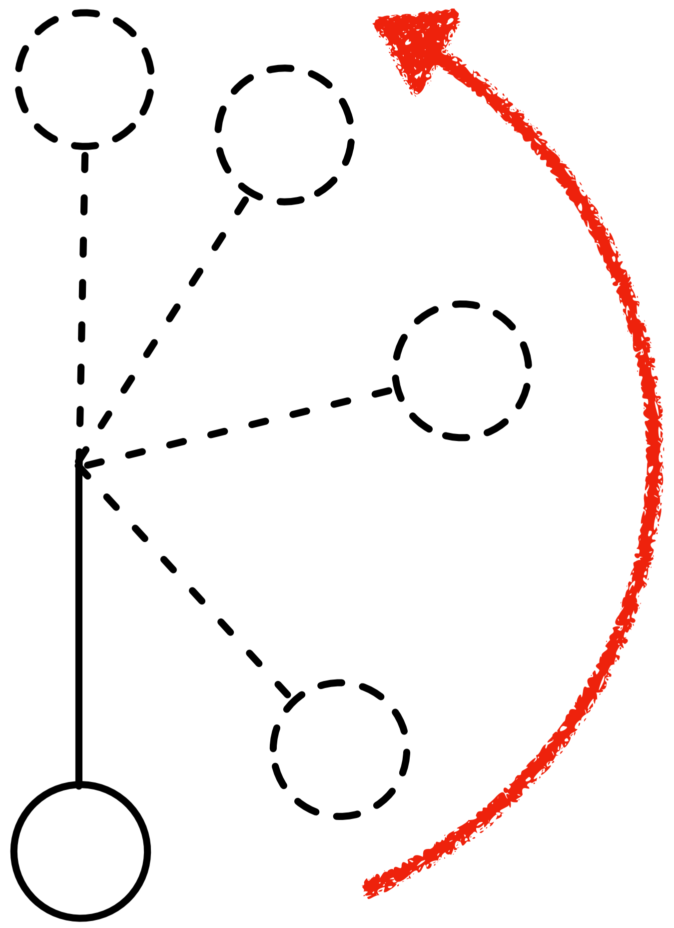
Model Predictive Control and the Unreasonable Effectiveness of Replanning

Sanjiban Choudhury

Goal: Design a complete planning/control architecture for real robot systems?

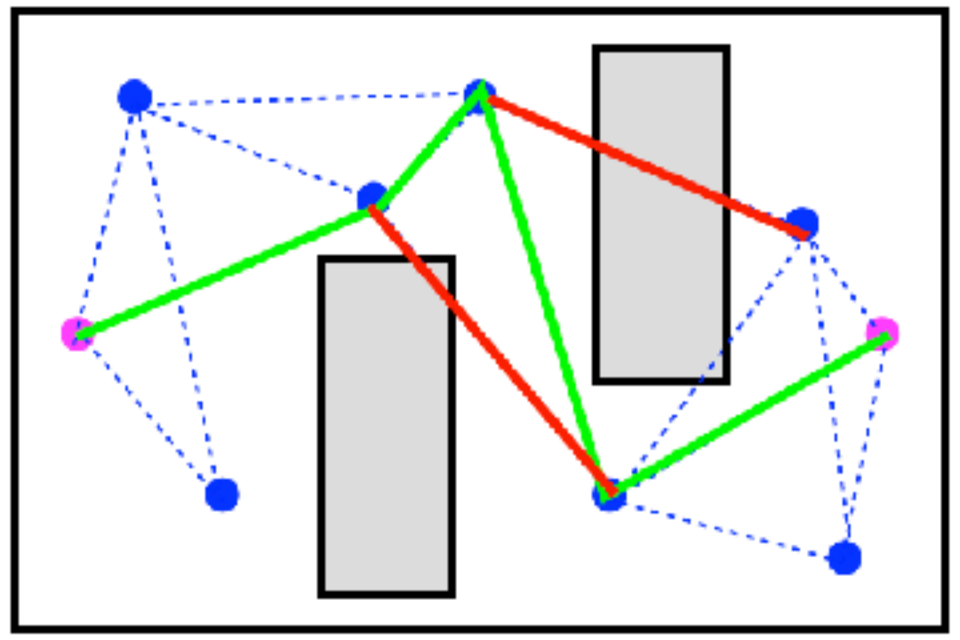
Landscape of Planning / Control Algorithms

Low-level control



LQR

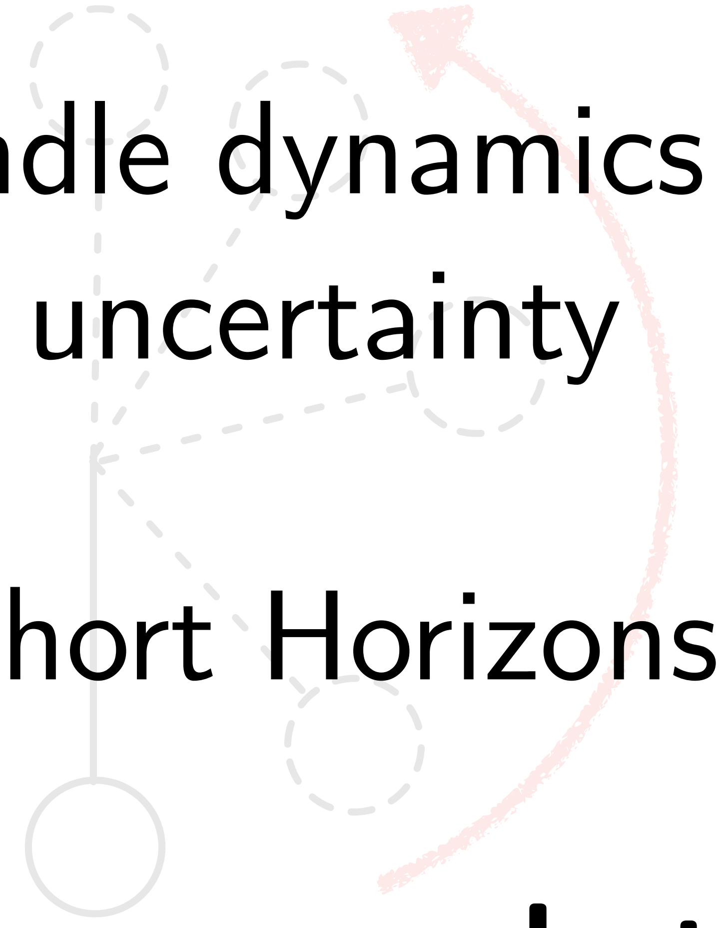
High-level path planning



LazySP / Halton Sampling

Landscape of Planning / Control Algorithms

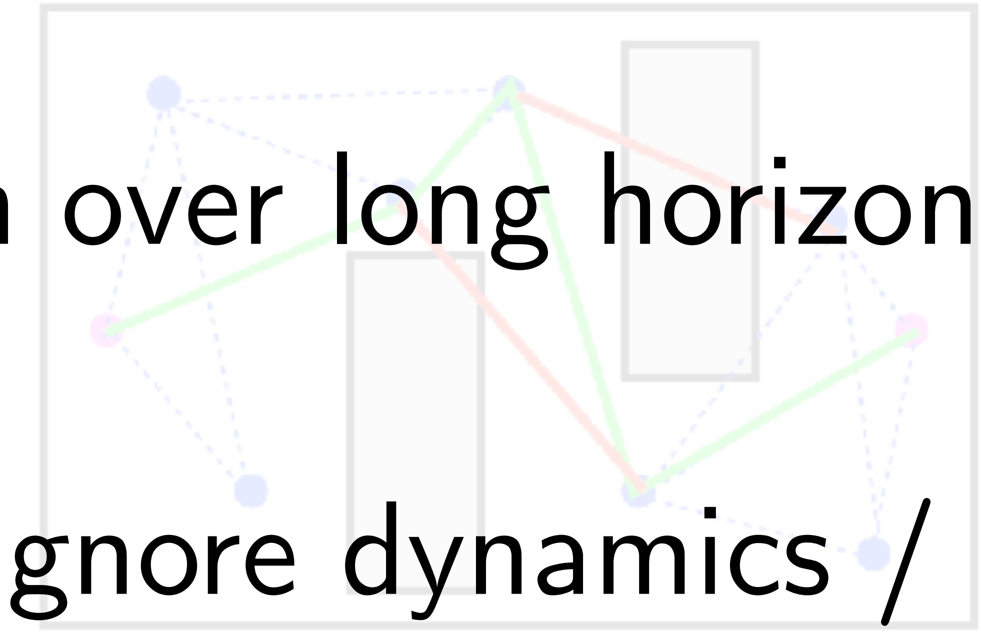
Low-level control



- ✓ Handle dynamics + uncertainty
- ✗ Short Horizons
- ✗ **Linear, quadratic**

LQR

High-level path planning

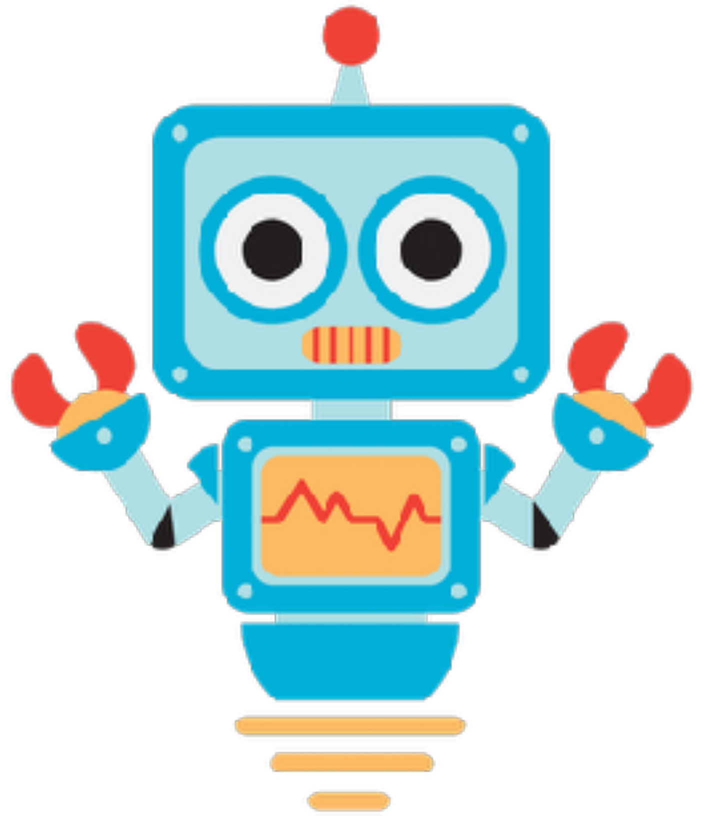


- ✓ Plan over long horizons
- ✗ Ignore dynamics / uncertainty

LazySP / Halton Sampling

LQR is cute...
But what if my
robot is not linear?

Goal: Solve a *general* continuous time MDP



Nonlinear!

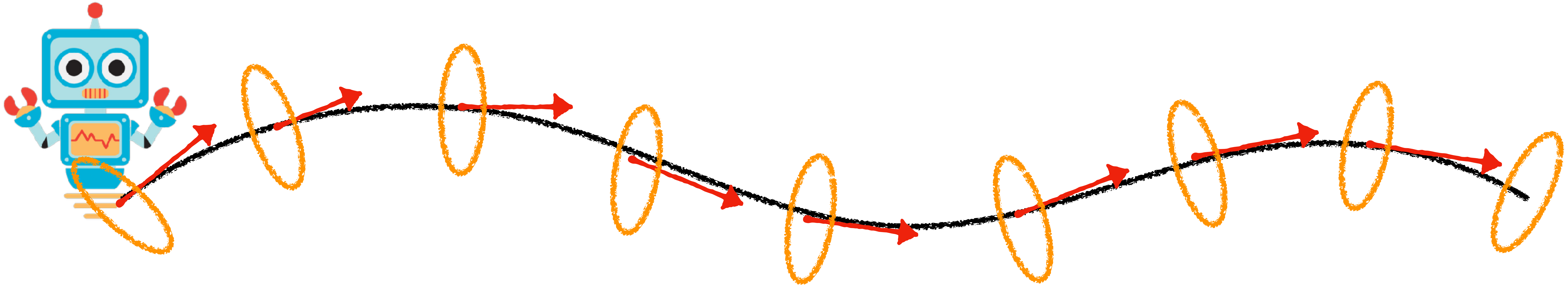
$$\min_{x_{0:T-1}, u_{0:T-1}} \sum_{t=0}^{T-1} c(x_t, u_t)$$

$$x_{t+1} = f(x_t, u_t)$$

Nonlinear!

Iterative LQR (ILQR) to the rescue!

Three simple steps!



Step 1: Forward pass - roll out current guess $u(t)$

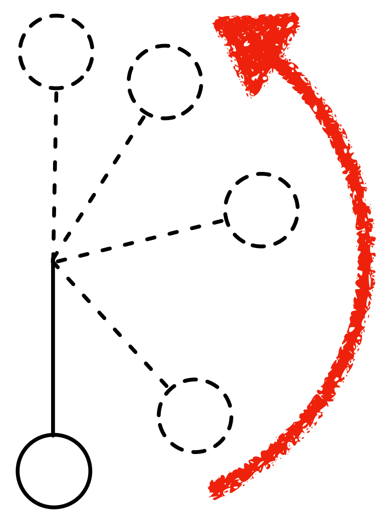
Step 2: Linearize dynamics, quadricize cost around roll out

Step 3: Backwards pass - compute LQR gains K_t at each time

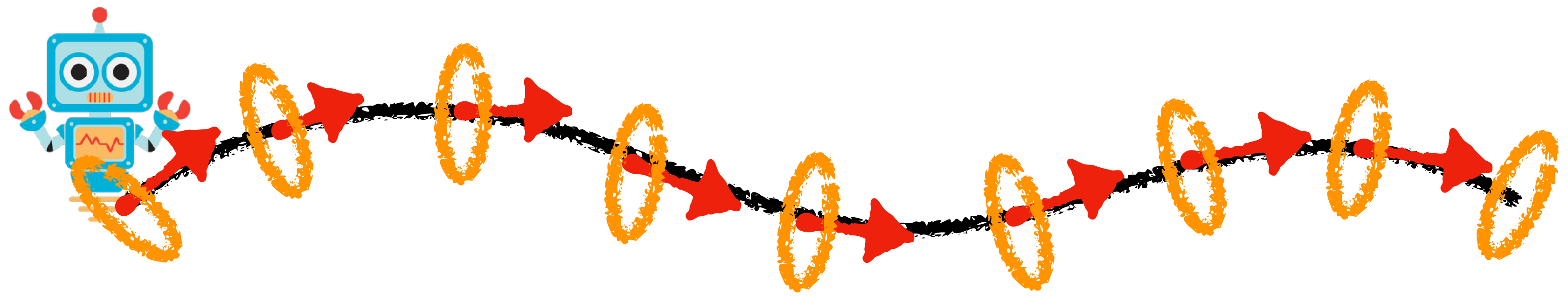
Repeat!

Landscape of Planning / Control Algorithms

Low-level control

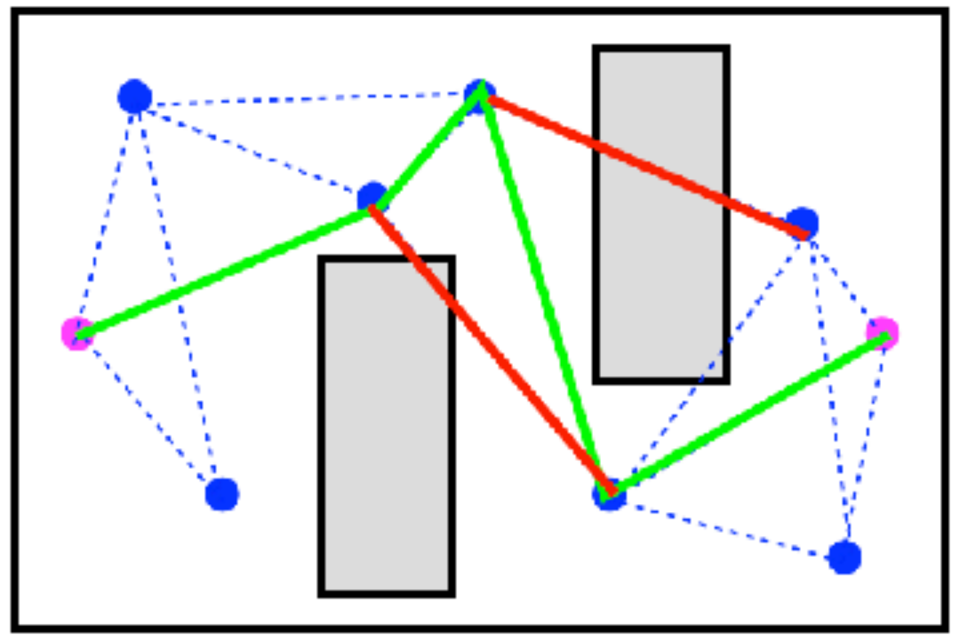


LQR



iLQR

High-level path planning



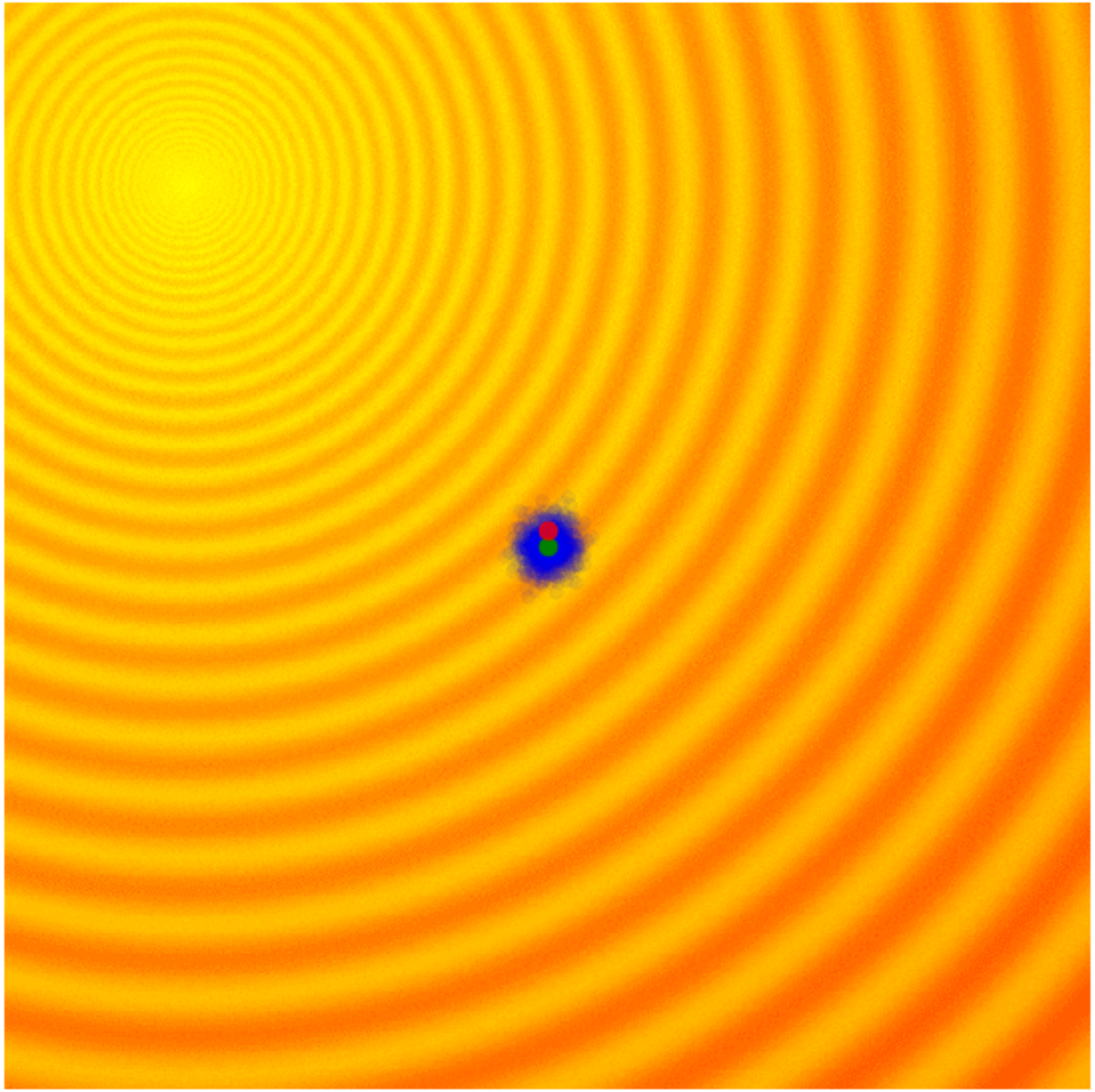
LazySP / Halton Sampling

iLQR seems
hard to implement

...

is there a simple
brute force
approach?

Cross Entropy Search



Credit: https://blog.otoro.net/2017/10/29/visual-evolution-strategies/

Let’s formalize!

The Cross Entropy Algorithm

D_θ

INIT D_θ

The Cross Entropy Algorithm

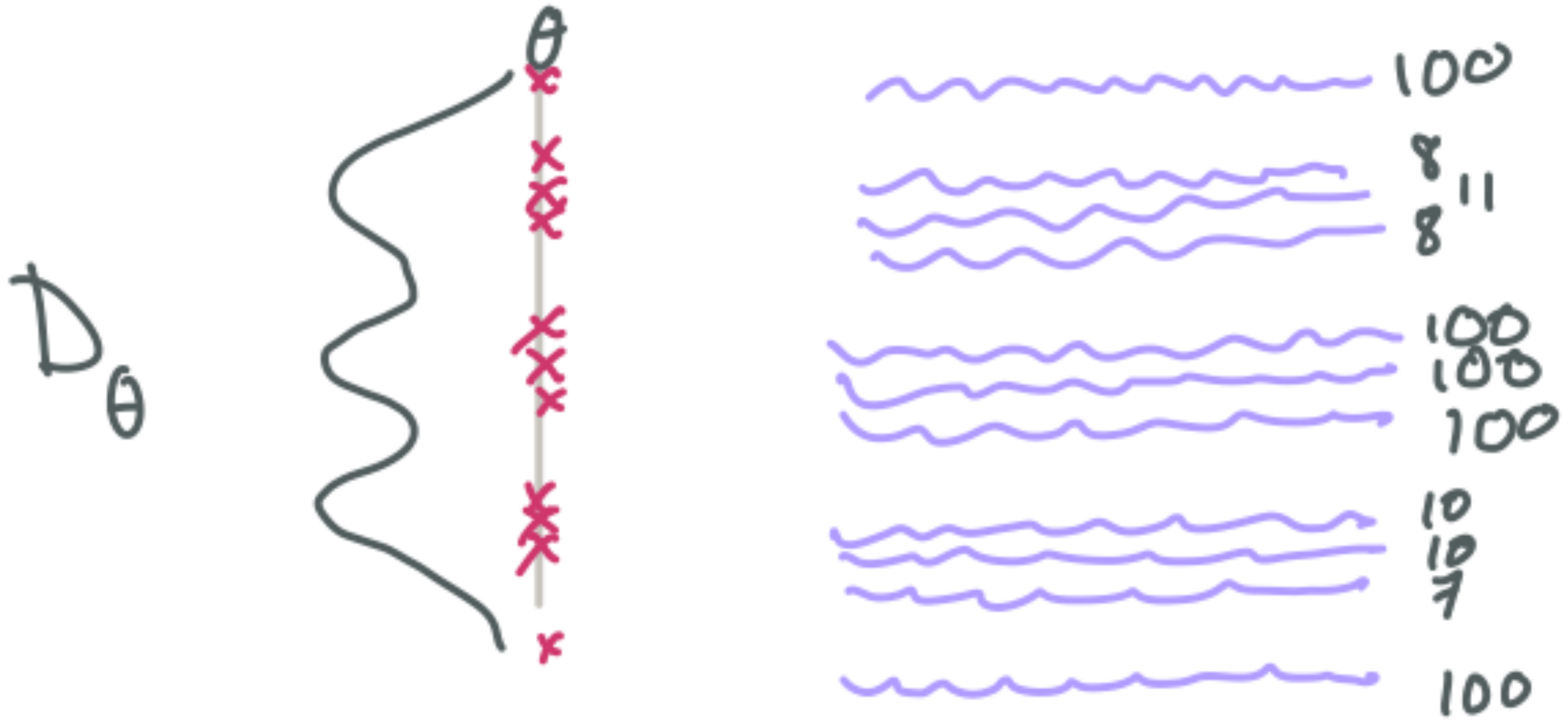
INIT D_θ

SAMPLE k TIMES
to get $\{\theta_i\}_{i=1}^{k}$

The Cross Entropy Algorithm

INIT D_θ

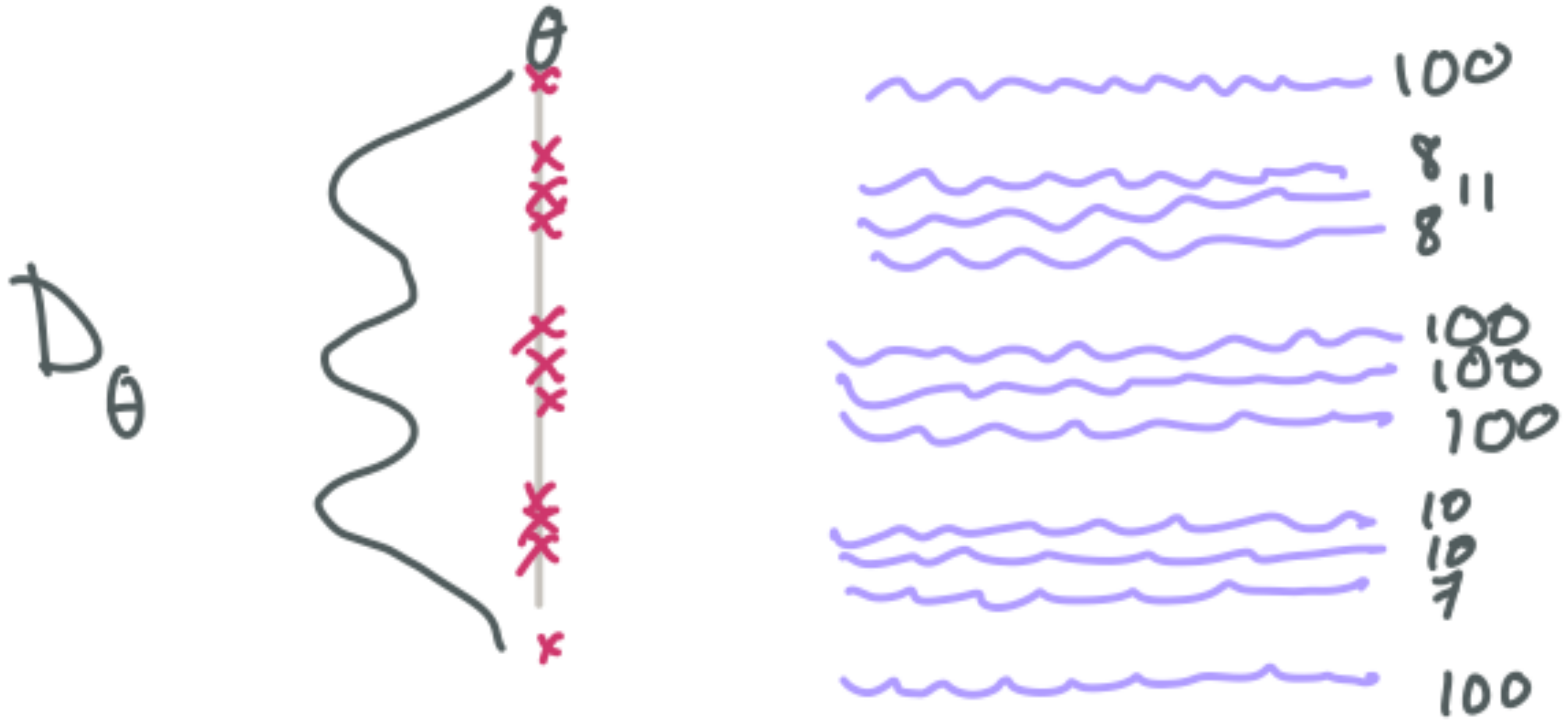
SAMPLE k TIMES
to get $\{\theta_i\}_{i=1}^{k}$



EVALUATE EACH θ_i:

- EXECUTE POLICY MULTIPLE TIMES

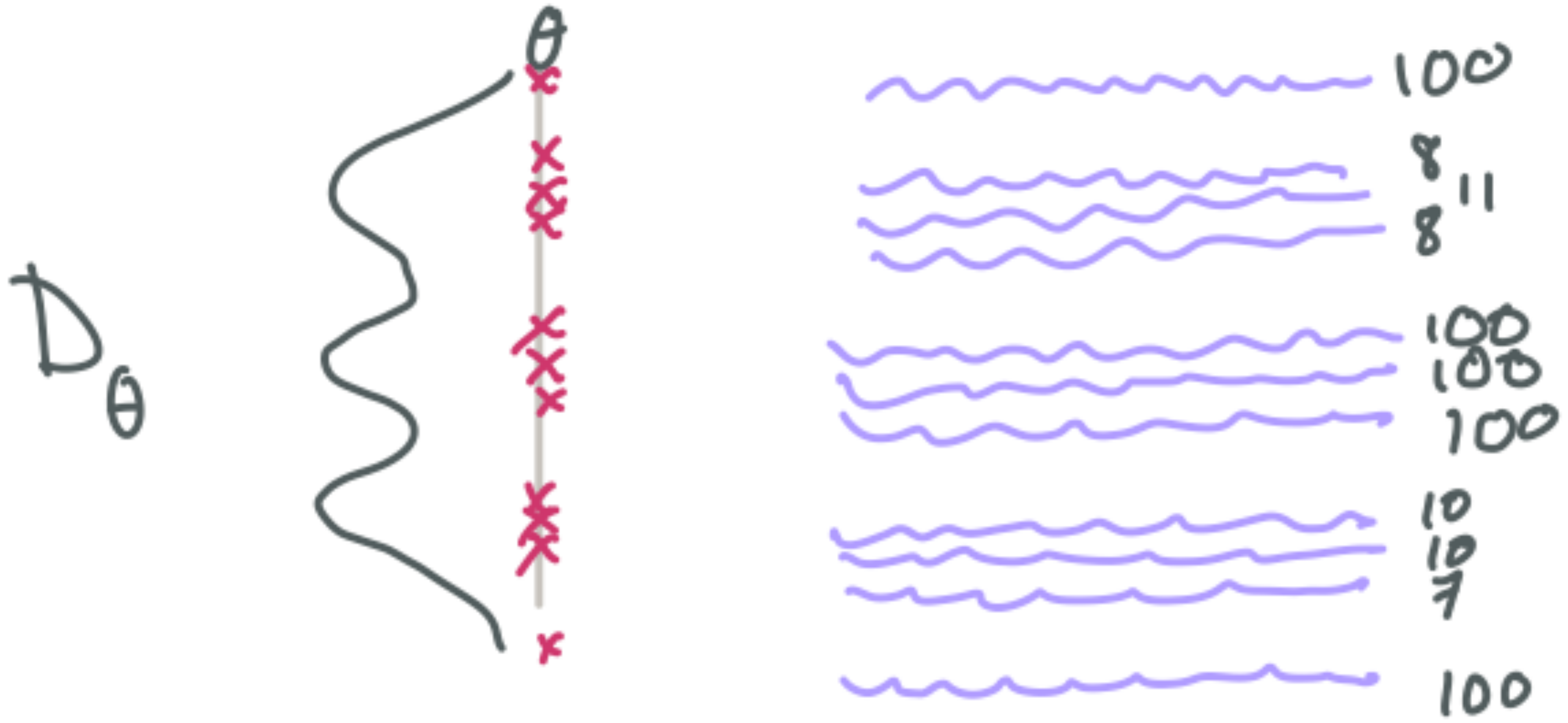
The Cross Entropy Algorithm



EVALUATE EACH θ_i:

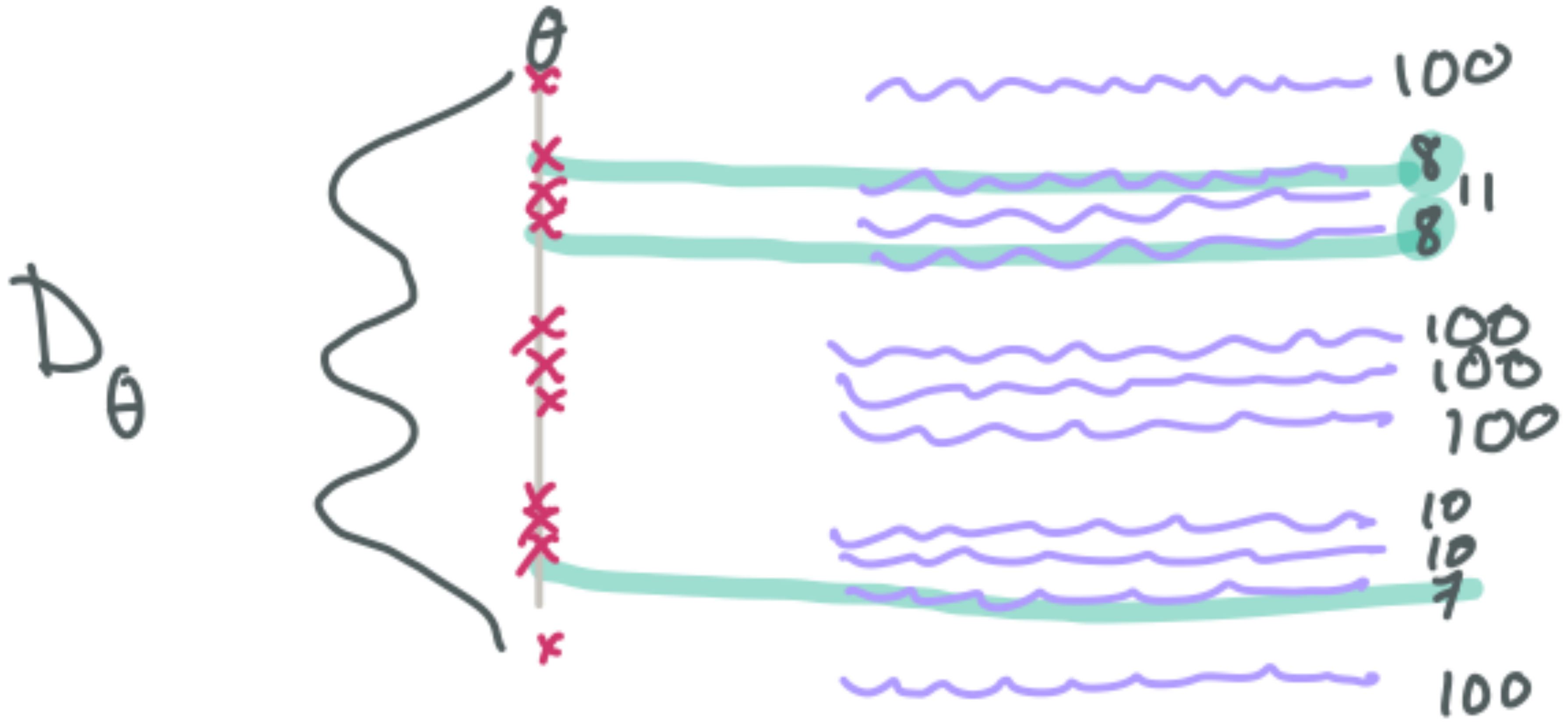
- EXECUTE POLICY MULTIPLE TIMES

The Cross Entropy Algorithm



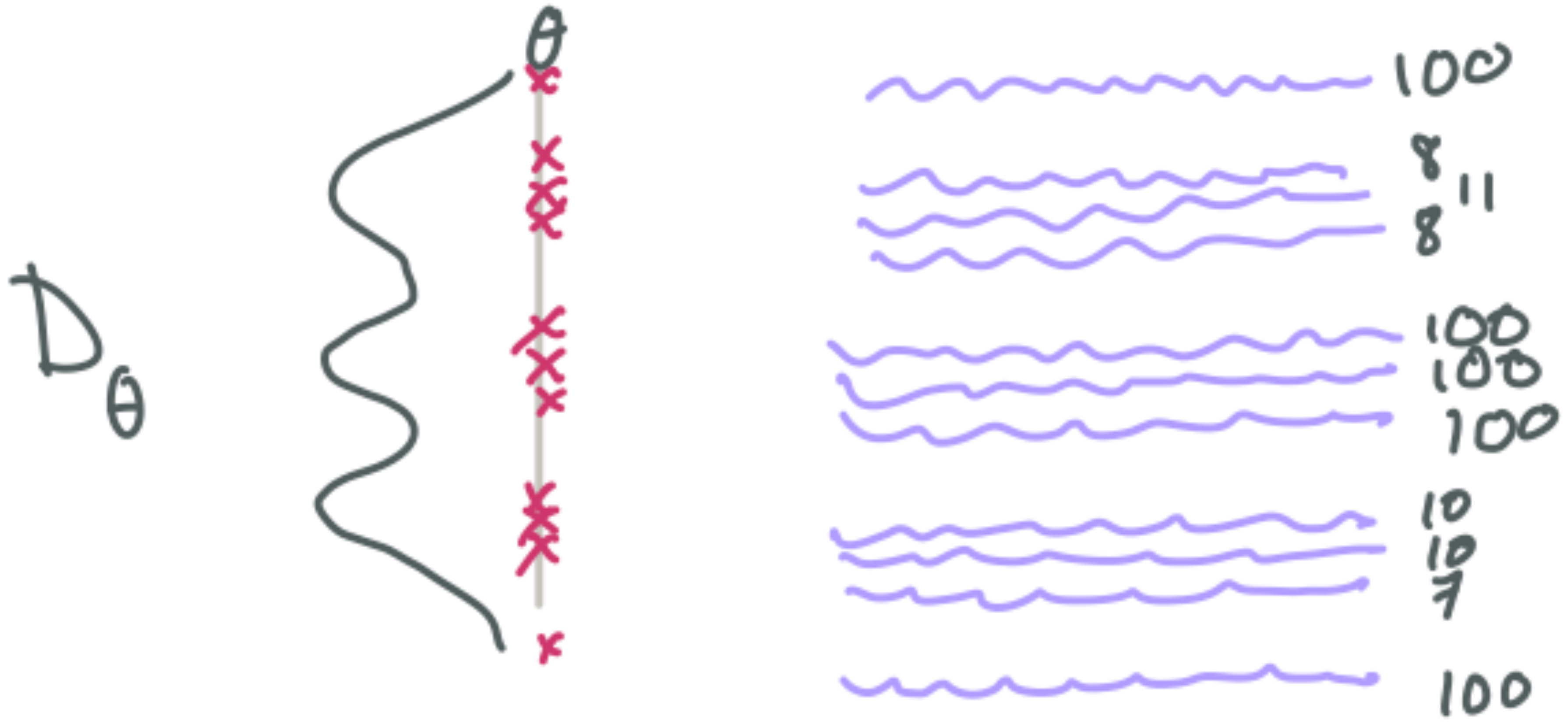
EVALUATE EACH θ_i:

- EXECUTE POLICY MULTIPLE TIMES



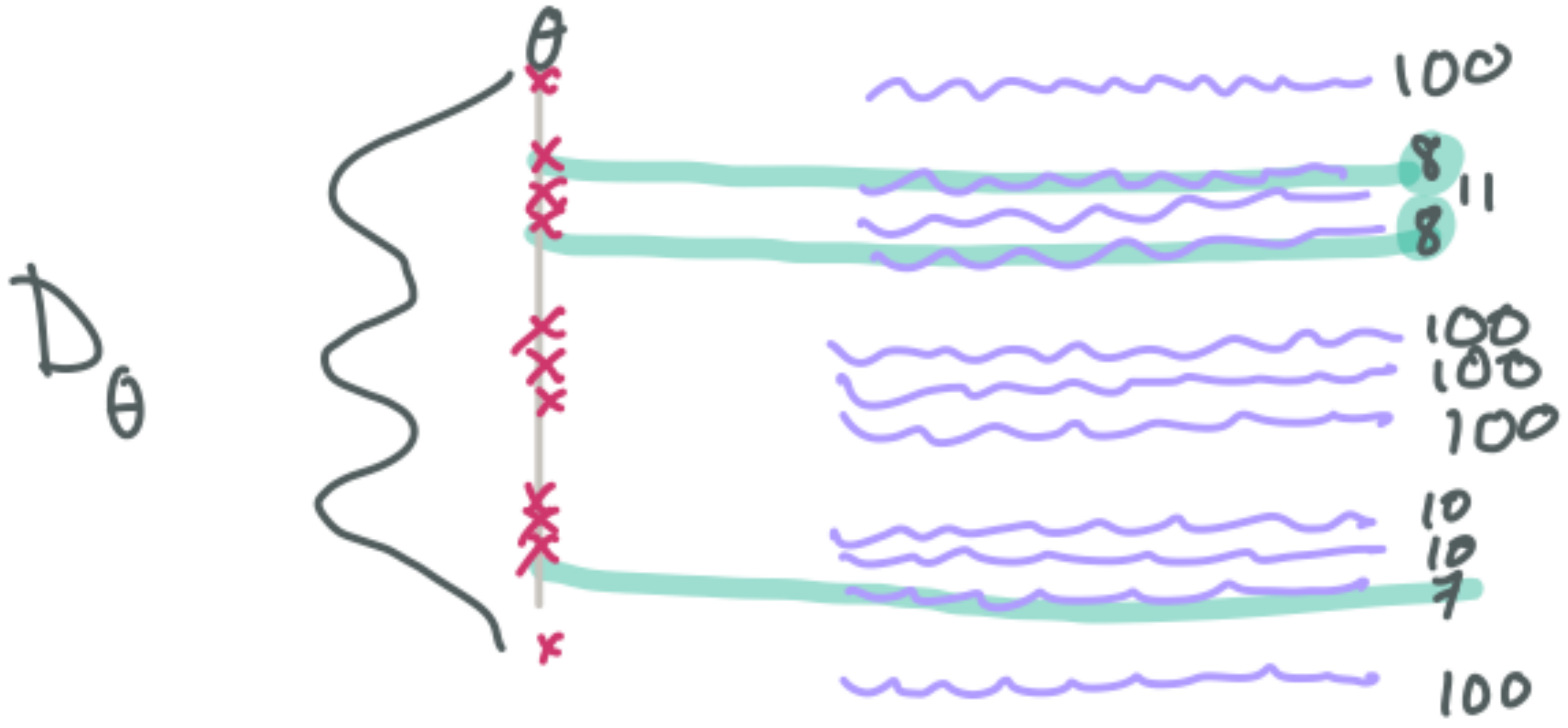
FIND TOP 'E' ELITES (e.g. 25%)

The Cross Entropy Algorithm



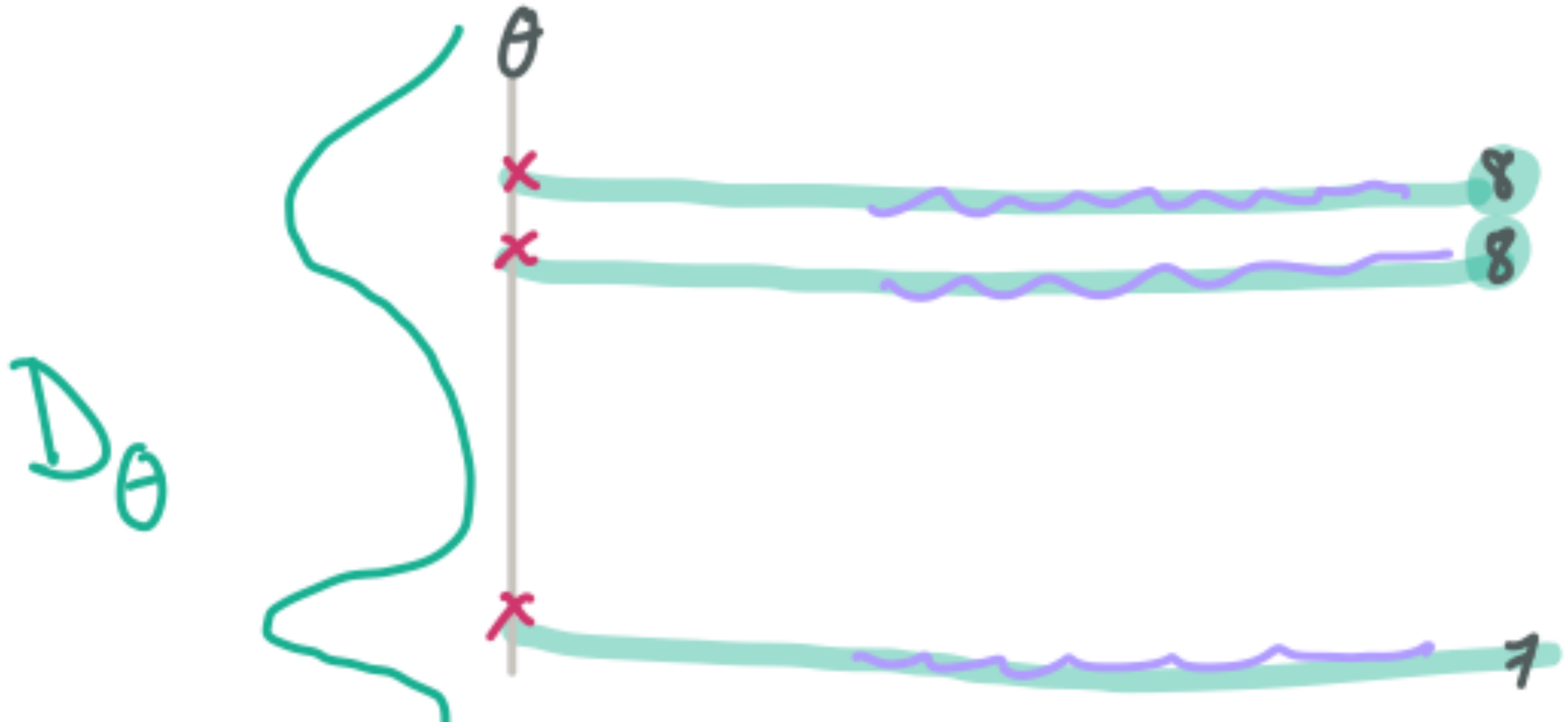
EVALUATE EACH θ_i:

- EXECUTE POLICY MULTIPLE TIMES



FIND TOP 'E' ELITES

(e.g. 25%)



FIT A NEW DISTRIBUTION

D_θ

Cross Entropy for Gaussian

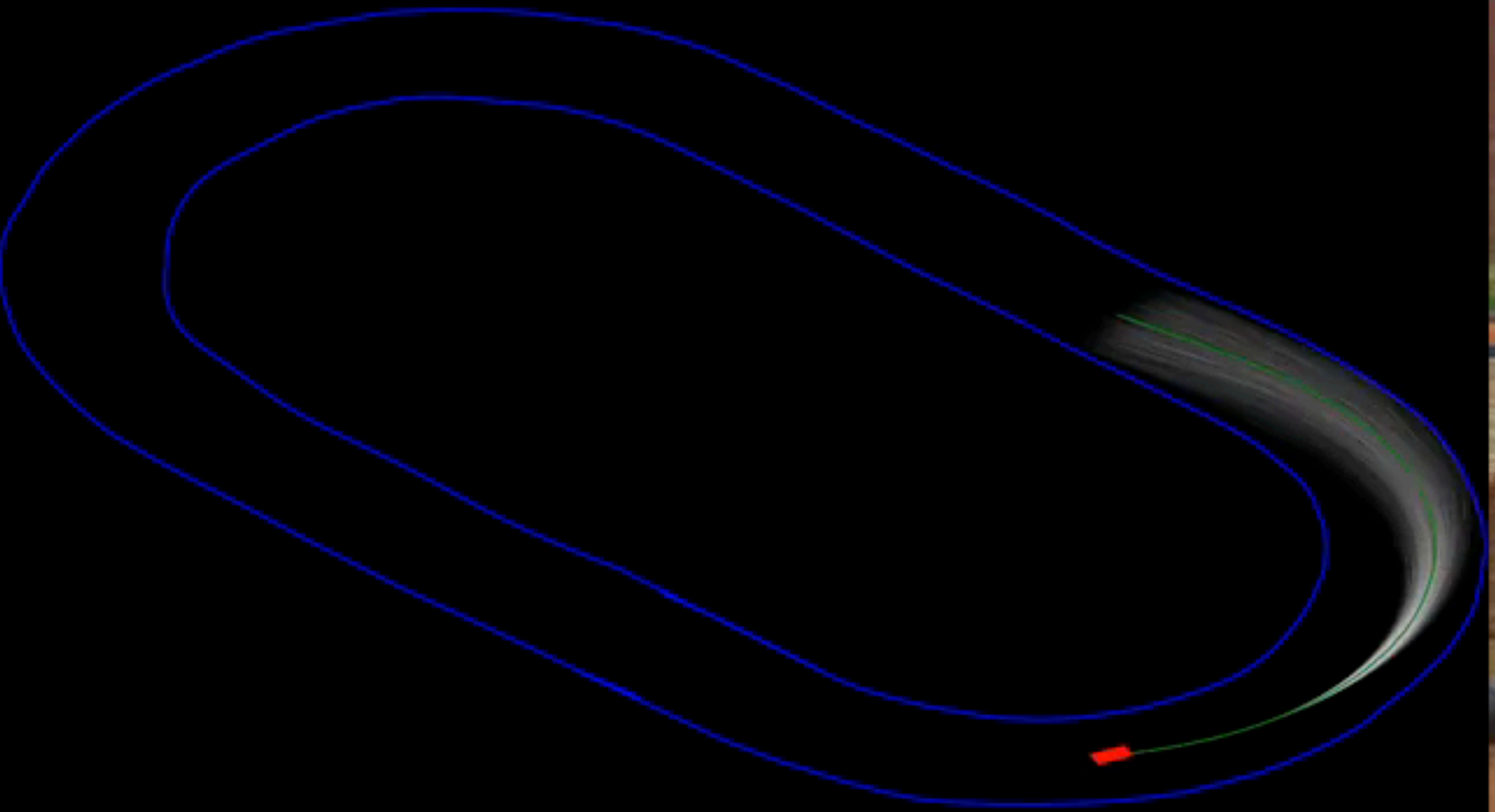
Gaussian Distribution $$D_\theta := \mathcal{N}(\mu, \Sigma)$$

Mean $$\mu^t = \frac{1}{e}\sum_{i=1}^{e}\theta_i$$

Variance $$\Sigma^t = \frac{1}{e}\sum_{i=1}^{e}(\theta_i - \mu^t)^2$$

GeorgiaTech AutoRally

2560, 2.5 second trajectories sampled
with cost-weighted average @ 60 Hz



Cross Entropy in Action!

Practical Issues and Fixes

Issue 1: What happens to the variance?

$$\Sigma^t = \frac{1}{e}\sum_{i=1}^{e}(\theta_i - \mu^t)^2$$

Collapses too quickly!

Simple fix: Add a bit of noise to the variance

$$\Sigma^t = \frac{1}{e}\sum_{i=1}^{e}(\theta_i - \mu^t)^2 + \Sigma_{noise}$$

Issue 2: What if we have a bad batch of samples?

$$\mu^t = \frac{1}{e}\sum_{i=1}^{e}\theta_i$$

The elites can be bad, and the mean can slingshot into a bad value

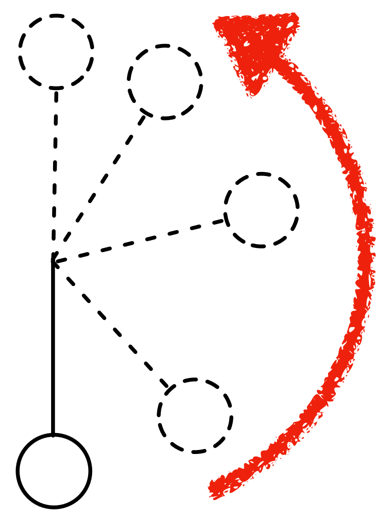
Simple fix: Slowly update mean

$$\mu^t = \mu^{t-1} + \eta\frac{1}{e}\sum_{i=1}^{e}\theta_i$$

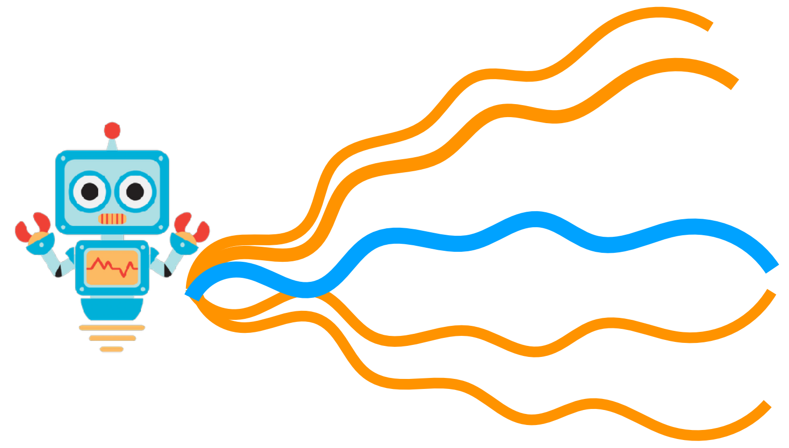
Landscape of Planning / Control Algorithms

Low-level control

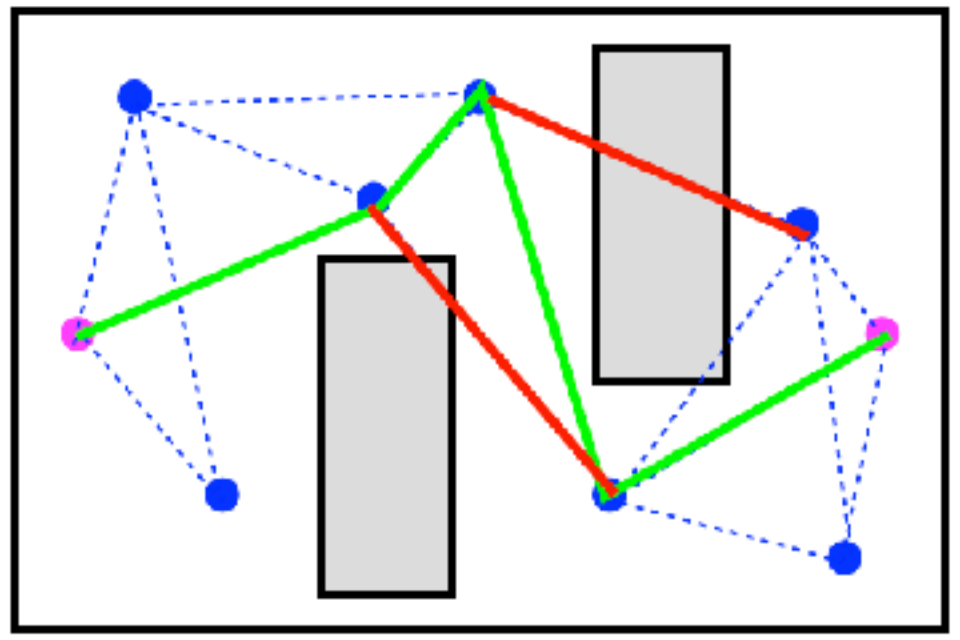
High-level path planning

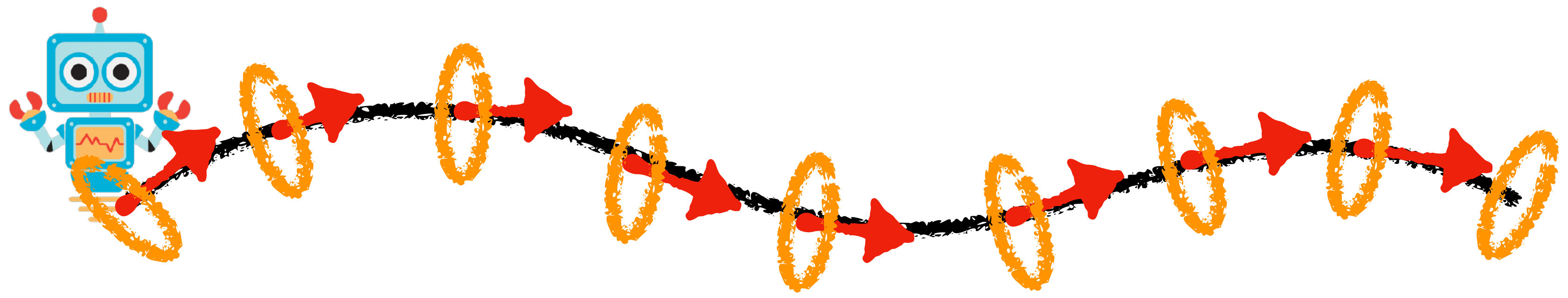


LQR



Cross Entropy

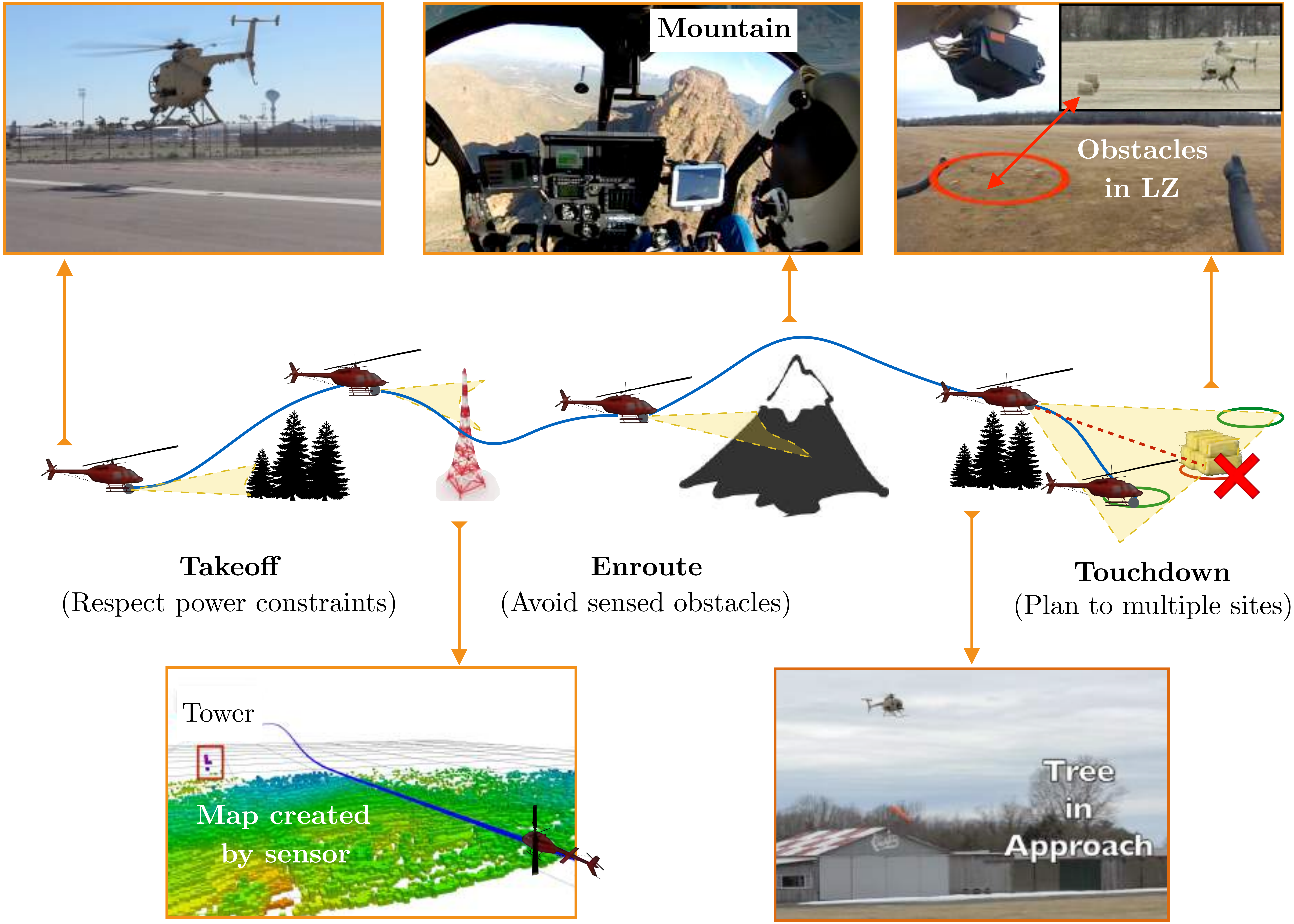




iLQR

LazySP / Halton Sampling

Let's apply what we know!


Mountain
Obstacles
in LZ
Takeoff
(Respect power constraints)
Enroute
(Avoid sensed obstacles)
Touchdown
(Plan to multiple sites)
Tower
Map created
by sensor
Tree
in
Approach

The Big Challenges

Problem 1: Don't know the terrain ahead of time!

Problem 2: Don't have a perfect dynamics model!

Problem 3: Not enough time to plan all the way to the goal!

The Unreasonable Effectiveness of Replanning

The Big Challenges

Problem 1: Don't know the terrain ahead of time!

Problem 2: Don't have a perfect dynamics model!

Problem 3: Not enough time to plan all the way to the goal!

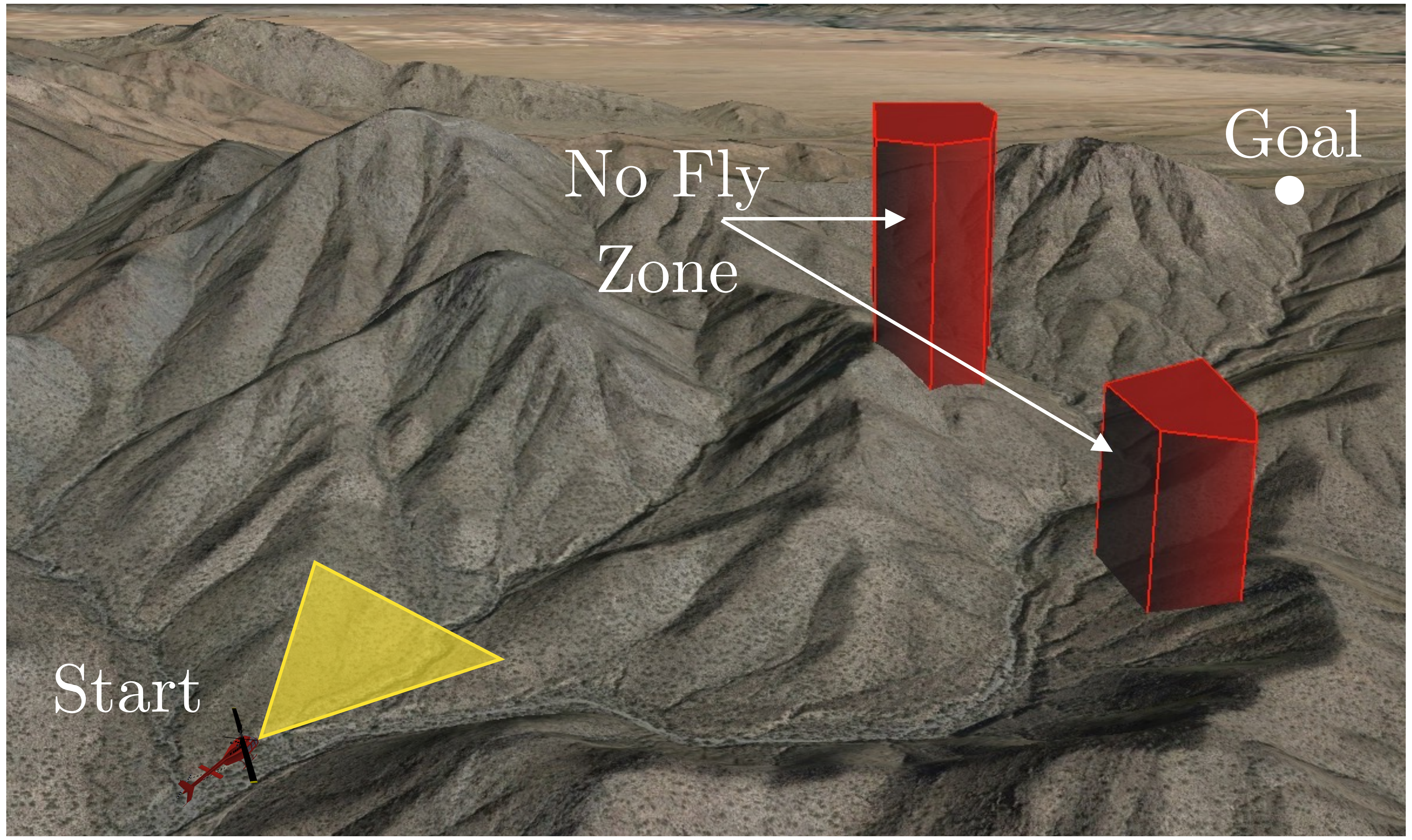
Activity!

Think-Pair-Share!

Think (30 sec): The helicopter can only sense 1km. How should it plan through unknown terrain? What assumptions can it make?

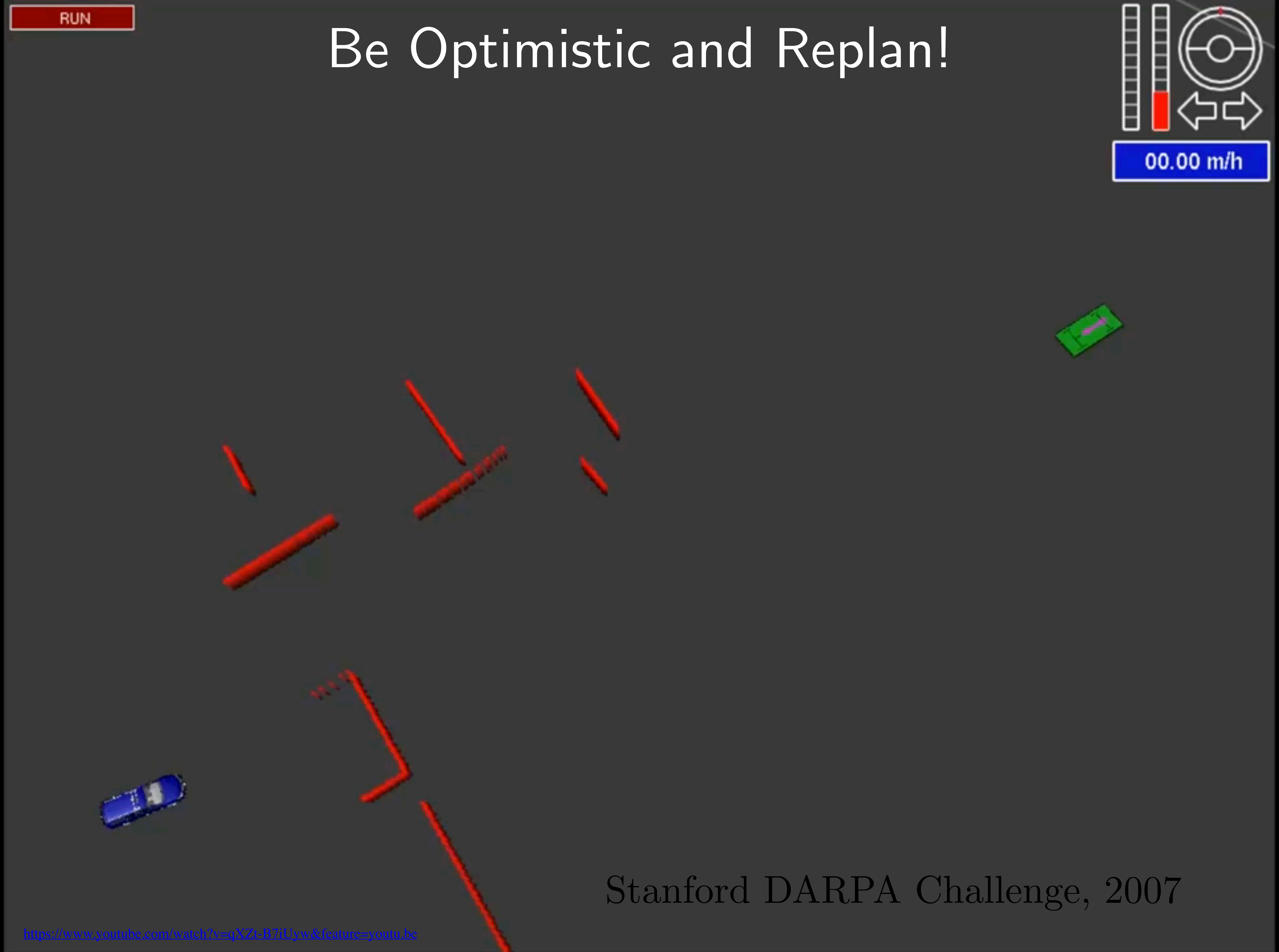
Pair: Find a partner

Share (45 sec): Partners exchange ideas



Be Optimistic and Replan!

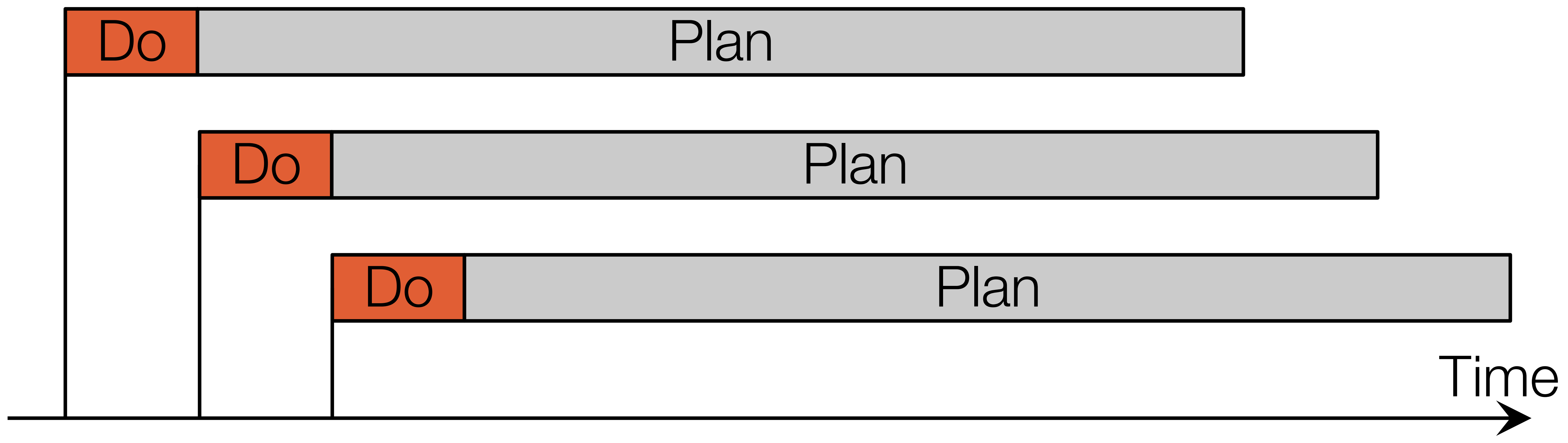
Be Optimistic and Replan!



Stanford DARPA Challenge, 2007

https://www.youtube.com/watch?v=qXZt-B7iUyw&feature=youtu.be

Model Predictive Control (MPC)



Step 1: Using your current information, solve an optimization problem

Step 2: Execute the first control and gain new information

Step 3: Repeat!

Why does this work?

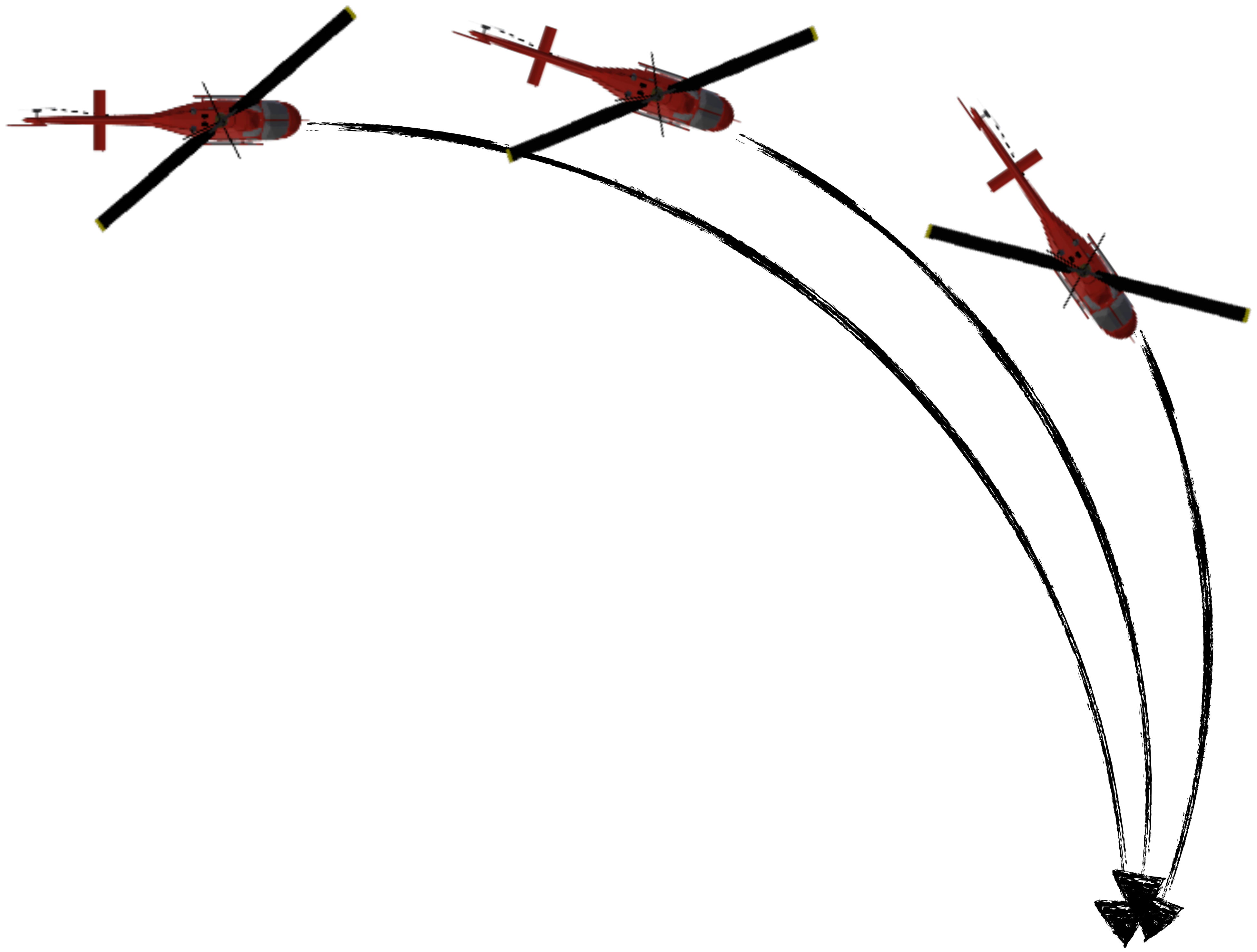
The Big Challenges

Problem 1: Don't know the terrain ahead of time!

Problem 2: Don't have a perfect dynamics model!

Problem 3: Not enough time to plan all the way to the goal!

Problem 2: Don't have a perfect dynamics model!



Let's say there is an unknown gust of wind pushing you off the path

MPC works in many cases!
(For bounded error in dynamics, the policy has bounded sub optimality)

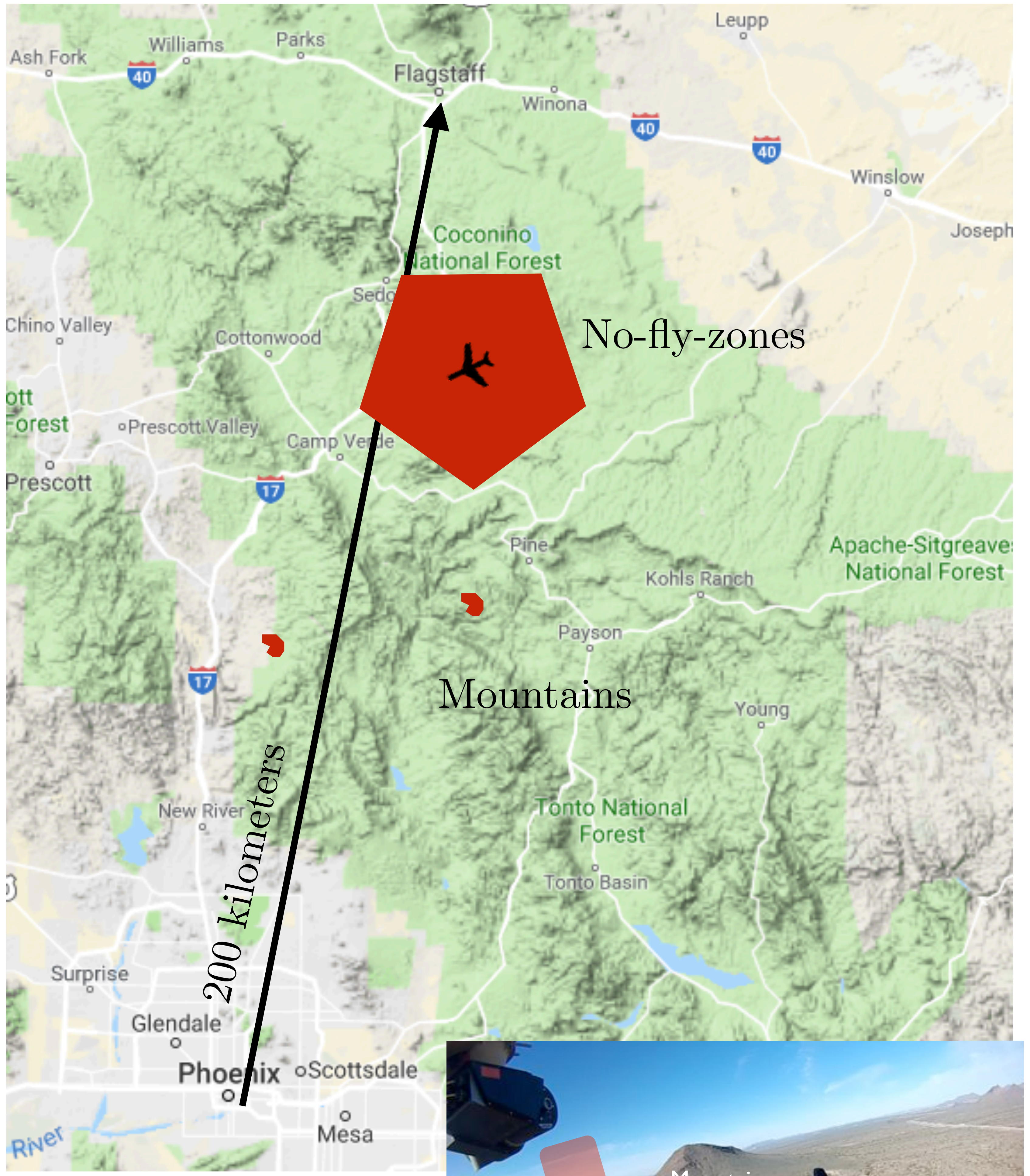
The Big Challenges

Problem 1: Don't know the terrain ahead of time!

Problem 2: Don't have a perfect dynamics model!

Problem 3: Not enough time to plan all the way to the goal!

Problem 3: Not enough time to plan all the way to goal!



Example mission:

Fly from Phoenix to Flagstaff as fast as possible (200 km)

Problem:

Take forever to plan at high resolution ALL the way to goal

When does hierarchical planning work?
When can it fail?