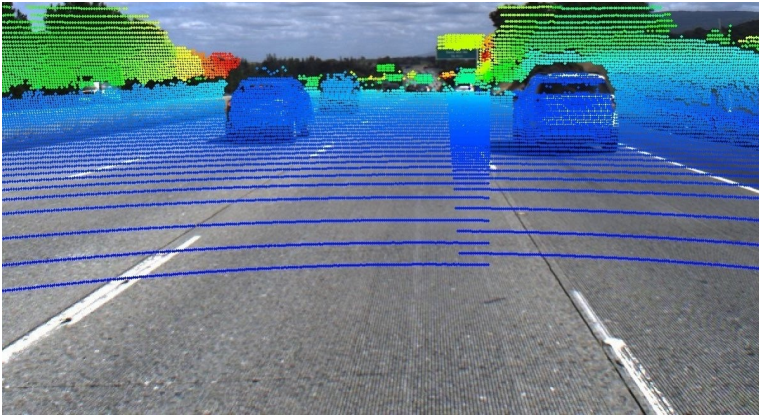
Sense-Plan-Act Framework

Robotics has three primary subpieces:

1. Sensing → from measurements
2. Planning → from models
3. Acting → control in the world



Sensing: Why is it nontrivial?



- Sensors have overwhelming amounts of information
- Partially observed
- Noisy and prone to drift

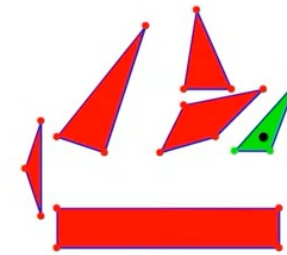
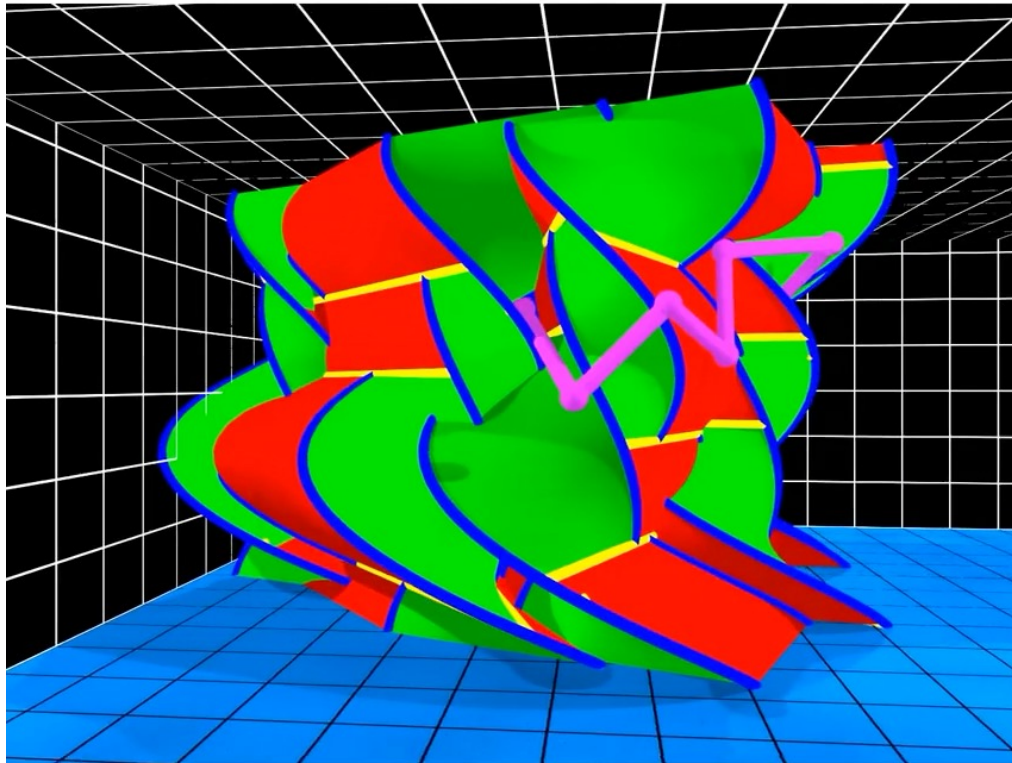
Acting: Why is it nontrivial?

- Robot systems in the real world are subject to significant perturbations/noise → need to be stable in the face of these perturbations



Planning: Why is it nontrivial?

- Searching/Optimization through a complex non-convex space
- Combination of discrete/continuous optimization

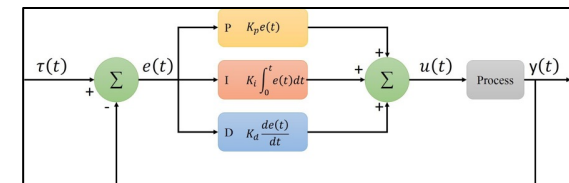
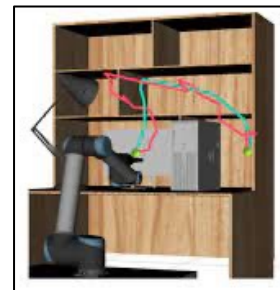
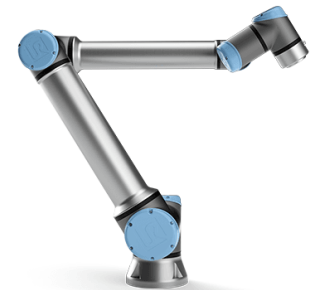
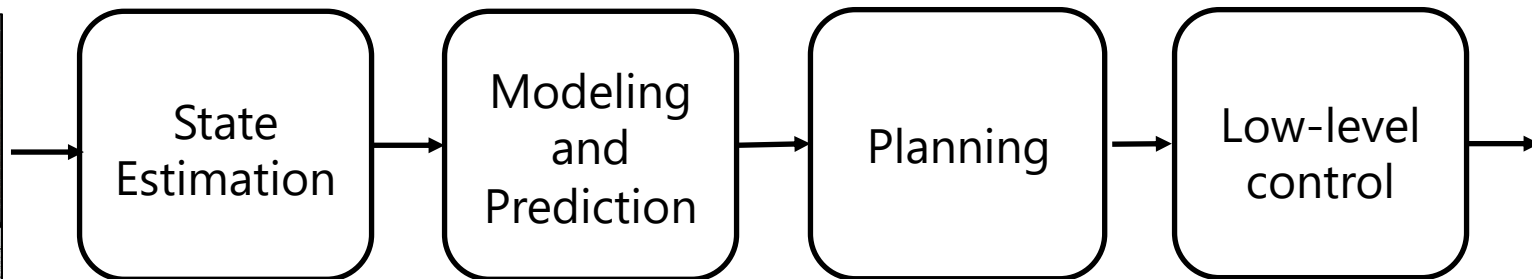
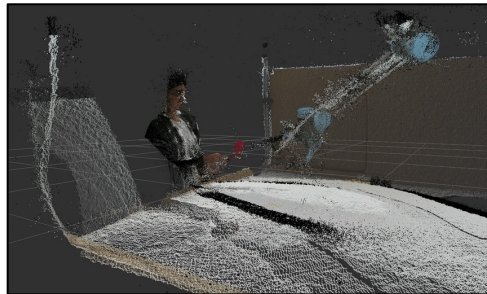
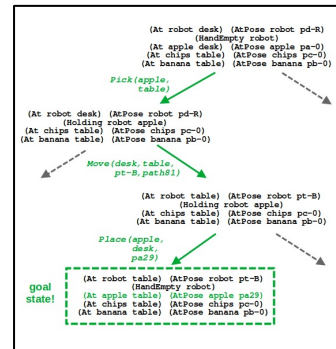


Overview

Robotics: Integrated System Research

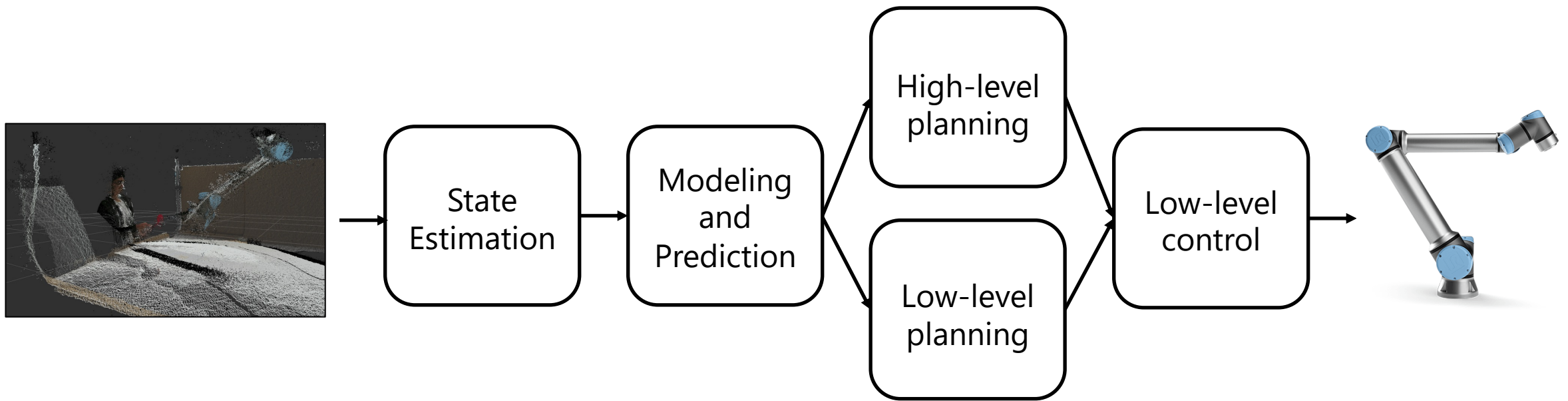
Focus on addressing all problems at once

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} = \boldsymbol{\tau}_g(\mathbf{q}) + \mathbf{B}\mathbf{u},$$

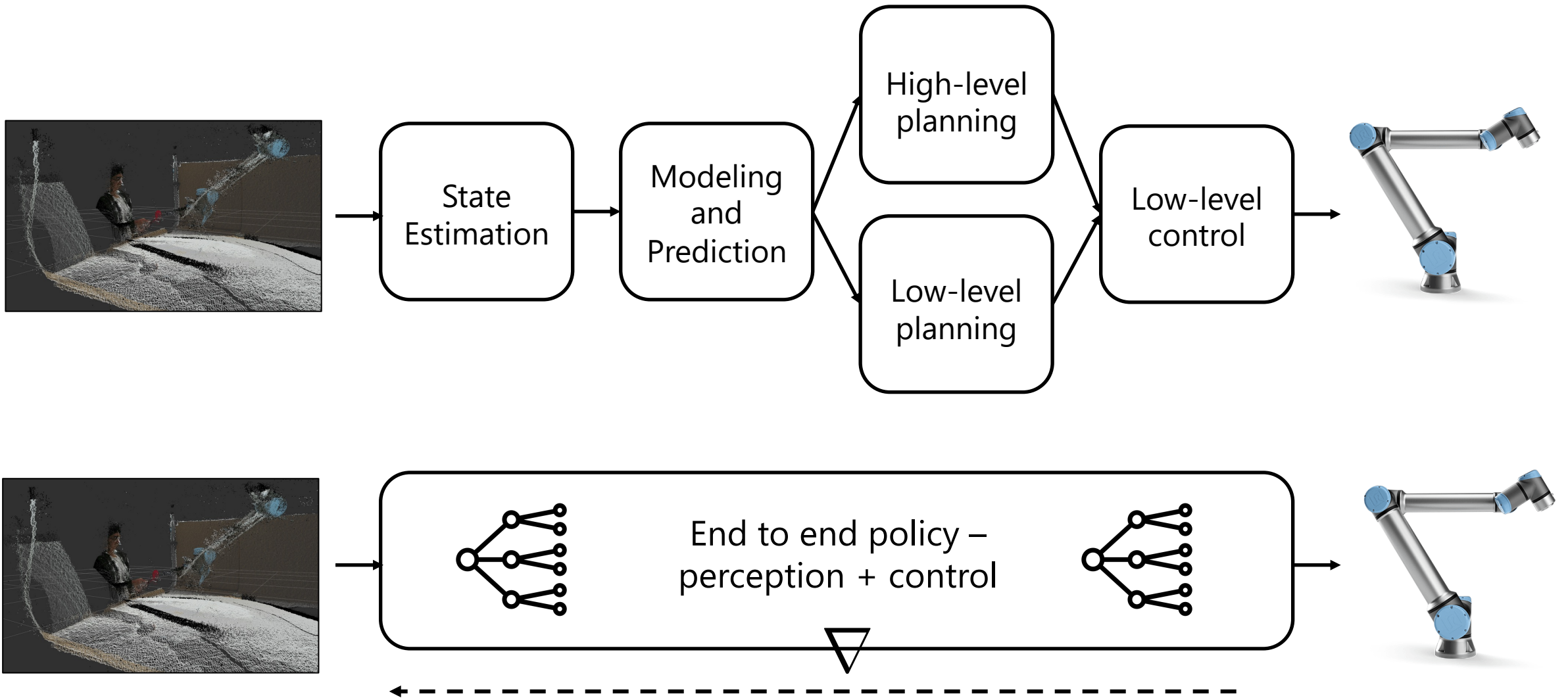


How do we tractably
solve the task?: The
model-free way

What does a typical model based look like?



End-to-End Learning Based Control for Robots



Why might we want/not want to do this?

Modules compensate
for each other

Avoids hand-designing
and supervising interfaces

Often more
performant/less biased

Lack of Interpretability

Lack of Reusability

Often data inefficient

Class Outline

State Estimation

Robotic System Design

Filtering

Localization

SLAM

Control

Feedback Control

PID Control

MPC

LQR

Planning

Search

Heuristic Search

Motion Planning

Lazy Search

Learning

Imitation Learning

Policy Gradient

Actor-Critic

Model-Based RL