Learning Model Predictive Control for Iterative Tasks

Theory and Applications

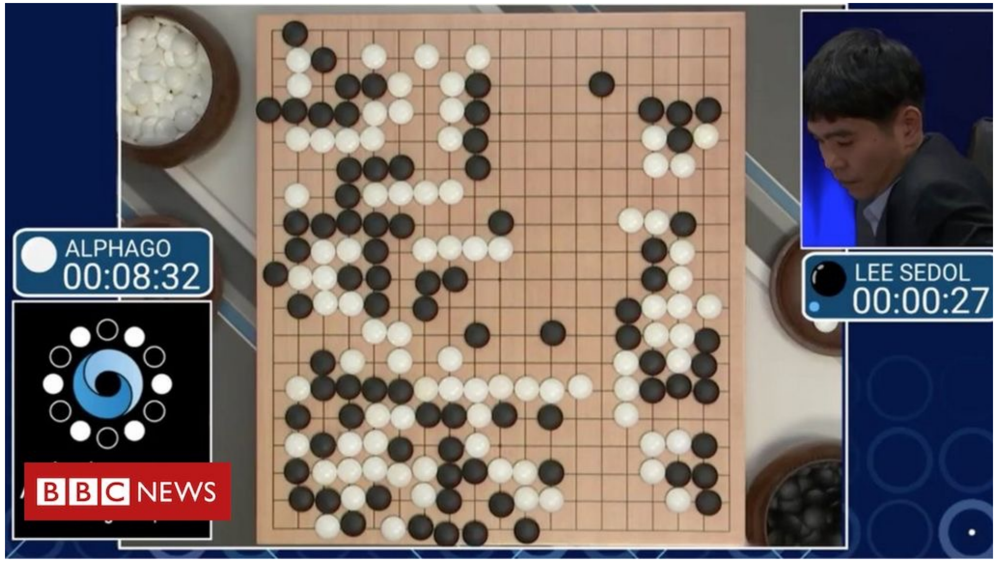
Ugo Rosolia
Research Scientist @Amazon

Work done @Caltech and @UCB

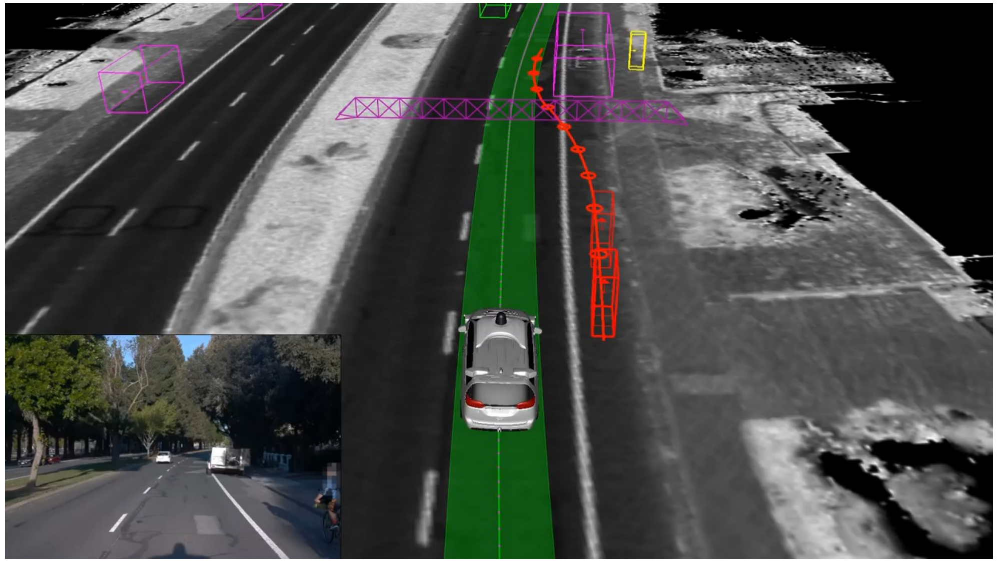
Feb 22nd, 2022

Success Stories from AI

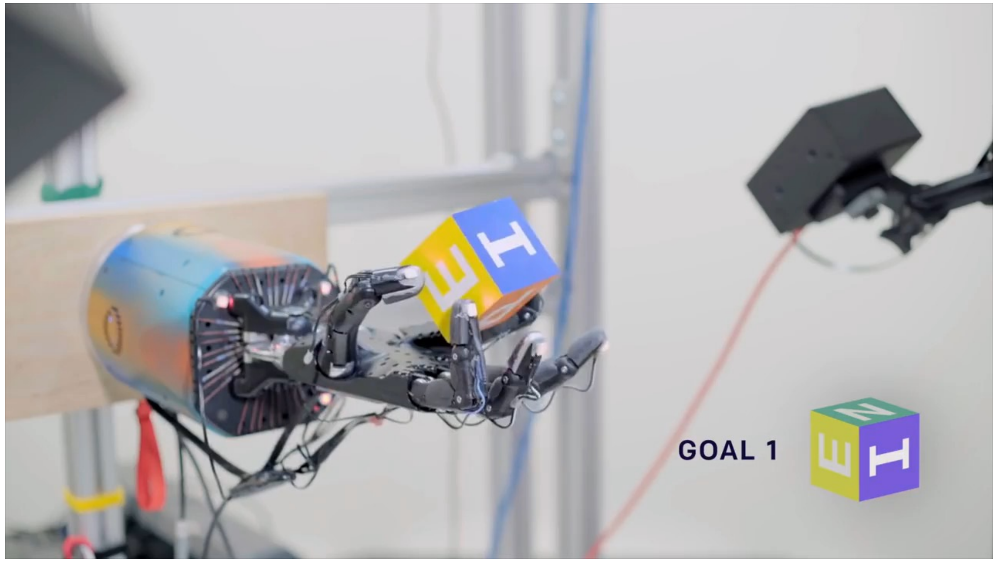
Alpha GO



Waymo's Perception Module



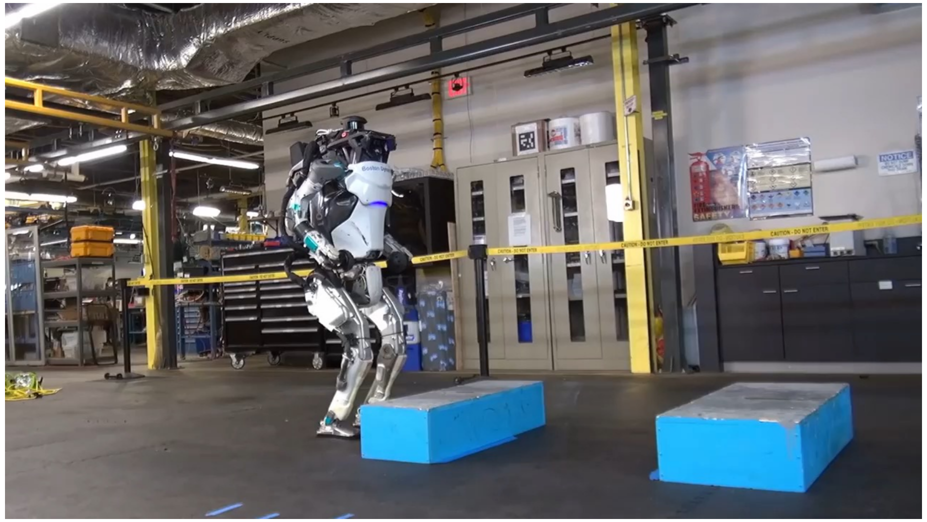
OpenAI



Google

Success Stories from Control Theory

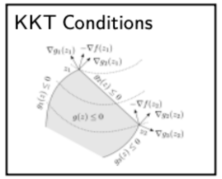
Boston Dynamics



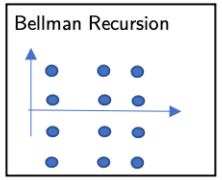
Stanford Dynamic Design Lab

Standard Control Pipeline

Optimal Trajectory

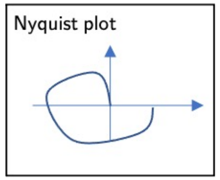


Optimization

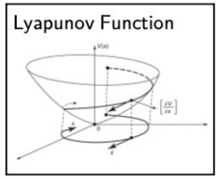


Dynamic Programming

Trajectory Tracking

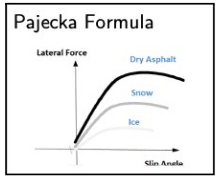


Frequency Domain

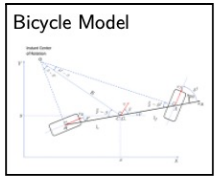


Nonlinear Control

System Identification

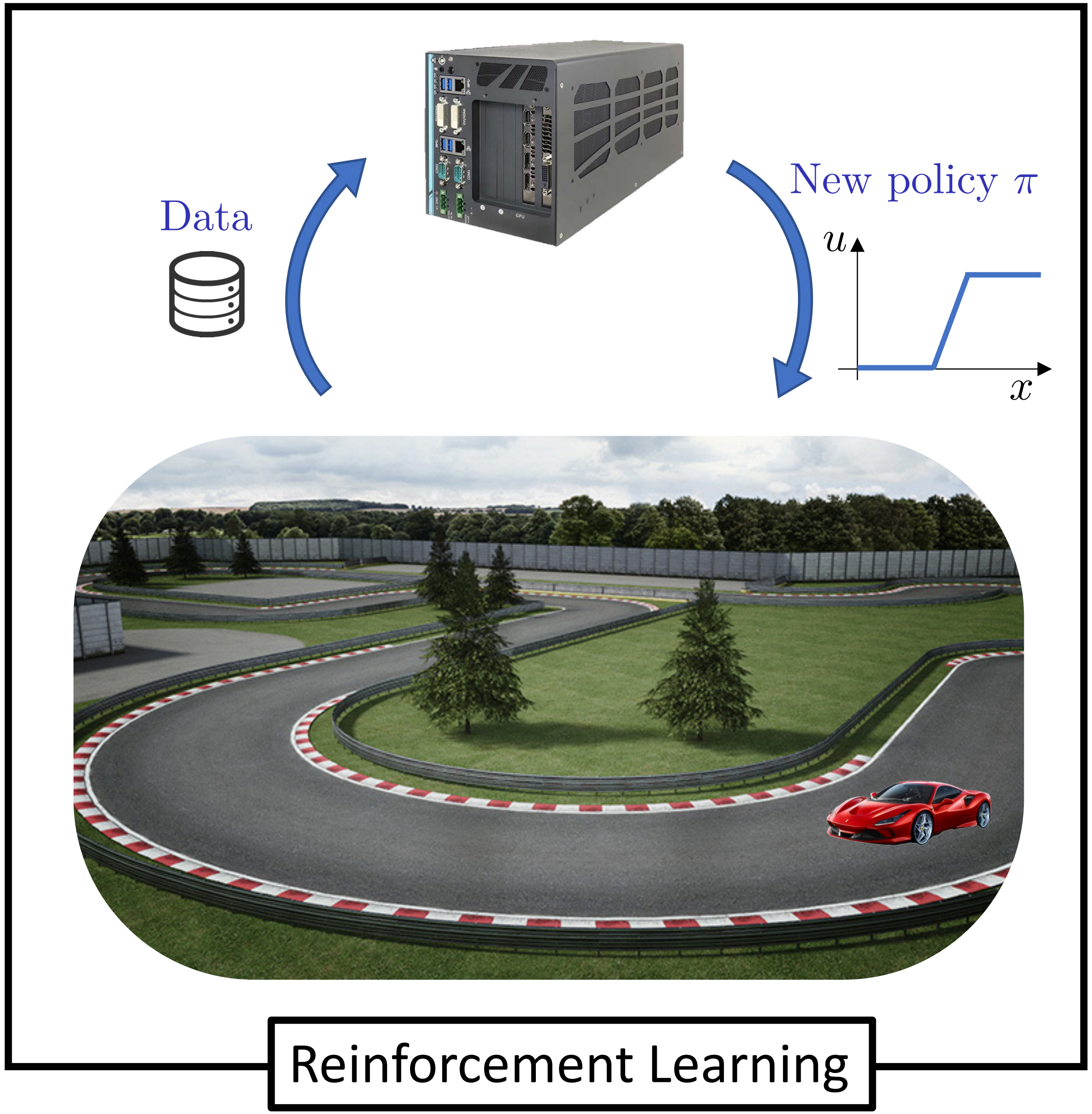


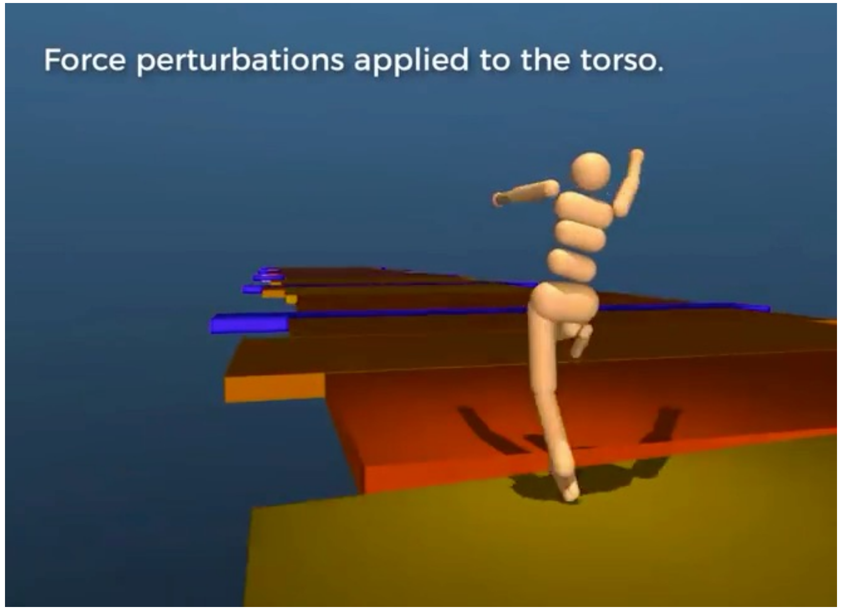
Tire Dynamics



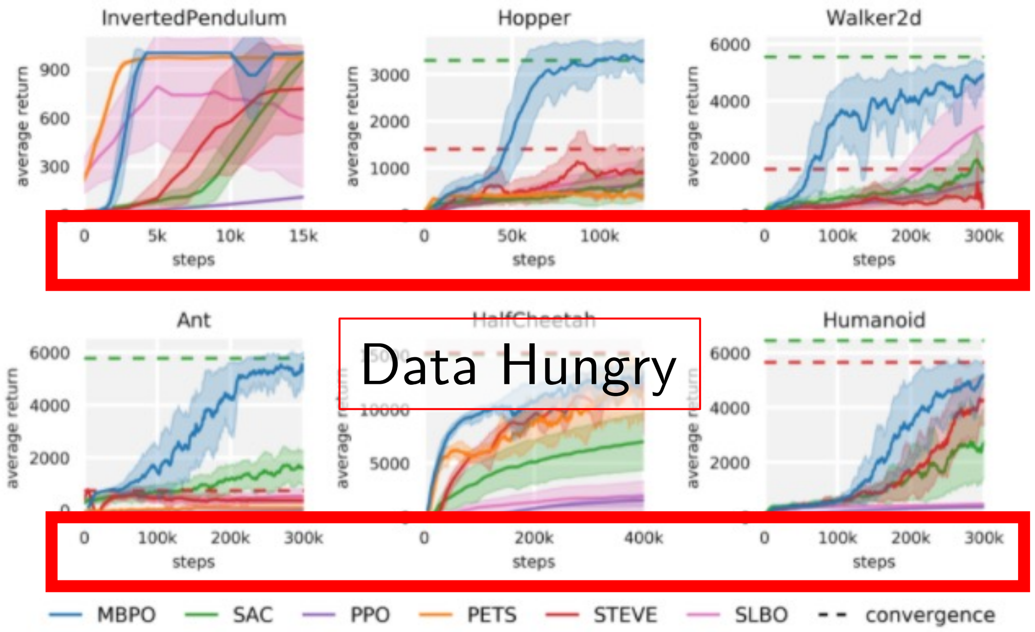
Vehicle Dynamics

Can we simplify the control design?



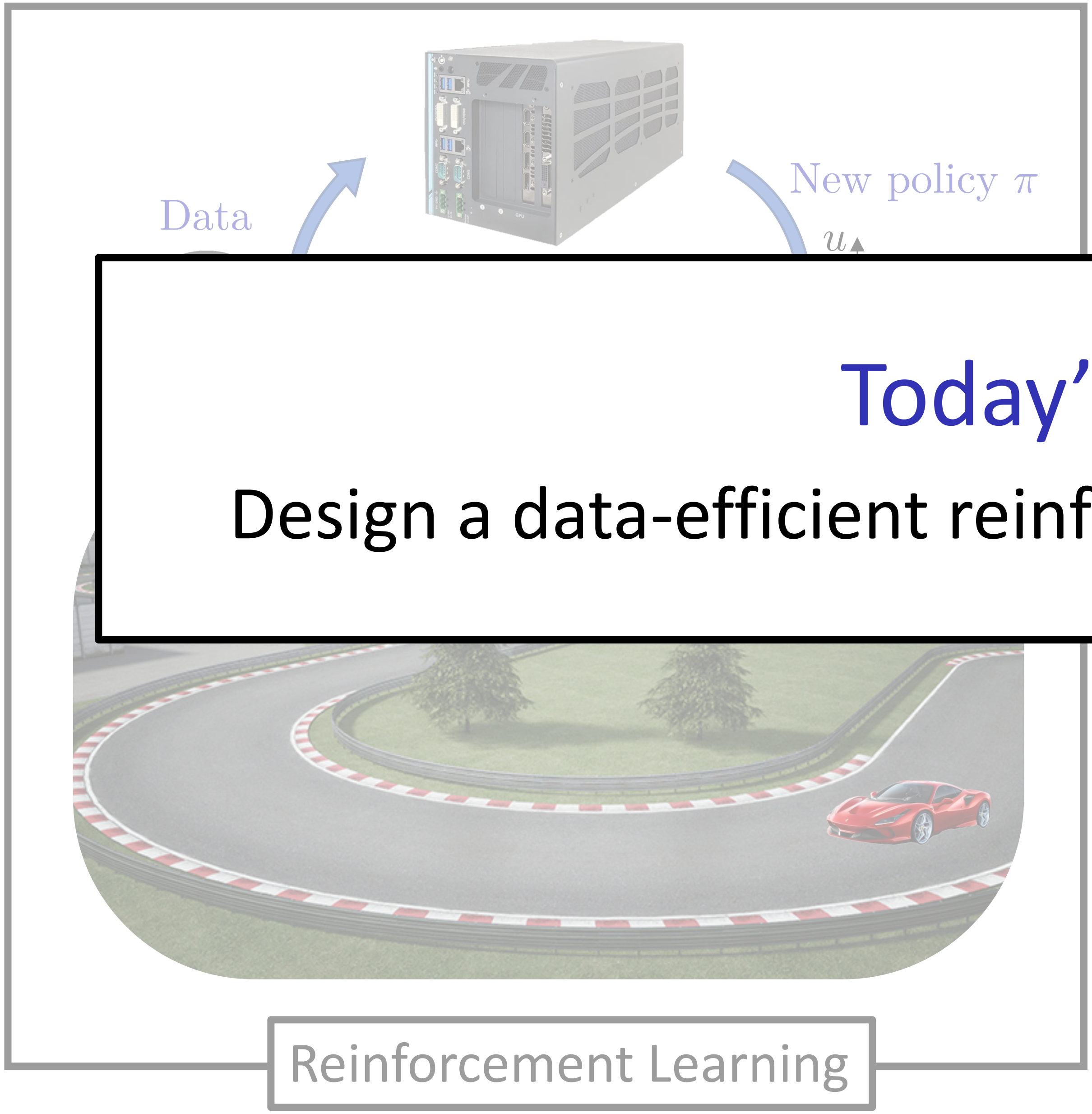


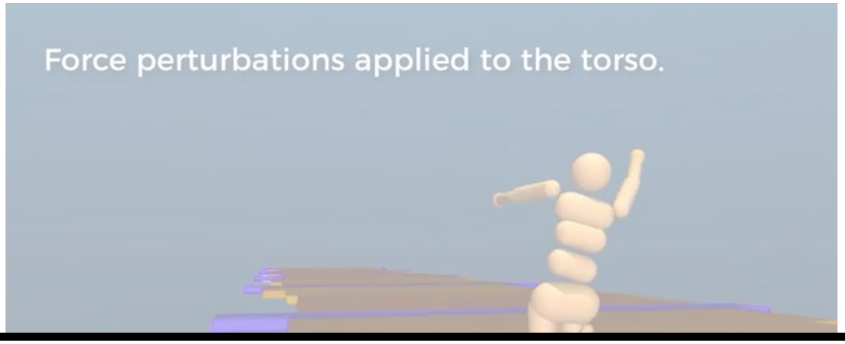
DeepMind



M. Janner, J. Fu, M. Zhang, and S. Levine. "When to trust your model: Model-based policy optimization." arXiv preprint arXiv:1906.08253 (2019).

Can we simplify the control design?





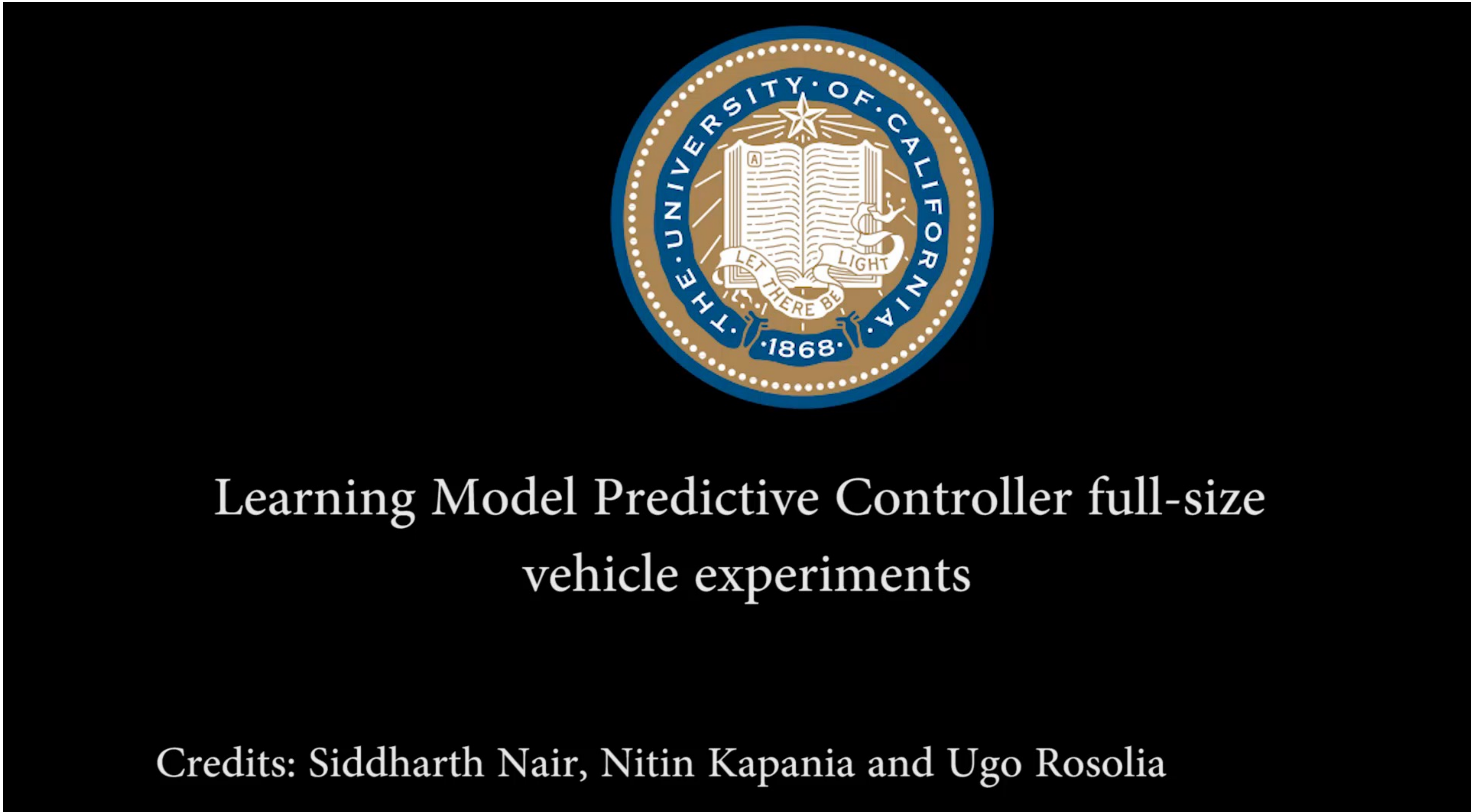
DeepMind

Today's goals:

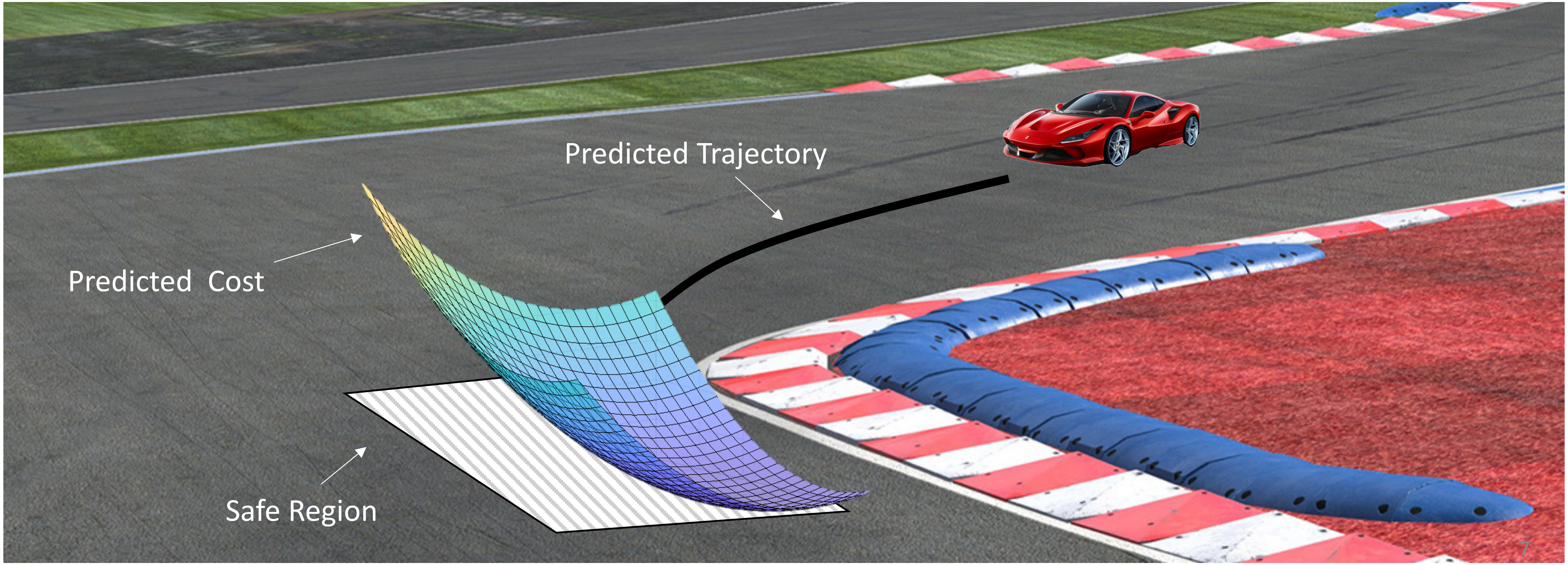
Design a data-efficient reinforcement learning algorithm

M. Janner, J. Fu, M. Zhang, and S. Levine. "When to trust your model: Model-based policy optimization." arXiv preprint arXiv:1906.08253 (2019).

Today's Example



Lesson from Model Predictive Control (MPC)



- Predicted trajectory given by Prediction Model
- Safe region estimated by the Safe Set
- Predicted cost estimated by Value Function

Three key components to learn

Prediction Model

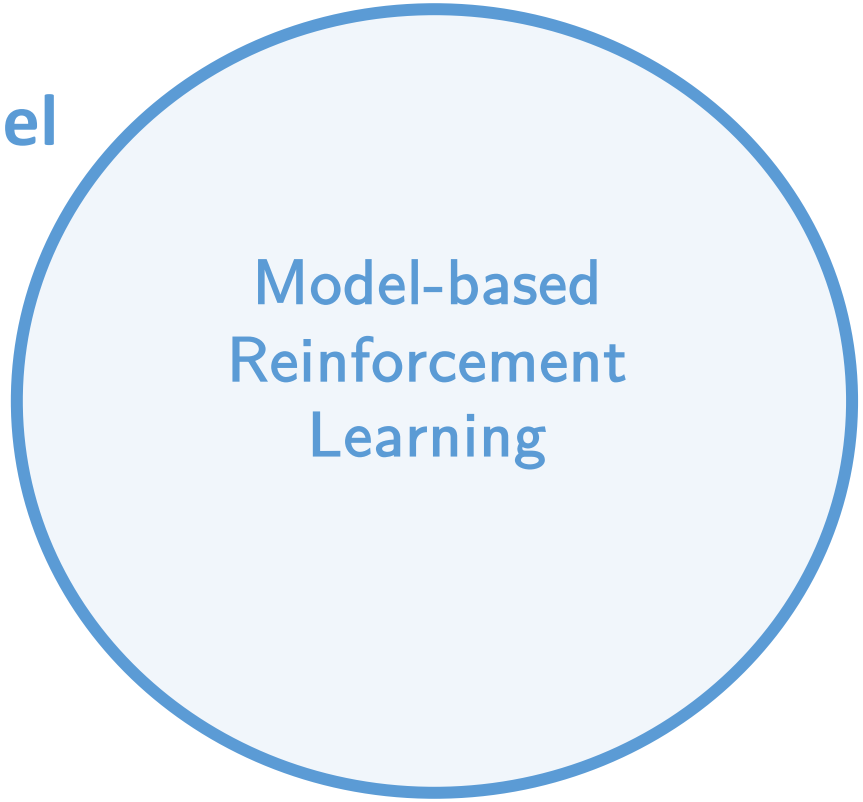
Value Function

Safe Set

Three key components to learn

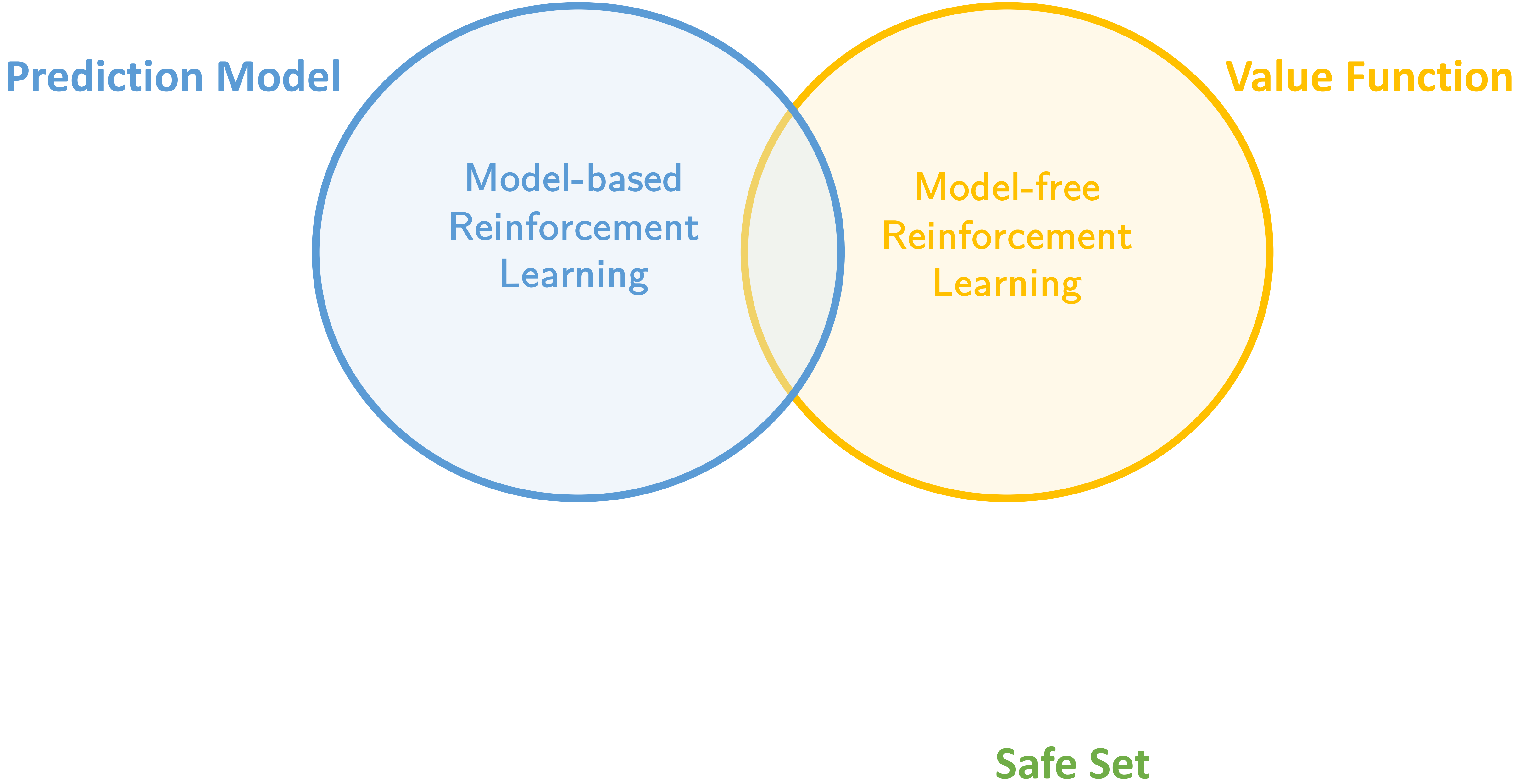
Prediction Model

Value Function

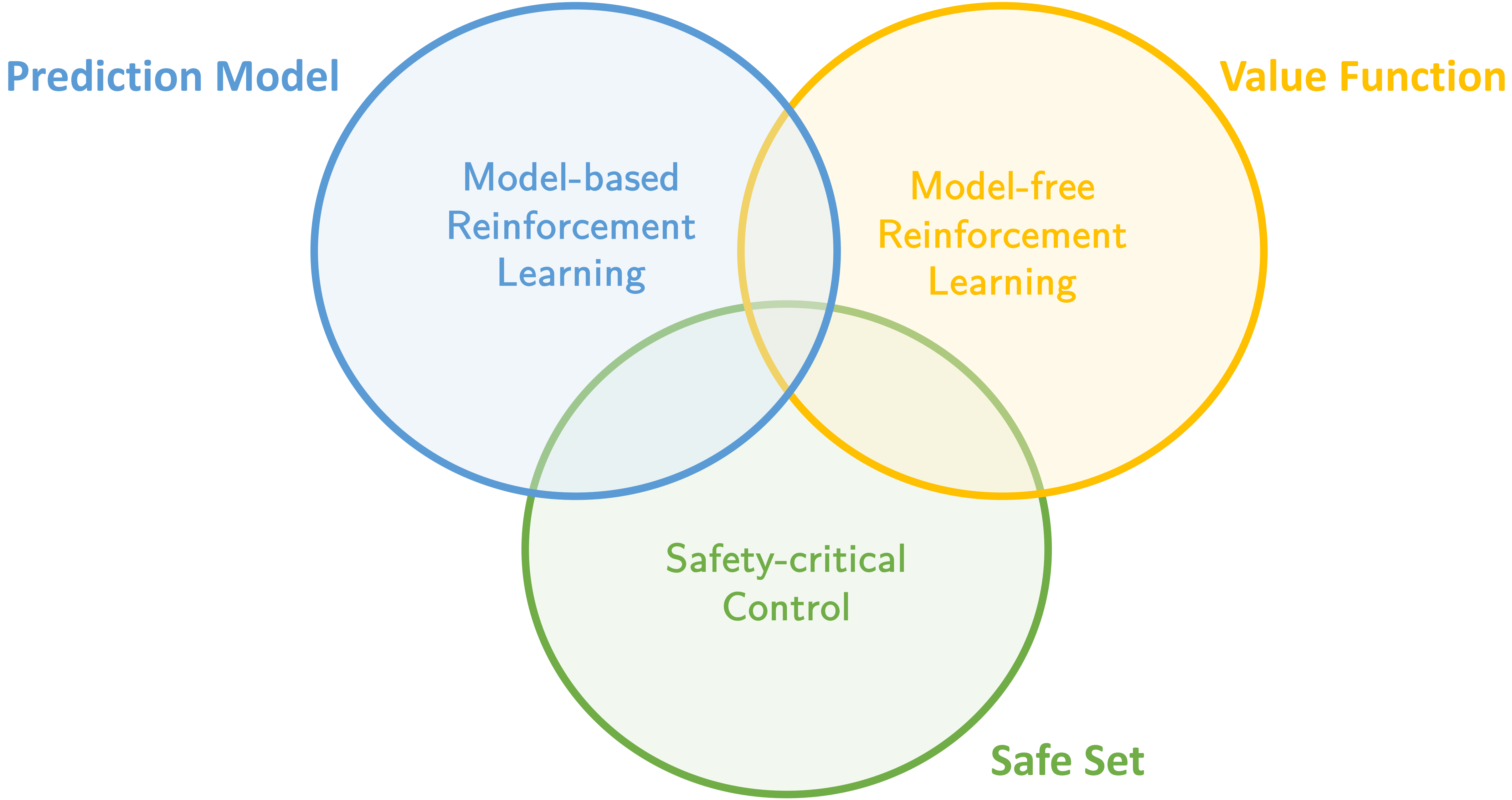


Safe Set

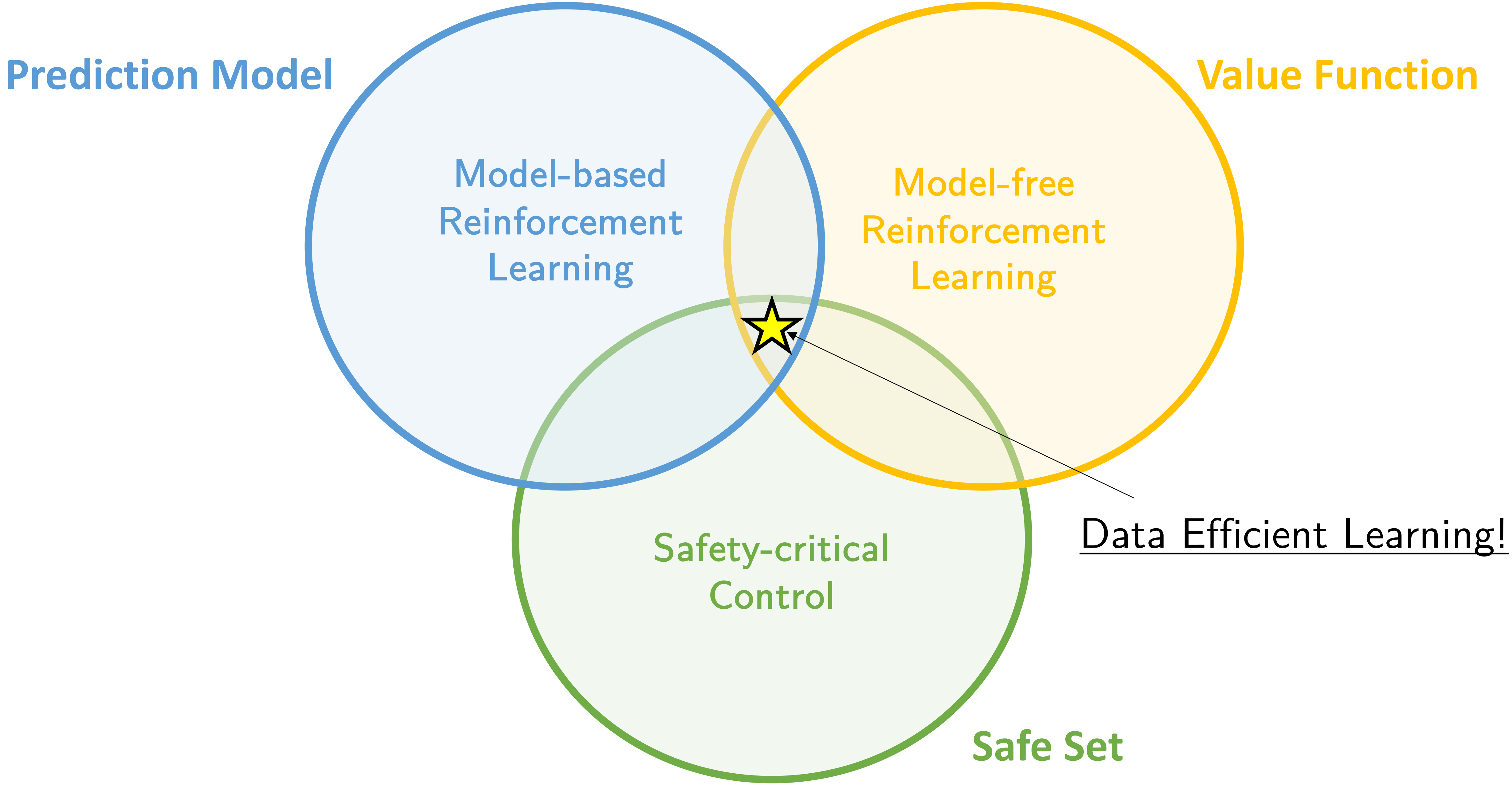
Three key components to learn



Three key components to learn



Three key components to learn



Outline

- Iterative Control Design for Deterministic Systems
- Autonomous Racing Experiments
- Uncertain Systems
- Multi-modal uncertainty and future steps

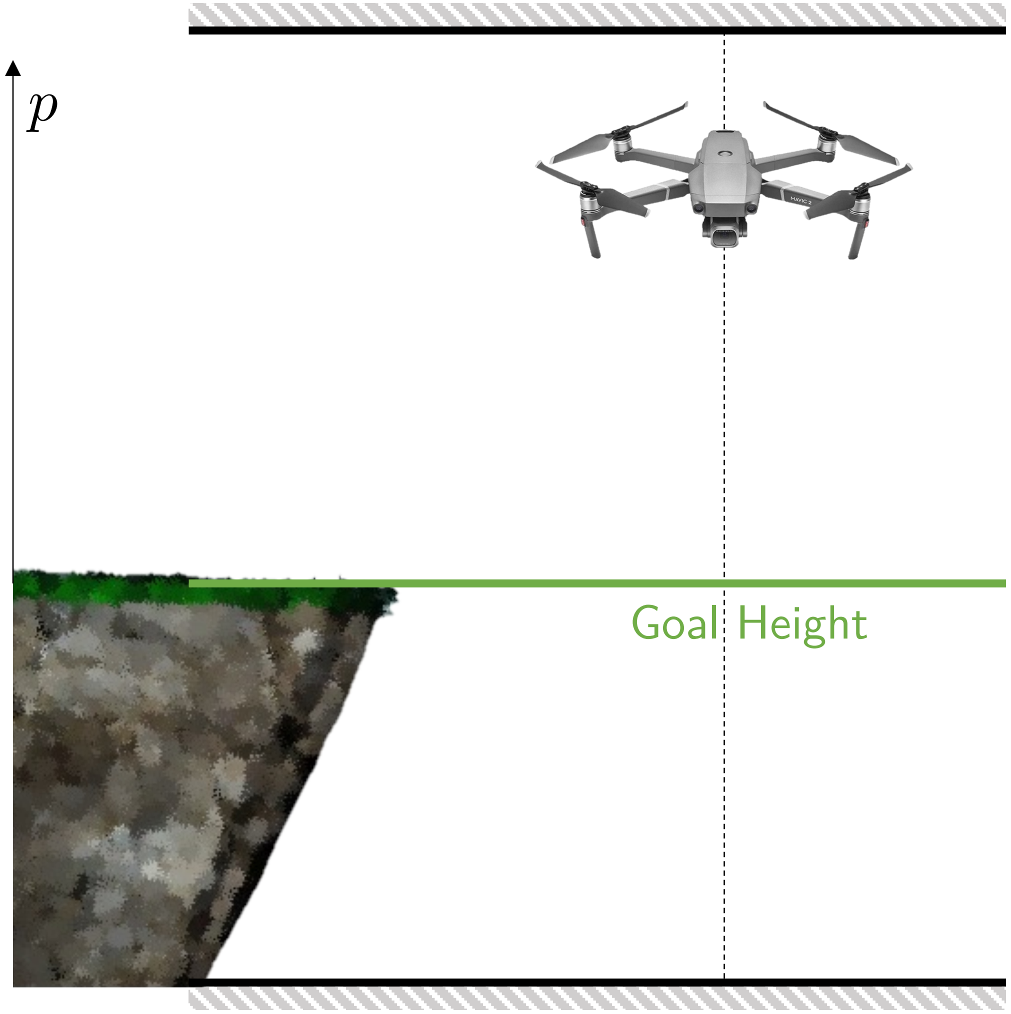
Outline

- Iterative Control Design for Deterministic Systems
- Autonomous Racing Experiments
- Uncertain Systems
- Multi-modal uncertainty and future steps

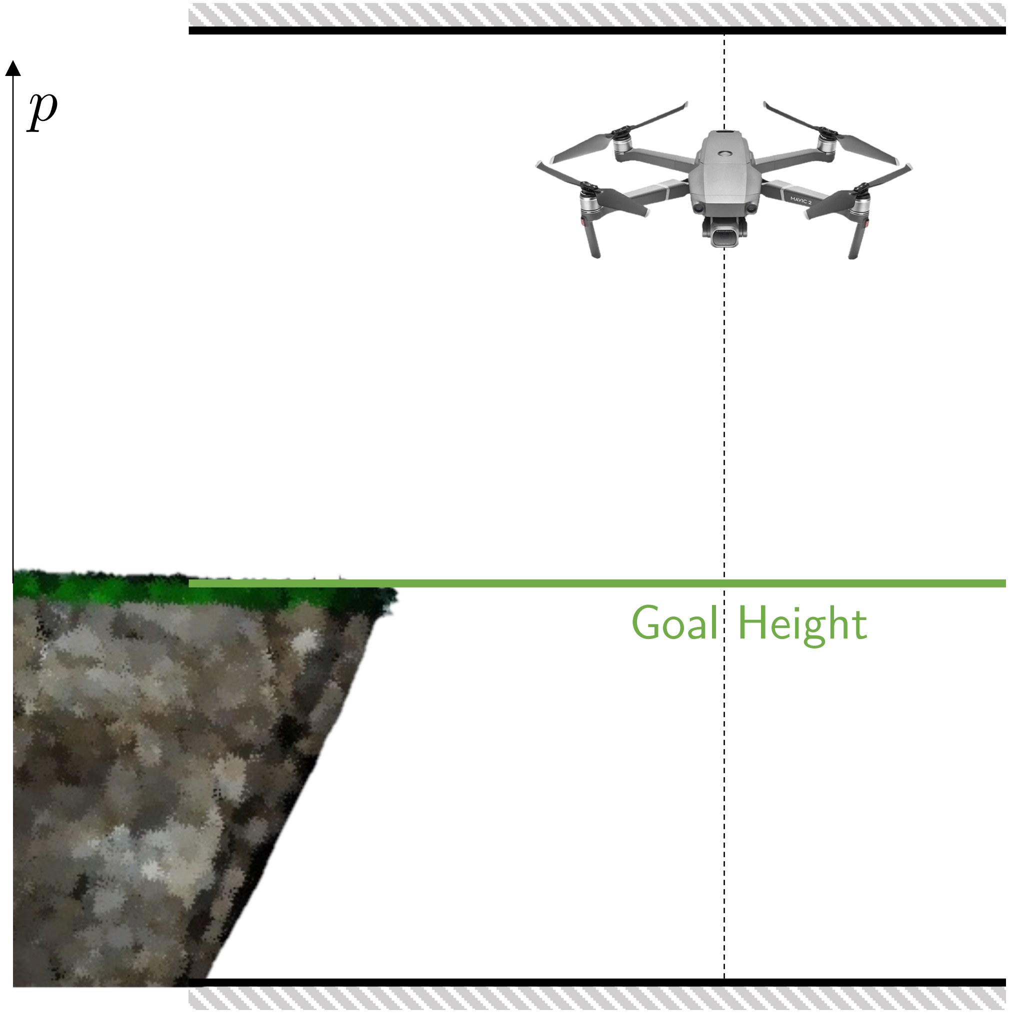
Iterative Tasks

Iterative data collection and policy update

Iterative Tasks – Drone Example



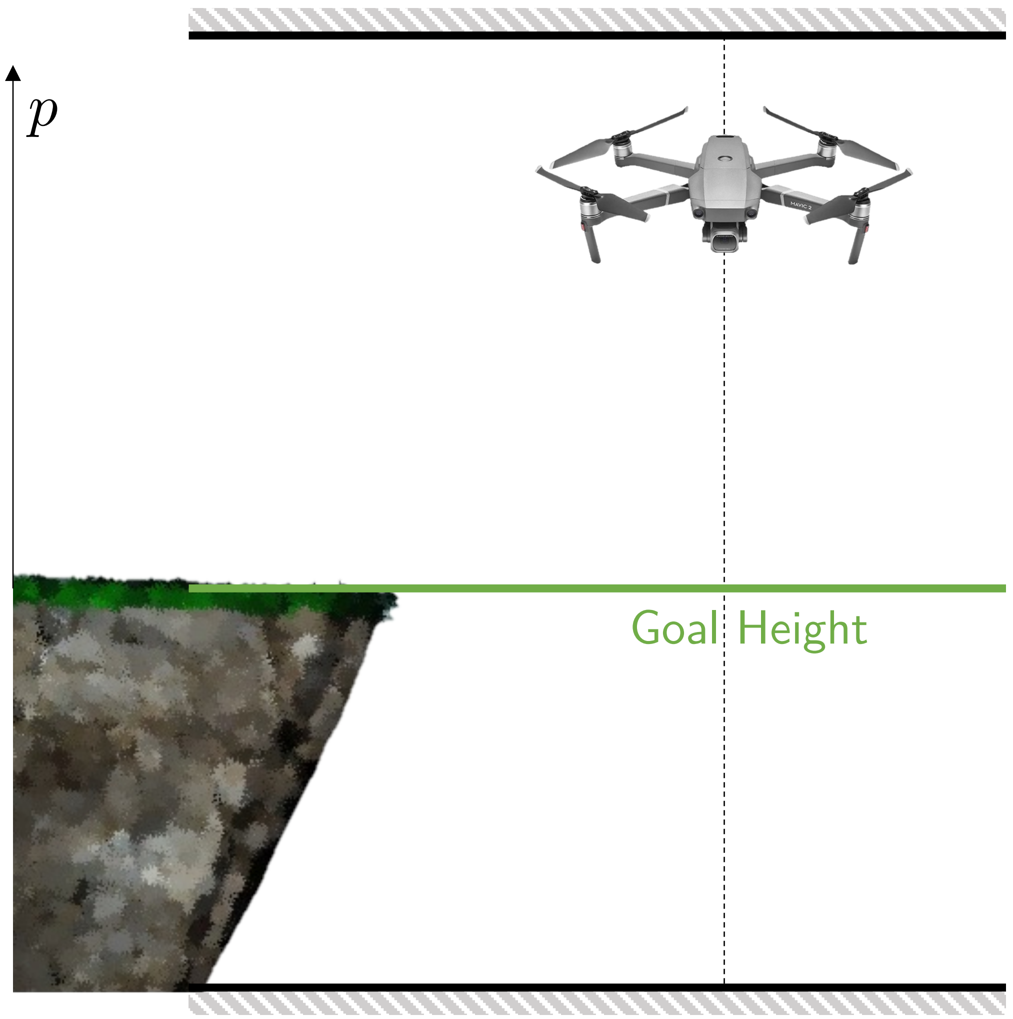
Iterative Tasks – Drone Example



- State

$$x = \begin{bmatrix} p \\ v \end{bmatrix} = \begin{bmatrix} \text{position} \\ \text{velocity} \end{bmatrix}$$

Iterative Tasks – Drone Example

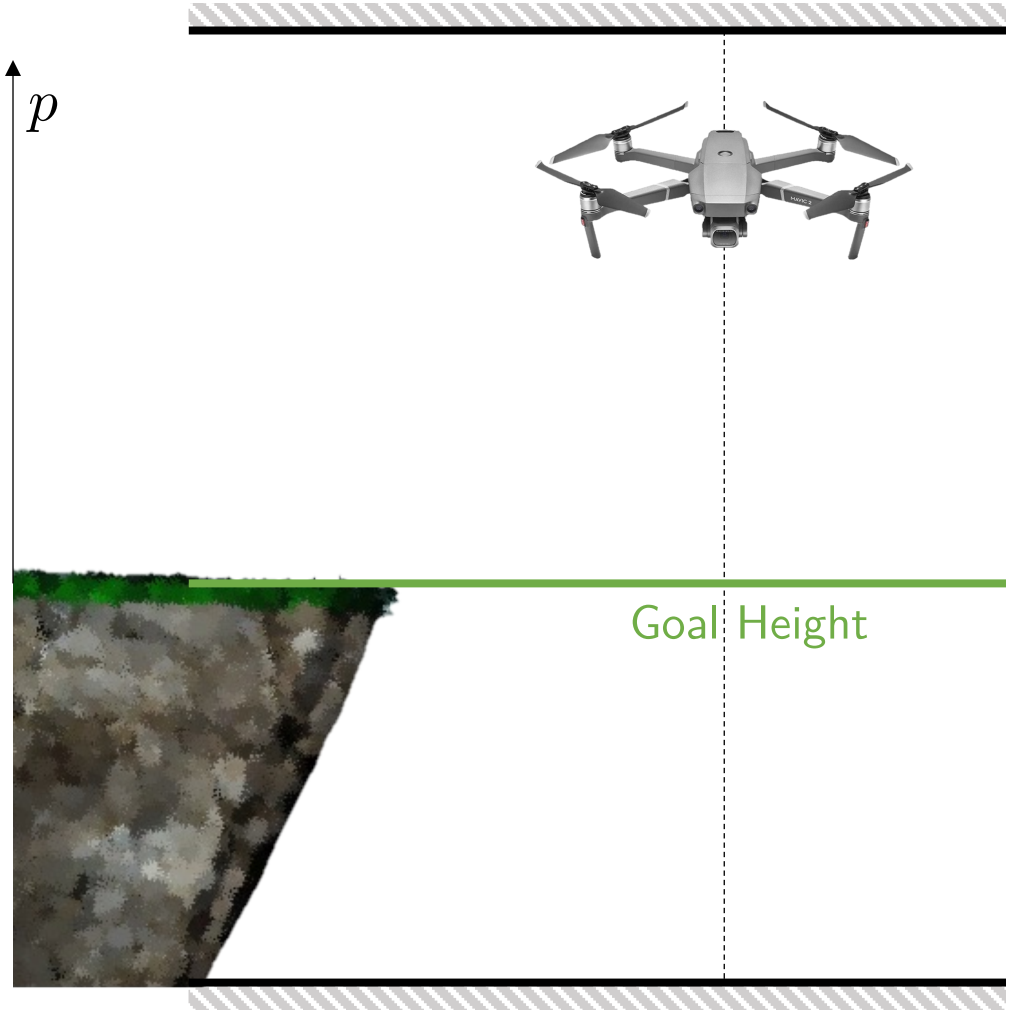


- State

$$x = \begin{bmatrix} p \\ v \end{bmatrix} = \begin{bmatrix} \text{position} \\ \text{velocity} \end{bmatrix}$$

- Input $u = a =$ acceleration

Iterative Tasks – Drone Example



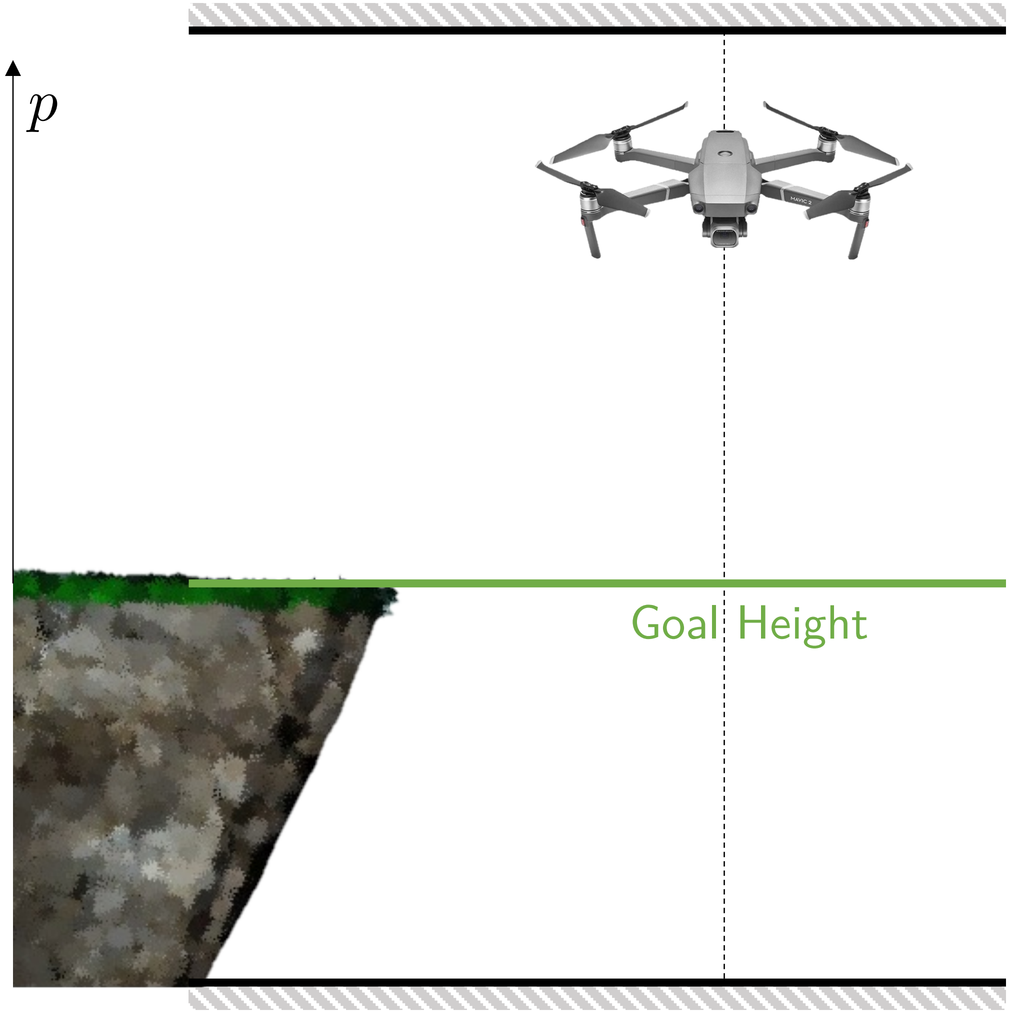
- State

$$x = \begin{bmatrix} p \\ v \end{bmatrix} = \begin{bmatrix} \text{position} \\ \text{velocity} \end{bmatrix}$$

- Input $u = a =$ acceleration
- Dynamics

$$\begin{bmatrix} p_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & dt \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_k \\ v_k \end{bmatrix} + \begin{bmatrix} 0 \\ a_k \end{bmatrix}$$

Iterative Tasks – Drone Example



- State

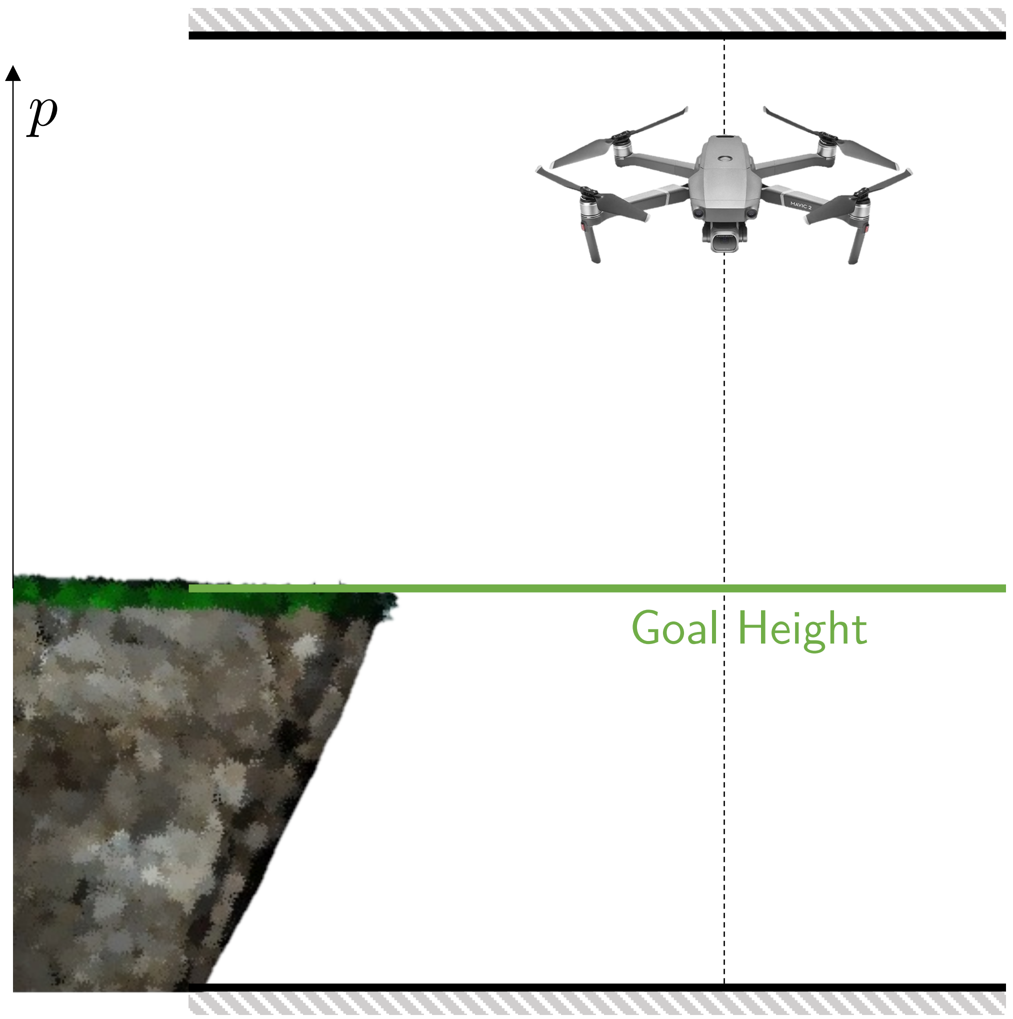
$$x = \begin{bmatrix} p \\ v \end{bmatrix} = \begin{bmatrix} \text{position} \\ \text{velocity} \end{bmatrix}$$

- Input $u = a =$ acceleration
- Dynamics

$$\begin{bmatrix} p_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & dt \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_k \\ v_k \end{bmatrix} + \begin{bmatrix} 0 \\ a_k \end{bmatrix}$$

- Cost $x_k^\top Q x_k + u_k^\top R u_k$

Iterative Tasks – Drone Example



- State

$$x = \begin{bmatrix} p \\ v \end{bmatrix} = \begin{bmatrix} \text{position} \\ \text{velocity} \end{bmatrix}$$

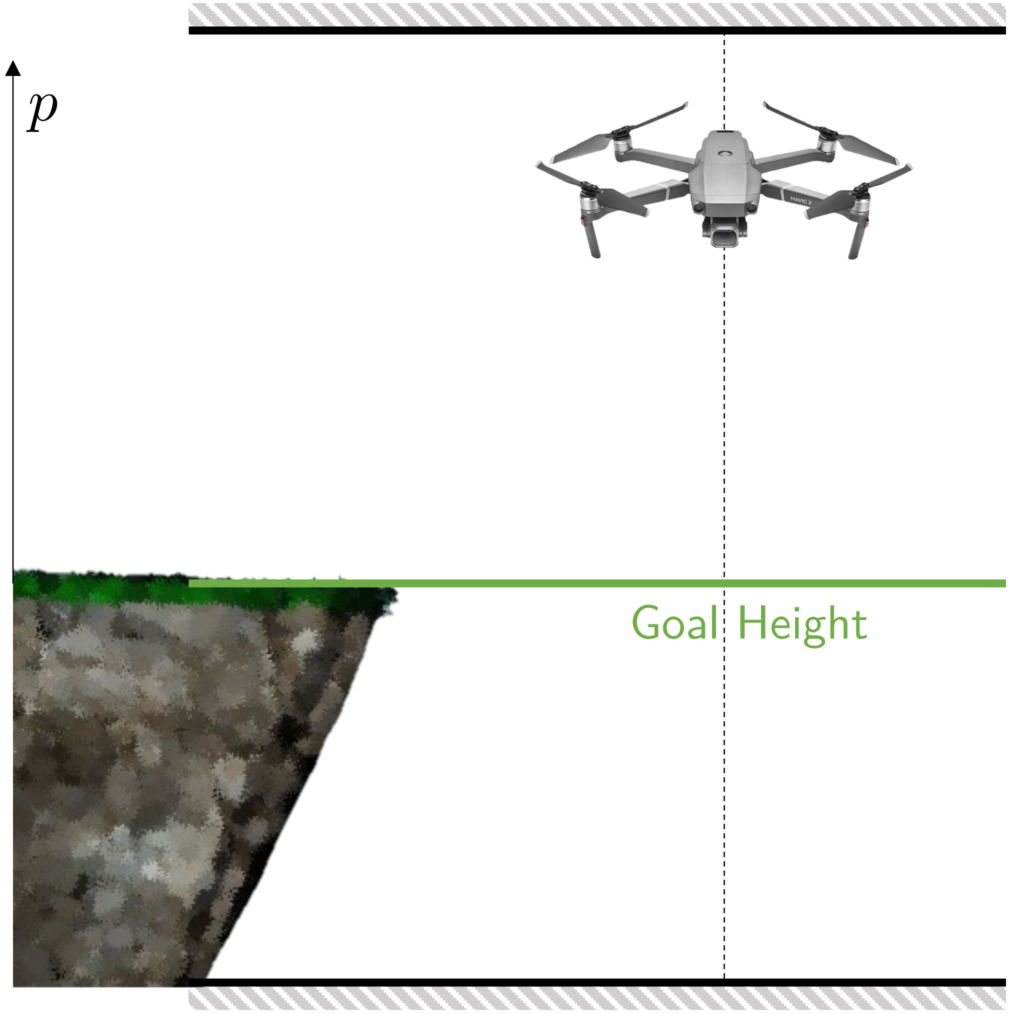
- Input $u = a =$ acceleration
- Dynamics

$$\begin{bmatrix} p_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & dt \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_k \\ v_k \end{bmatrix} + \begin{bmatrix} 0 \\ a_k \end{bmatrix}$$

- Cost $x_k^\top Q x_k + u_k^\top R u_k$
- Constraints

$$\begin{bmatrix} -5 \\ -5 \\ -0.5 \end{bmatrix} \leq \begin{bmatrix} p_k \\ v_k \\ a_k \end{bmatrix} \leq \begin{bmatrix} 5 \\ 5 \\ 0.5 \end{bmatrix}$$

Iterative Tasks – Drone Example



- State

$$x = \begin{bmatrix} p \\ v \end{bmatrix} = \begin{bmatrix} \text{position} \\ \text{velocity} \end{bmatrix}$$

- Input $u = a =$ acceleration
- Dynamics

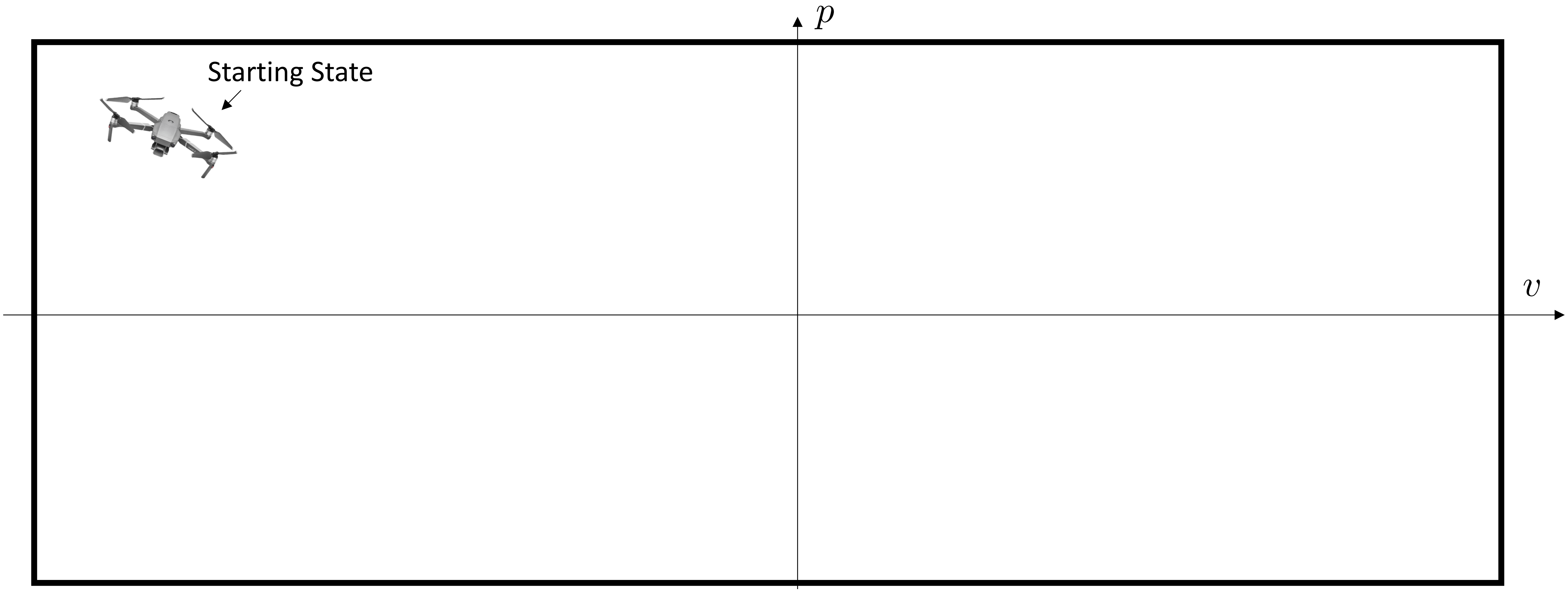
$$\begin{bmatrix} p_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & dt \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_k \\ v_k \end{bmatrix} + \begin{bmatrix} 0 \\ a_k \end{bmatrix}$$

- Cost $x_k^\top Q x_k + u_k^\top R u_k$
- Constraints

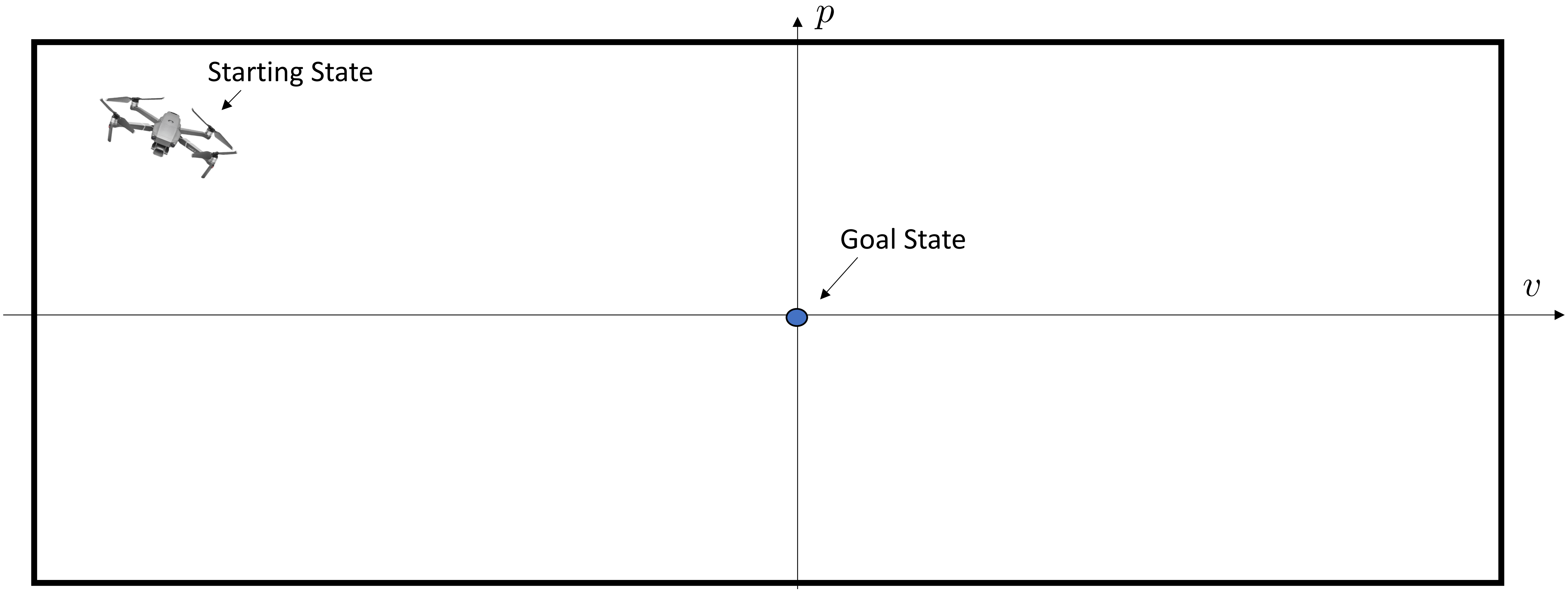
$$\begin{bmatrix} -5 \\ -5 \\ \boxed{-0.5} \end{bmatrix} \leq \begin{bmatrix} p_k \\ v_k \\ \boxed{a_k} \end{bmatrix} \leq \begin{bmatrix} 5 \\ 5 \\ \boxed{0.5} \end{bmatrix}$$

Limited actuation!

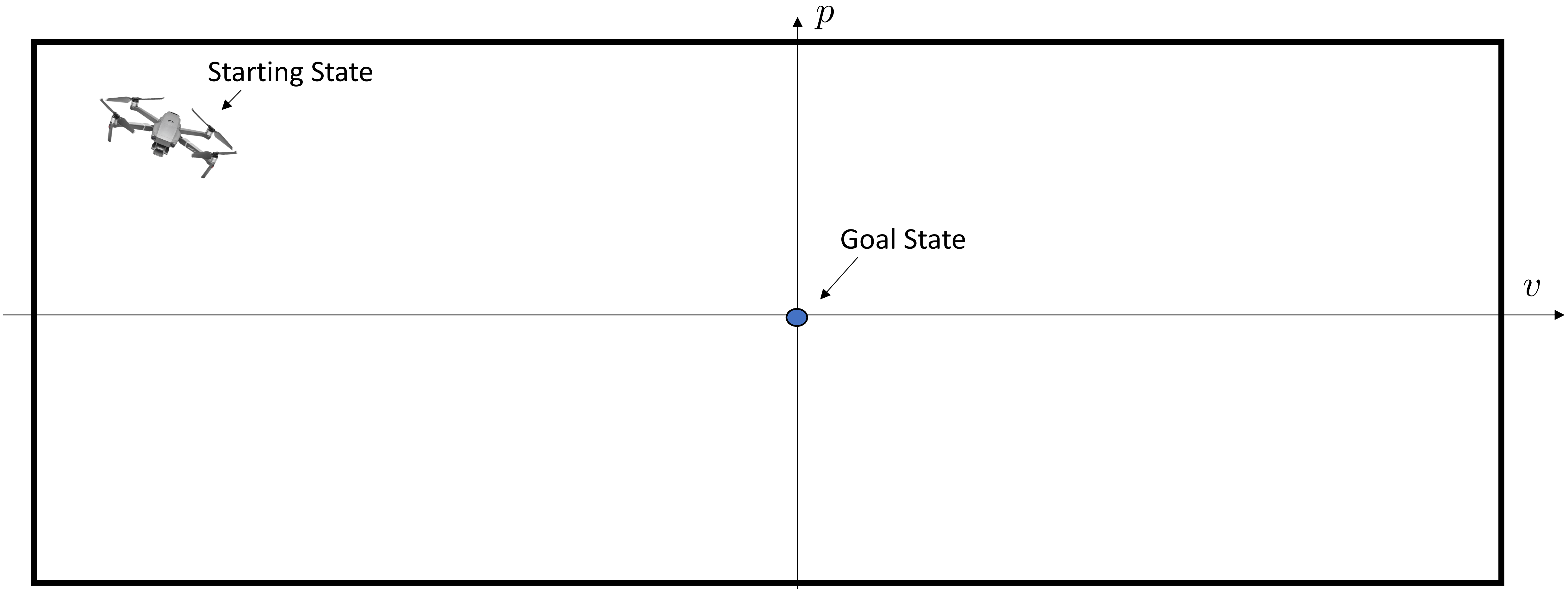
Iterative Tasks – Drone Example



Iterative Tasks – Drone Example

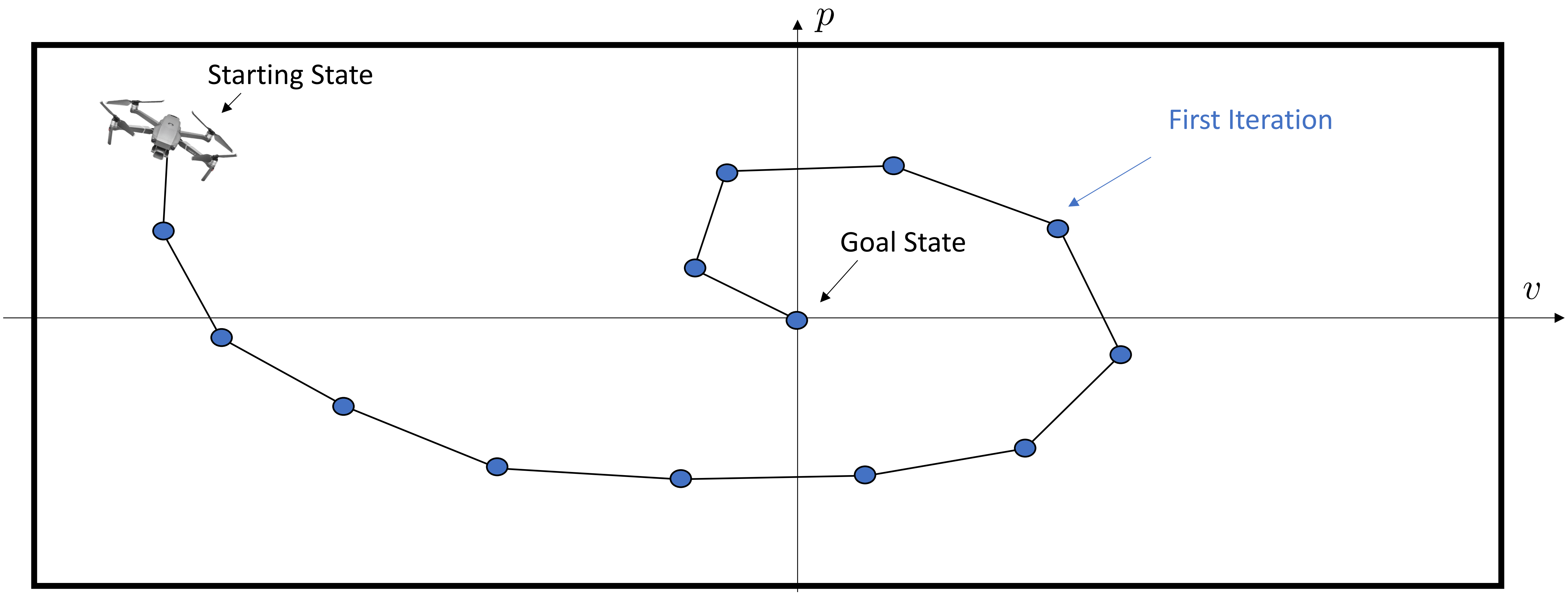


Iterative Tasks – Drone Example



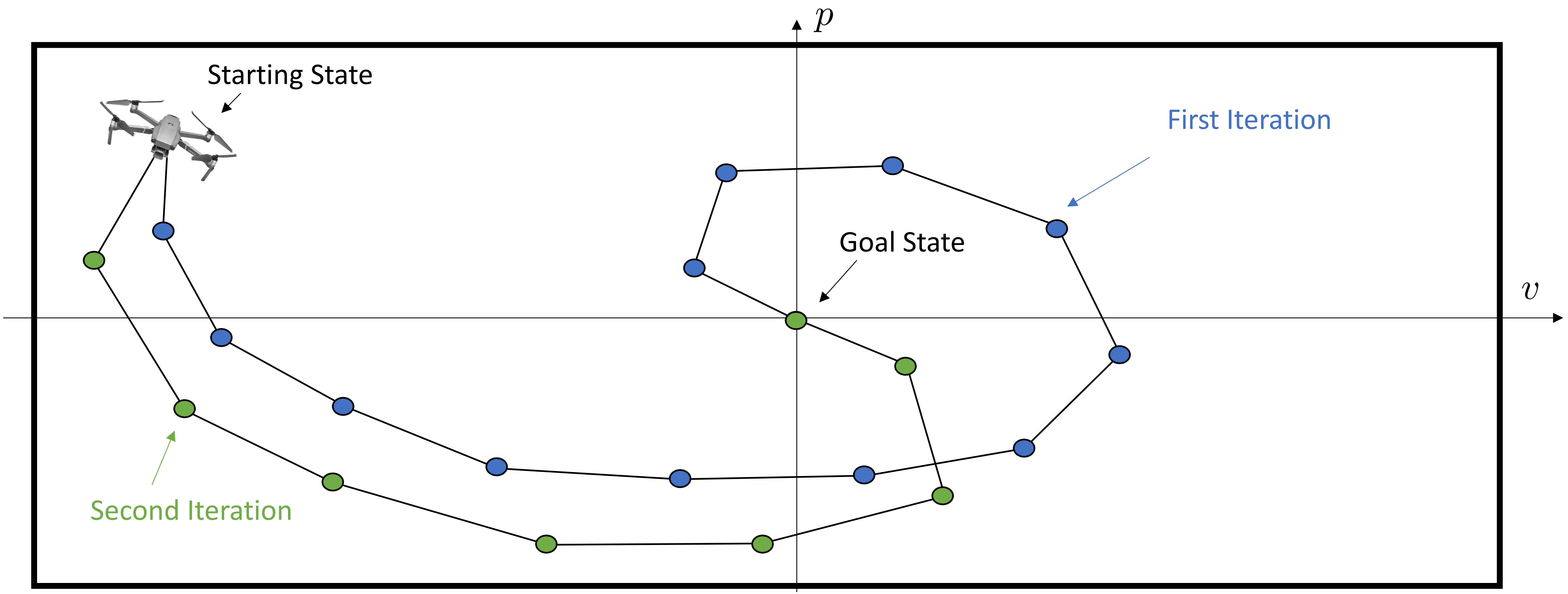
- Iteration $=$ one execution of the task

Iterative Tasks – Drone Example



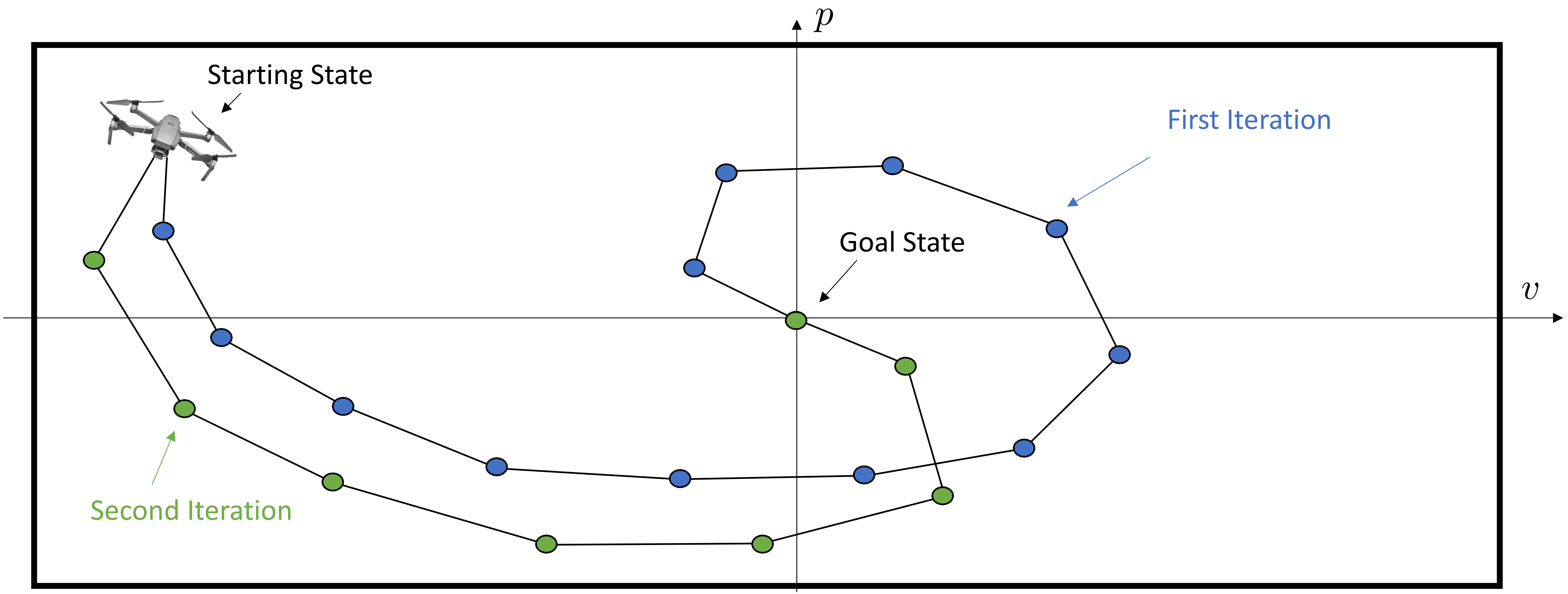
- Iteration $=$ one execution of the task

Iterative Tasks – Drone Example



- Iteration $=$ one execution of the task

Iterative Tasks – Drone Example



- Iteration $=$ one execution of the task
- Objective: Drive the drone <u>optimally</u> from the starting state to the goal state

Learning Model Predictive Control (LMPC)

Exploit historical data

Learning Model Predictive Control (LMPC) – Key Idea

Given $j-1$ trajectories, we define the following optimization problem:

Learning Model Predictive Control (LMPC) – Key Idea

Given $j-1$ trajectories, we define the following optimization problem:

$$J(x(t)) = \min_{u_0,\dots,u_{N-1}} \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + V^{j-1}(x_N)$$

Value Function

Learning Model Predictive Control (LMPC) – Key Idea

Given $j-1$ trajectories, we define the following optimization problem:

$$\begin{aligned} J(x(t)) = \min_{u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + V^{j-1}(x_N) \\ \text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\ & x_0 = x(t), \end{aligned}$$

Value Function

Prediction Model

Learning Model Predictive Control (LMPC) – Key Idea

Given $j-1$ trajectories, we define the following optimization problem:

$$
\begin{aligned}
J(x(t)) = \min_{u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + V^{j-1}(x_N) \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\
& x_0 = x(t), \\
& x_k \in \mathcal{X}, \;\; u_k \in \mathcal{U}, \\
& x_N \in \mathcal{SS}^{j-1}, \\
& \forall k \in [0, \cdots, N-1]
\end{aligned}
$$

Value Function

Prediction Model

Safe Set

Learning Model Predictive Control (LMPC) – Key Idea

Given $j-1$ trajectories, we define the following optimization problem:

$$
\begin{aligned}
J(x(t)) = \min_{u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k \right) + V^{j-1}(x_N) \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\
& x_0 = x(t), \\
& x_k \in \mathcal{X}, \; u_k \in \mathcal{U}, \\
& x_N \in \mathcal{SS}^{j-1}, \\
& \forall k \in [0, \cdots, N-1]
\end{aligned}
$$

Value Function

Prediction Model

Safe Set

Then apply to the system the control input $u(t) = u_0^*$

Learning Model Predictive Control (LMPC) – Key Idea

Given $j-1$ trajectories, we define the following optimization problem:

$$
\begin{aligned}
J(x(t)) = \min_{u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + \underbrace{V^{j-1}(x_N)}_{\text{Value Function}} \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\
& x_0 = x(t), \\
& x_k \in \mathcal{X},\ u_k \in \mathcal{U}, \\
& x_N \in \underbrace{\mathcal{SS}^{j-1}}_{\text{Safe Set}}, \\
& \forall k \in [0, \cdots, N-1]
\end{aligned}
$$

Then apply to the system the control input $u(t) = u_0^*$

Learning Model Predictive Control (LMPC) – Key Idea

Given $j-1$ trajectories, we define the following optimization problem:

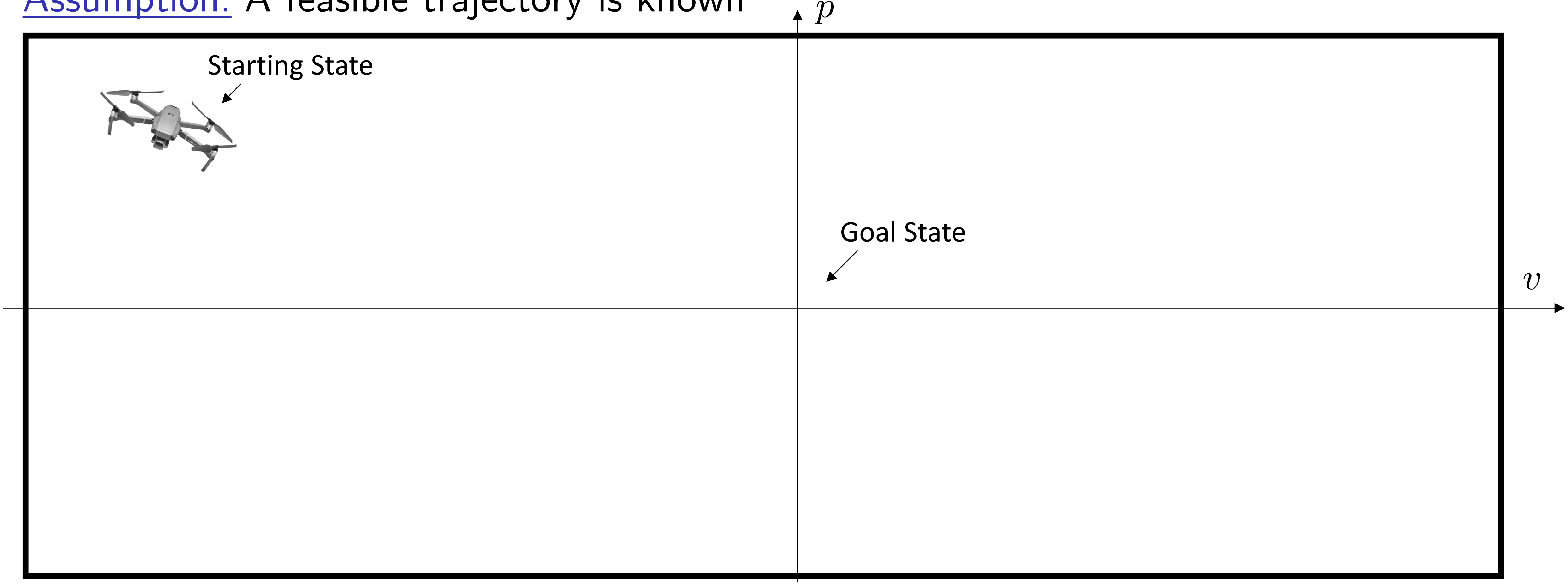
$$
\begin{aligned}
J(x(t)) = \min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + V^{j-1}(x_N) \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\
& x_0 = x(t), \\
& x_k \in \mathcal{X},\ u_k \in \mathcal{U}, \\
& x_N \in \ \mathcal{SS}^{j-1}, \\
& \forall k \in [0, \cdots, N-1]
\end{aligned}
$$

Safe Set

Then apply to the system the control input $u(t) = u_0^*$

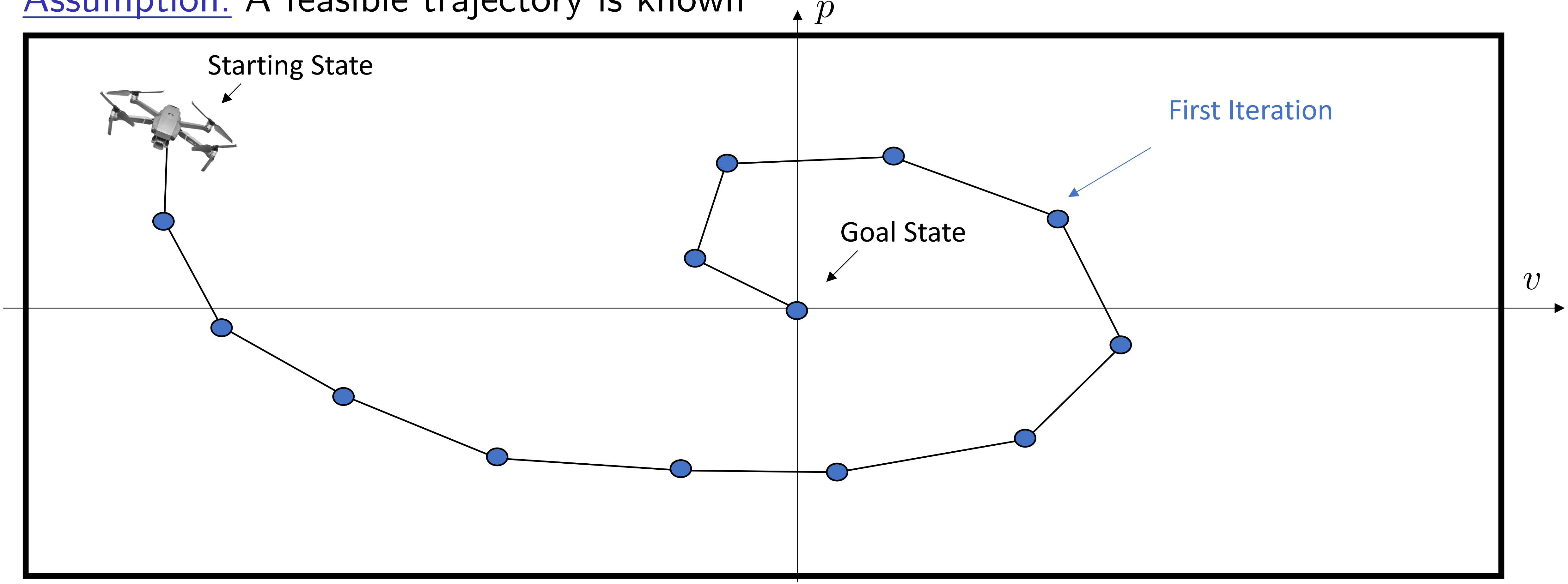
Iteration 1

Assumption: A feasible trajectory is known



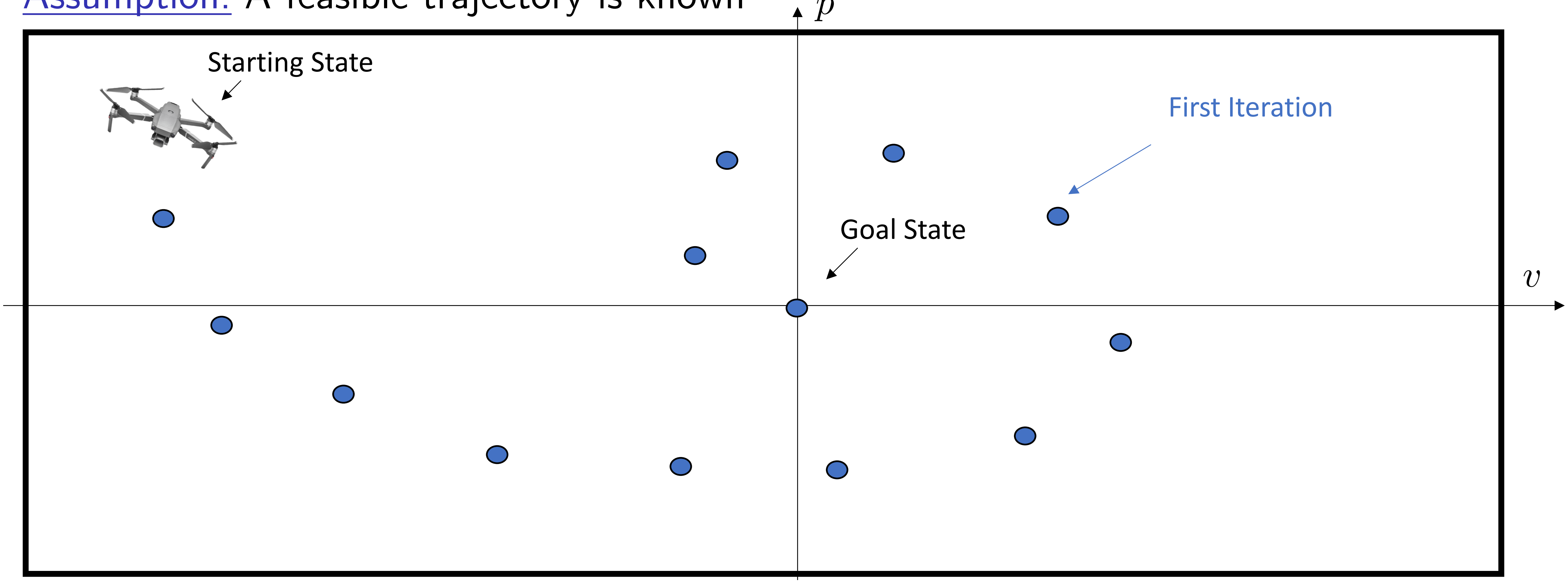
Iteration 1

Assumption: A feasible trajectory is known



Iteration 1

Assumption: A feasible trajectory is known

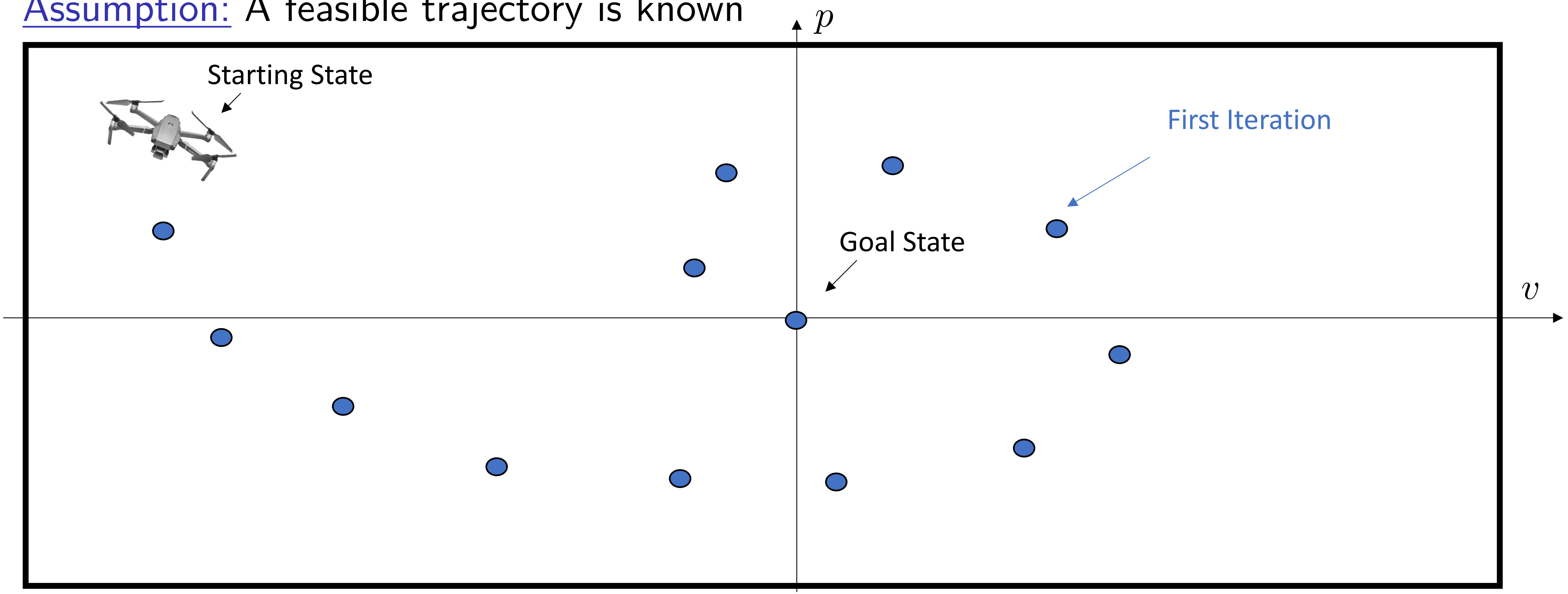


Definition: Sampled Safe Set

$$\mathcal{SS}^1 = \{\text{Stored Data}\}$$

Iteration 1

Assumption: A feasible trajectory is known



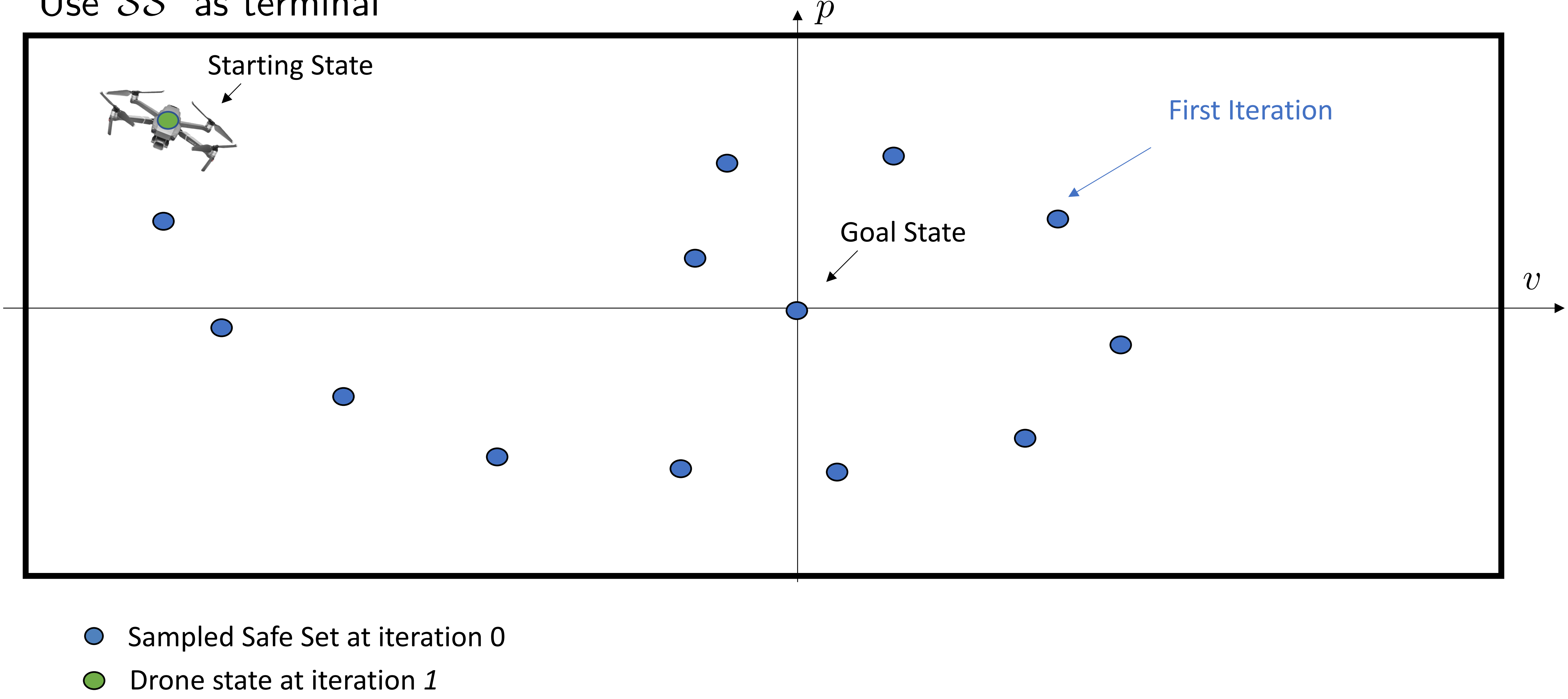
Definition: Sampled Safe Set

$$\mathcal{SS}^1 = \{\text{Stored Data}\}$$

Set of states from which the task can be completed!

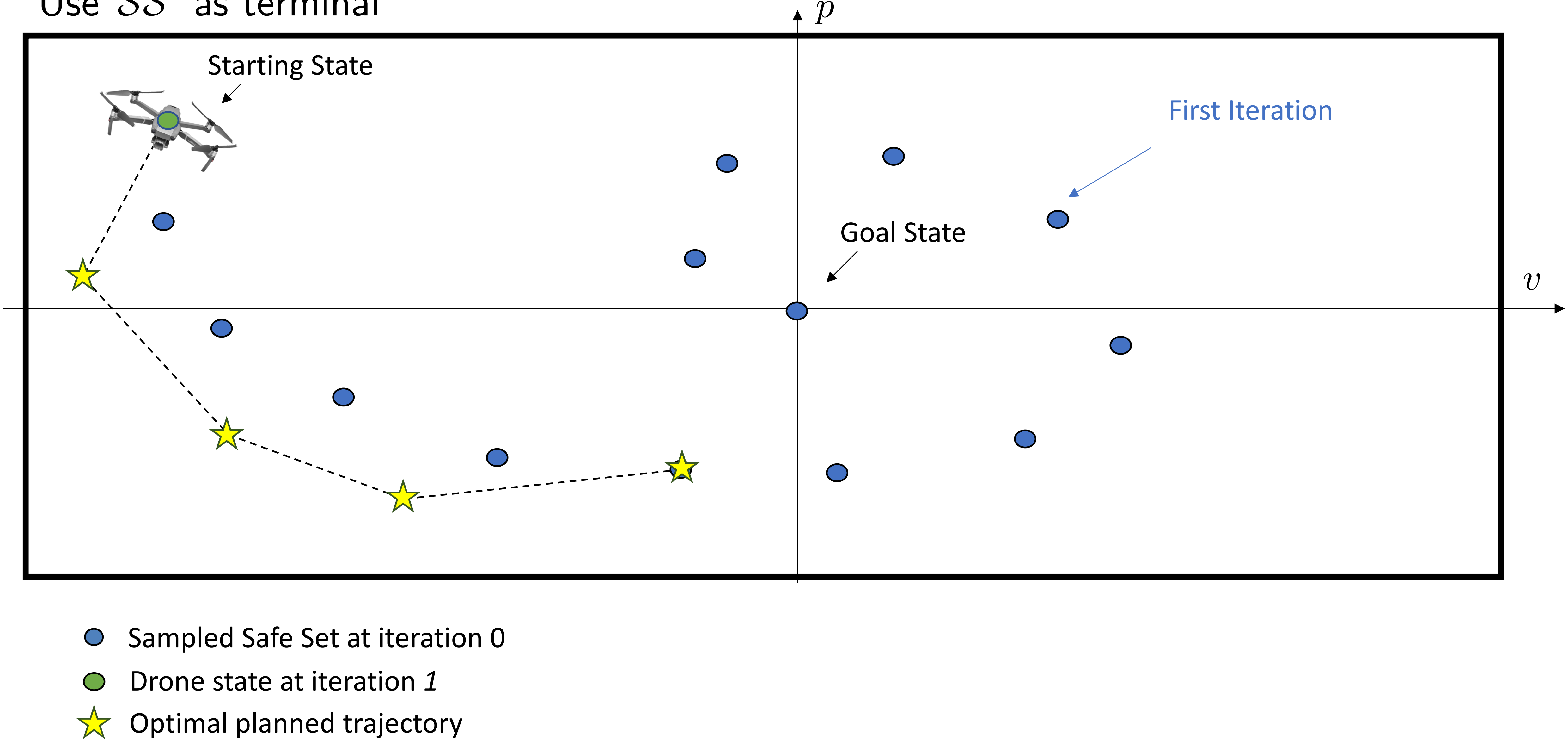
Iteration 2, Step 0

Use $\mathcal{SS}^1$ as terminal



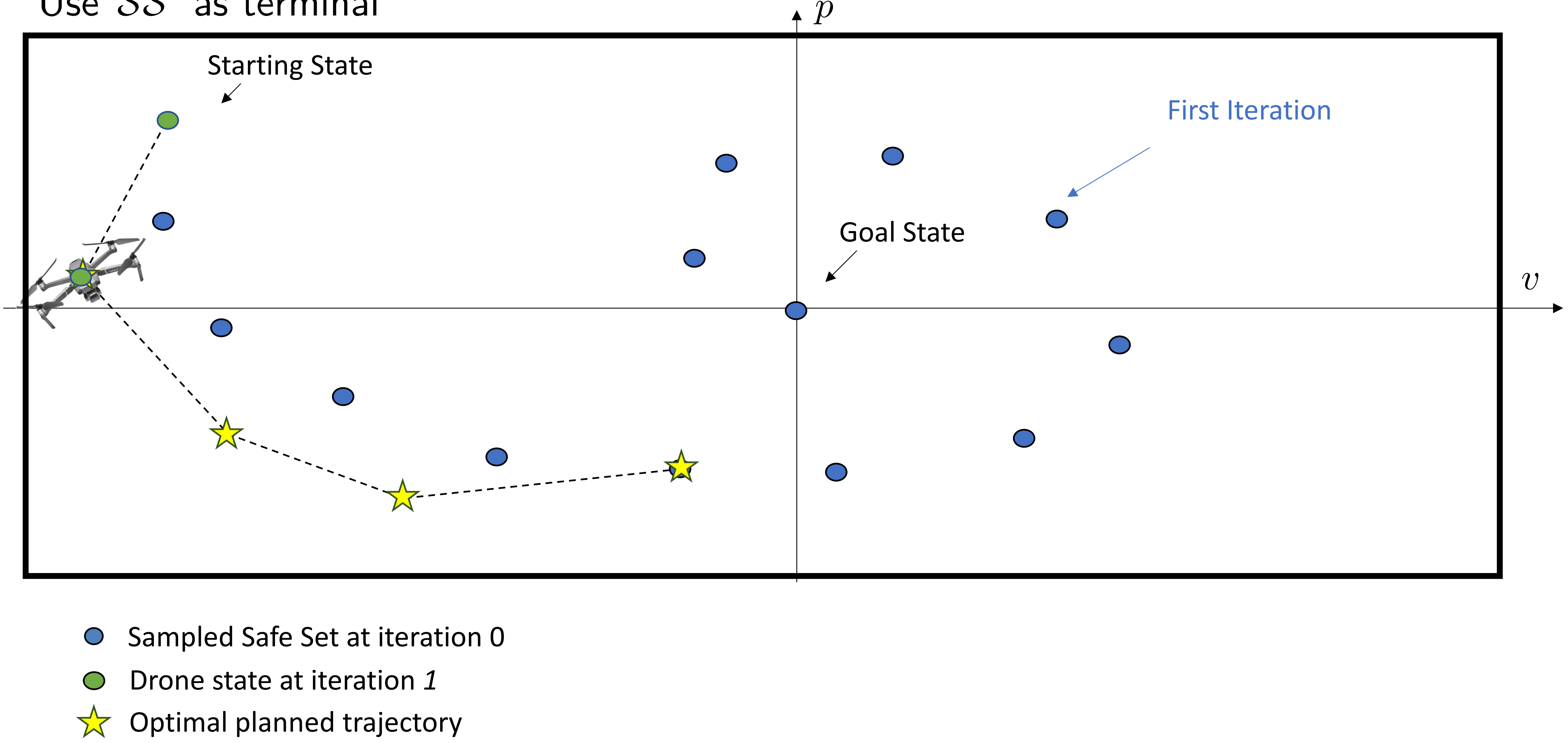
Iteration 2, Step 0

Use $\mathcal{SS}^1$ as terminal



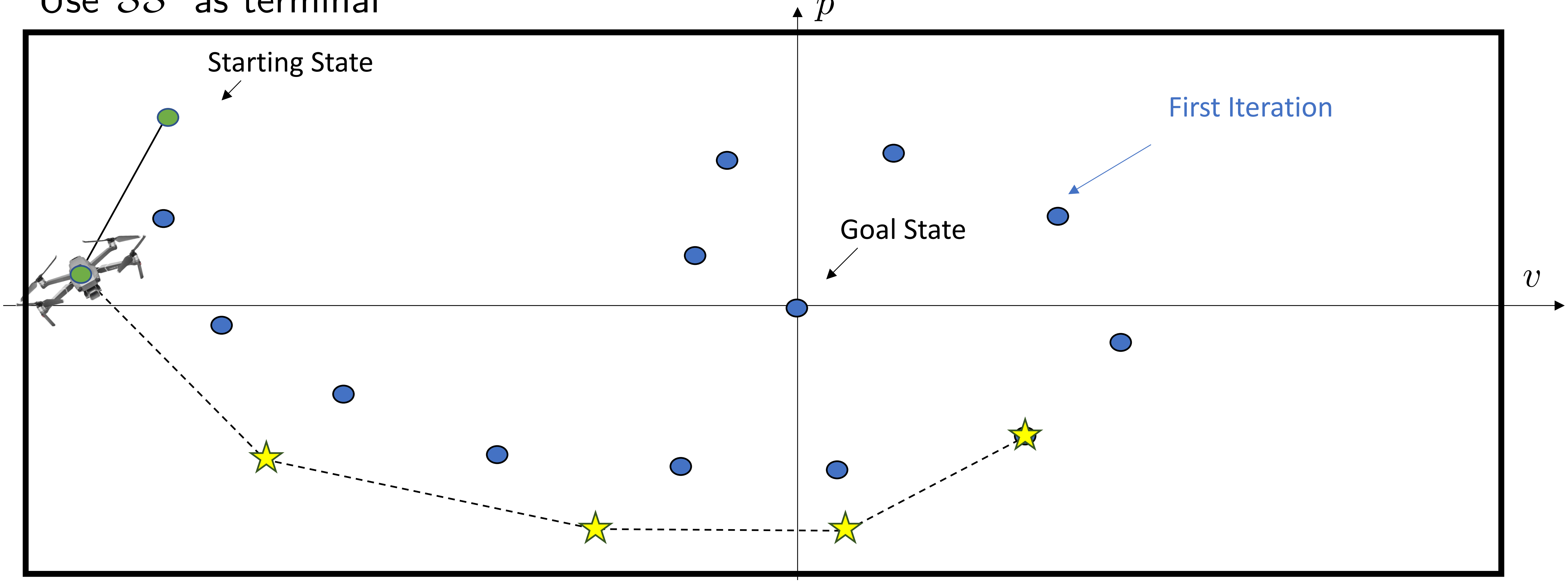
Iteration 2, Step 1

Use $\mathcal{SS}^1$ as terminal



Iteration 2, Step 1

Use $\mathcal{SS}^1$ as terminal



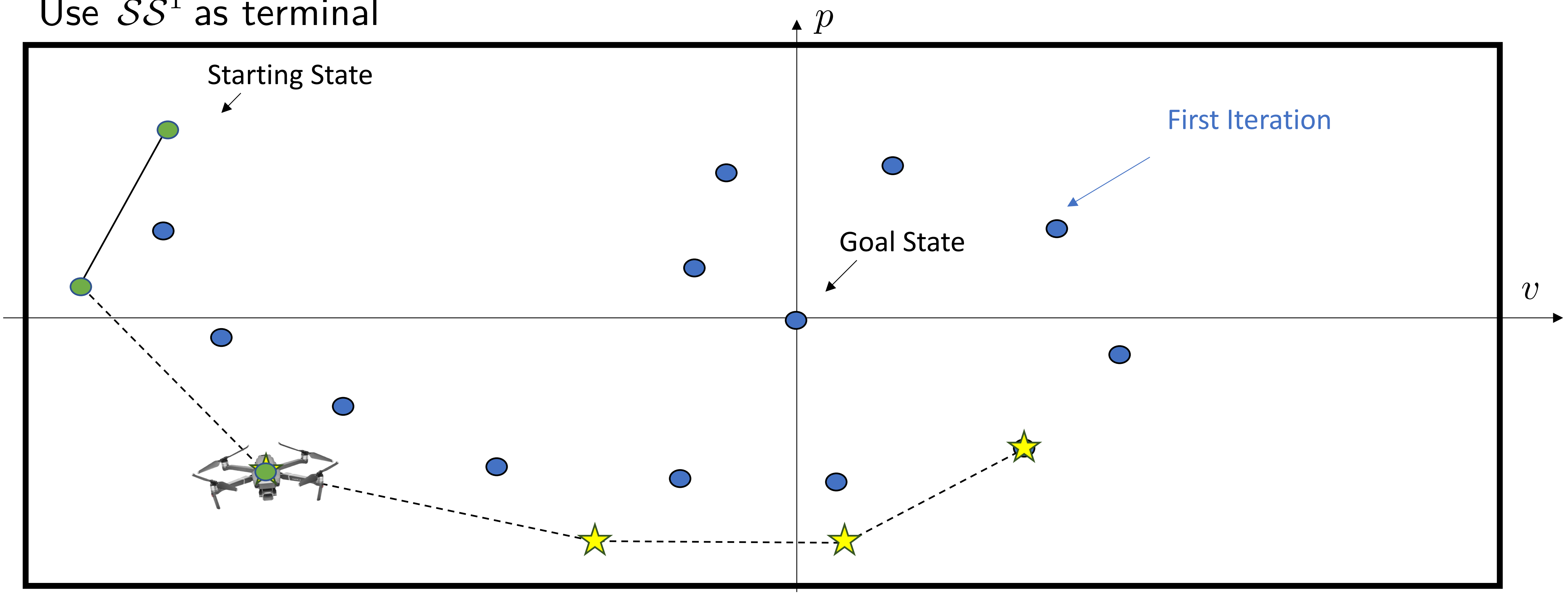
Sampled Safe Set at iteration 0

Drone state at iteration *1*

Optimal planned trajectory

Iteration 2, Step 2

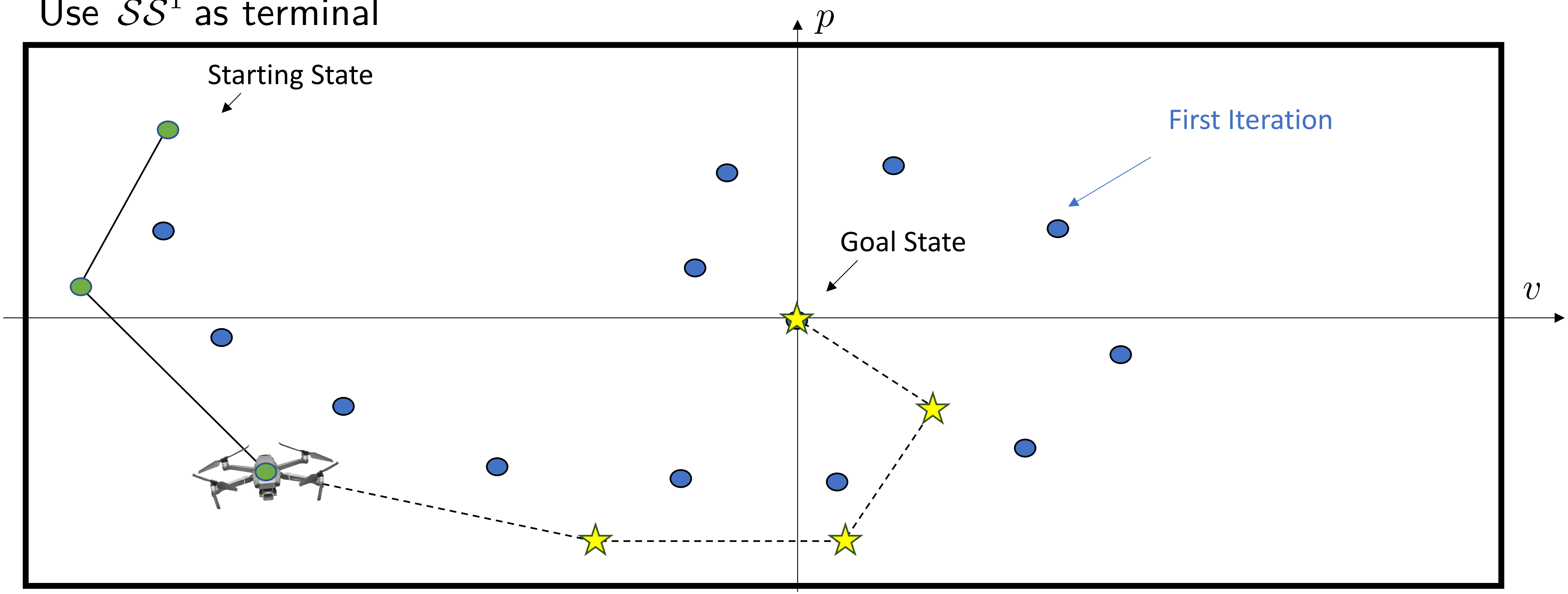
Use $\mathcal{SS}^1$ as terminal



- Sampled Safe Set at iteration 0
- Drone state at iteration *1*
- Optimal planned trajectory

Iteration 2, Step 2

Use $\mathcal{SS}^1$ as terminal



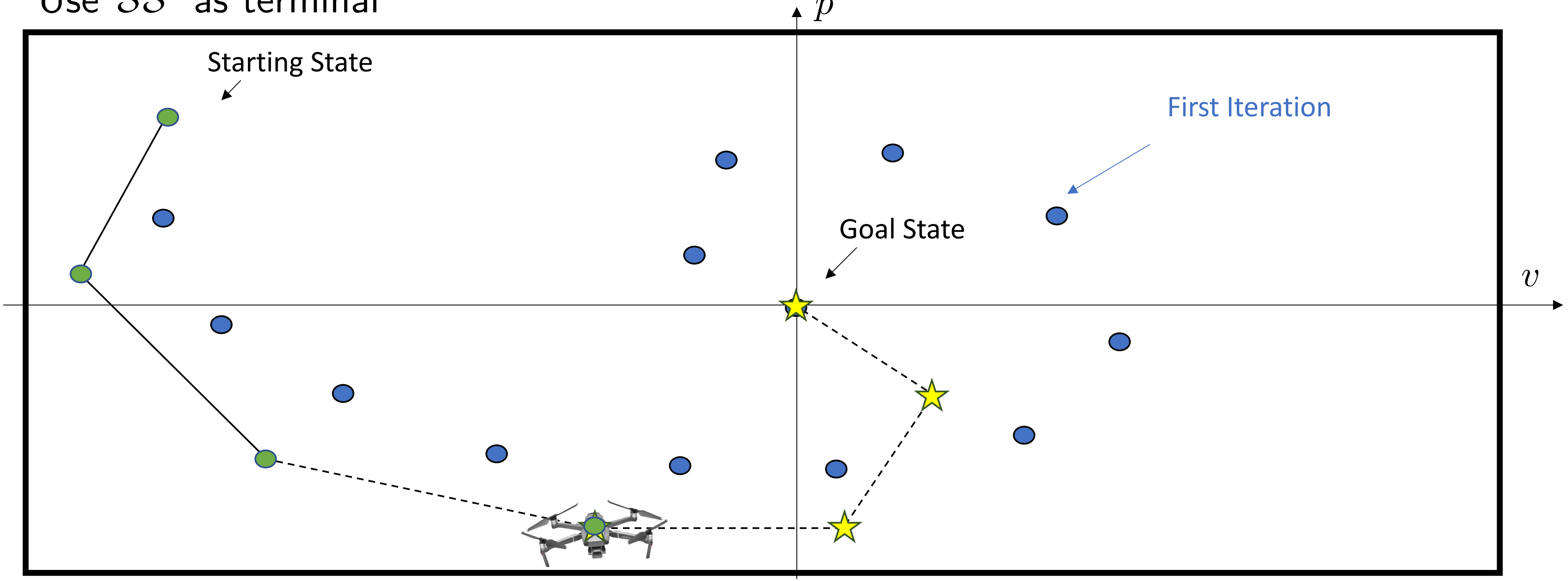
Sampled Safe Set at iteration 0

Drone state at iteration *1*

Optimal planned trajectory

Iteration 2, Step 3

Use $\mathcal{SS}^1$ as terminal



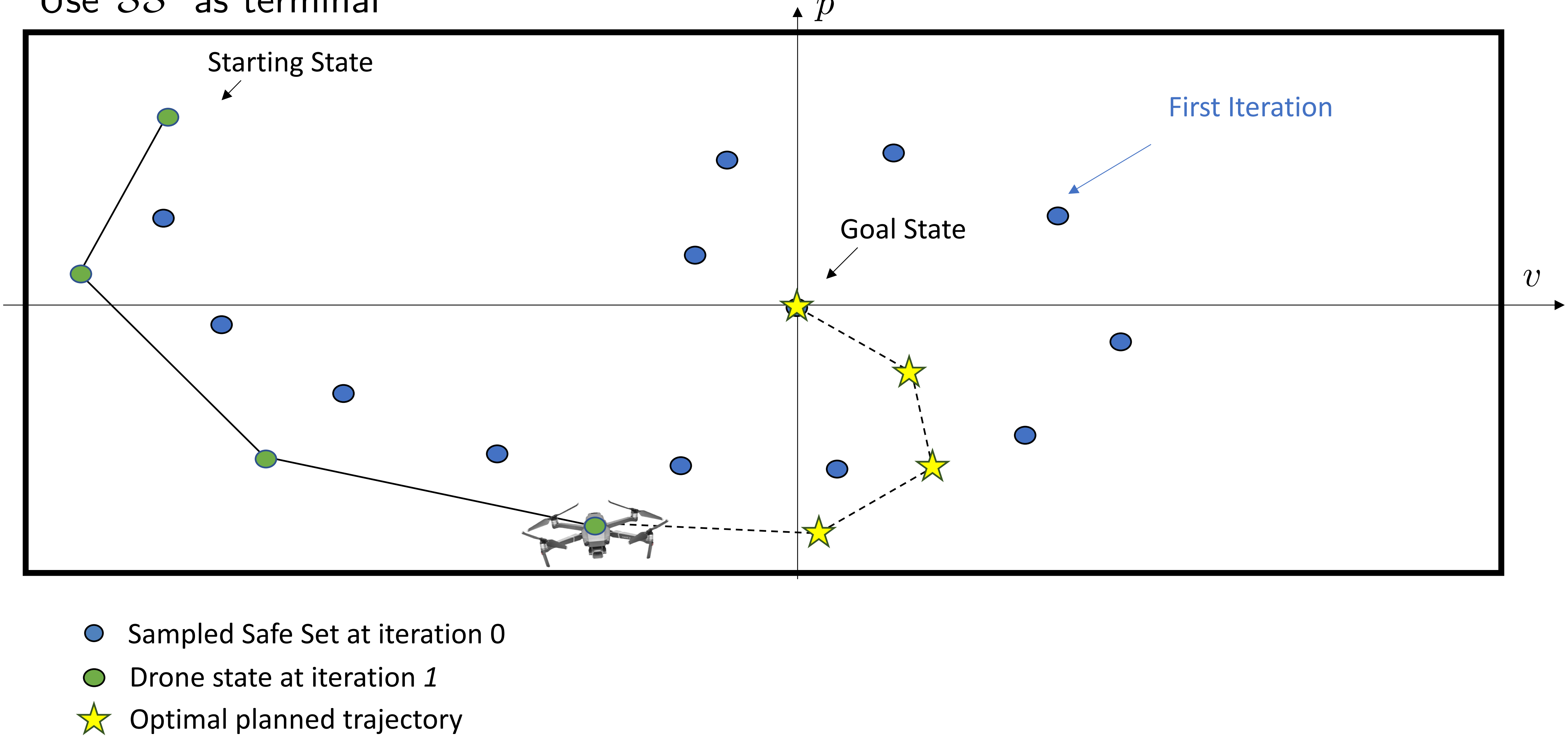
Sampled Safe Set at iteration 0

Drone state at iteration *1*

Optimal planned trajectory

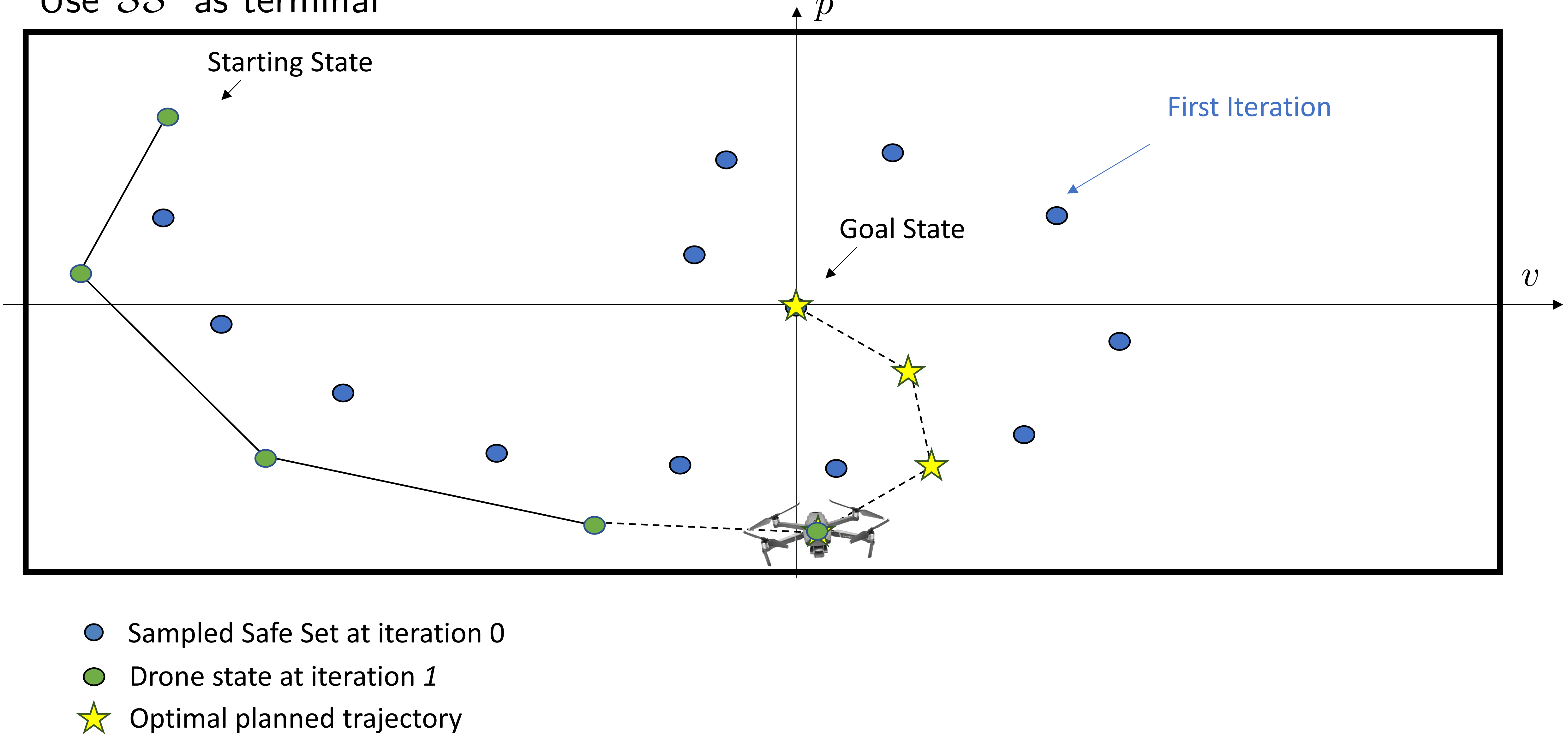
Iteration 2, Step 3

Use $\mathcal{SS}^1$ as terminal



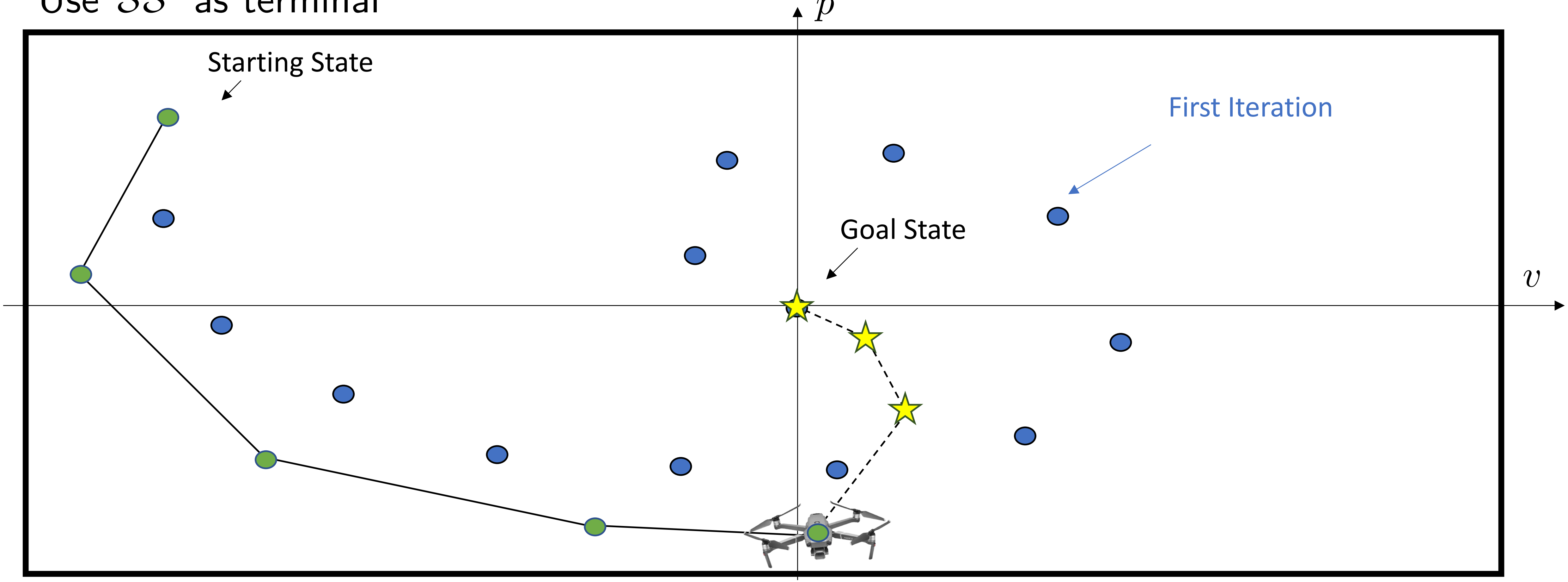
Iteration 2, Step 4

Use $\mathcal{SS}^1$ as terminal



Iteration 2, Step 4

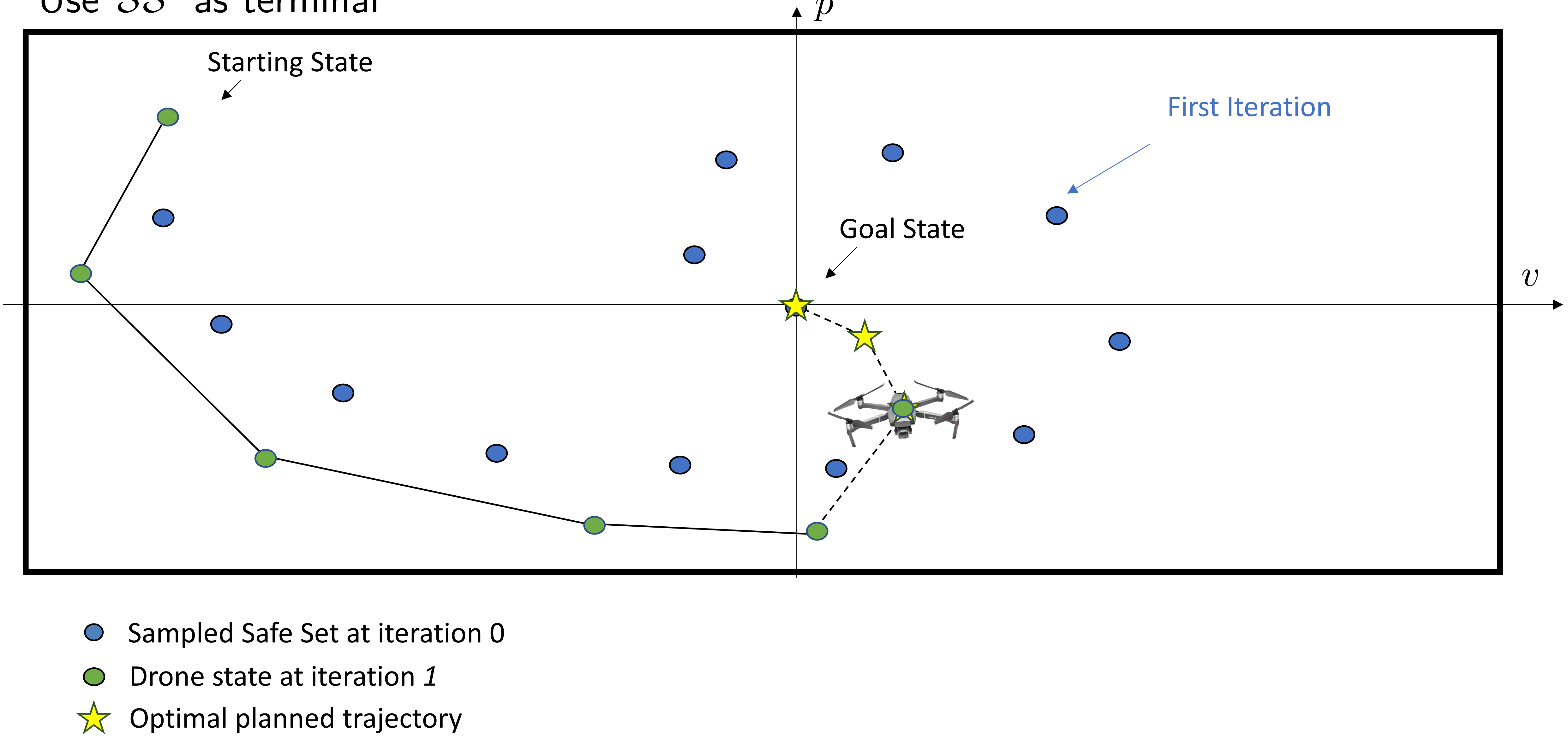
Use $\mathcal{SS}^1$ as terminal



- Sampled Safe Set at iteration 0
- Drone state at iteration *1*
- Optimal planned trajectory

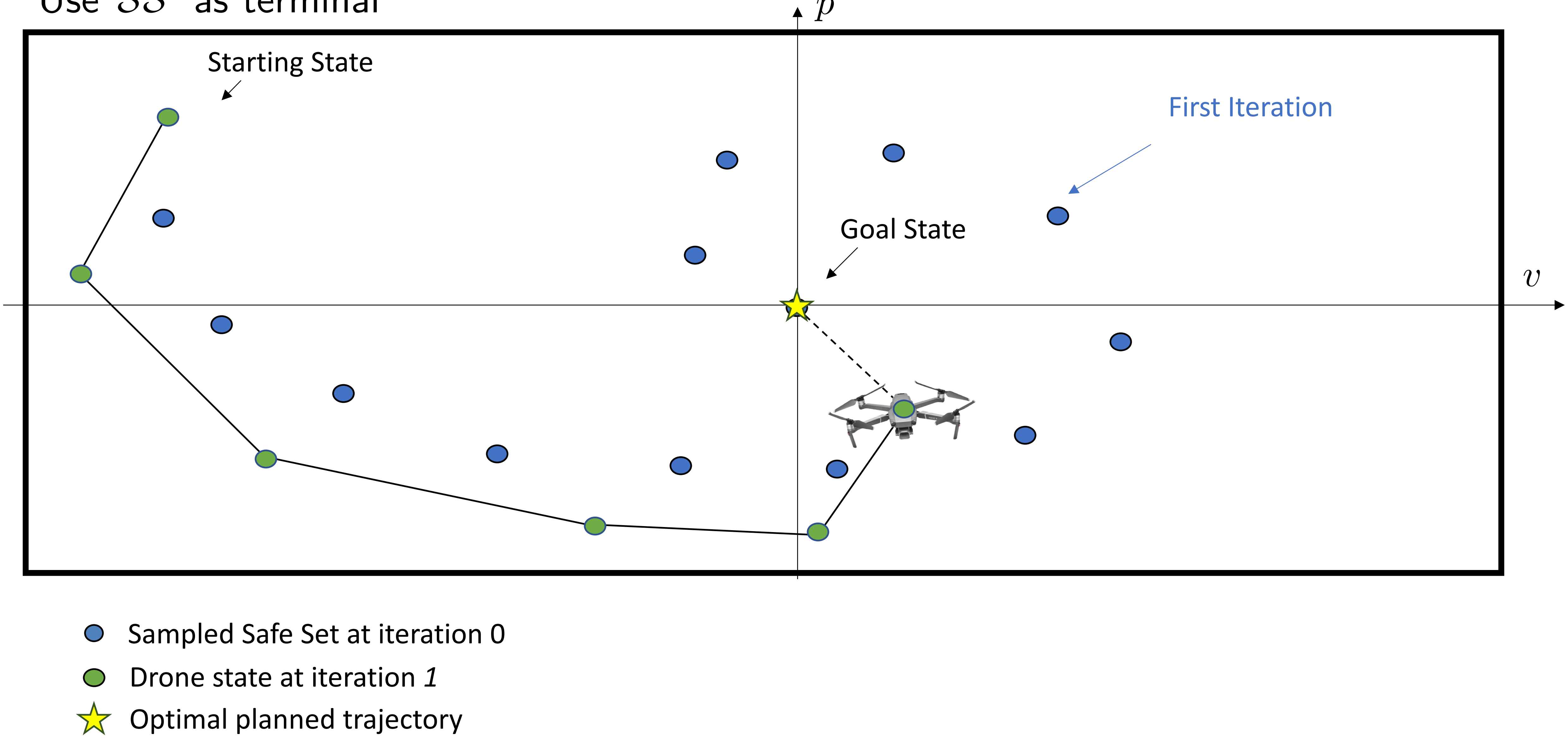
Iteration 2, Step 5

Use $\mathcal{SS}^1$ as terminal



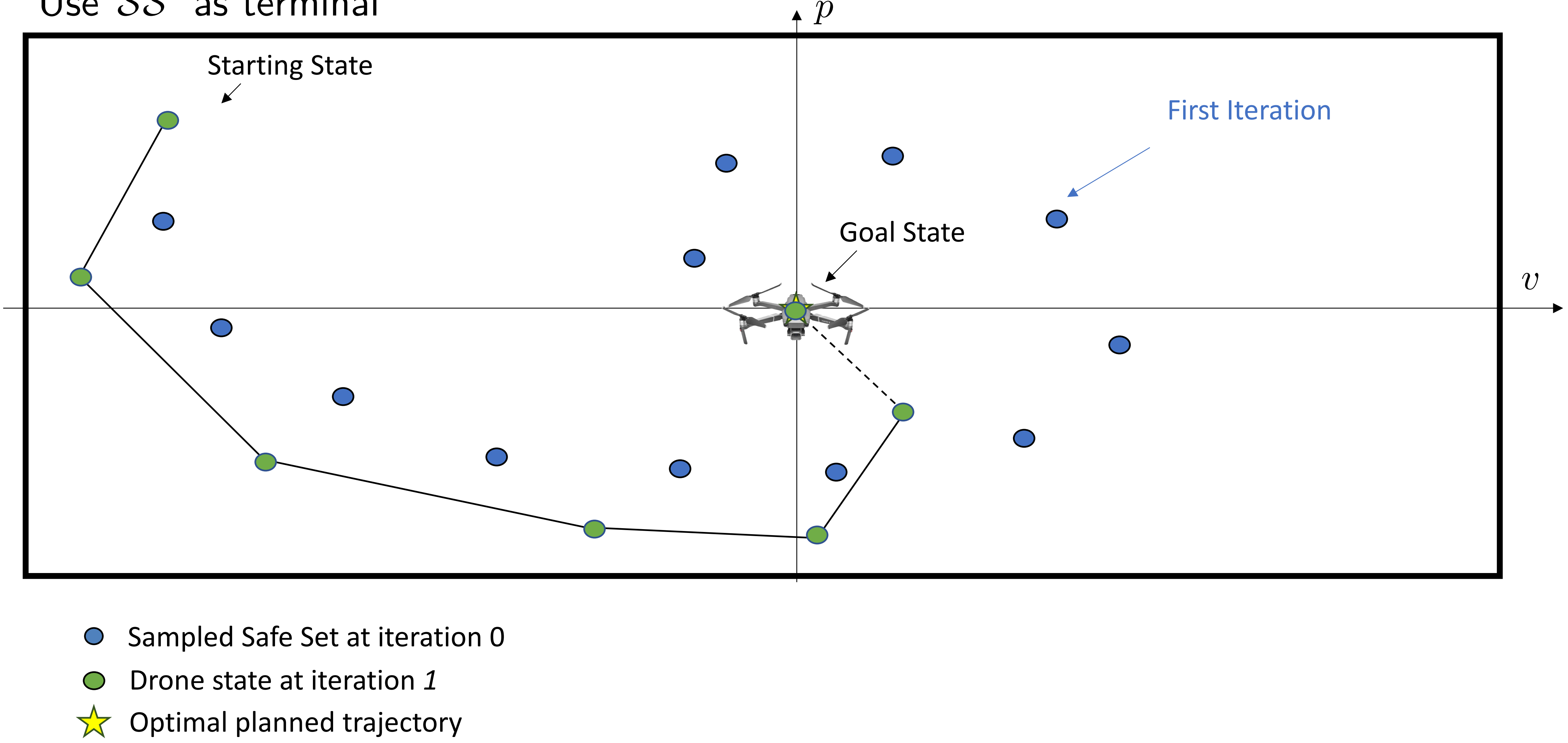
Iteration 2, Step 5

Use $\mathcal{SS}^1$ as terminal



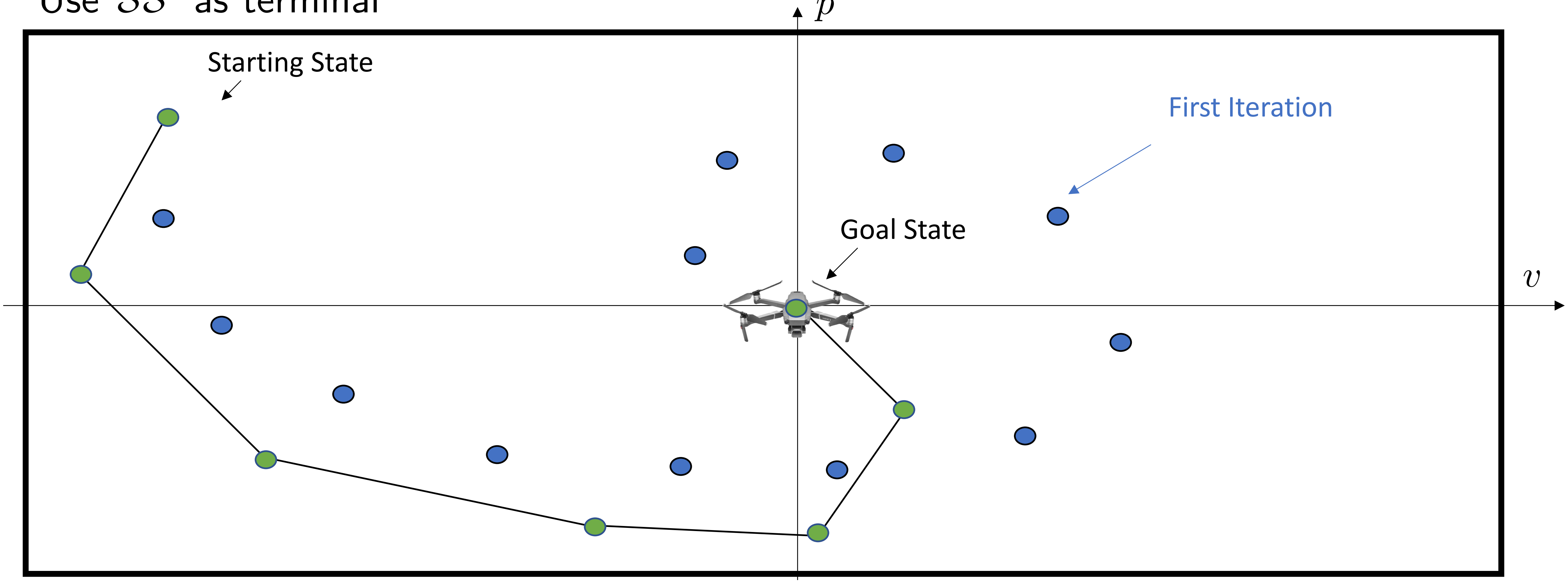
Iteration 2, Step 5

Use $\mathcal{SS}^1$ as terminal



Iteration 2, Step 5

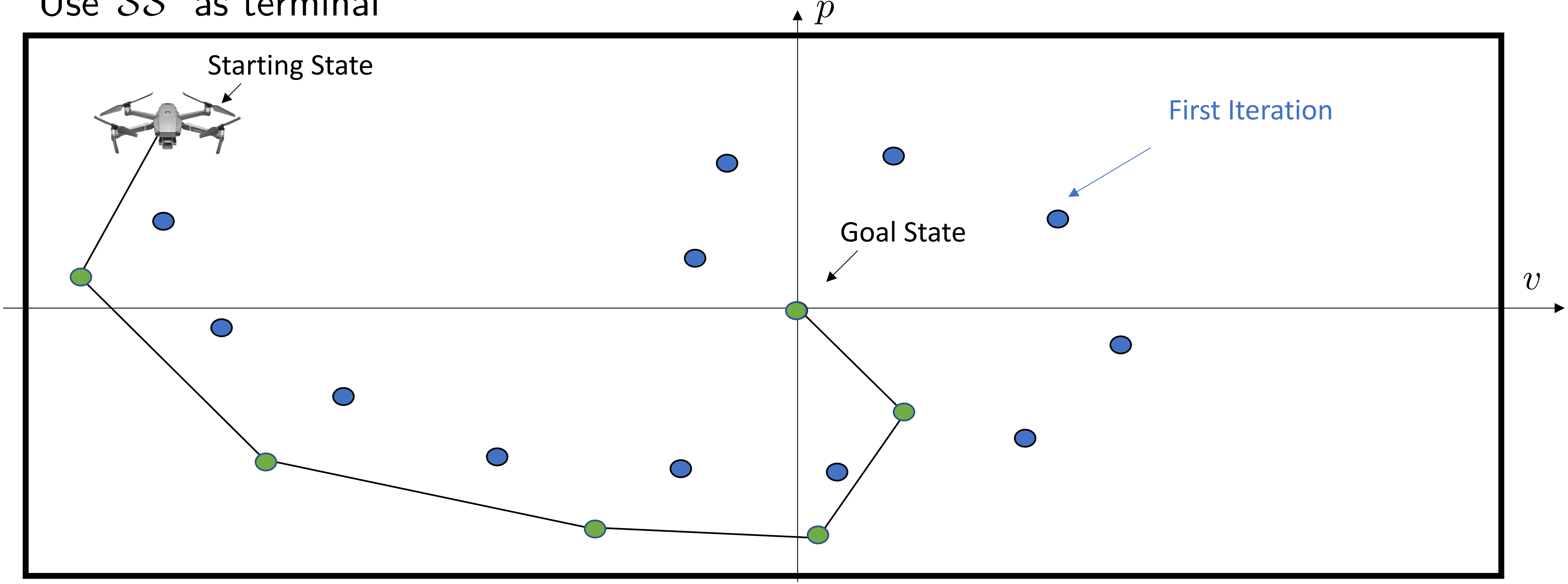
Use $\mathcal{SS}^1$ as terminal



Sampled Safe Set at iteration 0
Drone state at iteration *1*
Optimal planned trajectory

Iteration 2, Step 5

Use $\mathcal{SS}^1$ as terminal



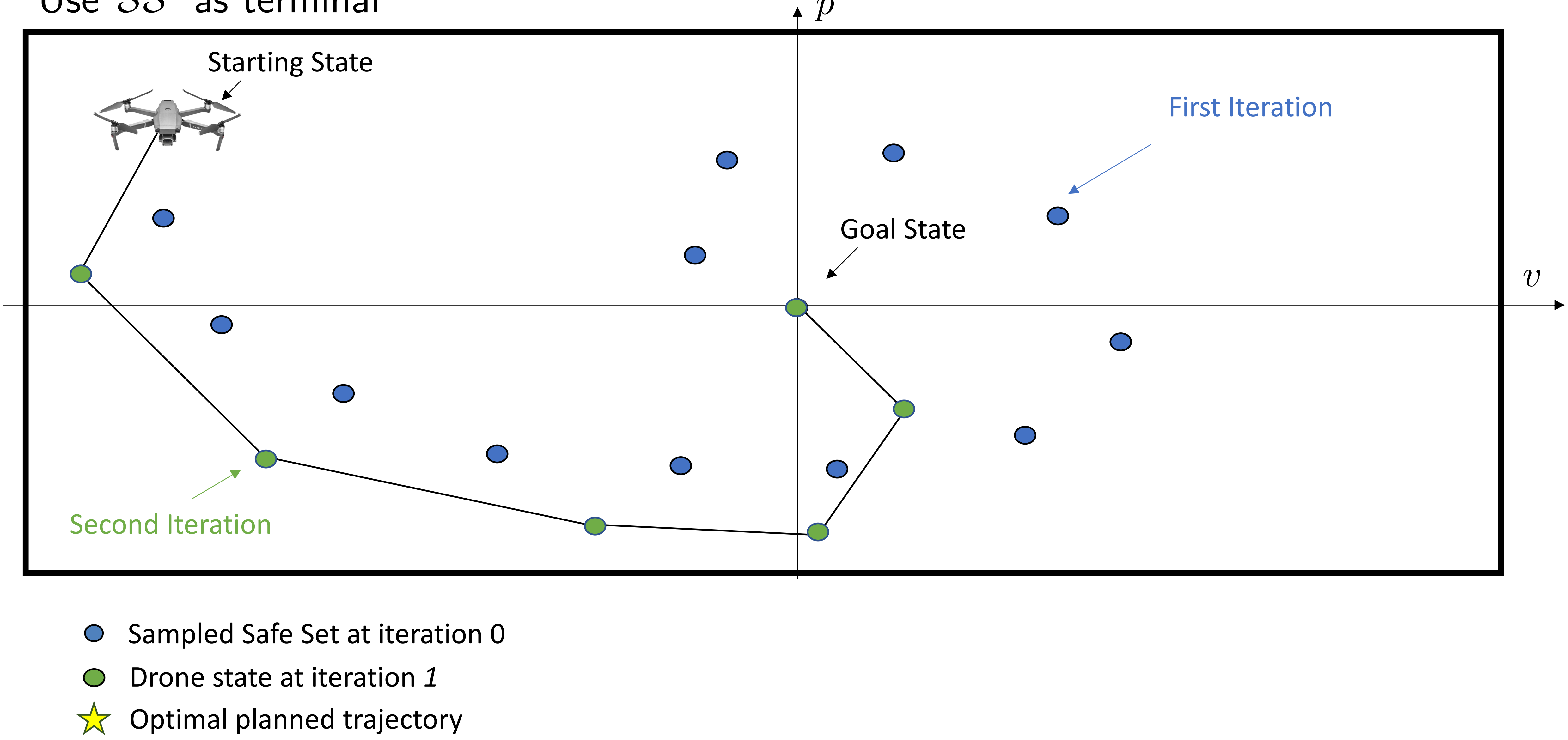
Sampled Safe Set at iteration 0

Drone state at iteration *1*

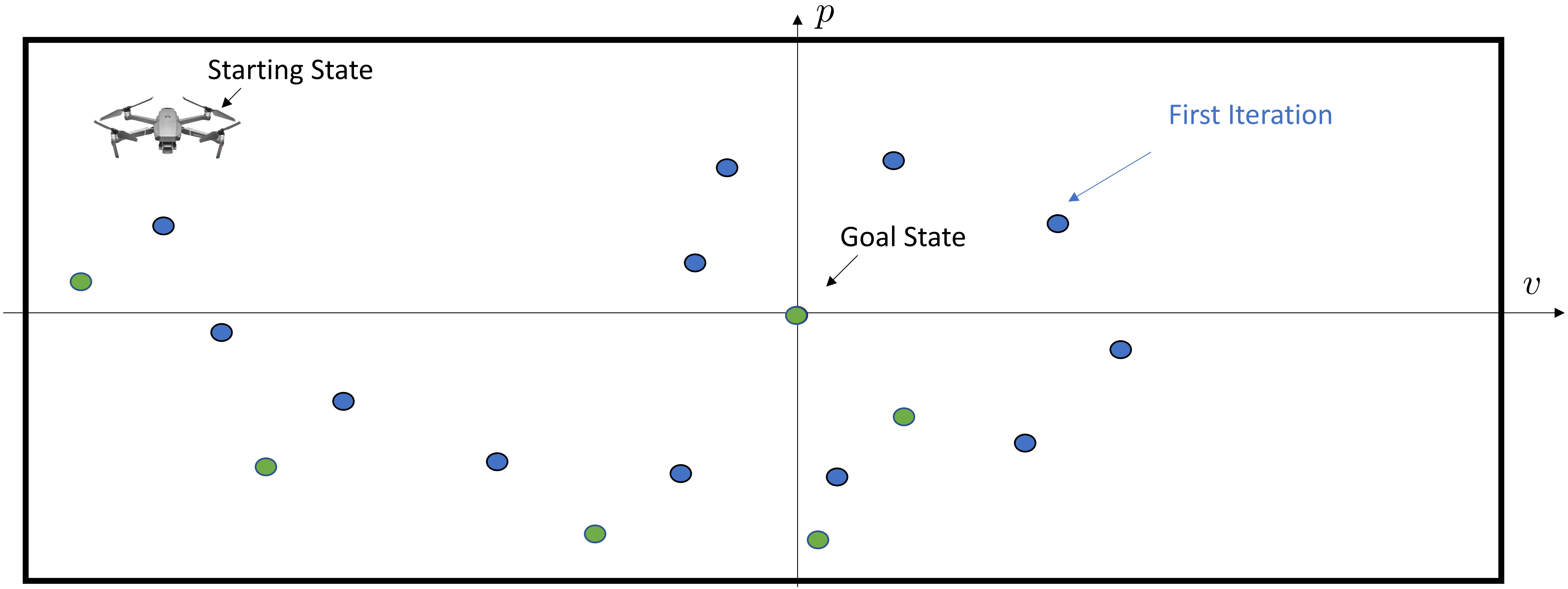
Optimal planned trajectory

Iteration 2, Step 5

Use $\mathcal{SS}^1$ as terminal



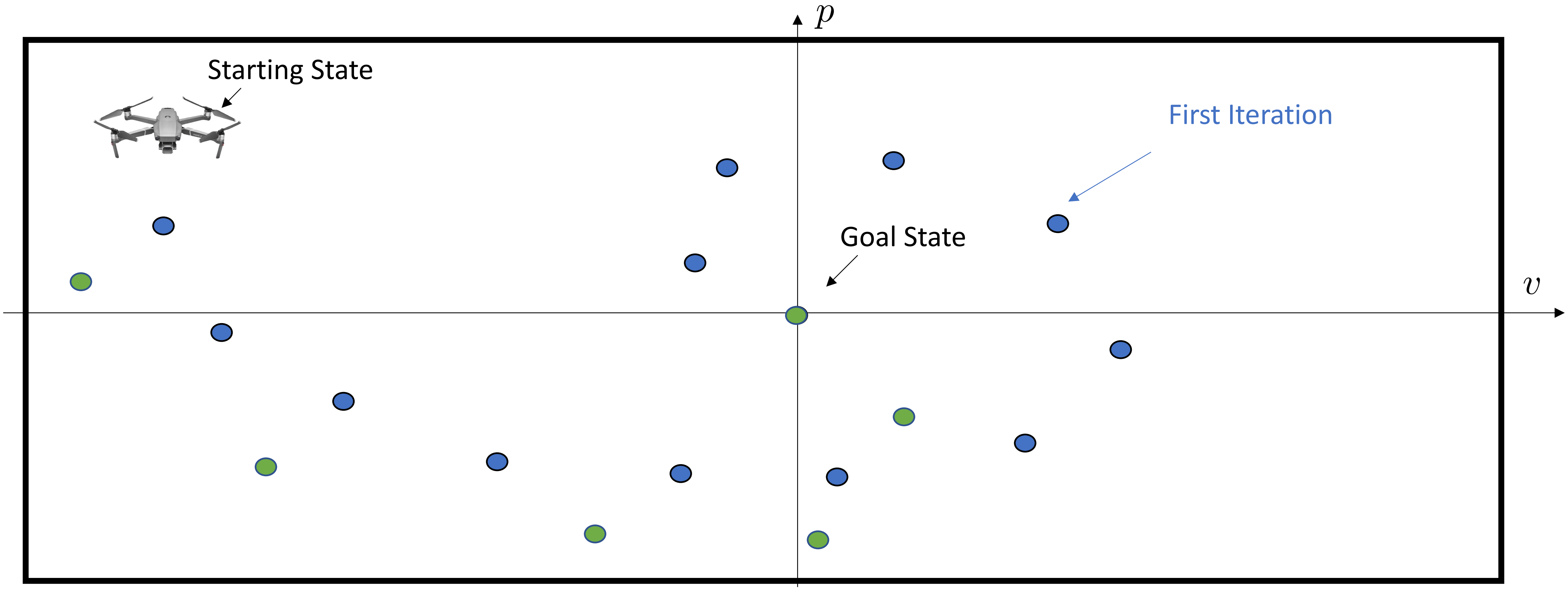
Iteration 3



Definition: Sampled Safe Set

$$\mathcal{SS}^j = \{\text{Stored Data at all iterations}\}$$

Iteration 3

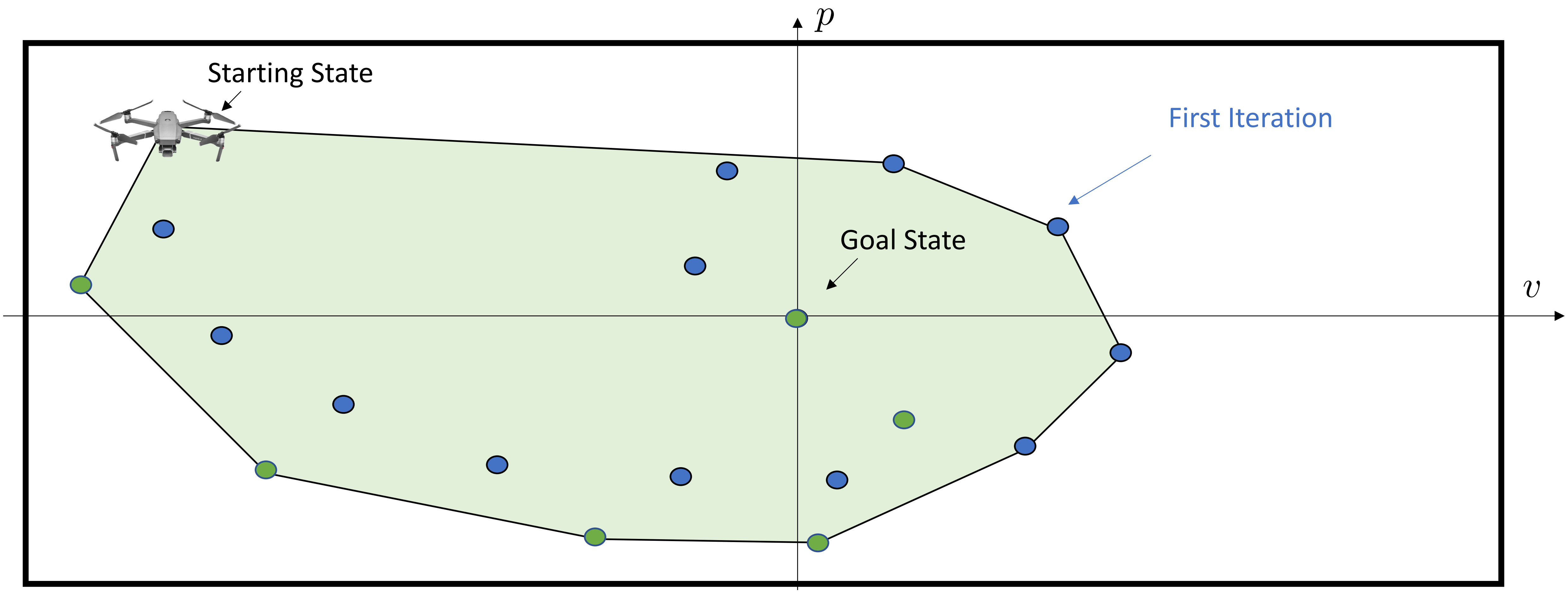


Definition: Sampled Safe Set

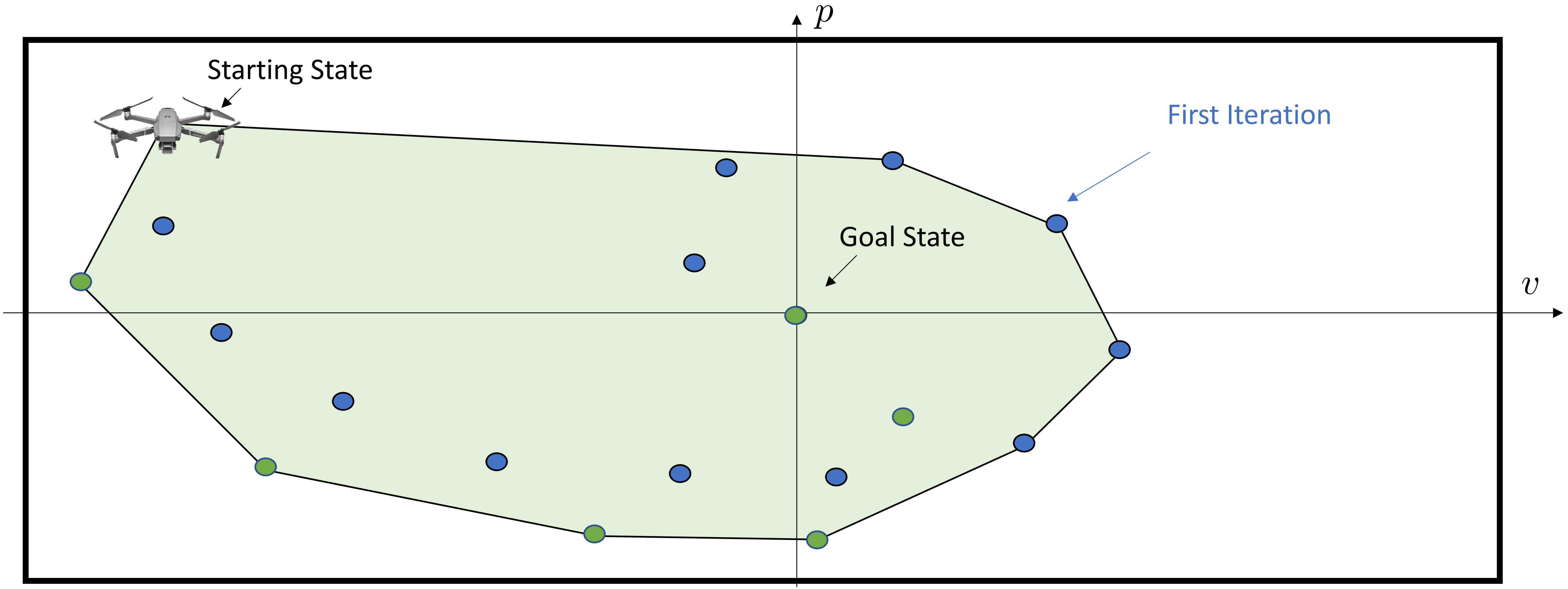
$$\mathcal{SS}^j = \{\text{Stored Data at all iterations}\}$$

Set of states from which the task can be completed!

Iteration 3



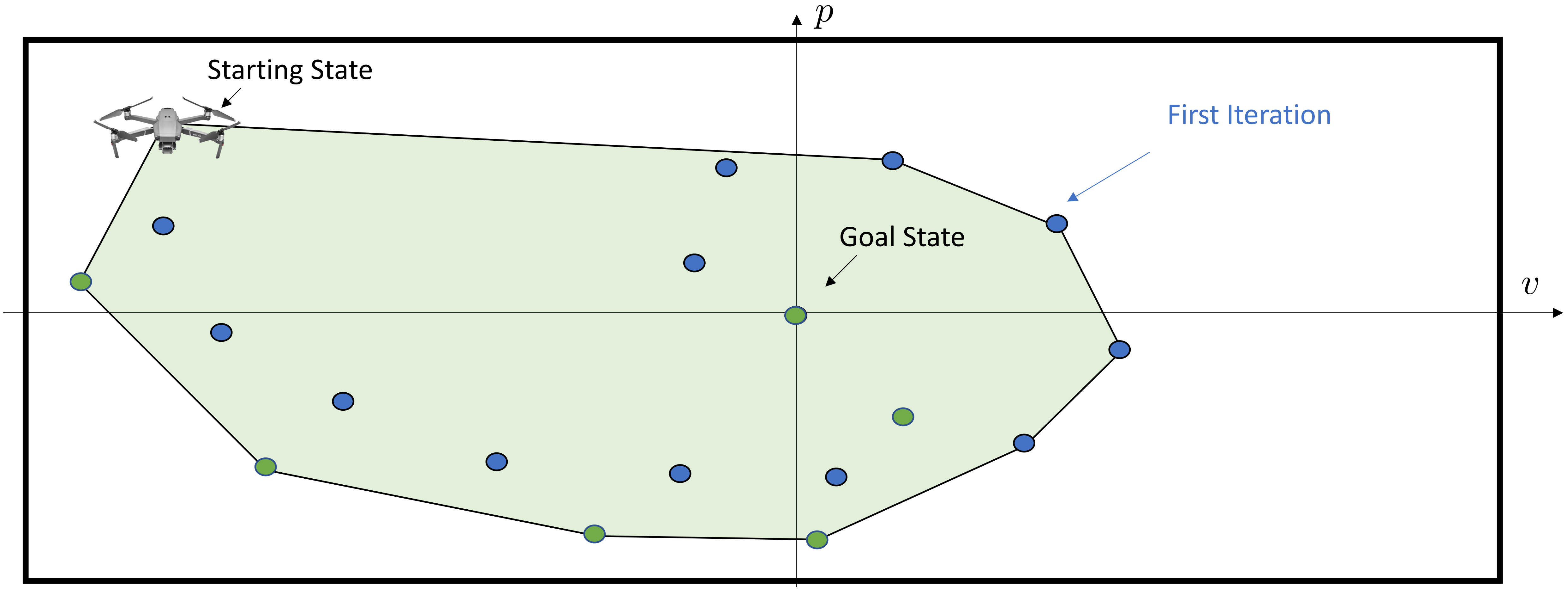
Iteration 3



Definition: Convex Safe Set

$$\mathcal{CS}^j = \text{Conv}(\{\text{Stored Data at all iterations}\})$$

Iteration 3



Definition: Convex Safe Set

$$\mathcal{CS}^j = \text{Conv}(\{\text{Stored Data at all iterations}\})$$

Set of states from which the task can be completed!

Learning Model Predictive Control (LMPC) – Key Idea

Given $j-1$ trajectories, we define the following optimization problem:

$$
\begin{aligned}
J(x(t)) = \min_{u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + V^{j-1}(x_N) \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\
& x_0 = x(t), \\
& x_k \in \mathcal{X}, \;\; u_k \in \mathcal{U}, \\
& x_N \in \; \mathcal{SS}^{j-1}, \\
& \forall k \in [0, \cdots, N-1]
\end{aligned}
$$

Safe Set

Then apply to the system the control input $u(t) = u_0^*$

Learning Model Predictive Control (LMPC) – Key Idea

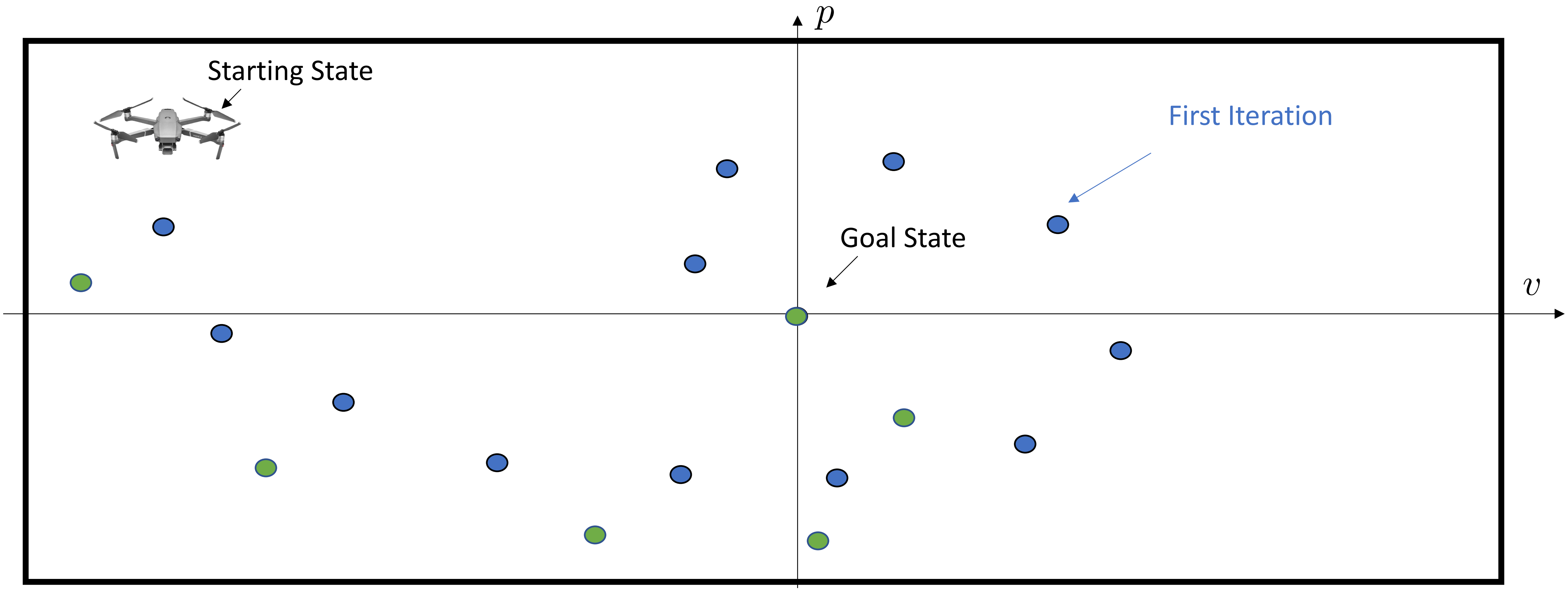
Given $j-1$ trajectories, we define the following optimization problem:

$$
\begin{aligned}
J(x(t)) = \min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + \boxed{V^{j-1}(x_N)} \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\
& x_0 = x(t), \\
& x_k \in \mathcal{X},\ u_k \in \mathcal{U}, \\
& x_N \in \ \mathcal{SS}^{j-1}, \\
& \forall k \in [0, \cdots, N-1]
\end{aligned}
$$

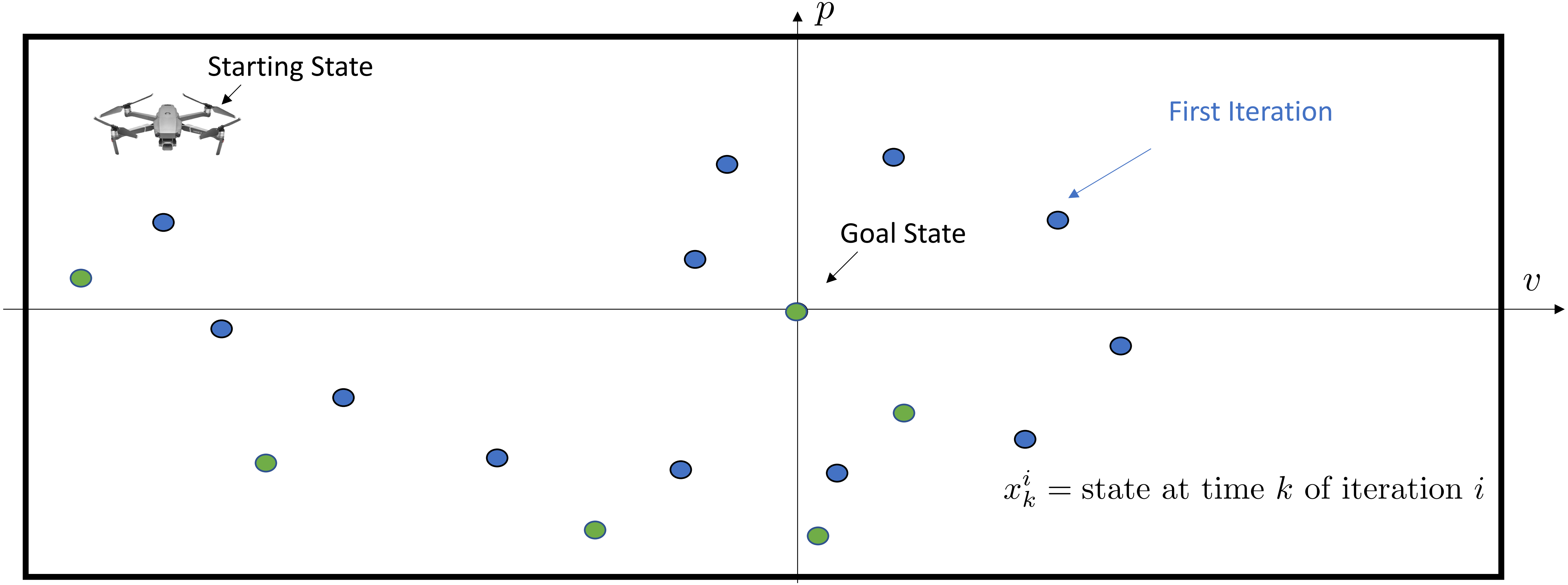
Value Function

Then apply to the system the control input $u(t) = u_0^*$

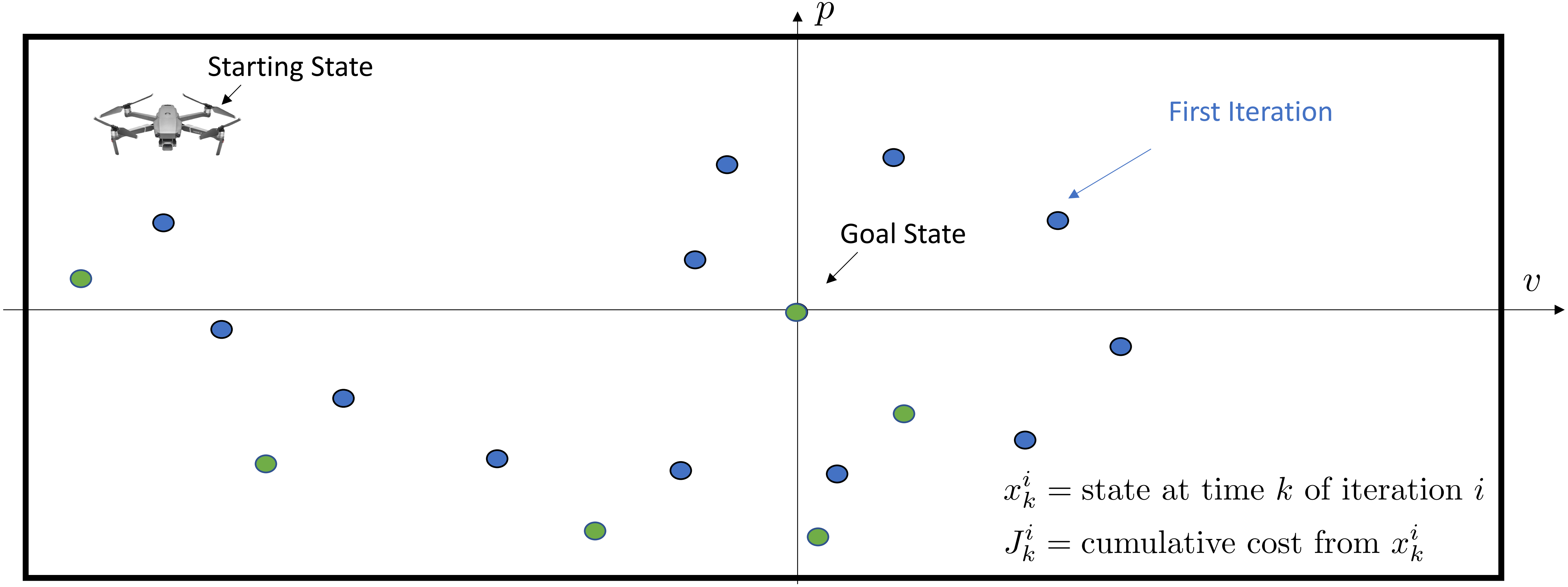
Value Function Estimation



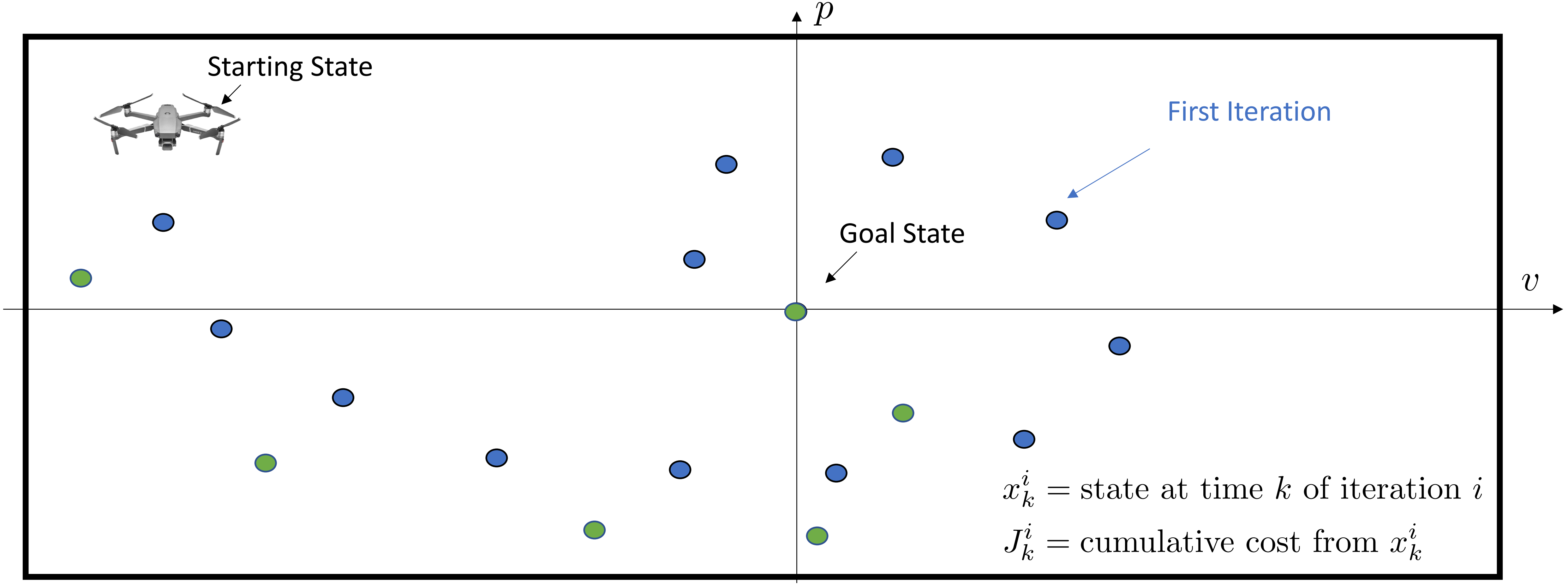
Value Function Estimation



Value Function Estimation



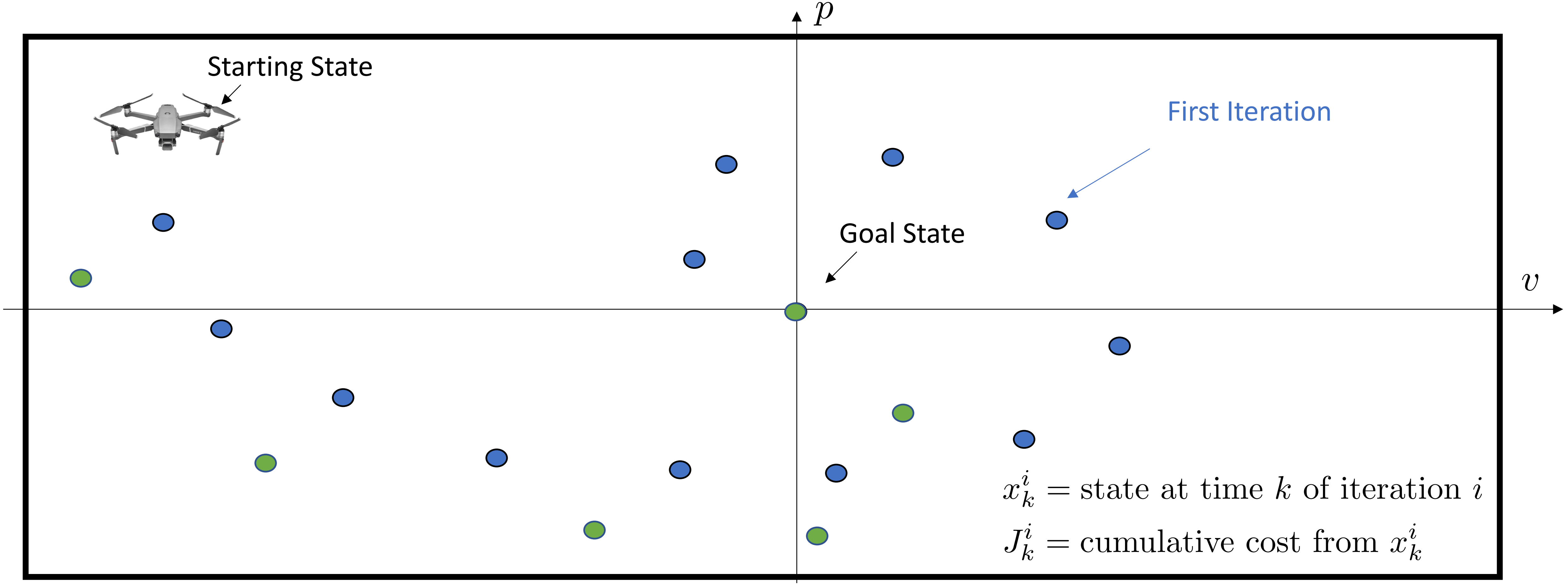
Value Function Estimation



Value Function

$$V^j(x) = \text{Interpolation of the set of data } \{\{(J_k^i, x_k^i)\}_{i=0}^{j}\}_{k=0}^{K}$$

Value Function Estimation

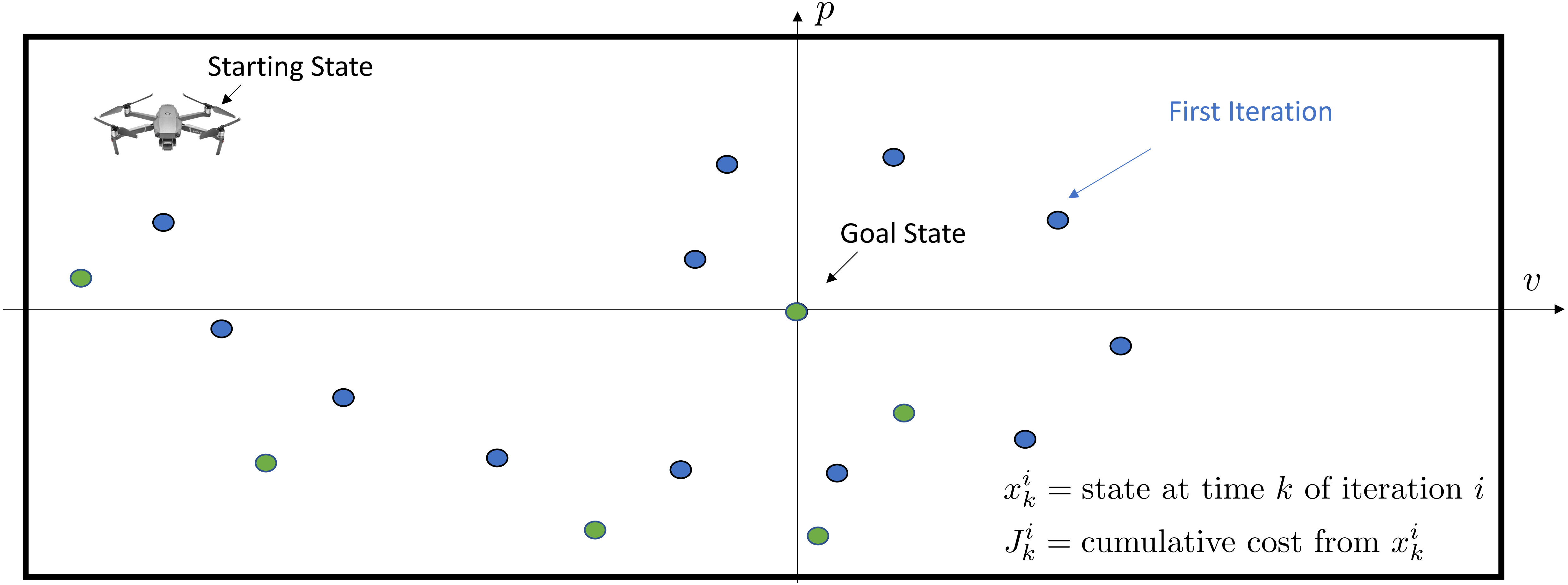


Value Function

$$V^j(x) = \min_{\lambda_k^i \in [0,1]} \; \sum_k \sum_k J_k^k \lambda_k^i$$

$$\text{s.t} \qquad \sum_k \sum_i x_k^i \lambda_k^i = x, \; \sum_k \sum_i \lambda_k^i = 1$$

Value Function Estimation



Value Function

$$V^j(\textcolor{red}{x}) = \min_{\lambda_k^i \in [0,1]} \; \sum_k \sum_k J_k^k \lambda_k^i$$

$$\text{s.t} \quad \sum_k \sum_i x_k^i \lambda_k^i = \textcolor{red}{x}, \; \sum_k \sum_i \lambda_k^i = 1$$

Upper-bound on future cumulated cost

LMPC Summary

At each time t of iteration j, solve

$$
\begin{aligned}
J(x(t)) = \min_{u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + V^{j-1}(x_N) \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\
& x_0 = x(t), \\
& x_k \in \mathcal{X},\ u_k \in \mathcal{U}, \\
& x_N \in \ \mathcal{SS}^{j-1}, \\
& \forall k \in [0, \cdots, N-1]
\end{aligned}
$$

LMPC Summary

At each time *t* of iteration *j*, solve

$$
\begin{aligned}
J(x(t)) = \min_{u_0,\dots,u_{N-1}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + \boxed{V^{j-1}(x_N)} \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\
& x_0 = x(t), \\
& x_k \in \mathcal{X},\ u_k \in \mathcal{U}, \\
& x_N \in \boxed{\mathcal{SS}^{j-1}}, \\
& \forall k \in [0, \cdots, N-1]
\end{aligned}
$$

Constructed using historical data

Guarantees for constrained (linear) systems [1,2]

The properties of the (convex) safe set and (convex) V-function allows us to guarantee:

- Safety: constraint satisfaction at iteration *j* → satisfaction at iteration *j+1*
- Non-decreasing Performance: closed-loop cost at iteration *j* >= closed-loop cost at iteration *j+1*
- Performance Improvement: closed-loop cost strictly deceasing at each iteration (LICQ required)
- (Global) optimality: steady state trajectory is optimal for the original problem (LICQ required)

[1] U. Rosolia, F. Borrelli. “Learning model predictive control for iterative tasks. a data-driven control framework.” *IEEE Transactions on Automatic Control* (2018).
[2] U. Rosolia, F. Borrelli. “Learning model predictive control for iterative tasks: A computationally efficient approach for linear system.” *IFAC-PapersOnLine* (2017)
[3] U. Rosolia, Y. Lian, E. Maddalena, G. Ferrari-Trecate, and C. N. Jones. "On the Optimality and Convergence Properties of the Iterative Learning Model Predictive Controller." IEEE Transactions on Automatic Control (2022).

Practical Implementation

Learning MPC convex formulation

Linear(ized) LMPC

Given $j-1$ trajectories, we define the following optimization problem:

$$J(x(t)) = \min_{u_0,\ldots,u_{N-1}} \quad \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + \ V^{j-1}(x_N)$$

$$\begin{aligned}
\text{s.t.} \qquad & x_{k+1} = f(x_k, u_k), \\
& x_0 = x(t), \\
& x_k \in \mathcal{X}, \ u_k \in \mathcal{U}, \\
& x_N \in \mathcal{CS}^{j-1} \\
& \forall k \in [0, \cdots, N-1]
\end{aligned}$$

Linear(ized) LMPC

Given $j-1$ trajectories, we define the following optimization problem:

$$J(x(t)) = \min_{\substack{u_0,\ldots,u_{N-1} \\ \boxed{\lambda_0^0,\ldots,\lambda_K^{j-1}}}} \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + V^{j-1}(x_N)$$

$$\begin{aligned} \text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\ & x_0 = x(t), \\ & x_k \in \mathcal{X}, \ u_k \in \mathcal{U}, \\ \boxed{x_N \in \mathcal{CS}^{j-1}} \iff & \boxed{x_N = \sum_k \sum_i x_k^i \lambda_k^i, \sum_k \sum_i \lambda_k^i = 1, \lambda_k^i \geq 0} \\ & \forall k \in [0, \cdots, N-1] \end{aligned}$$

Linear(ized) LMPC

Given $j-1$ trajectories, we define the following optimization problem:

$$
\begin{aligned}
J(x(t)) = \min_{\substack{u_0,\dots,u_{N-1}\\ \boxed{\lambda_0^0,\dots,\lambda_K^{j-1}}}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + \boxed{\sum_k \sum_i J_k^i \lambda_k^i} \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\
& x_0 = x(t), \\
& x_k \in \mathcal{X},\ u_k \in \mathcal{U}, \\
& \boxed{x_N = \sum_k \sum_i x_k^i \lambda_k^i, \sum_k \sum_i \lambda_k^i = 1, \lambda_k^i \geq 0} \\
& \forall k \in [0, \cdots, N-1]
\end{aligned}
$$

$$\boxed{V^{j-1}(x_N)}$$

$$\boxed{x_N \in \mathcal{CS}^{j-1}} \iff \boxed{x_N = \sum_k \sum_i x_k^i \lambda_k^i, \sum_k \sum_i \lambda_k^i = 1, \lambda_k^i \geq 0}$$

Linear(ized) LMPC

Given $j-1$ trajectories, we define the following optimization problem:

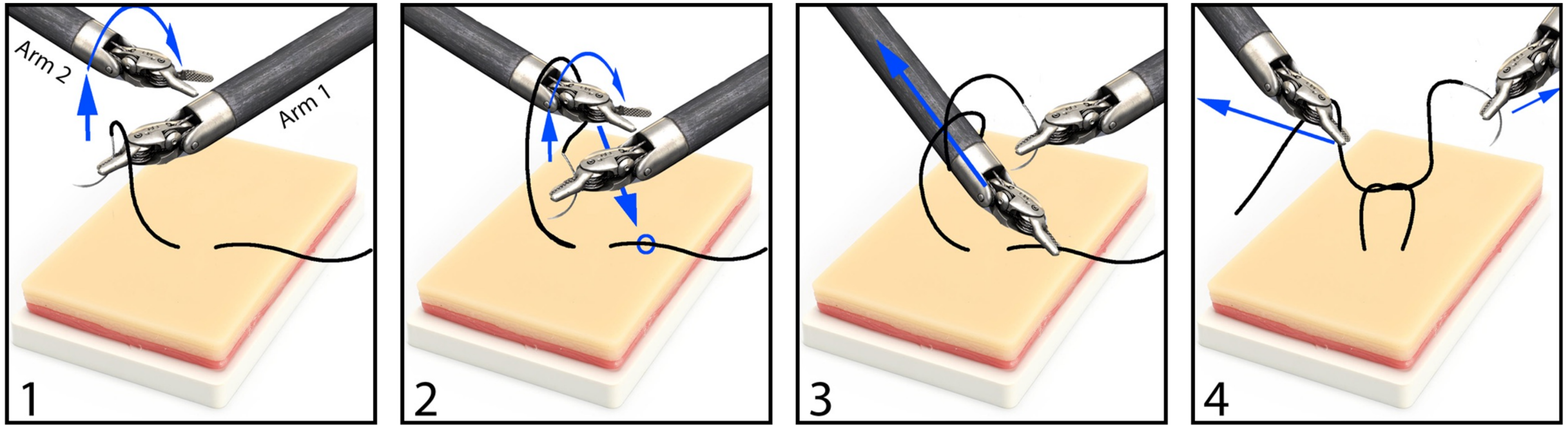
$$
\begin{aligned}
J(x(t)) = \min_{\substack{u_0,\dots,u_{N-1} \\ \boxed{\lambda_0^0,\dots,\lambda_K^{j-1}}}} \quad & \sum_{k=0}^{N-1} \left(x_k^\top Q x_k + u_k^\top R u_k\right) + \boxed{\sum_k \sum_i J_k^i \lambda_k^i} \Longleftrightarrow \boxed{V^{j-1}(x_N)} \\
\text{s.t.} \quad & x_{k+1} = f(x_k, u_k), \\
& x_0 = x(t), \\
& x_k \in \mathcal{X}, \; u_k \in \mathcal{U}, \\
\boxed{x_N \in \mathcal{CS}^{j-1}} \Longleftrightarrow \quad & \boxed{x_N = \sum_k \sum_i x_k^i \lambda_k^i, \sum_k \sum_i \lambda_k^i = 1, \lambda_k^i \geq 0} \\
& \forall k \in [0, \cdots, N-1]
\end{aligned}
$$

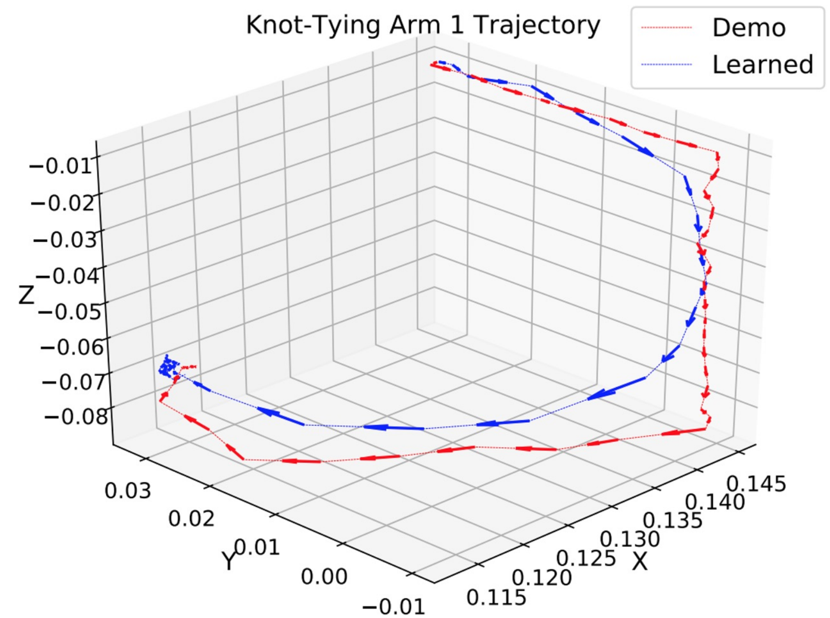
- Convex optimization problem over inputs and lambdas
- Safety and performance improvement guarantees still hold (simple proofs as before)
- Converges to global optimal solution (Constraints Qualification Condition required)

Terminal Components via DNN

Brijen

Ashwin





- Safe Set constructed using non-parametric estimation
- Model ensemble and input sampling strategies for MPC
- Knot tying task on real surgical robot with inefficient demos (red)
- Constraints: stay within 1 cm tube of reference trajectory
- SAVED successfully smooths + optimizes demos

"Safety Augmented Value Estimation from Demonstrations (SAVED): Safe Deep Model-Based RL for Sparse Cost Robotic Tasks.", B. Thananjeyan*, A. Balakrishna*, U. Rosolia, F. Li, R. McAllister, J. E. Gonzalez, S. Levine, F. Borrelli, K. Goldberg *IEEE Robotics and Automation Letters (RA-L)* (2020)

*= equal contribution

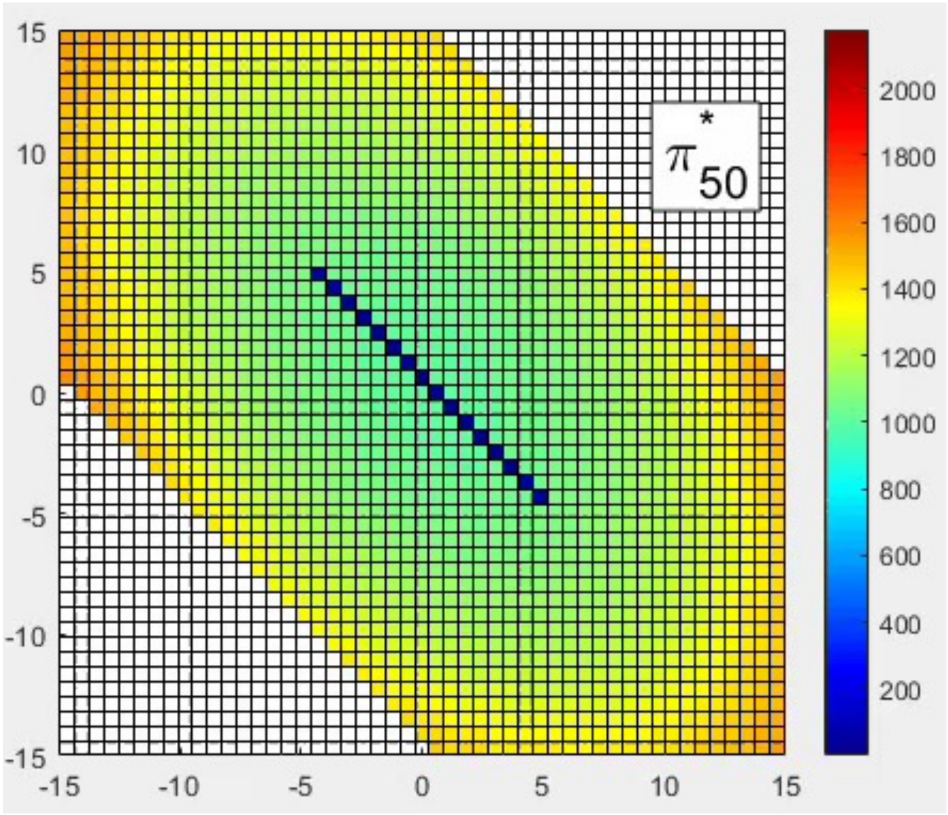
Comparison with Approximate DP (aka RL)

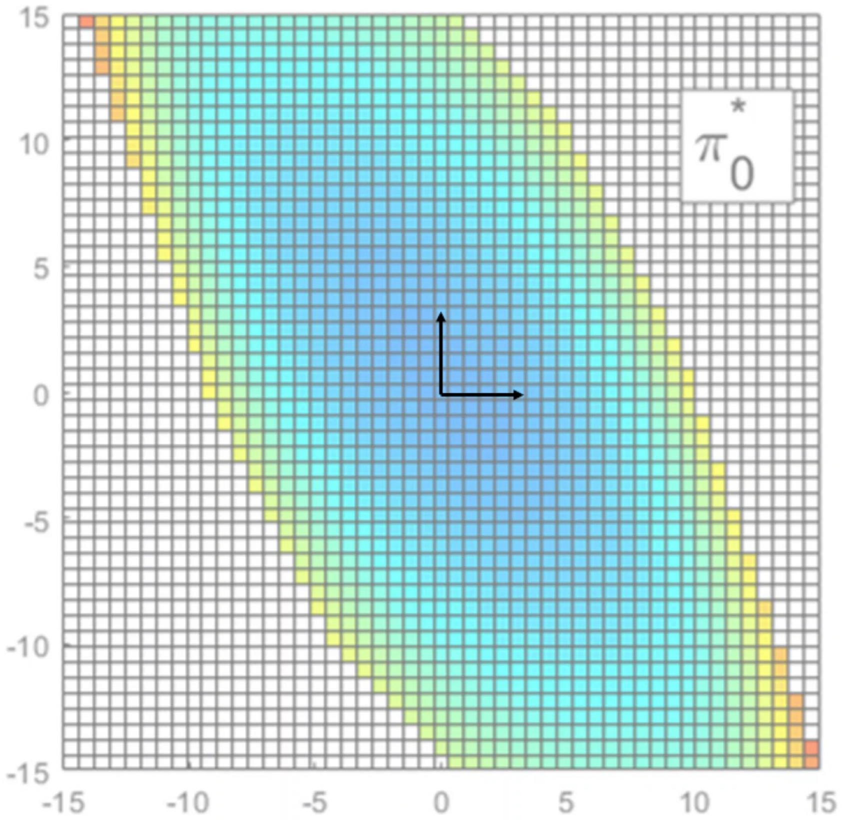
- Some references:
  - Bertsekas paper connecting MPC and ADP [1], books on RL and OC [2,3]
  - Lewis and Vrabie survey [4]
  - Recht survey [5]

- Learning MPC highlights
  - Continuous state and action formulation
  - Constraints satisfaction
  - V-function constructed locally based on cost/model driven exploration
  - V-function at stored state is "exact" and upperbounds at intermediate points

[1] D. Bertsekas, "Dynamic programming and suboptimal control: A survey from ADP to MPC." European Journal of Control 11.4-5 (2005)
[2] D. Bertsekas, "Reinforcement learning and optimal control." Athena Scientific, 2019.
[3] D. Bertsekas, "Distributed Reinforcement Learning" http://web.mit.edu/dimitrib/www/RL_2_Rollout_&_PI.pdf
[4] F. Lewis, Frank, and D. Vrabie. "Reinforcement learning and adaptive dynamic programming for feedback control." IEEE circuits and systems magazine 9.3 (2009)
[5] R. Benjamin. "A tour of reinforcement learning: The view from continuous control." Annual Review of Control, Robotics, and Autonomous Systems 2 (2019)

Learning MPC = Forward Value Iteration





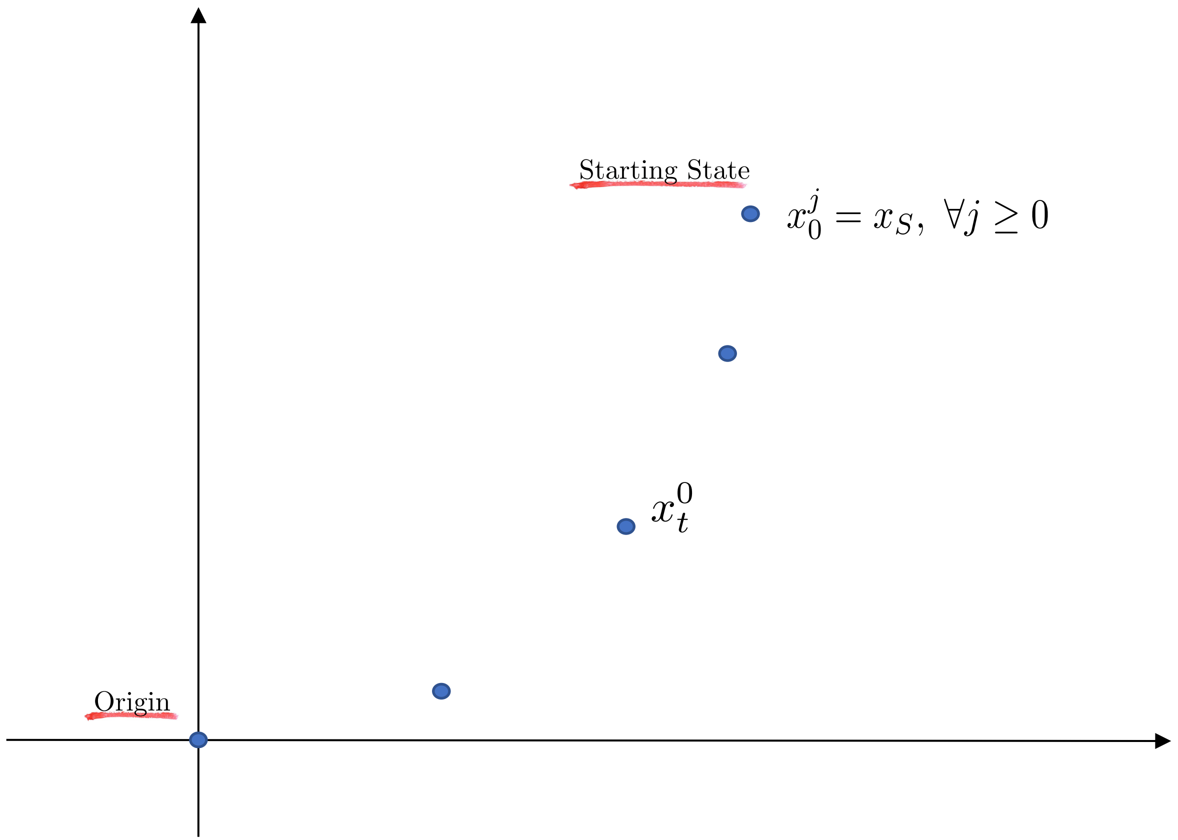
Dynamic Programming:

- Gridding, global properties
- Backward, one-step iteration

LMPC:

- No Gridding, local properties
- Forward, multi-step prediction
- LICQ required for optimality

Example I: Constrained LQR



Infinite Time Optimal Control Problem

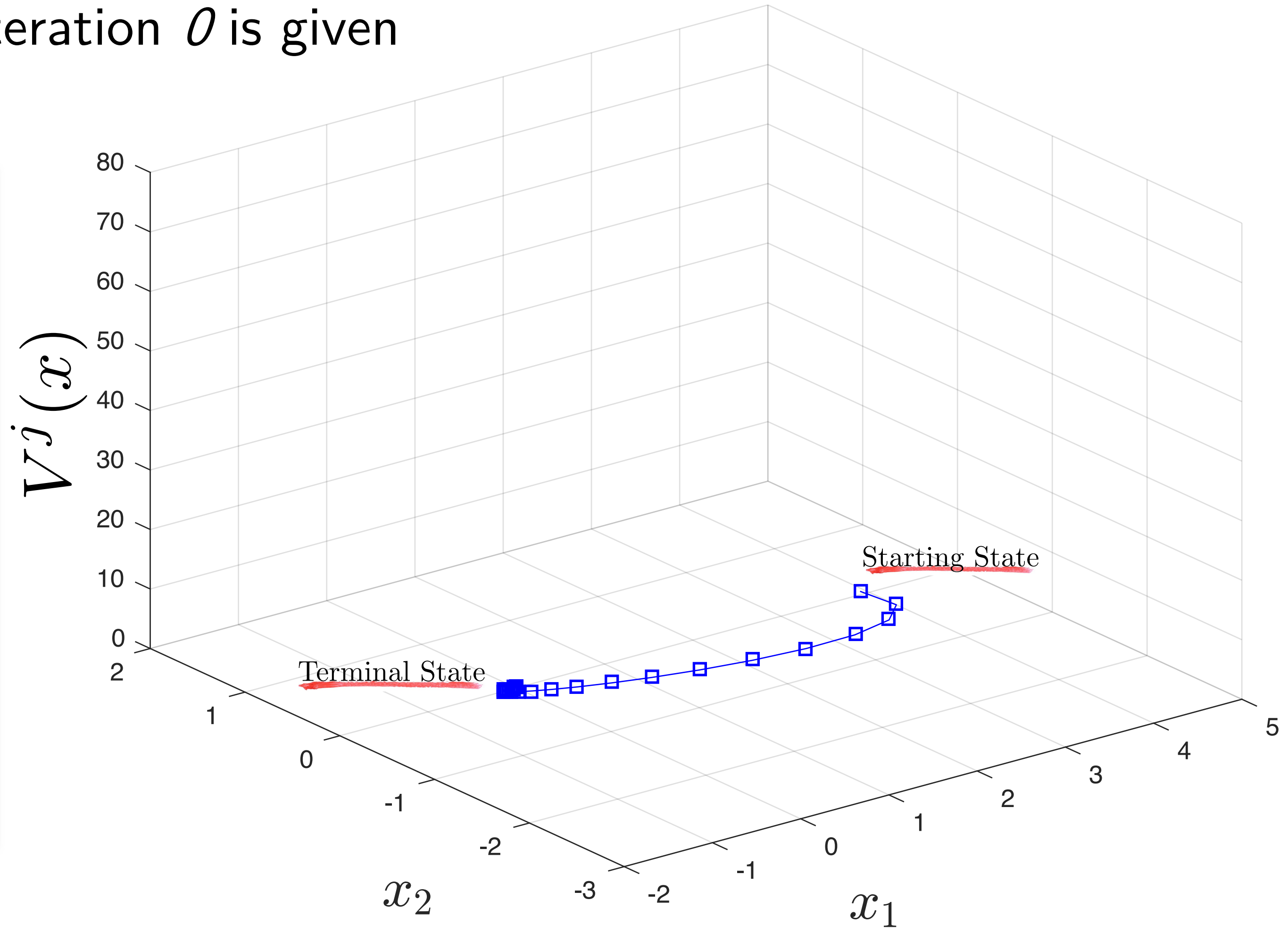
The goal of the control design is to solve the following constrained LQR problem for the double integrator system,

$$\begin{aligned} \min_{u_0, u_1, \dots} \quad & \sum_{k=0}^{\infty} x_k^\top Q x_k + u_k^\top R u_k \\ \text{s.t.} \quad & x_0 = x_S, \\ & x_{k+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_k, \\ & x_k \in \mathcal{X}, \ u_k \in \mathcal{U}, \ \forall k \geq 0 \end{aligned}$$

Example I: Constrained LQR

Assumption: A first feasible trajectory at iteration *0* is given

Iterative LMPC



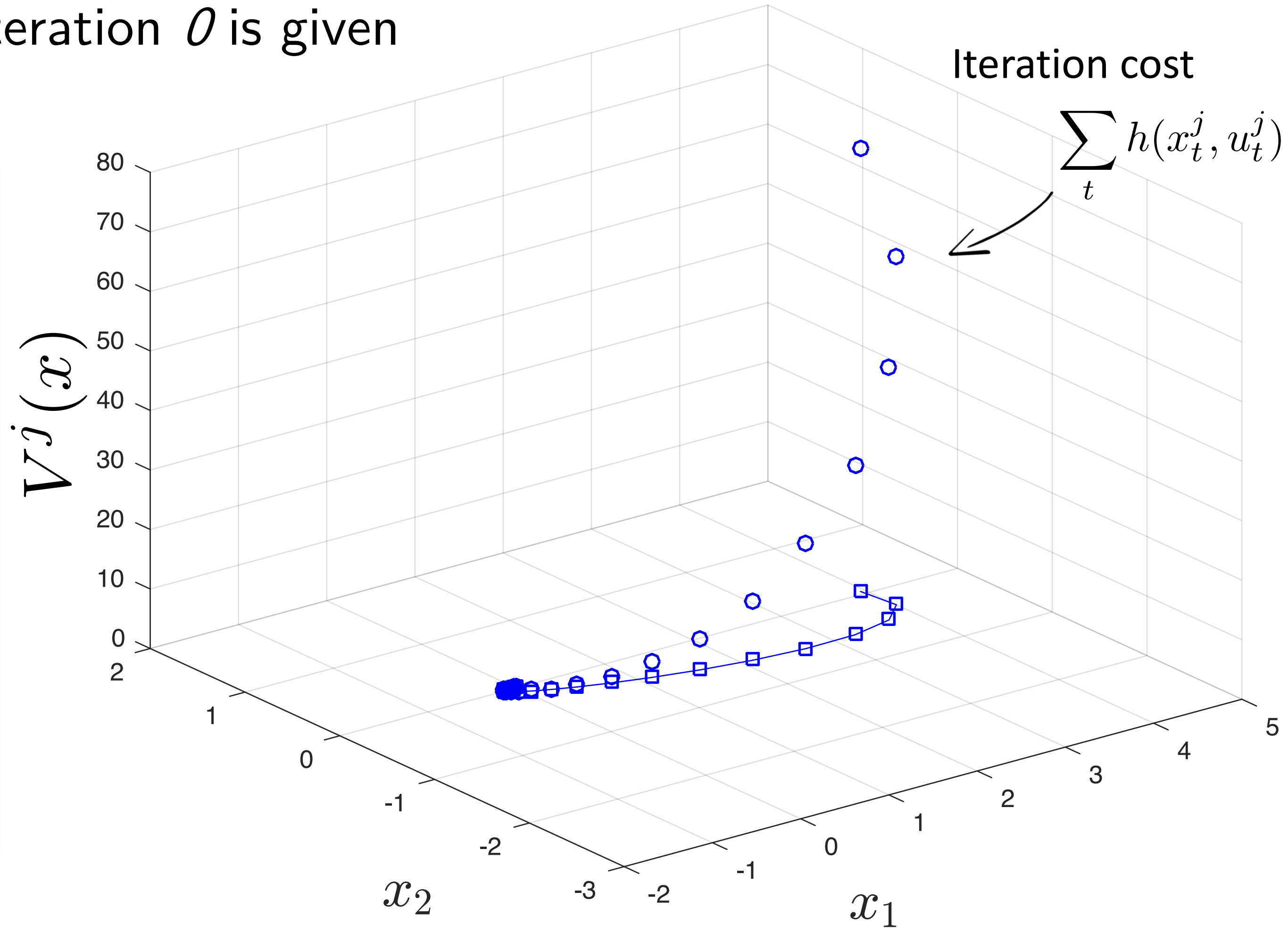
Example I: Constrained LQR

Assumption: A first feasible trajectory at iteration *0* is given

Iterative LMPC

Step 0: Set iteration counter $j=0$

→ Step 1: Compute the roll-out cost for the recorded data up to iteration j



Example I: Constrained LQR

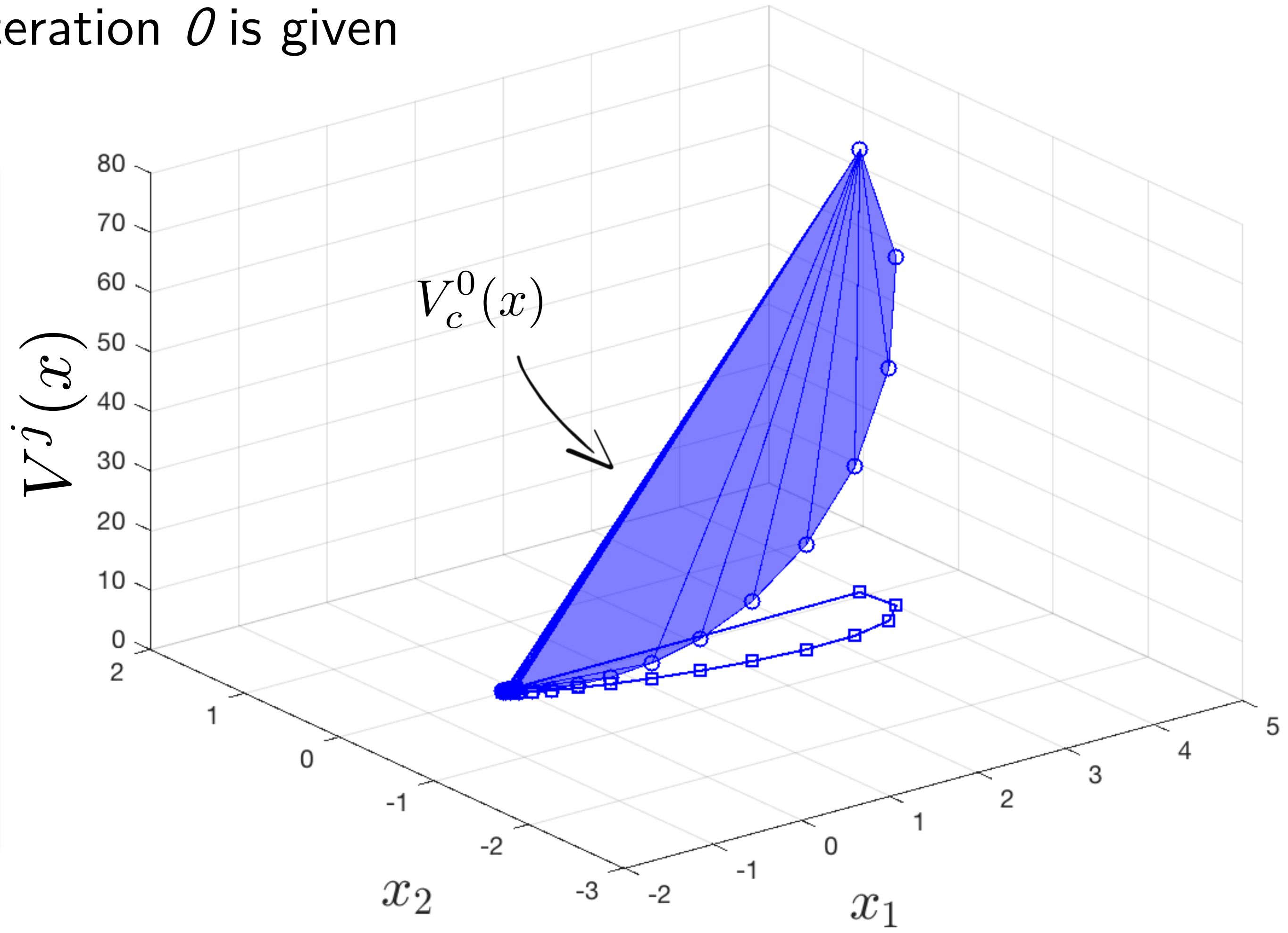
Assumption: A first feasible trajectory at iteration *0* is given

Iterative LMPC

Step 0: Set iteration counter *j=0*

Step 1: Compute the roll-out cost for the recorded data up to iteration *j*

→ Step 2: Define V^j which interpolates linearly the roll-out cost



Example I: Constrained LQR

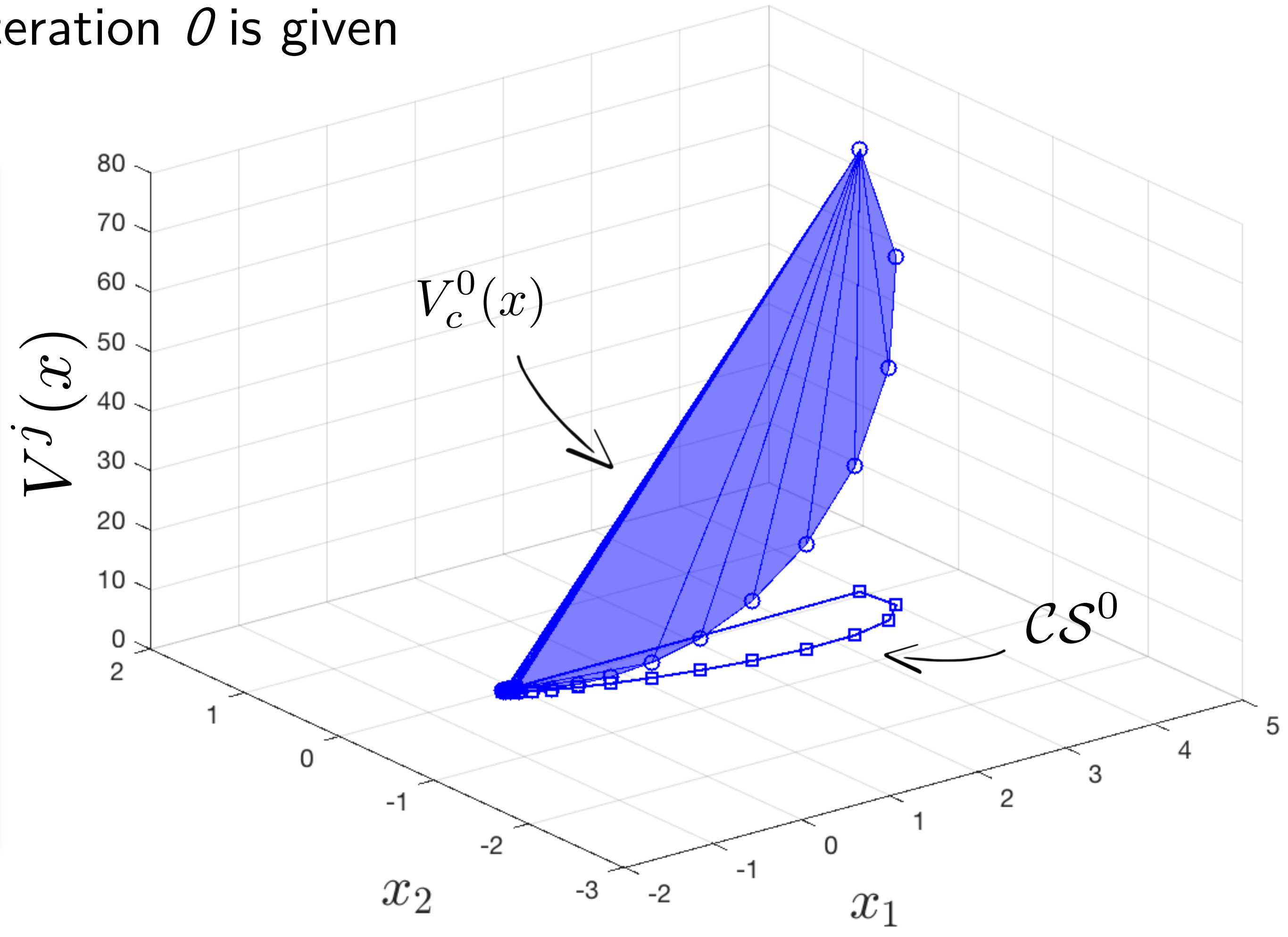
Assumption: A first feasible trajectory at iteration *0* is given

Iterative LMPC

Step 0: Set iteration counter j=*0*

Step 1: Compute the roll-out cost for the recorded data up to iteration j

→ Step 2: Define V^j which interpolates linearly the roll-out cost



Example I: Constrained LQR

Assumption: A first feasible trajectory at iteration *0* is given

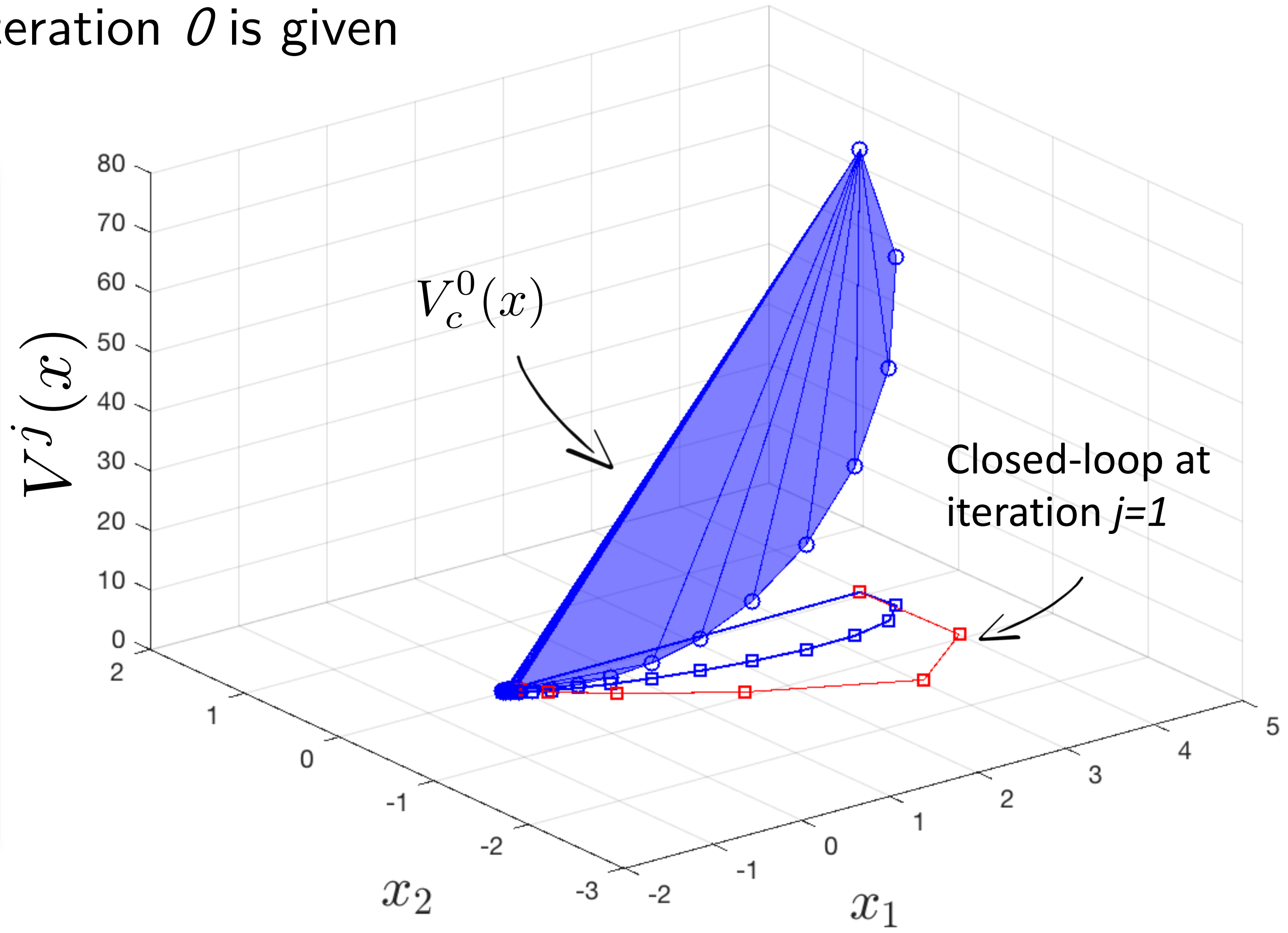
Iterative LMPC

Step 0: Set iteration counter *j=0*

Step 1: Compute the roll-out cost for the recorded data up to iteration *j*

Step 2: Define V^j which interpolates linearly the roll-out cost

Step 3: Run a closed-loop simulation at iteration *j+1*



Example I: Constrained LQR

Assumption: A first feasible trajectory at iteration *0* is given

Iterative LMPC

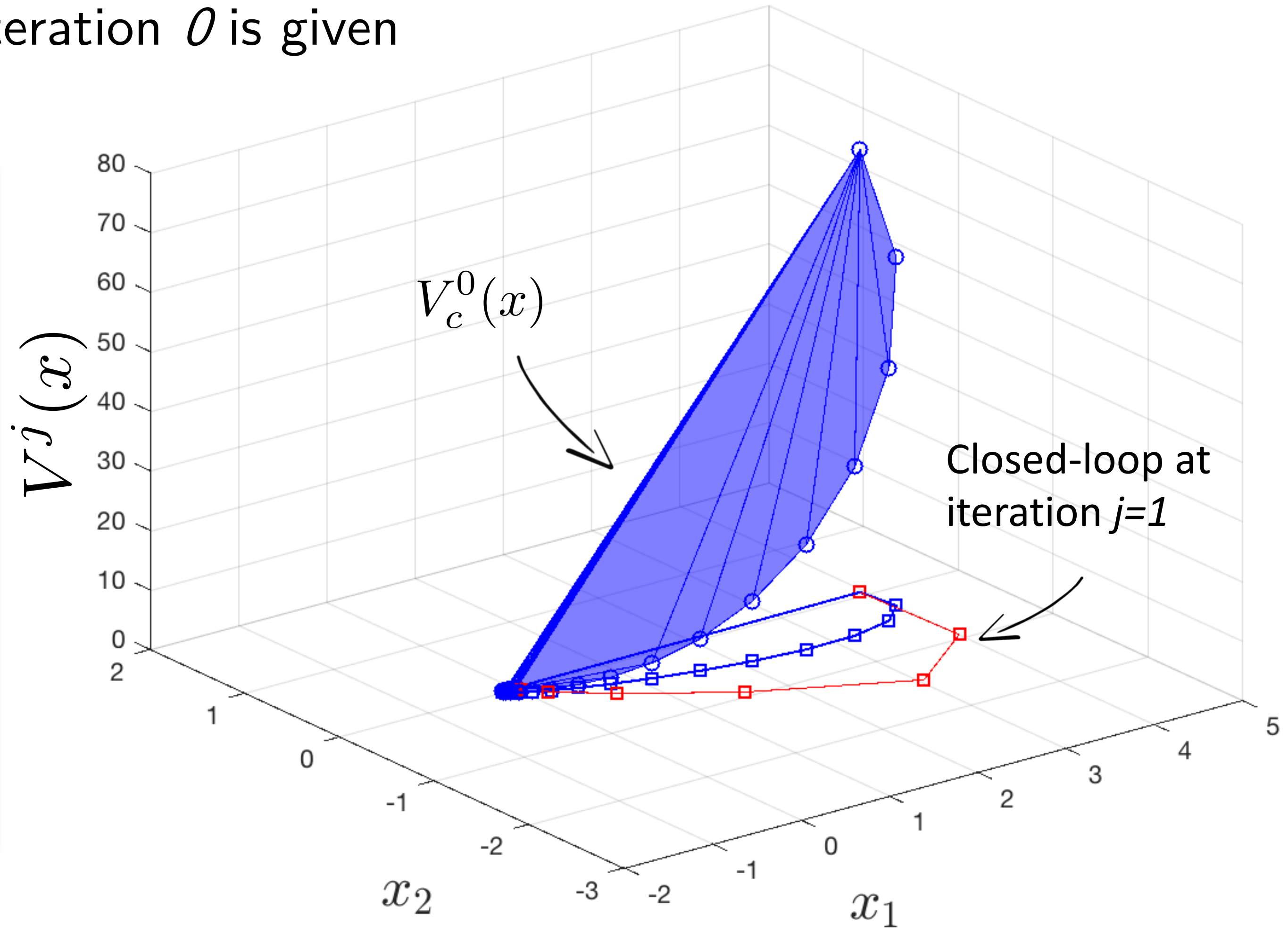
Step 0: Set iteration counter j=*0*

Step 1: Compute the roll-out cost for the recorded data up to iteration j

Step 2: Define V^j which interpolates linearly the roll-out cost

Step 3: Run a closed-loop simulation at iteration j+*1*

→ Step 5: Set iteration counter $j = j$+*1.* Go to Step 1



Example I: Constrained LQR

Assumption: A first feasible trajectory at iteration *0* is given

Iterative LMPC

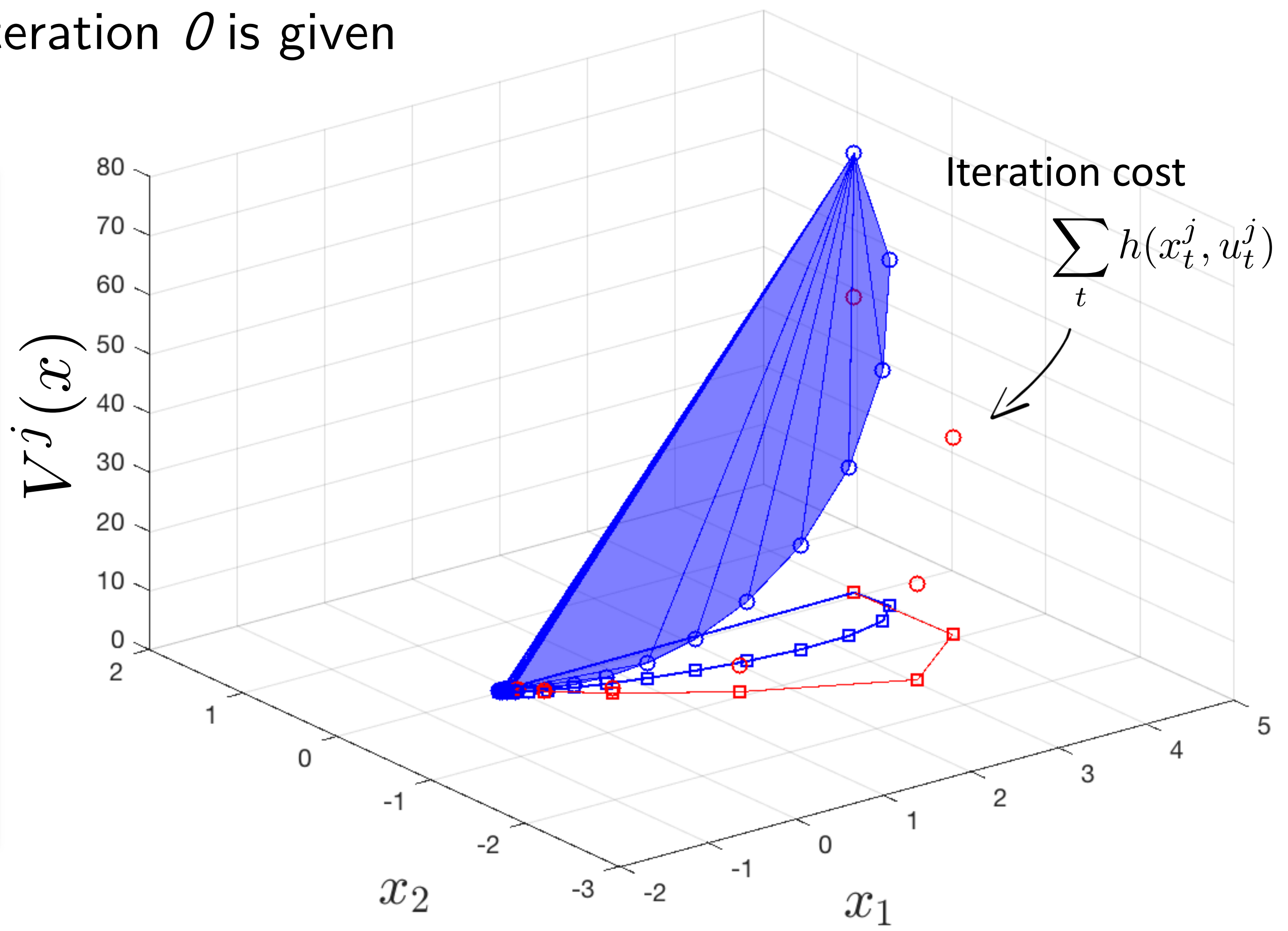
Step 0: Set iteration counter j=0

Step 1: Compute the roll-out cost for the recorded data up to iteration j

Step 2: Define V^j which interpolates linearly the roll-out cost

Step 3: Run a closed-loop simulation at iteration j+1

Step 5: Set iteration counter $j = j$+1. Go to Step 1



Example I: Constrained LQR

Assumption: A first feasible trajectory at iteration *0* is given

Iterative LMPC

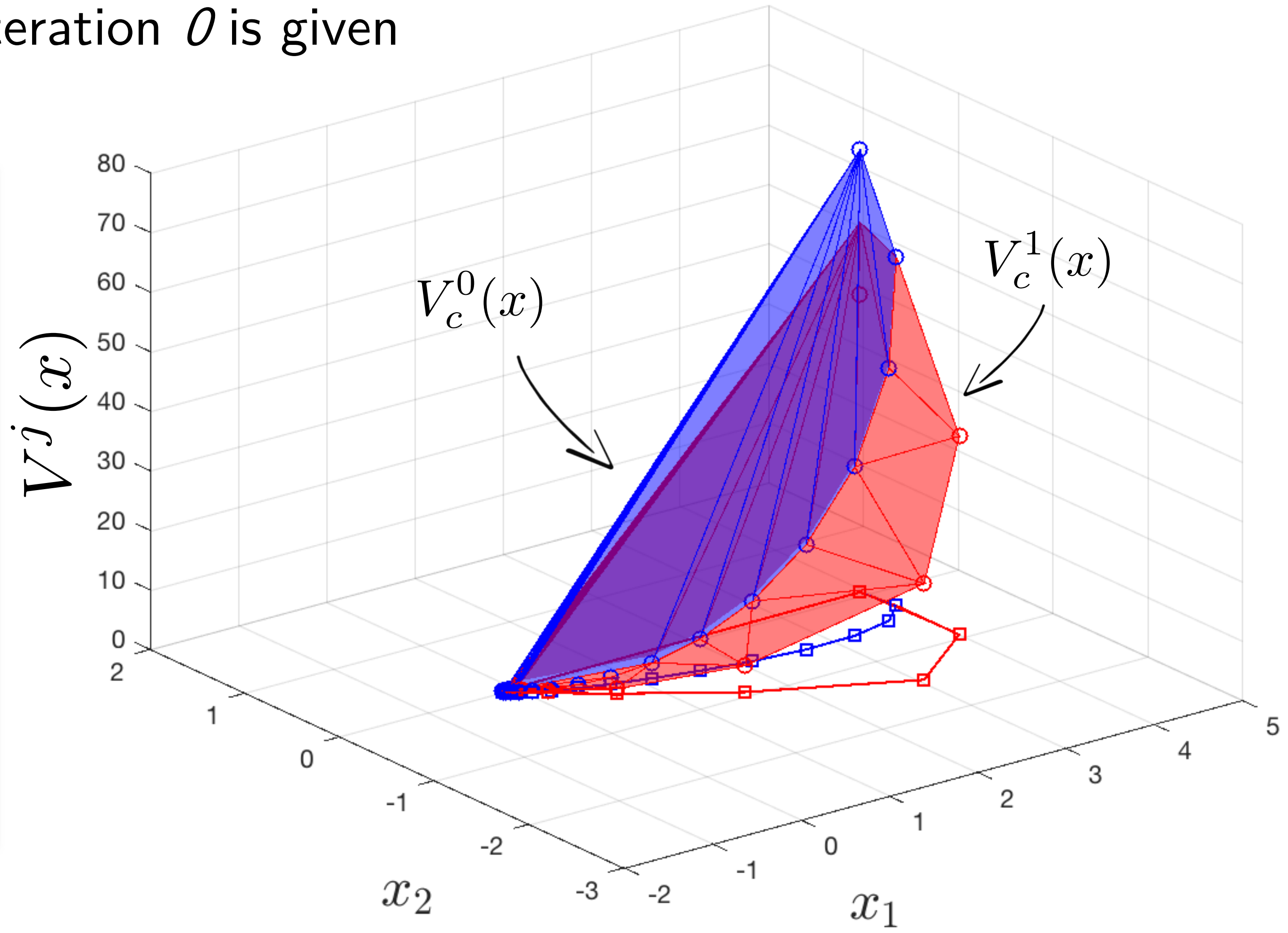
Step 0: Set iteration counter *j=0*

Step 1: Compute the roll-out cost for the recorded data up to iteration *j*

→ Step 2: Define V^j which interpolates linearly the roll-out cost

Step 3: Run a closed-loop simulation at iteration *j+1*

Step 5: Set iteration counter *j = j+1*. Go to Step 1



Example I: Constrained LQR

Assumption: A first feasible trajectory at iteration *0* is given

Iterative LMPC

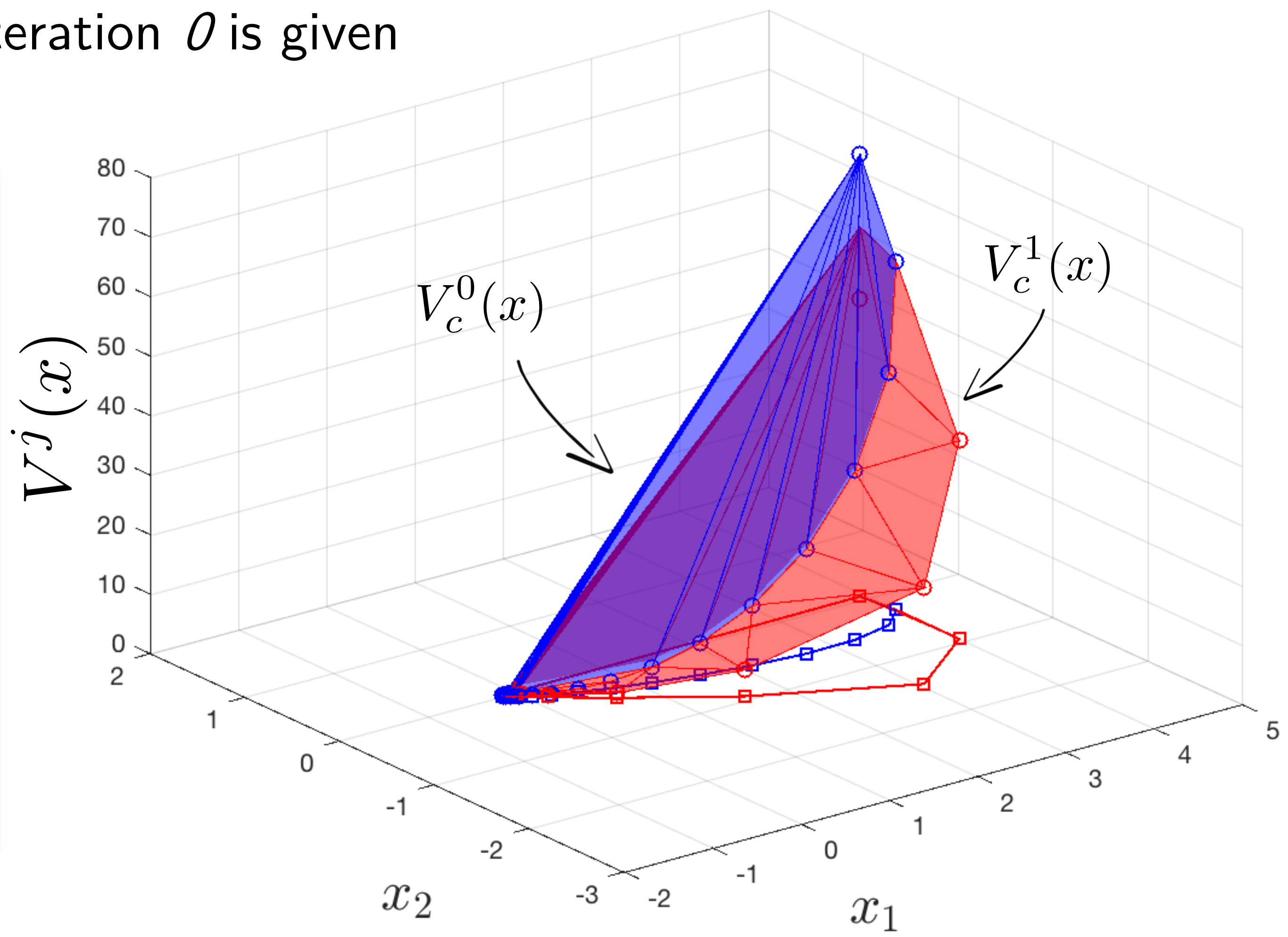
Step 0: Set iteration counter *j=0*

Step 1: Compute the roll-out cost for the recorded data up to iteration *j*

Step 2: Define V^j which interpolates linearly the roll-out cost

Step 3: Run a closed-loop simulation at iteration *j+1*

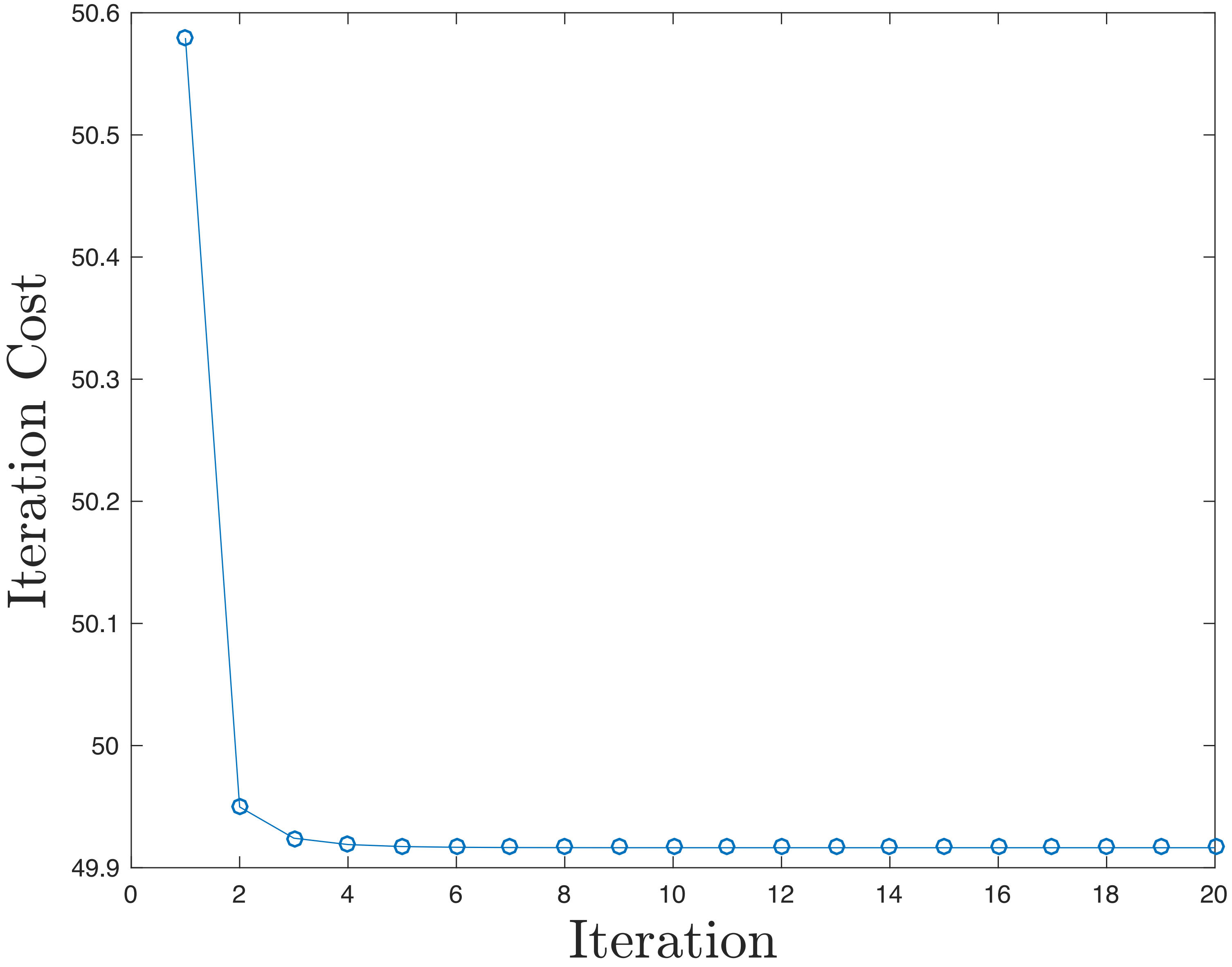
Step 5: Set iteration counter *j = j+1.* Go to Step 1



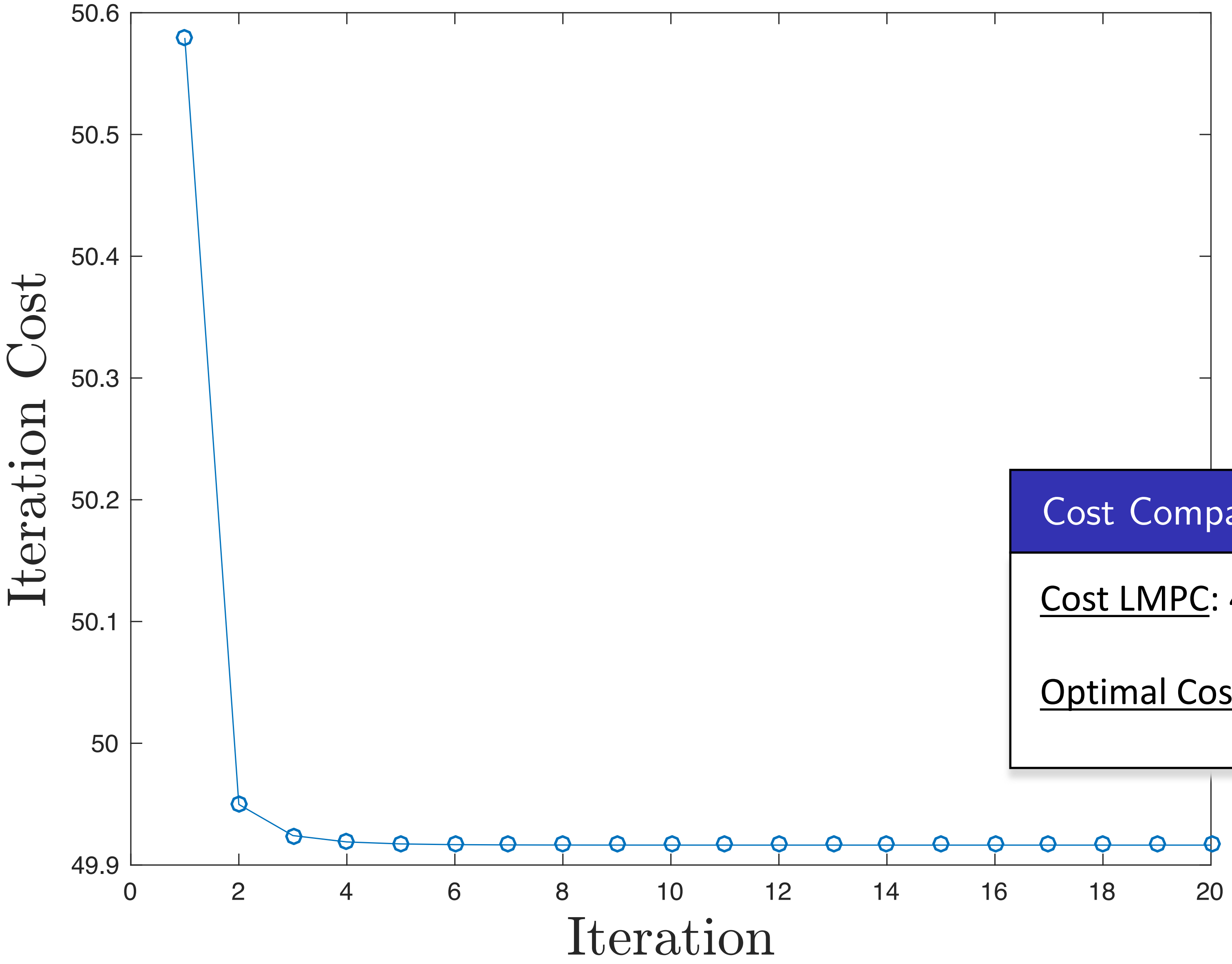
Key Messages:

- The cost function is defined on a **subset** of the state space.
- The LMPC **explores** the state space in order to enlarge the terminal cost domain.

Iteration Cost



Iteration Cost



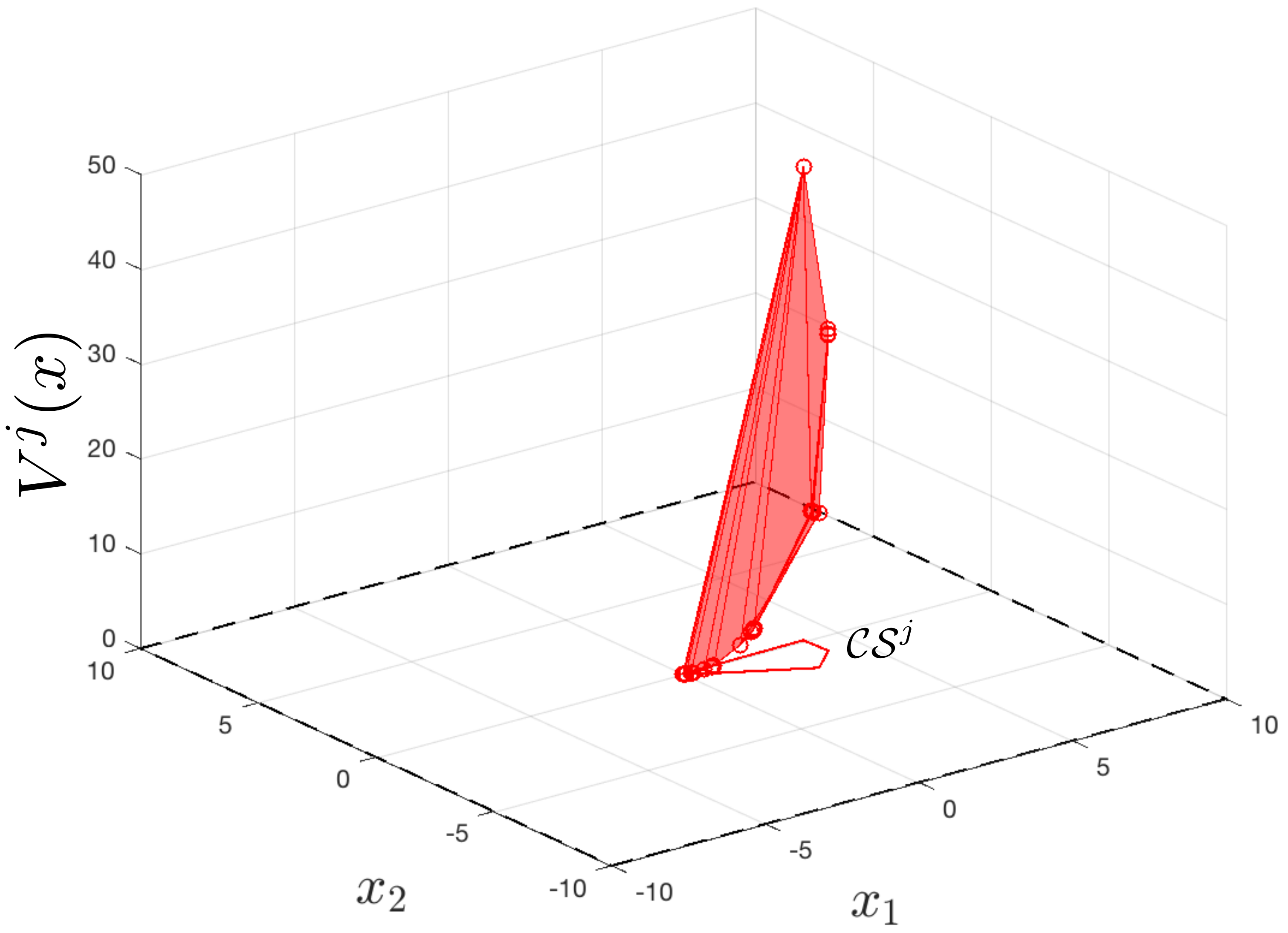
Cost Comparison

Cost LMPC: 49.9164

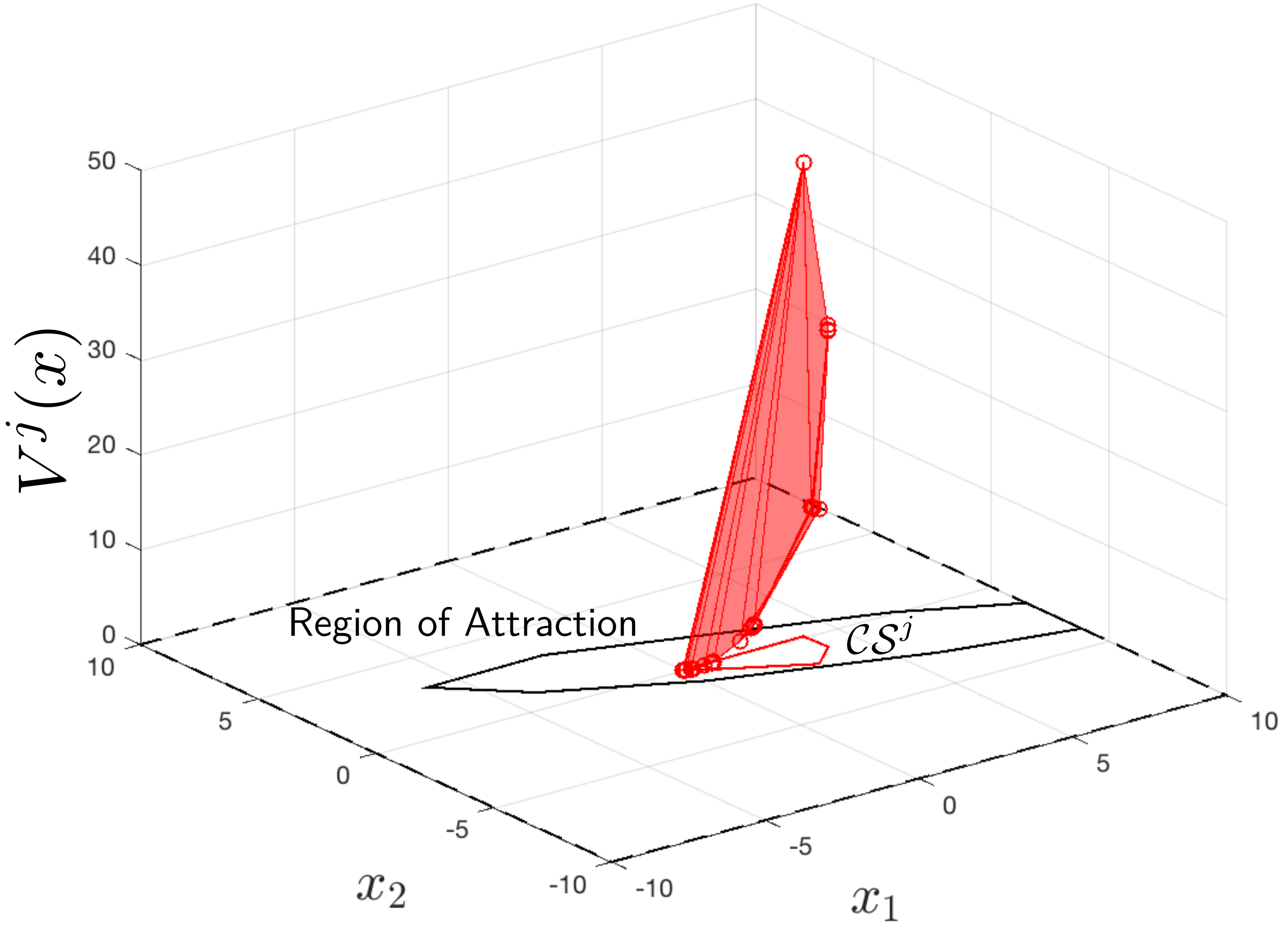
Optimal Cost: 49.9164

Different initial conditions at each iteration

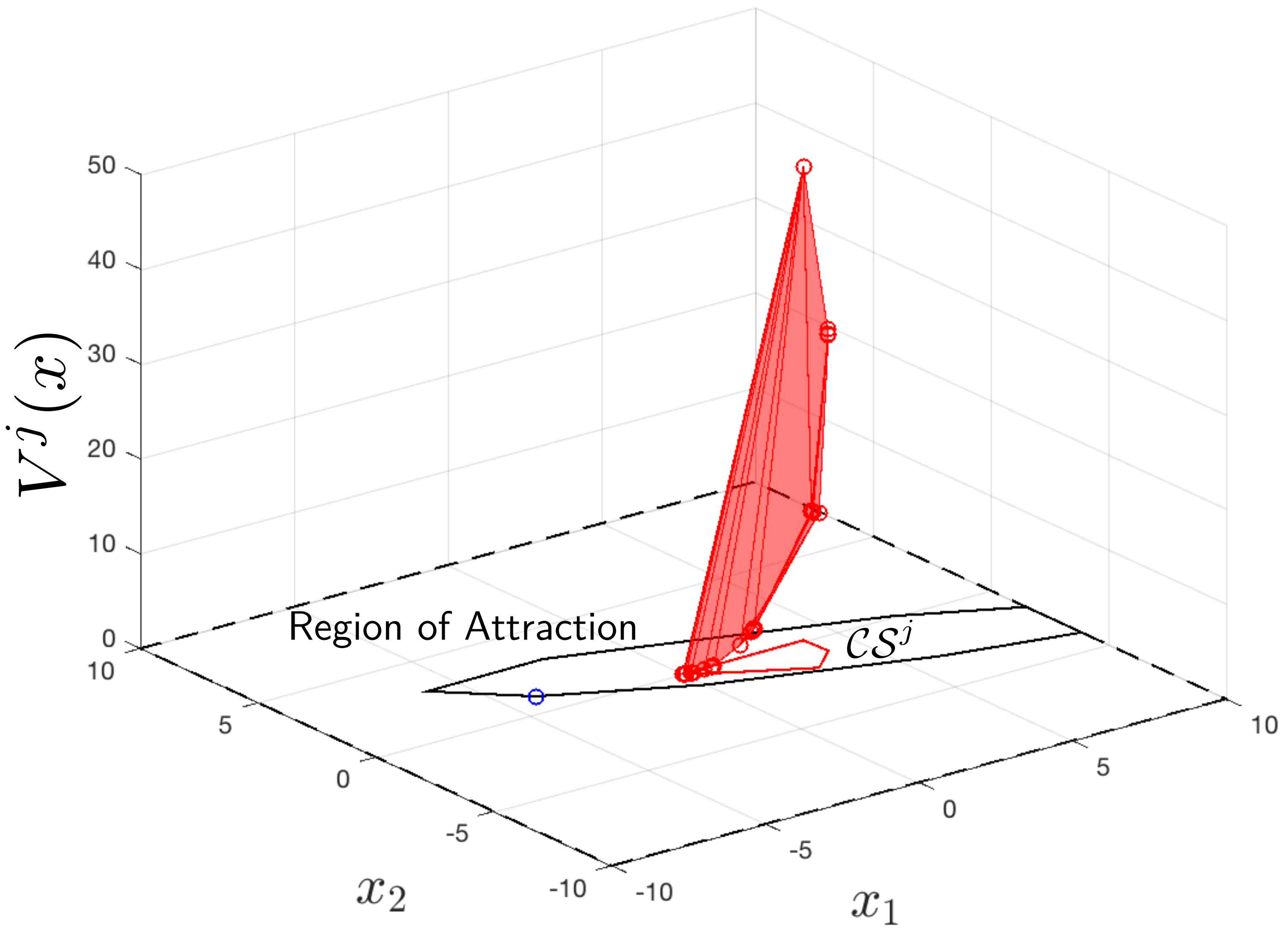
Different initial conditions at each iteration



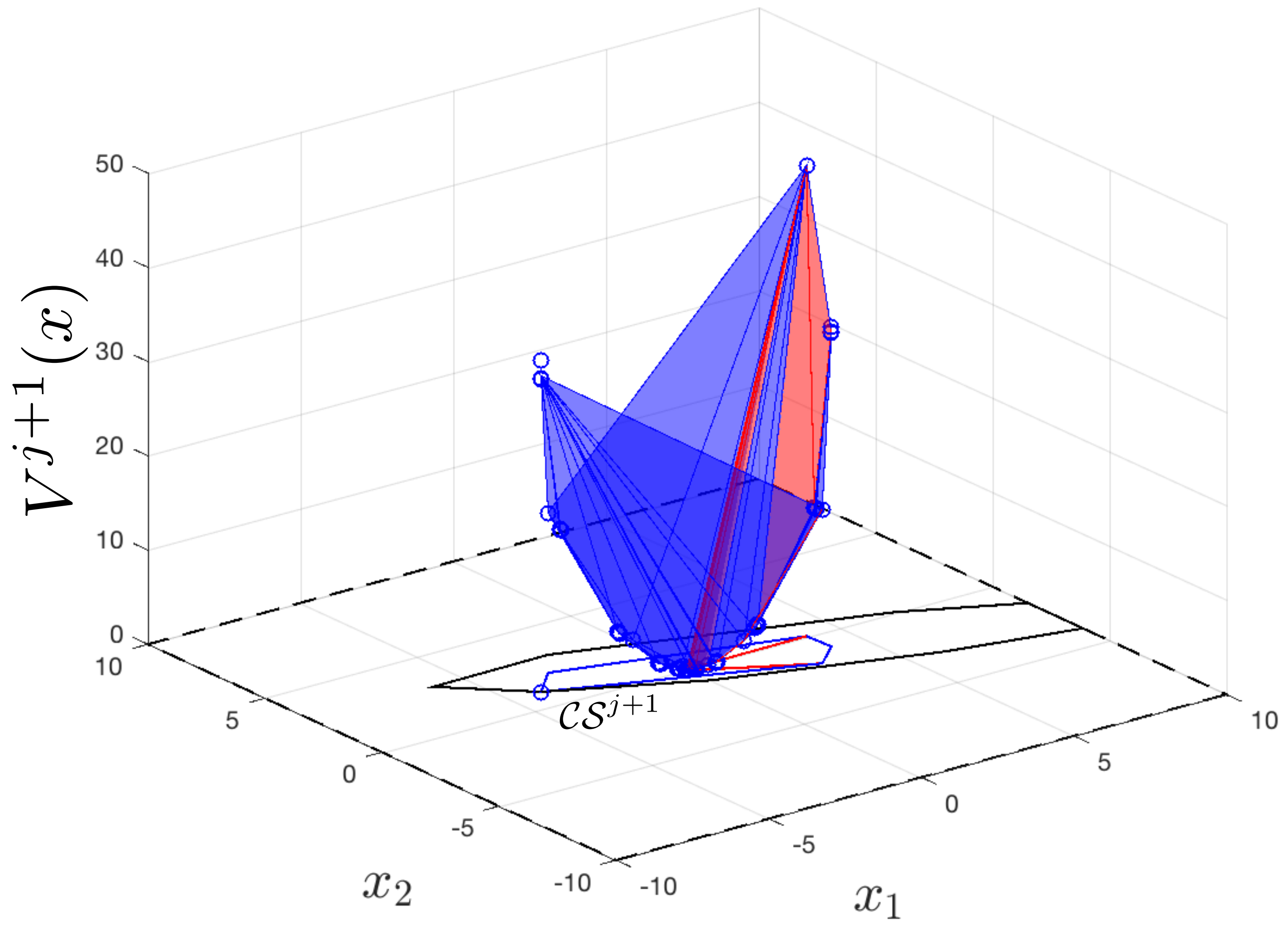
Different initial conditions at each iteration



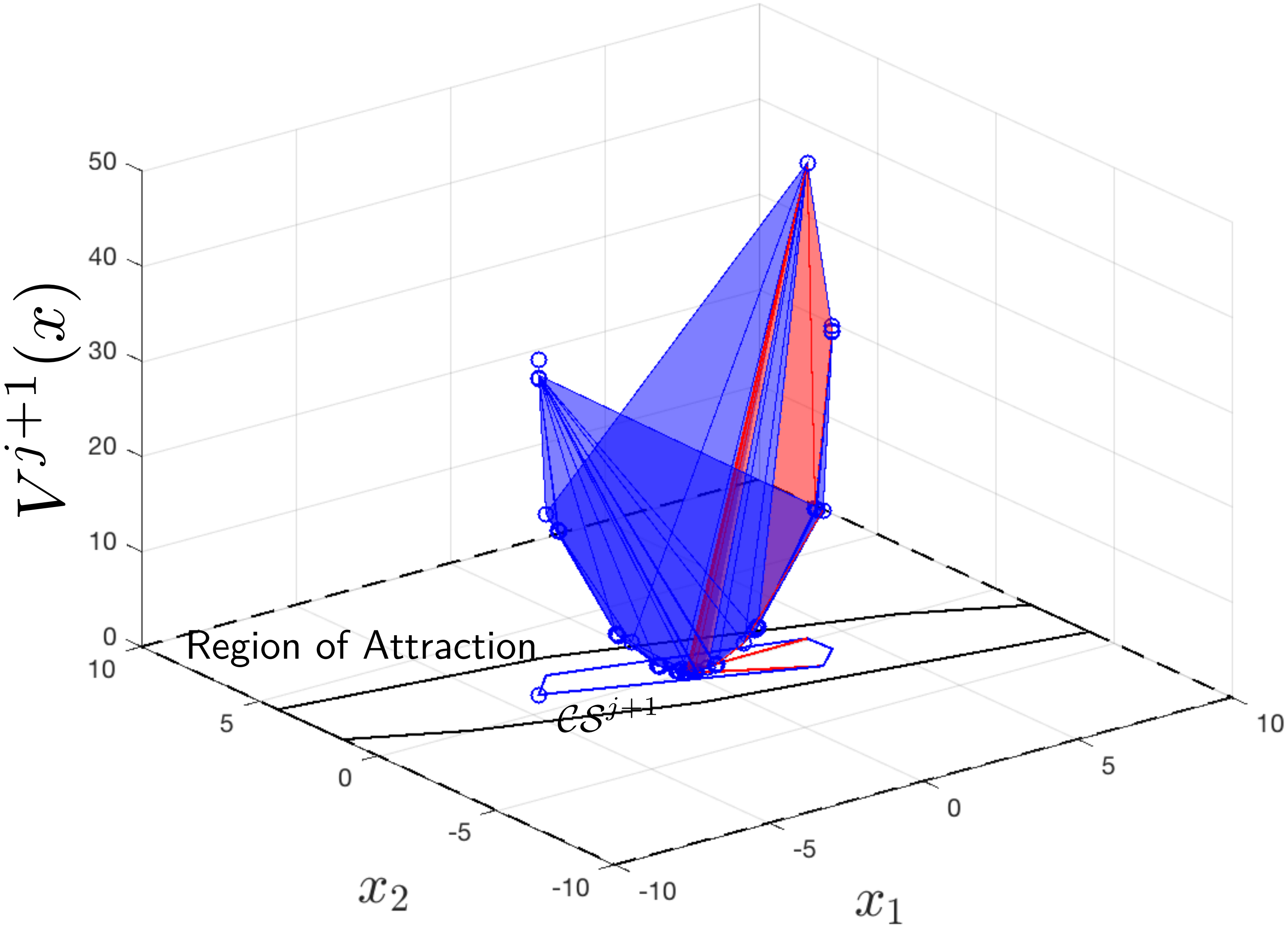
Different initial conditions at each iteration



Different initial conditions at each iteration



Different initial conditions at each iteration



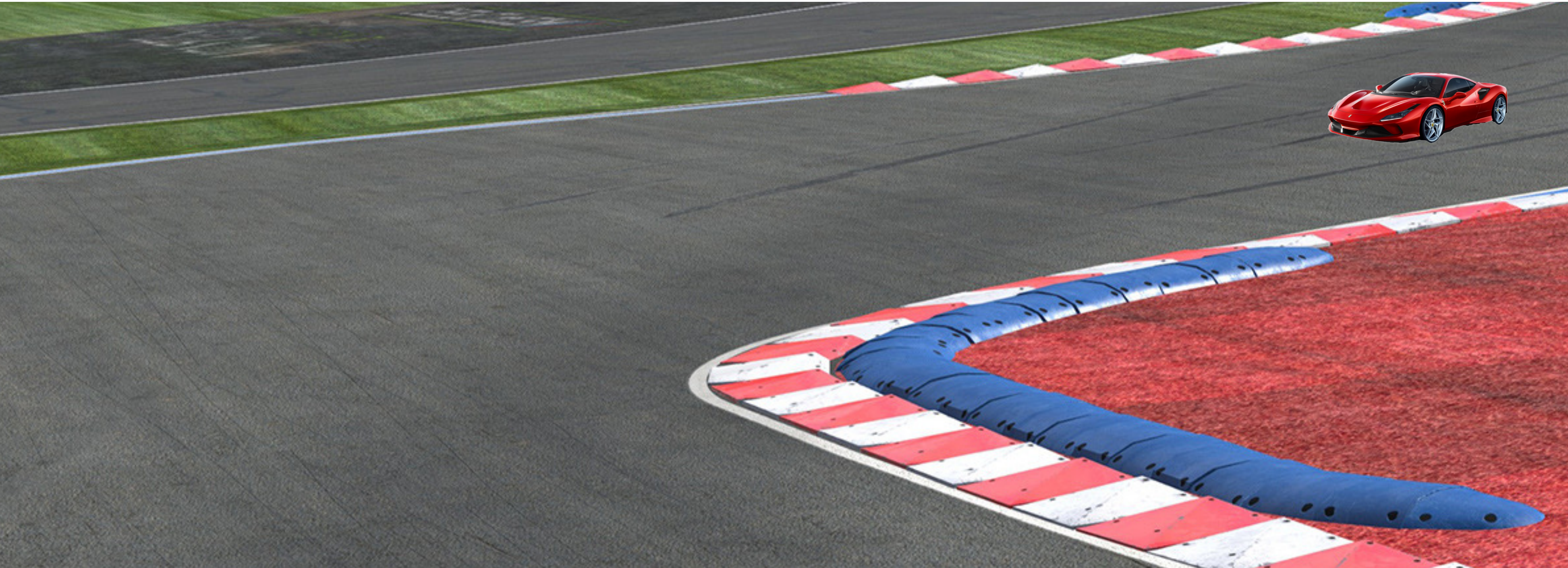
Outline

- Iterative Control Design for Deterministic Systems
- Autonomous Racing Experiments
- Uncertain Systems
- Multi-modal uncertainty and future steps

Autonomous Racing

Goal: Minimize lap time

Requirement: Guarantee safety



Learning Model Predictive Controller

Given $j-1$ trajectories, we define the following optimization problem:

$$J_{0\to N}^{\mathrm{LMPC},j}(x_t) = \min_{u_t,\ldots,u_{N-1}} \sum_{k=0}^{N-1} h(x_k, u_k) + V^{j-1}(x_N)$$

s.t.

$$x_{k+1} = A_k x_k + B_k u_k + C_k,$$

$$x_t = x_t,$$

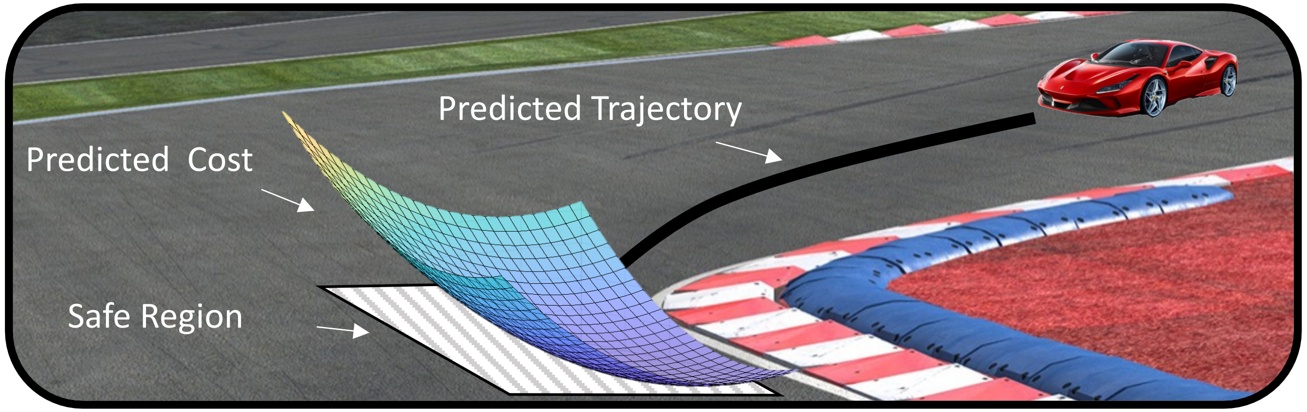
$$x_k \in \mathcal{X},\ u_k \in \mathcal{U},\ \forall k \in [0, \cdots, N-1]$$

$$x_N \in \mathcal{CS}^{j-1},$$

Value Function

Prediction Model

Safe Set



Learning Model Predictive Controller

Given $j-1$ trajectories, we define the following optimization problem:

$$J_{0\to N}^{\mathrm{LMPC},j}(x_t) = \min_{u_t,\dots,u_{N-1}} \sum_{k=0}^{N-1} h(x_k, u_k) + V^{j-1}(x_N)$$

s.t.

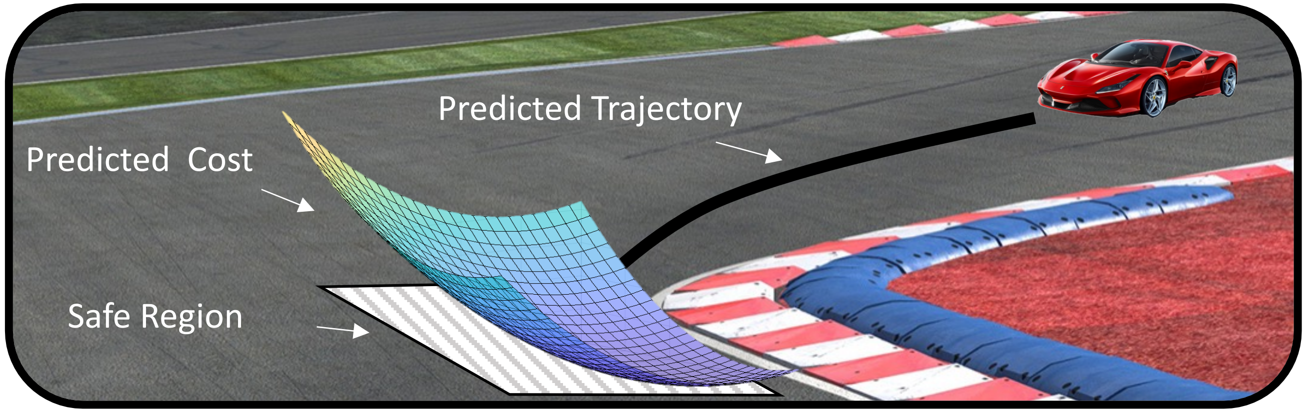
$$x_{k+1} = A_k x_k + B_k u_k + C_k,$$

$$x_t = x_t,$$

$$x_k \in \mathcal{X},\ u_k \in \mathcal{U},\ \forall k \in [0,\cdots,N-1]$$

$$x_N \in \ \mathcal{CS}^{j-1},$$

Prediction Model



System ID in Autonomous Racing

- Nonlinear Dynamical System,

$$
\begin{aligned}
\ddot{x} &= \dot{y}\dot{\psi} + \tfrac{1}{m}\textstyle\sum_i F_{x_i} \\
\ddot{y} &= -\dot{x}\dot{\psi} + \tfrac{1}{m}\textstyle\sum_i F_{y_i} \\
\ddot{\psi} &= \tfrac{1}{I_z}(a(F_{y_{1,2}}) - b(F_{y_{2,3}}) + c(-F_{x_{1,3}} + F_{x_{2,4}}) \\
\dot{X} &= \dot{x}\cos\psi - \dot{y}\sin\psi, \quad \dot{Y} = \dot{x}\sin\psi + \dot{y}\cos\psi
\end{aligned}
$$

System ID in Autonomous Racing

- Nonlinear Dynamical System,

$$
\begin{aligned}
\ddot{x} &= \dot{y}\dot{\psi} + \frac{1}{m}\sum_i F_{x_i} \\
\ddot{y} &= -\dot{x}\dot{\psi} + \frac{1}{m}\sum_i F_{y_i} \\
\ddot{\psi} &= \frac{1}{I_z}(a(F_{y_{1,2}}) - b(F_{y_{2,3}}) + c(-F_{x_{1,3}} + F_{x_{2,4}}) \\
&\boxed{\dot{X} = \dot{x}\cos\psi - \dot{y}\sin\psi, \quad \dot{Y} = \dot{x}\sin\psi + \dot{y}\cos\psi}
\end{aligned}
$$

Kinematic Equations

System ID in Autonomous Racing

- Nonlinear Dynamical System,

$$
\begin{aligned}
\ddot{x} &= \dot{y}\dot{\psi} + \frac{1}{m}\sum_i F_{x_i} \\
\ddot{y} &= -\dot{x}\dot{\psi} + \frac{1}{m}\sum_i F_{y_i} \\
\ddot{\psi} &= \frac{1}{I_z}(a(F_{y_{1,2}}) - b(F_{y_{2,3}}) + c(-F_{x_{1,3}} + F_{x_{2,4}}) \\
\dot{X} &= \dot{x}\cos\psi - \dot{y}\sin\psi, \quad \dot{Y} = \dot{x}\sin\psi + \dot{y}\cos\psi
\end{aligned}
$$

Kinematic Equations

- Identifying the Dynamical System

$$
x^j_{k+1|t} = \begin{bmatrix} \dot{x}_{k+1|t} \\ \dot{y}_{k+1|t} \\ \ddot{\psi} \\ \psi_{k+1|t} \\ X_{k+1|t} \\ Y_{k+1|t} \end{bmatrix} = \begin{bmatrix} \\ \\ \\ \text{Linearized Kinematics} \\ \text{Linearized Kinematics} \\ \text{Linearized Kinematics} \end{bmatrix} x^j_{k|t} + \begin{bmatrix} \\ \\ \\ \text{Linearized Kinematics} \\ \text{Linearized Kinematics} \\ \text{Linearized Kinematics} \end{bmatrix} \begin{bmatrix} u^j_{k|t} \\ 1 \end{bmatrix}
$$

Linearization around predicted trajectory

System ID in Autonomous Racing

- Nonlinear Dynamical System,

$$
\begin{aligned}
\ddot{x} &= \dot{y}\dot{\psi} + \frac{1}{m}\sum_i F_{x_i} \\
\ddot{y} &= -\dot{x}\dot{\psi} + \frac{1}{m}\sum_i F_{y_i} \\
\ddot{\psi} &= \frac{1}{I_z}(a(F_{y_{1,2}}) - b(F_{y_{2,3}}) + c(-F_{x_{1,3}} + F_{x_{2,4}})
\end{aligned}
$$

Dynamic Equations

$$
\dot{X} = \dot{x}\cos\psi - \dot{y}\sin\psi, \quad \dot{Y} = \dot{x}\sin\psi + \dot{y}\cos\psi
$$

Kinematic Equations

- Identifying the Dynamical System

$$
x^j_{k+1|t} = \begin{bmatrix} \dot{x}_{k+1|t} \\ \dot{y}_{k+1|t} \\ \ddot{\psi} \\ \psi_{k+1|t} \\ X_{k+1|t} \\ Y_{k+1|t} \end{bmatrix} = \begin{bmatrix} \\ \\ \\ \text{Linearized Kinematics} \\ \text{Linearized Kinematics} \\ \text{Linearized Kinematics} \end{bmatrix} x^j_{k|t} + \begin{bmatrix} \\ \\ \\ \text{Linearized Kinematics} \\ \text{Linearized Kinematics} \\ \text{Linearized Kinematics} \end{bmatrix} \begin{bmatrix} u^j_{k|t} \\ 1 \end{bmatrix}
$$

Linearization around predicted trajectory

System ID in Autonomous Racing

- Nonlinear Dynamical System,

$$\boxed{\begin{aligned}\ddot{x} &= \dot{y}\dot{\psi} + \frac{1}{m}\sum_i F_{x_i}\\ \ddot{y} &= -\dot{x}\dot{\psi} + \frac{1}{m}\sum_i F_{y_i}\\ \ddot{\psi} &= \frac{1}{I_z}(a(F_{y_{1,2}}) - b(F_{y_{2,3}}) + c(-F_{x_{1,3}} + F_{x_{2,4}})\end{aligned}}$$

Dynamic Equations

$$\boxed{\dot{X} = \dot{x}\cos\psi - \dot{y}\sin\psi, \quad \dot{Y} = \dot{x}\sin\psi + \dot{y}\cos\psi}$$

Kinematic Equations

- Identifying the Dynamical System

Local Linear Regression

$$x^j_{k+1|t} = \begin{bmatrix} \dot{x}_{k+1|t} \\ \dot{y}_{k+1|t} \\ \dot{\psi} \\ \psi_{k+1|t} \\ X_{k+1|t} \\ Y_{k+1|t} \end{bmatrix} = \left[\begin{array}{c} \boxed{\underset{\Lambda_y}{\operatorname{argmin}} \sum_{i,s} K(x^j_{k|t} - x^i_s)||\Lambda_y \begin{bmatrix} x^i_s \\ u^i_s \\ 1 \end{bmatrix} - y^i_{s+1}||, \forall y \in \{\dot{x}, \dot{y}, \ddot{\psi}\}} \\ \boxed{\begin{matrix}\text{Linearized Kinematics} \\ \text{Linearized Kinematics} \\ \text{Linearized Kinematics}\end{matrix}} \; x^j_{k|t} + \boxed{\begin{matrix}\text{Linearized Kinematics} \\ \text{Linearized Kinematics} \\ \text{Linearized Kinematics}\end{matrix}} \end{array} \right] \begin{bmatrix} u^j_{k|t} \\ 1 \end{bmatrix}$$

Linearization around predicted trajectory

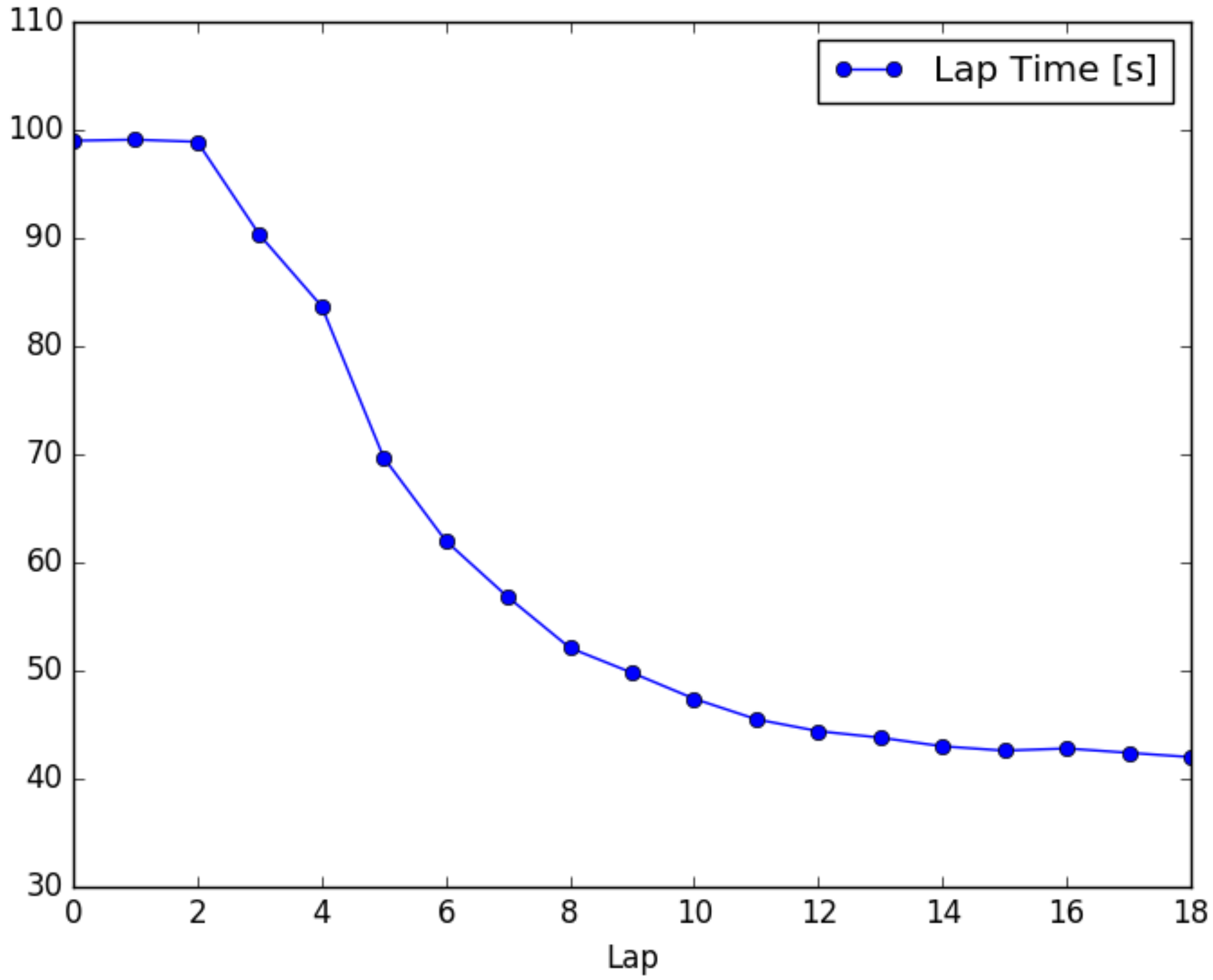
Hyundai California Proving Ground

Hyundai California Proving Ground

Learning Model Predictive Controller full-size vehicle experiments

Credits: Siddharth Nair, Nitin Kapania and Ugo Rosolia

Lap Time



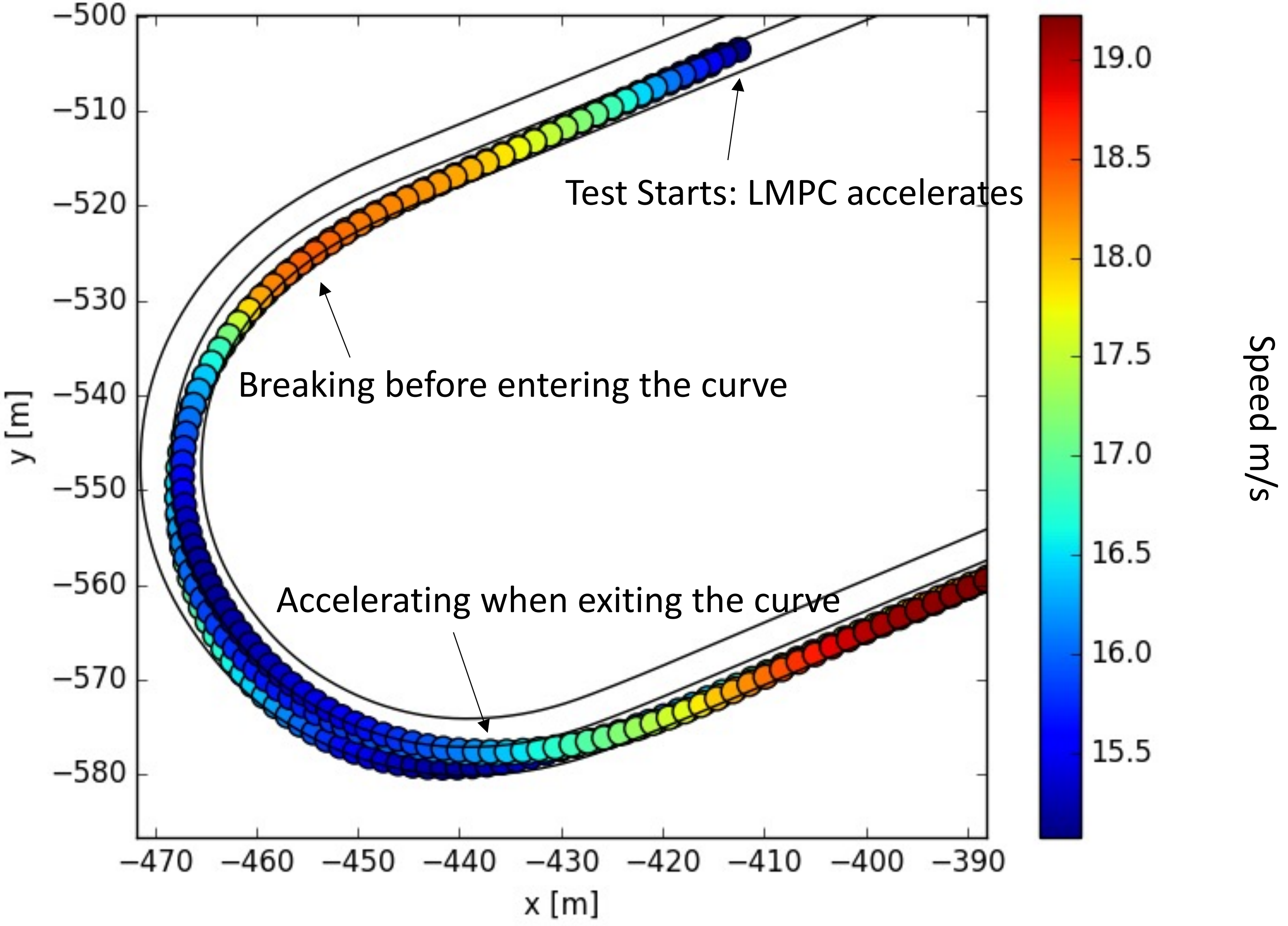
The control policy is constructed using ∼1k data points (last 2 laps)

The control action is computed using ∼100 data points

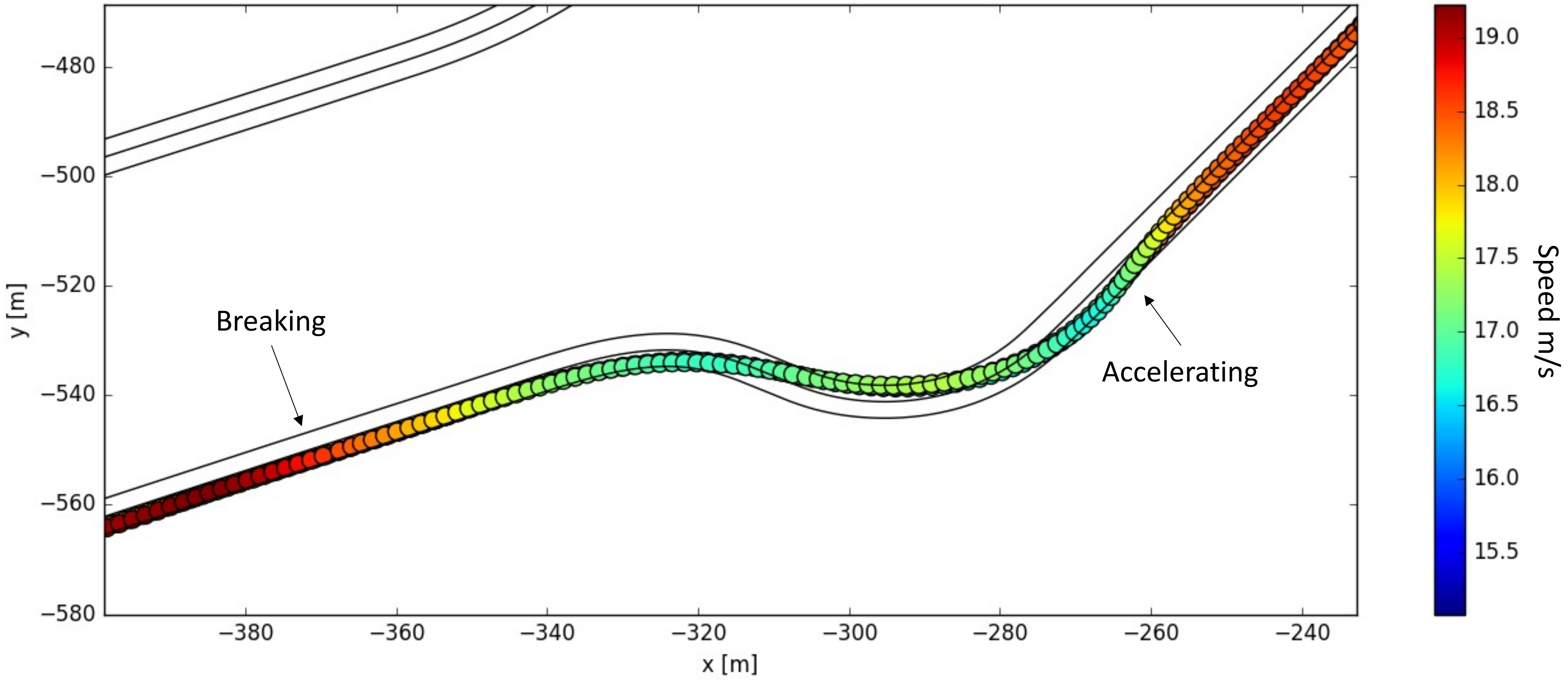
Learning Model Predictive Controller full-size vehicle experiments

Credits: Siddharth Nair, Nitin Kapania and Ugo Rosolia

Velocity Profile at Convergence (Curve 1)

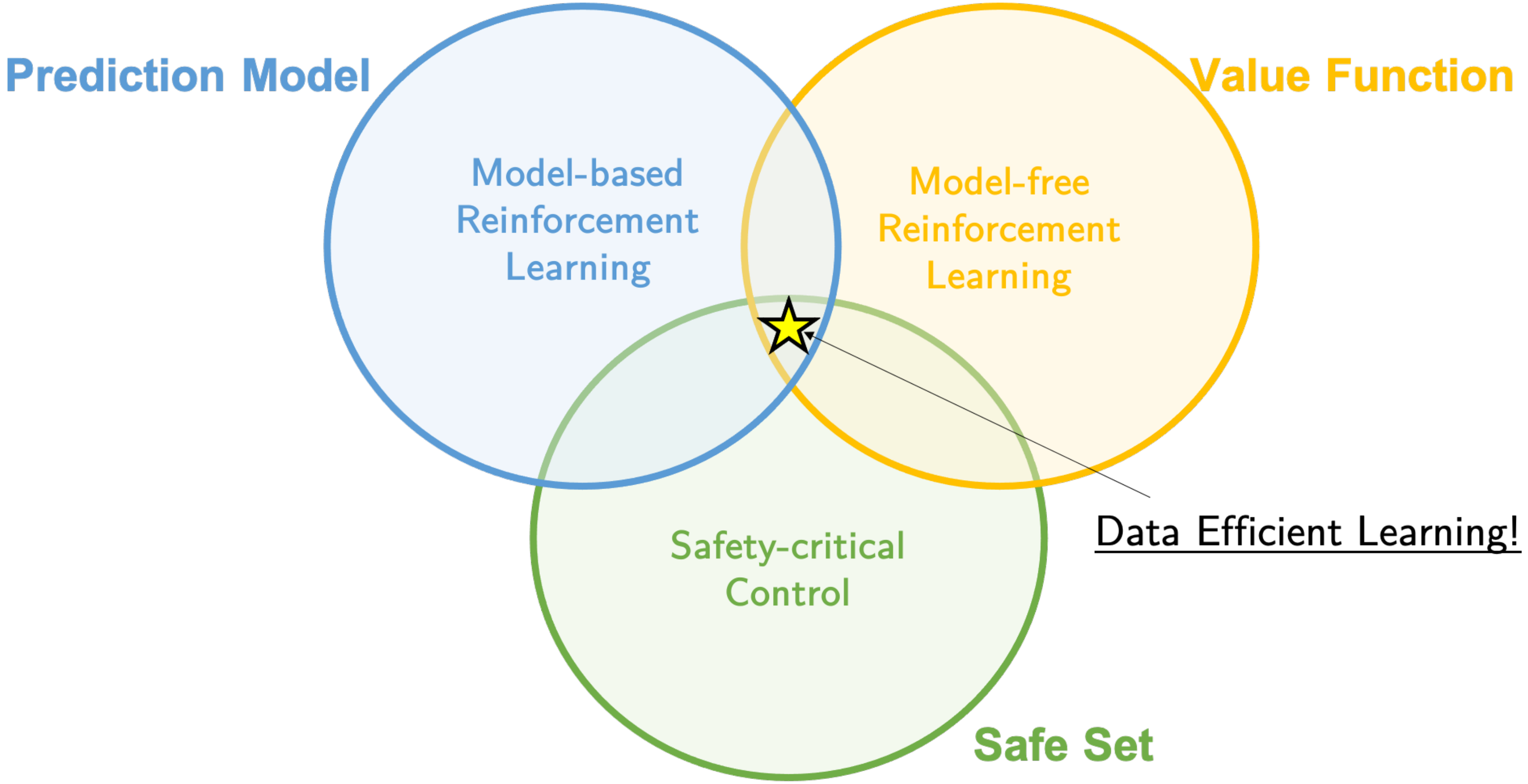


Velocity Profile at Convergence (Chicane)



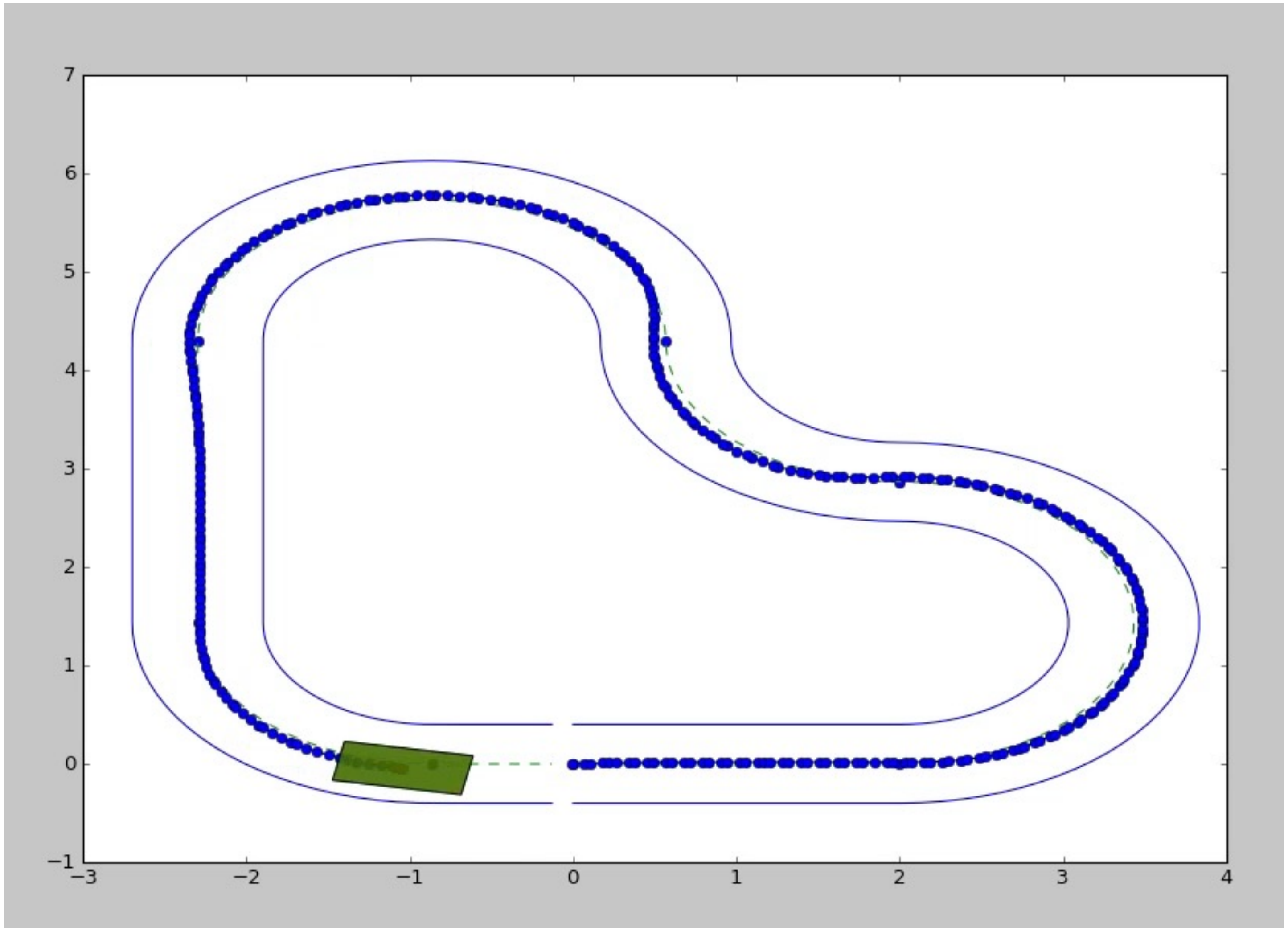
The key components

- Predicted trajectory given by **prediction model**
- Predicted cost estimated by **value function**
- Safe region estimated by the **safe set**



Do you need the safe set? – Yes
LMPC without Invariant Set

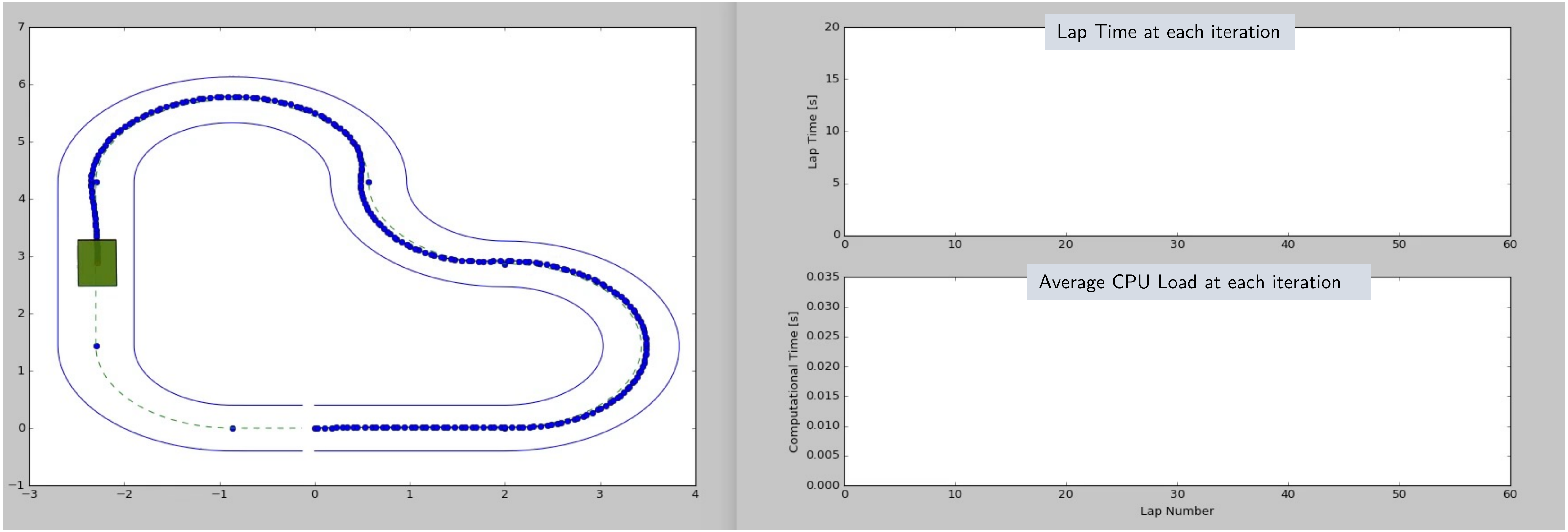
The controller extrapolates the Q-function on the Vx dimension



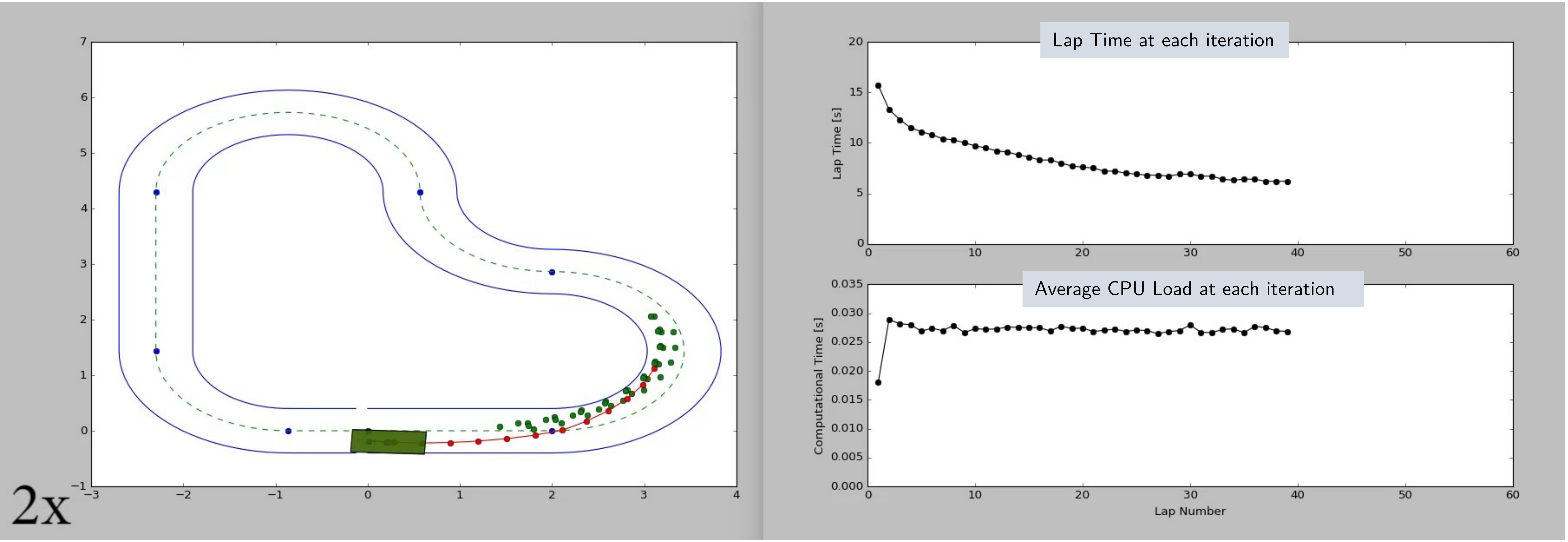
Do you need to Predict to Learn? Yes

When the LMPC horizon is $N = 1$ the controller

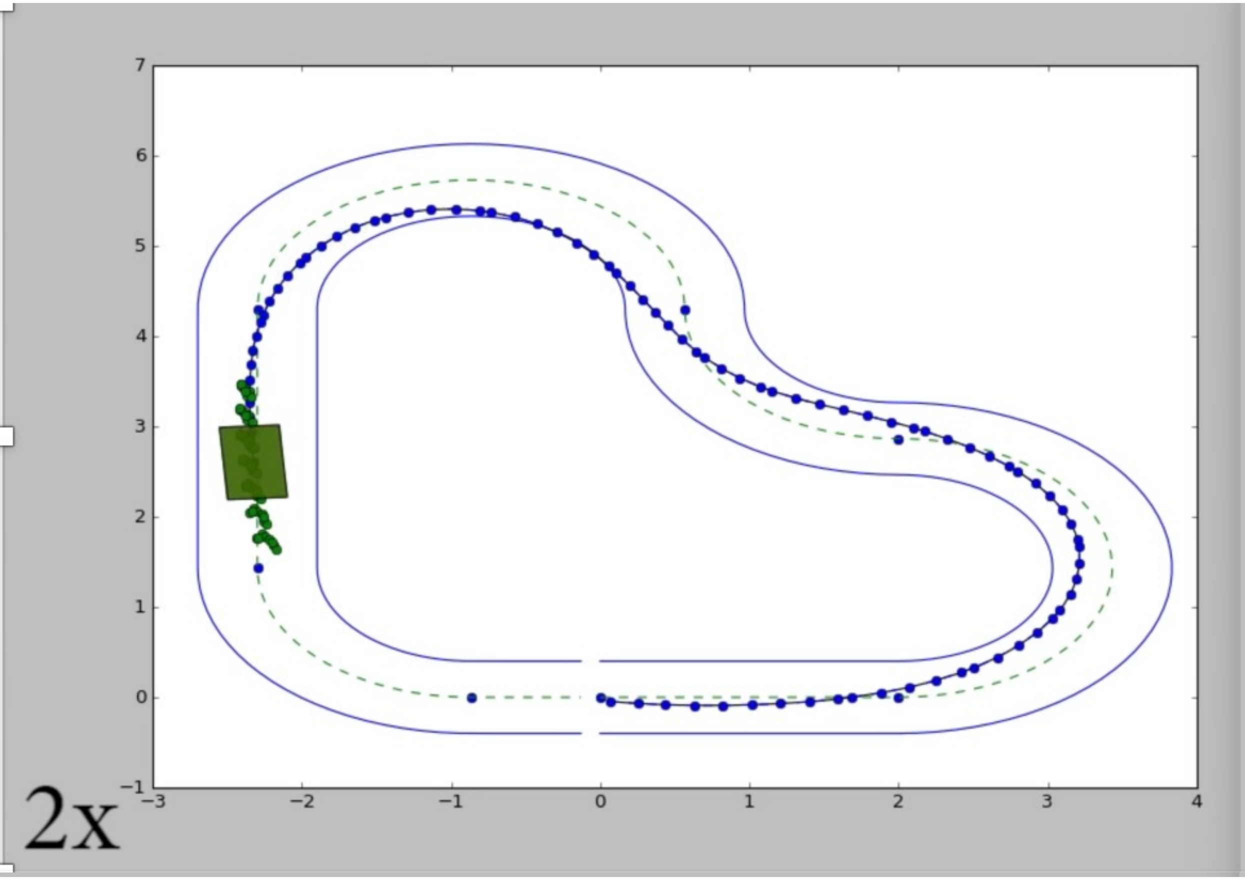
- solves the Bellman equation using the Q-function as value function approximation
- does not explore the state space as it cannot plan outside the safe set

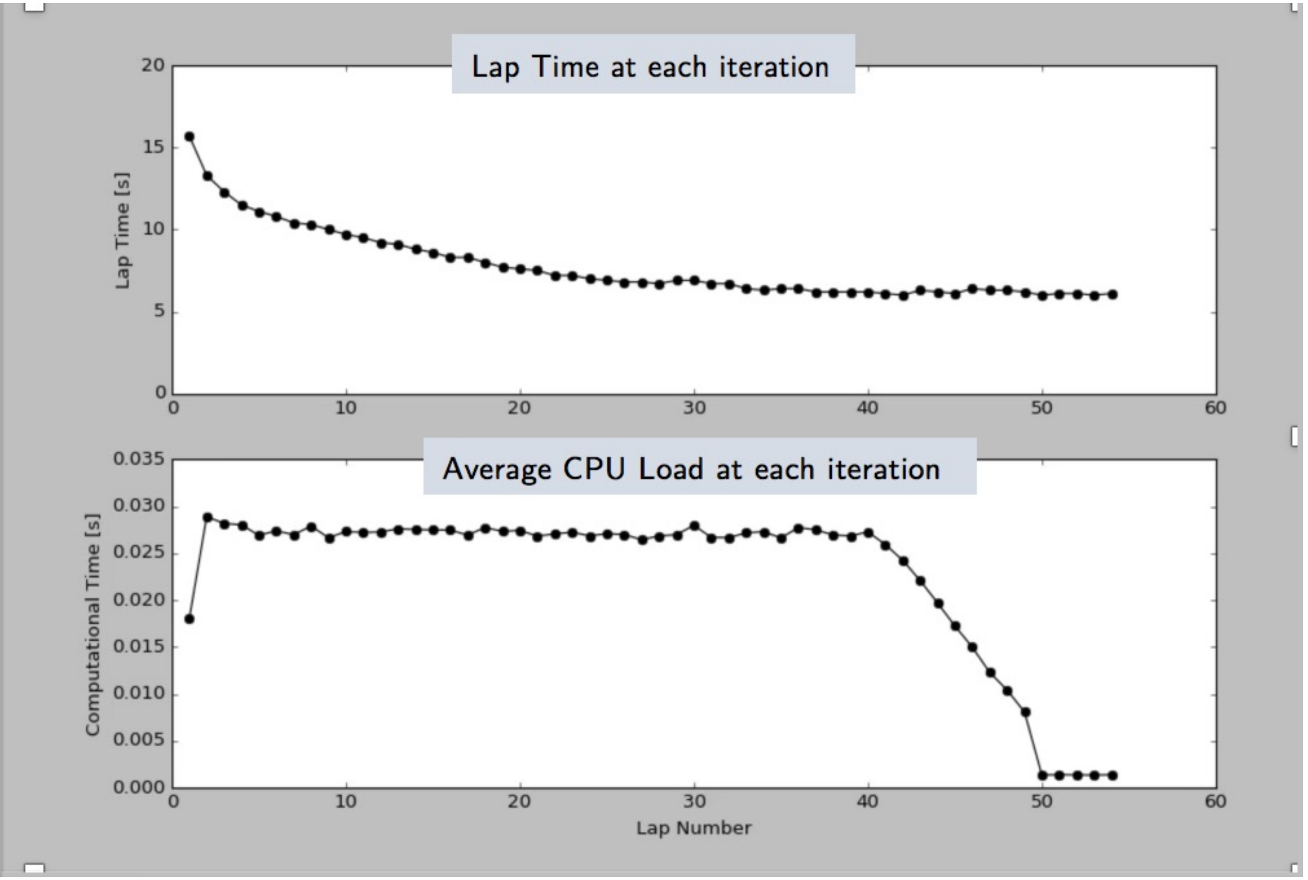


Do you need to Predict at Convergence? No



Do you need to Predict at Convergence? No

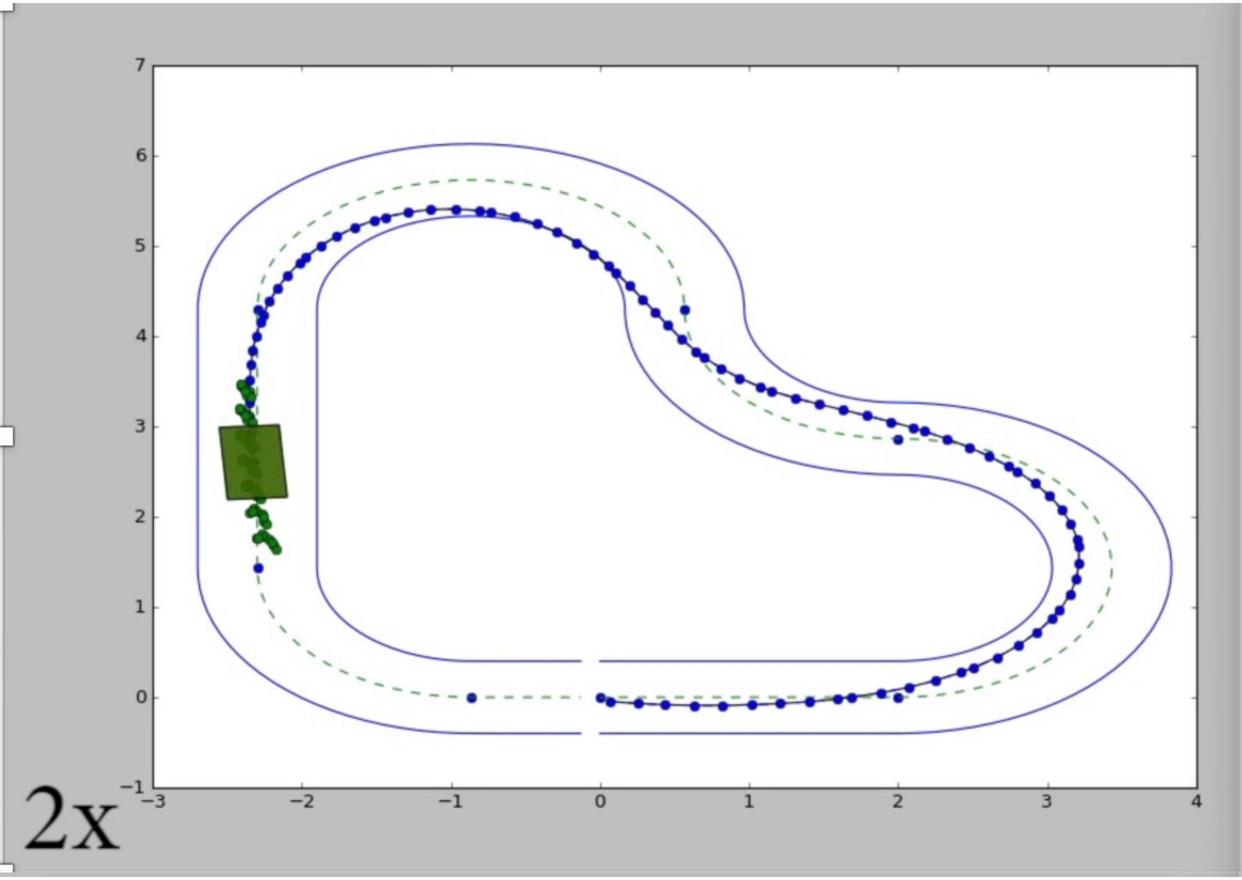


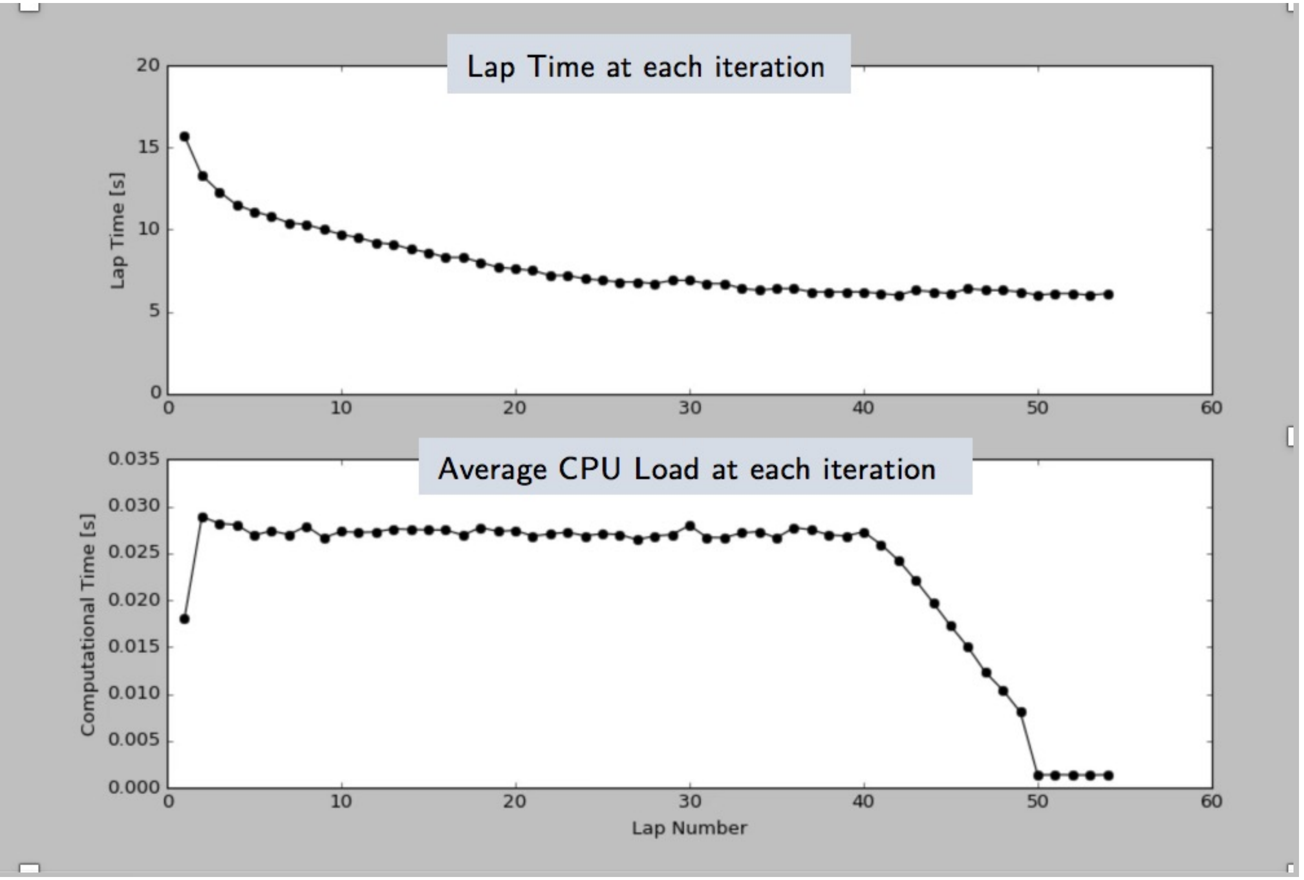


Value Function Approximation

$$
\begin{aligned}
[\lambda_0^{0,*}, \dots, \lambda_i^{j,*}] = \arg\min_{\lambda_i^j \in [0,1]} \quad & \textstyle\sum_i \sum_j J_i^j \lambda_i^j \\
\text{s.t} \quad & \textstyle\sum_i \sum_j x_i^j \lambda_i^j = x(t), \\
& \textstyle\sum_i \sum_j \lambda_i^j = 1
\end{aligned}
$$

Do you need to Predict at Convergence? No





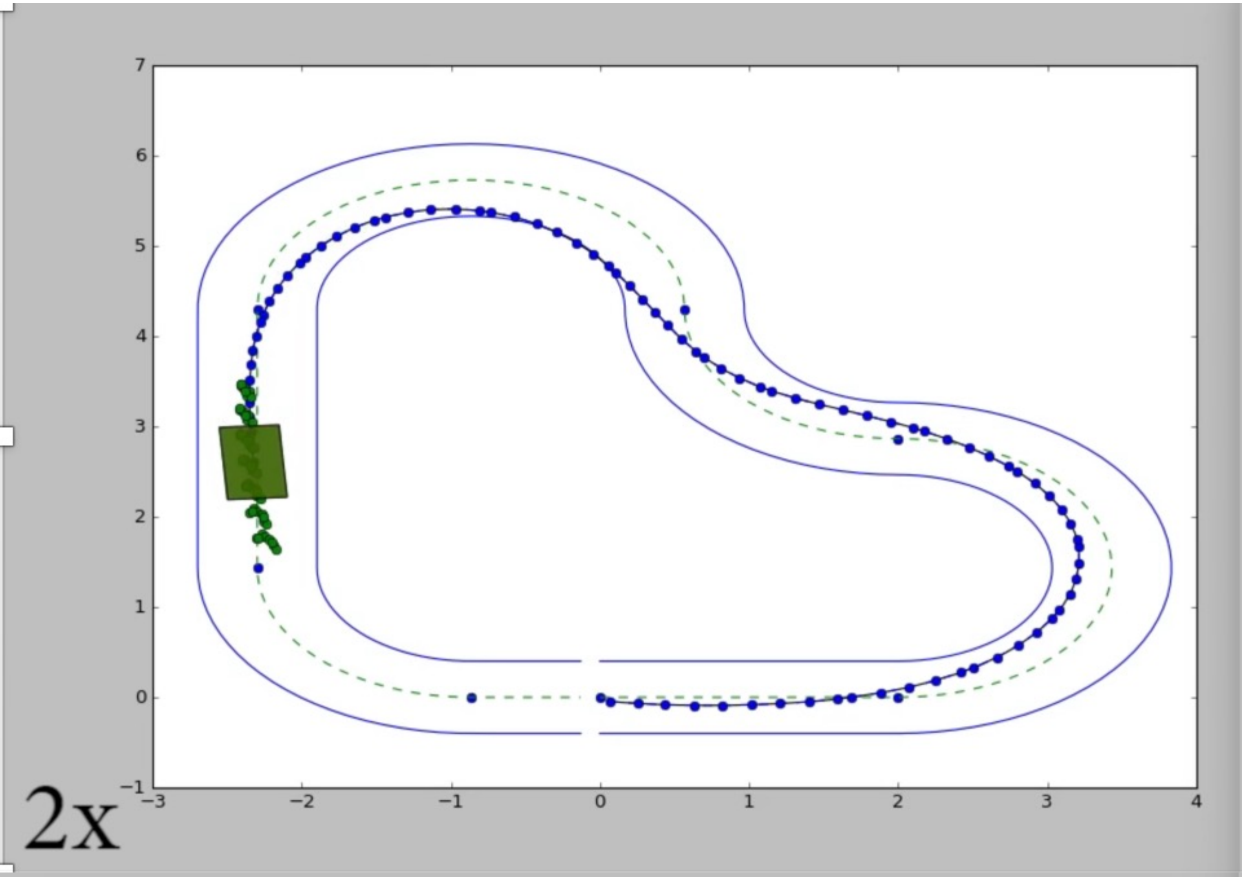
Value Function Approximation

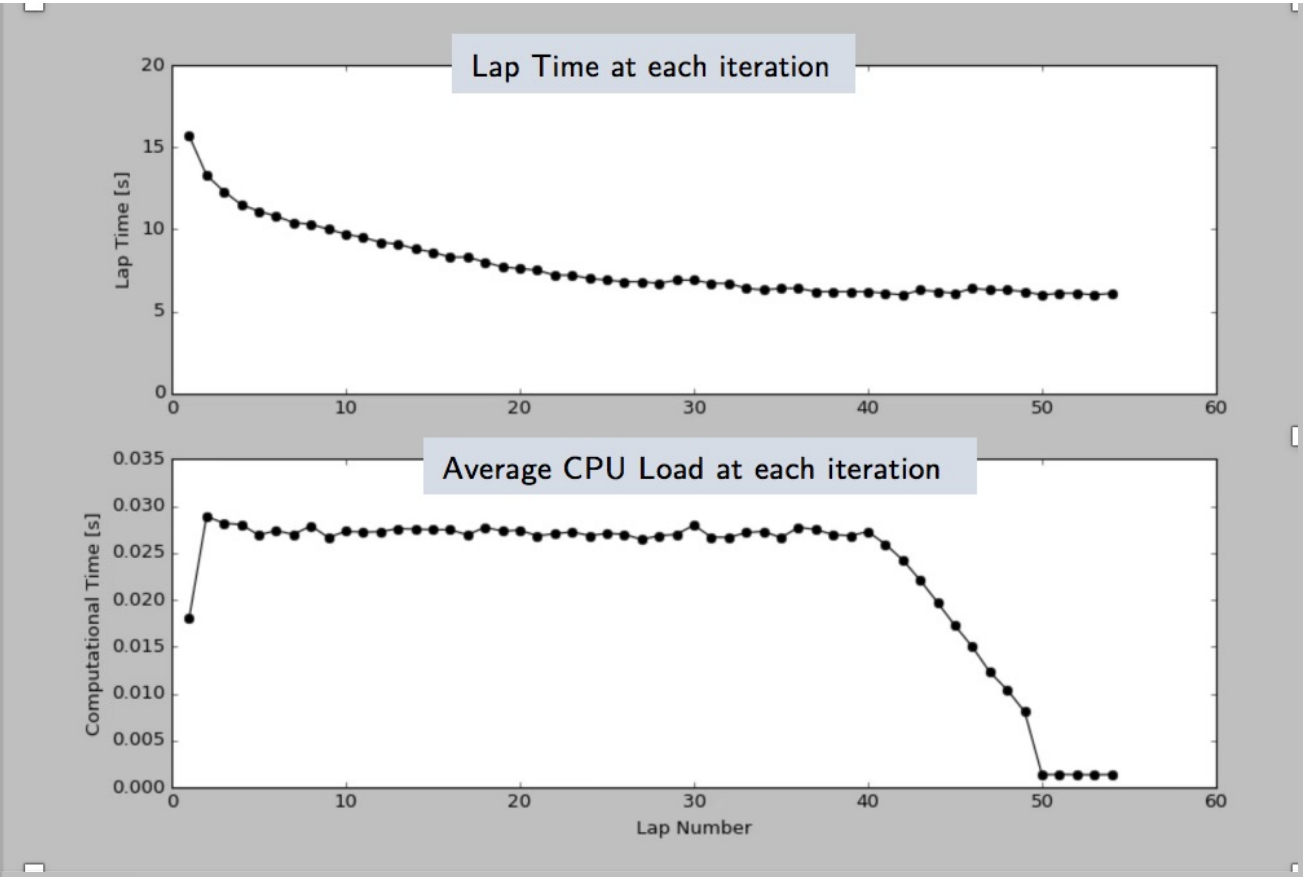
$$
\begin{aligned}
[\lambda_0^{0,*}, \ldots, \lambda_i^{j,*}] = \arg\min_{\lambda_i^j \in [0,1]} \quad & \textstyle\sum_i \sum_j J_i^j \lambda_i^j \\
\text{s.t} \quad & \textstyle\sum_i \sum_j x_i^j \lambda_i^j = x(t), \\
& \textstyle\sum_i \sum_j \lambda_i^j = 1
\end{aligned}
$$

Control Policy

$$\pi\big(x(t)\big) = \textstyle\sum_i \sum_j u_i^j \lambda_i^{j,*}$$

Do you need to Predict at Convergence? No





Value Function Approximation

$$
\begin{aligned}
[\lambda_0^{0,*}, \dots, \lambda_i^{j,*}] = \arg\min_{\lambda_i^j \in [0,1]} \quad & \textstyle\sum_i \sum_j J_i^j \lambda_i^j \\
\text{s.t} \quad & \textstyle\sum_i \sum_j x_i^j \lambda_i^j = x(t), \\
& \textstyle\sum_i \sum_j \lambda_i^j = 1
\end{aligned}
$$

Control Policy

Stored Data

$$\pi\big(x(t)\big) = \sum_i \sum_j u_i^j \lambda_i^{j,*}$$

Outline

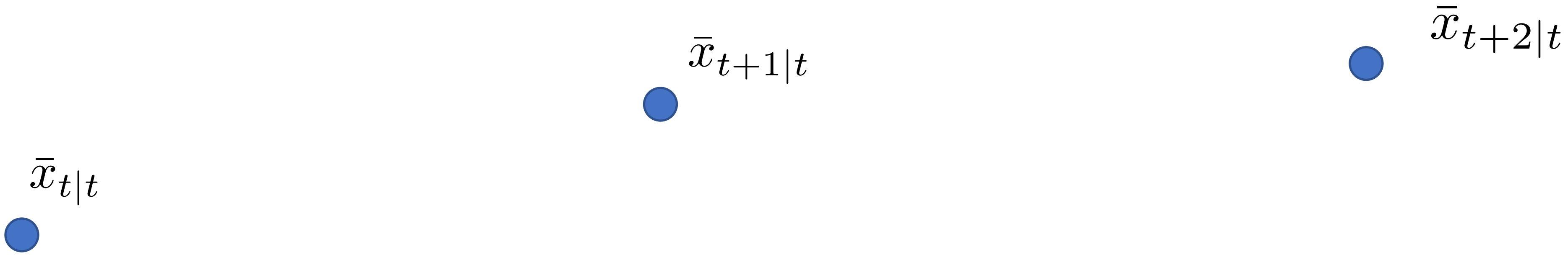
- Iterative Control Design for Deterministic Systems
- Autonomous Racing Experiments
- Uncertain Systems
- Multi-modal uncertainty and future steps

Outline

- Iterative Control Design for Deterministic Systems
- Autonomous Racing Experiments
- Uncertain Systems
- Multi-modal uncertainty and future steps

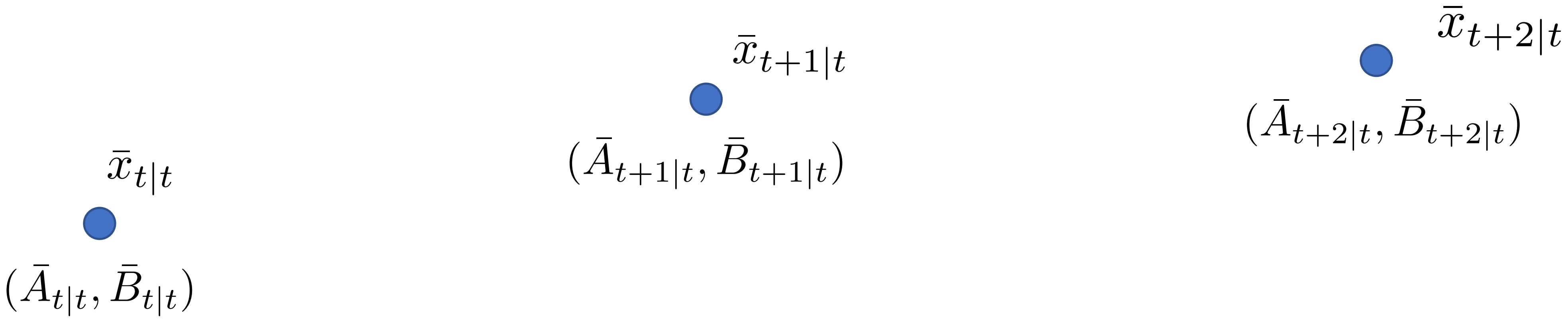
Model Estimation: An Iterative Linearization Strategy

Model Estimation: An Iterative Linearization Strategy



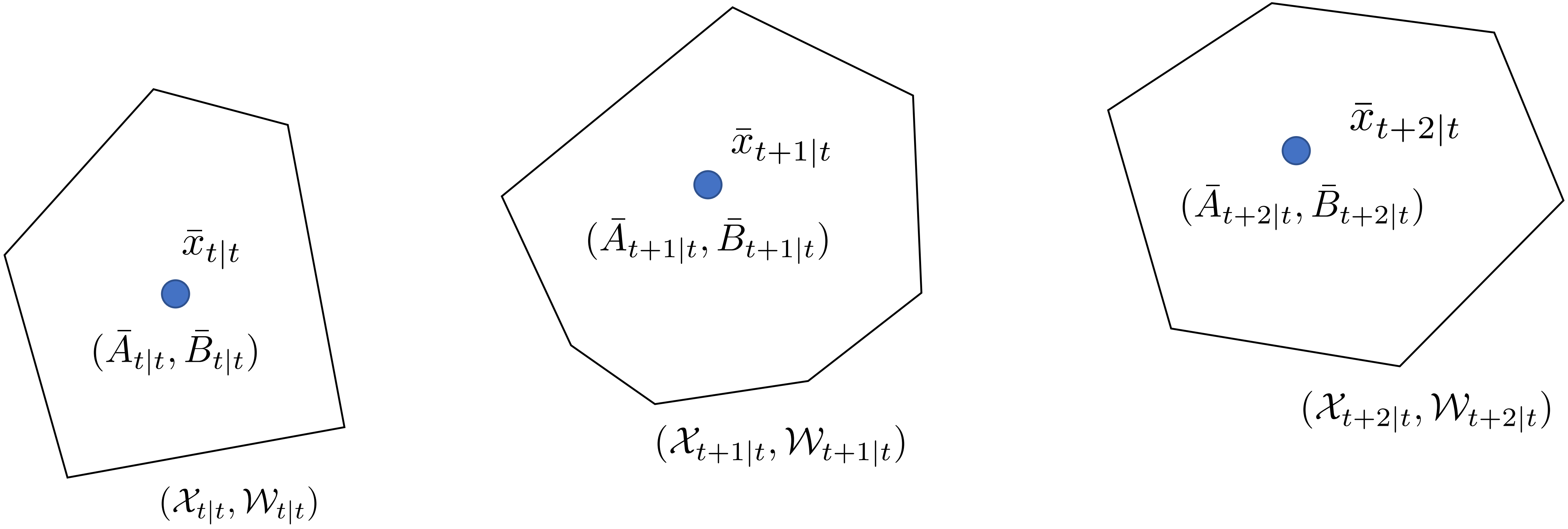
Model Estimation: An Iterative Linearization Strategy

- Linearize estimated dynamics around a candidate trajectory



Model Estimation: An Iterative Linearization Strategy

- Linearize estimated dynamics around a candidate trajectory
- Estimate confidence intervals where the system dynamics are accurate



Model Estimation: An Iterative Linearization Strategy

- Linearize estimated dynamics around a candidate trajectory
- Estimate confidence intervals where the system dynamics are accurate
- Probabilistic guarantees for closed-loop constraint satisfaction

$$
\begin{aligned}
J^*(x(t)) = \min_{\boldsymbol{u_t}} \max_{\boldsymbol{w_t}} \quad & \sum_{k=t}^{t+N-1} h(x_{k|t}, u_{k|t}) + Q(x_{t+T_t|t}) \\
\text{s.t.} \quad & x_{k+1|t} = \boxed{\bar{A}_{k|t}} x_{k|t} + \boxed{\bar{B}_{k|t}} u_{k|t} + w_{k|t} \\
& u_{k|t} \in \mathcal{U}_{k|t}, x_{k|t} \in \boxed{\mathcal{X}_{k|t}} \\
& x_{t|t} = x(t) \\
& x_{t+N|t} \in \mathcal{O} \\
& \forall w_{k|t} \in \boxed{\mathcal{W}_{k|t}}, \forall k = t, \ldots, t+N-1.
\end{aligned}
$$

Linearized Estimate

“Trust region” and uncertainty

Model Estimation: An Iterative Linearization Strategy

- Linearize estimated dynamics around a candidate trajectory
- Estimate confidence intervals where the system dynamics are accurate
- Probabilistic guarantees for closed-loop constraint satisfaction

$$
\begin{aligned}
J^*(x(t)) = \min_{\boldsymbol{u_t}} \max_{\boldsymbol{w_t}} \quad & \sum_{k=t}^{t+N-1} h(x_{k|t}, u_{k|t}) + Q(x_{t+T_t|t}) \\
\text{s.t.} \quad & x_{k+1|t} = \boxed{\bar{A}_{k|t}} x_{k|t} + \boxed{\bar{B}_{k|t}} u_{k|t} + w_{k|t} \\
& u_{k|t} \in \mathcal{U}_{k|t}, x_{k|t} \in \boxed{\mathcal{X}_{k|t}} \\
& x_{t|t} = x(t) \\
& x_{t+N|t} \in \mathcal{O} \\
& \forall w_{k|t} \in \boxed{\mathcal{W}_{k|t}}, \forall k = t, \ldots, t+N-1.
\end{aligned}
$$

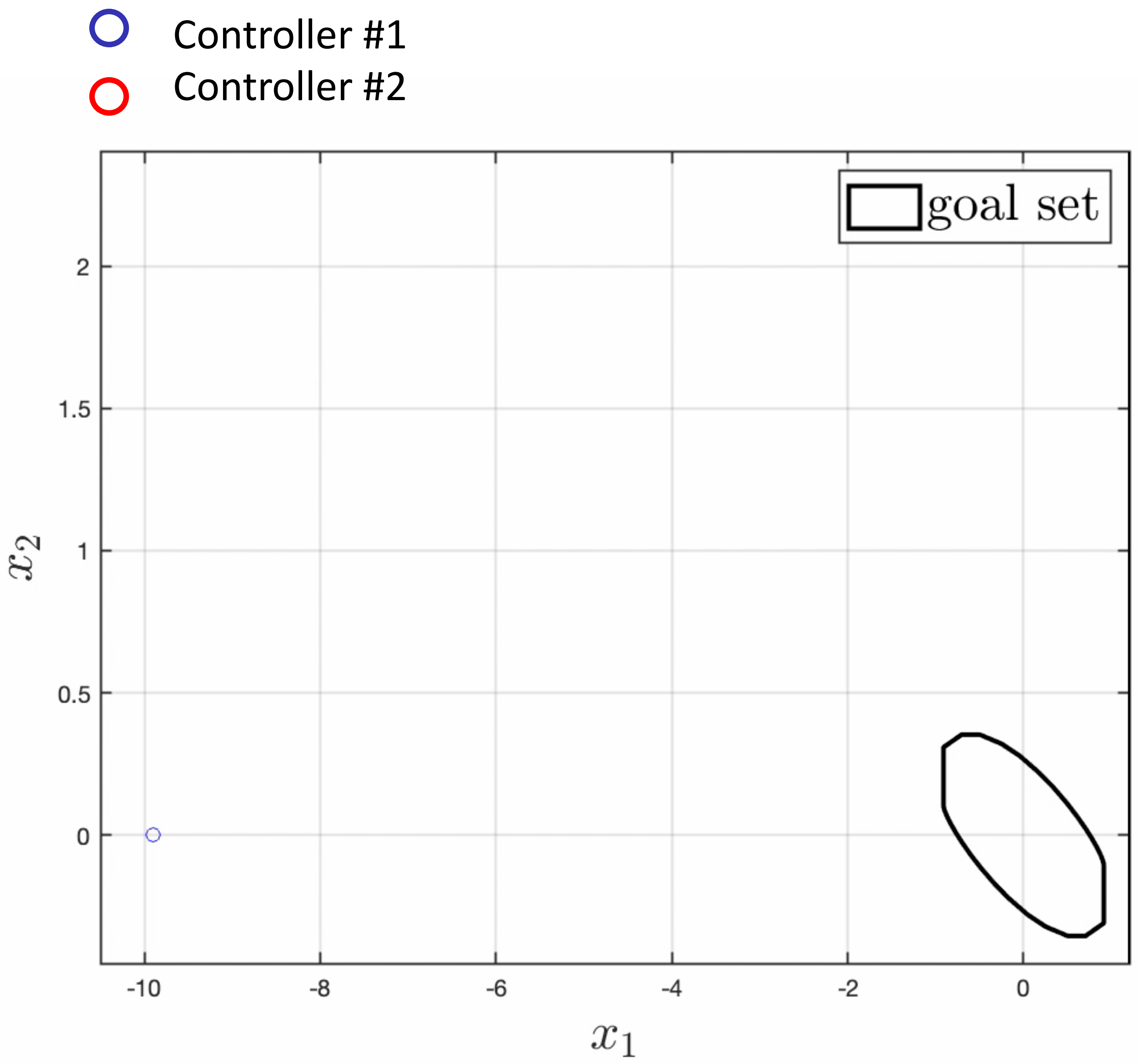
Linearized Estimate

“Trust region” and uncertainty

“Control of Unknown Nonlinear Systems with Linear Time-Varying MPC.” D. Papadimitriou, U. Rosolia, and F. Borrelli. in 2020 Conference on Decision and Control

Safe Sets and Value Functions Estimation via Sampling

Safe Sets and Value Functions Estimation via Sampling



- Collect several trajectories with the same controller
- Safe sets computed as before using multiple trajectories
- Value functions estimate either the mean or worst-case cost
- All statement hold with some probability that is proportional to the amount of data

U. Rosolia, and F. Borrelli. "Sample-based learning model predictive control for linear uncertain systems." In 2019 IEEE 58th Conference on Decision and Control (CDC). IEEE, 2019.
U. Rosolia, X. Zhang, and F. Borrelli. "Robust learning model predictive control for linear systems performing iterative tasks." IEEE Transactions on Automatic Control (2021).

Outline

- Iterative Control Design for Deterministic Systems
- Autonomous Racing Experiments
- Uncertain Systems
- Multi-modal uncertainty and future steps

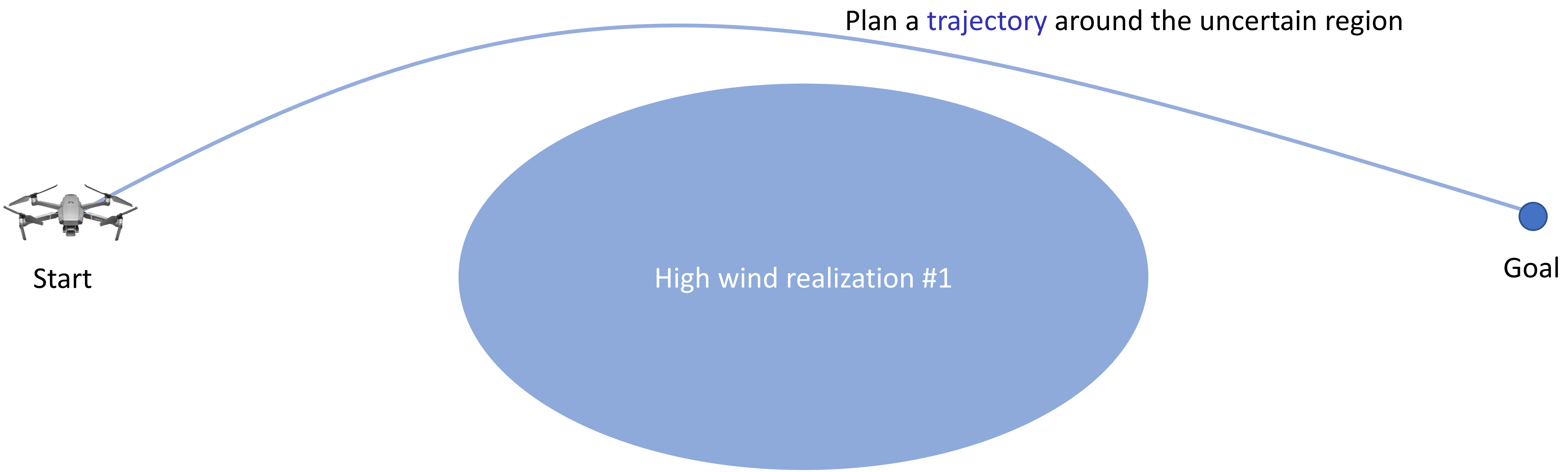
Outline

- Iterative Control Design for Deterministic Systems
- Autonomous Racing Experiments
- Uncertain Systems
- Multi-modal uncertainty and future steps

Why multi-modal uncertainty?

Why multi-modal uncertainty?

Why multi-modal uncertainty?

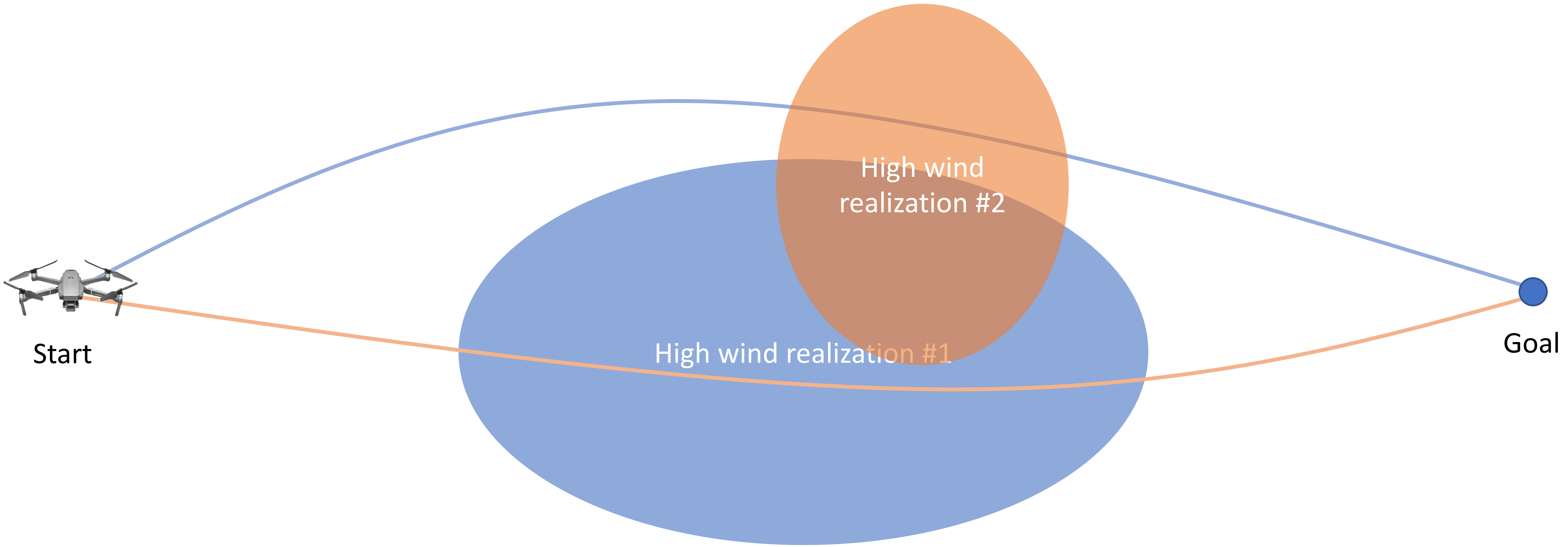


Why multi-modal uncertainty?

Why multi-modal uncertainty?

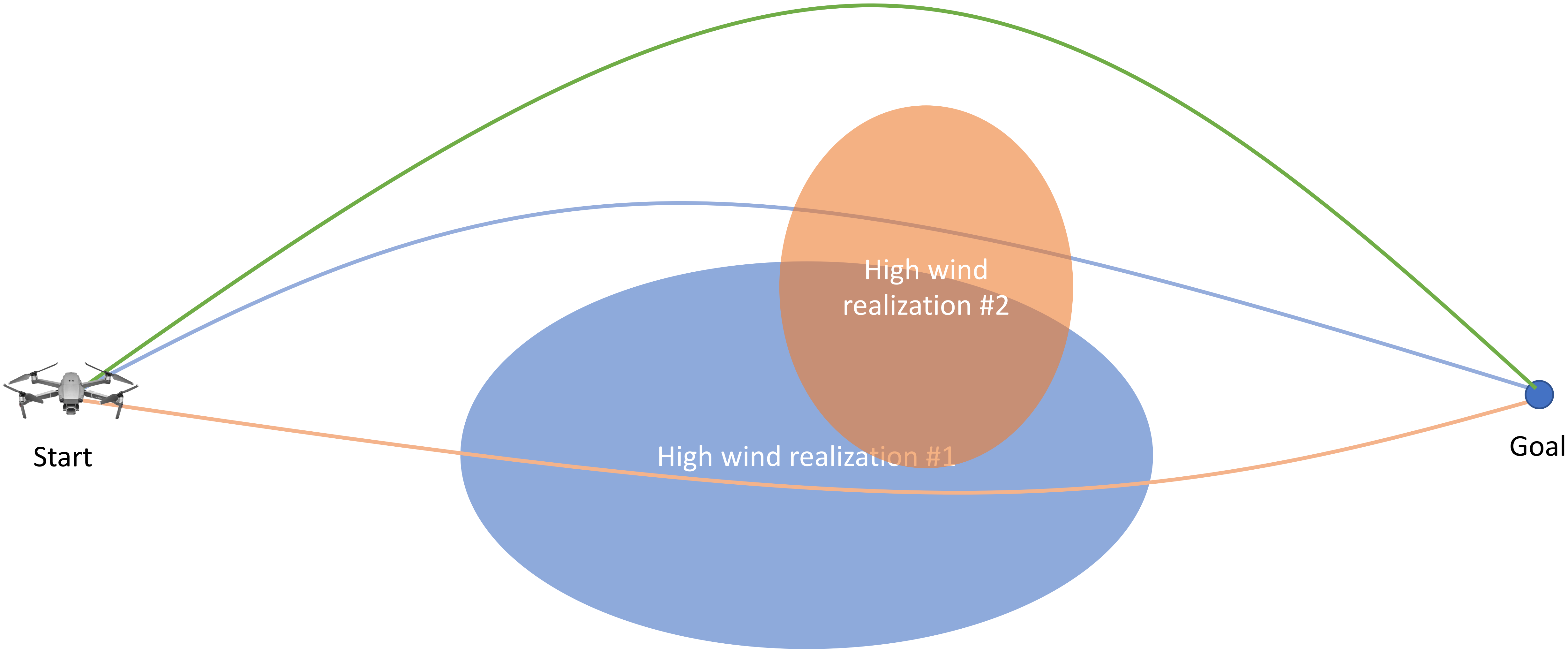
Plan a trajectory around the uncertain region

Why multi-modal uncertainty?

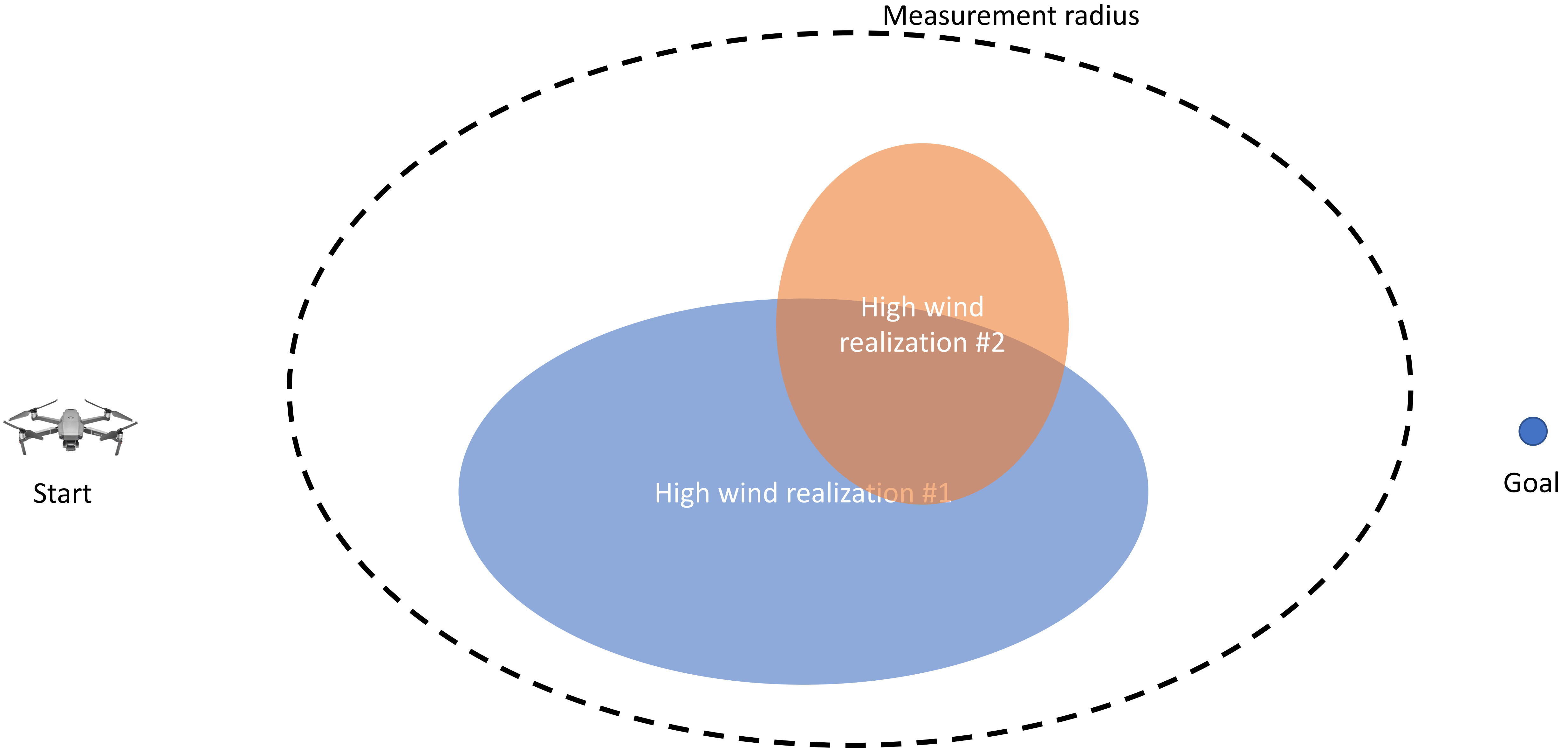


Why multi-modal uncertainty?

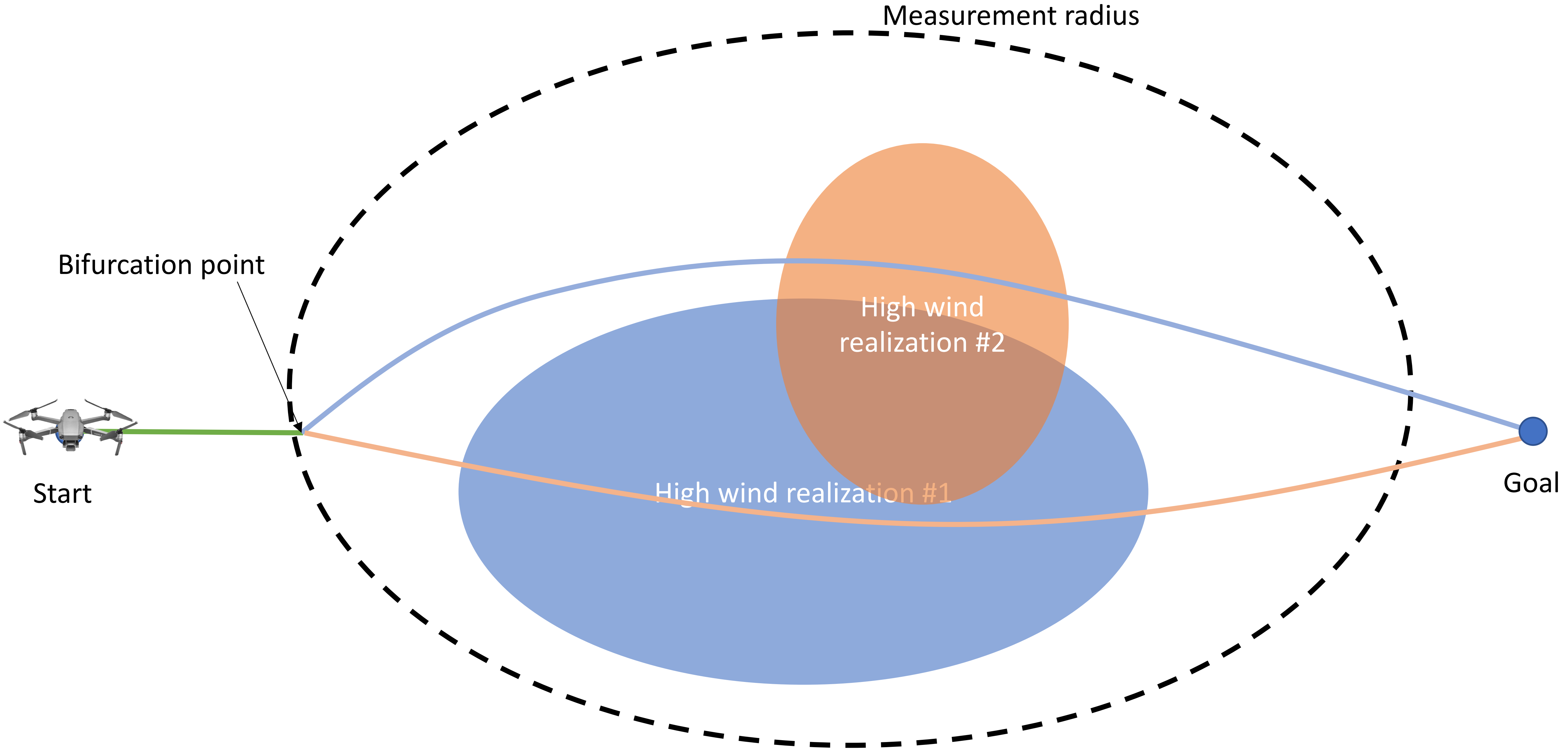
Plan a trajectory around the both uncertain regions



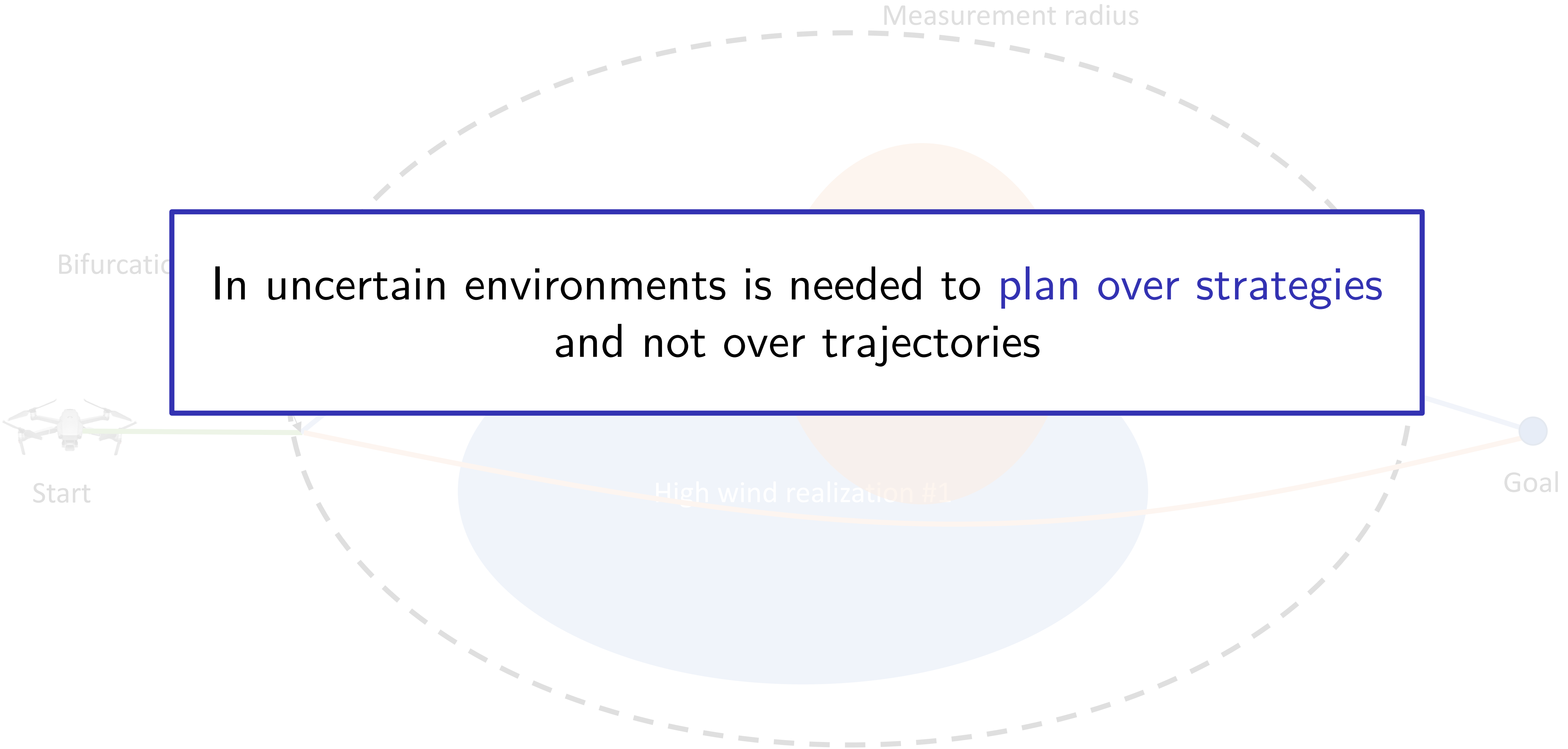
Why multi-modal uncertainty?



Why multi-modal uncertainty?

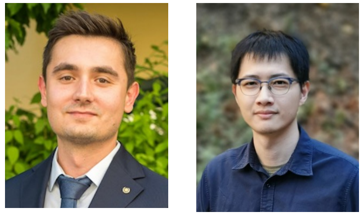


Why multi-modal uncertainty?

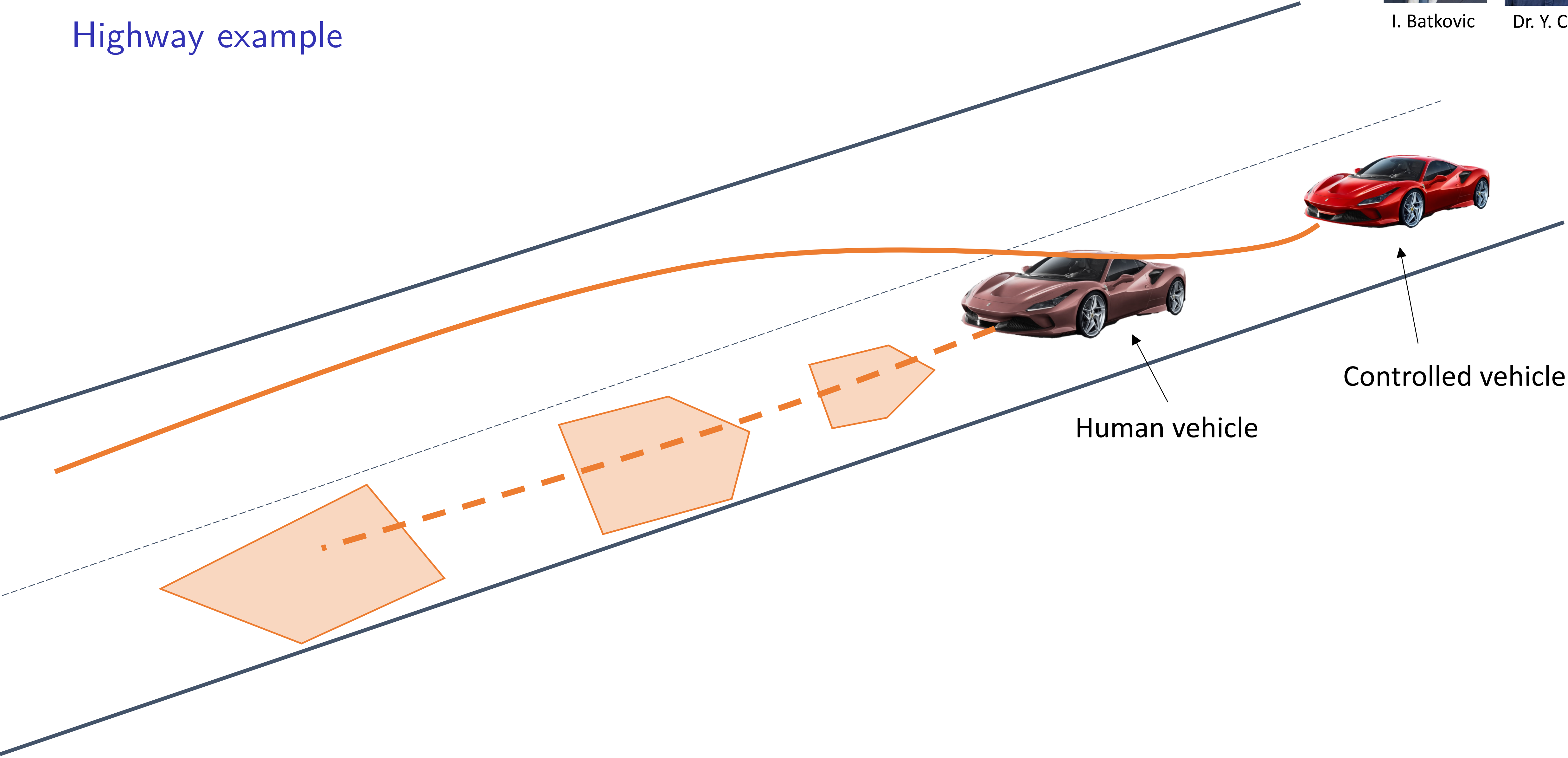


In uncertain environments is needed to **plan over strategies** and not over trajectories

Planning in Multi-modal Uncertain Environments


I. Batkovic Dr. Y. Chen

Highway example

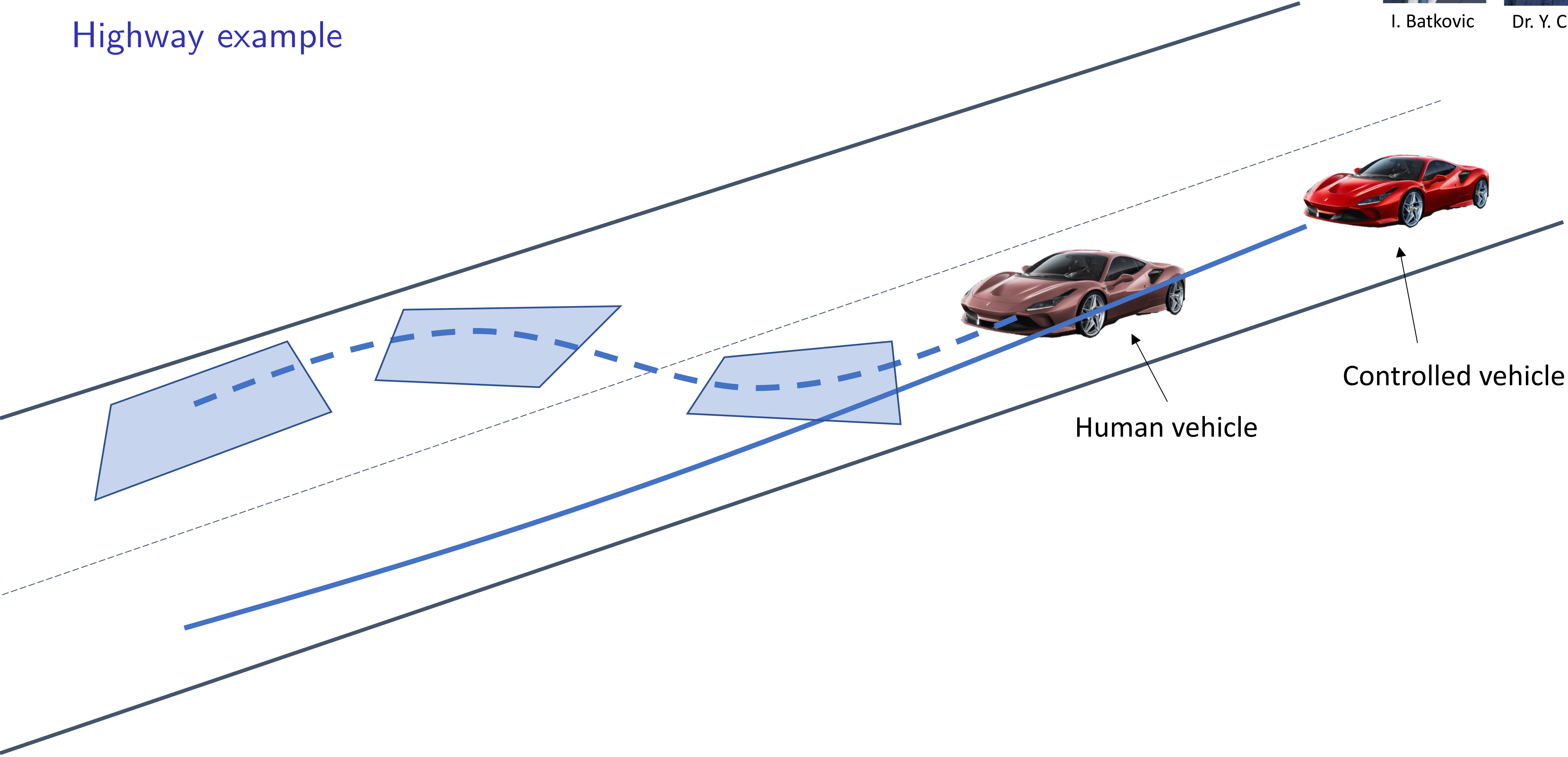


Planning in Multi-modal Uncertain Environments

I. Batkovic

Dr. Y. Chen

Highway example

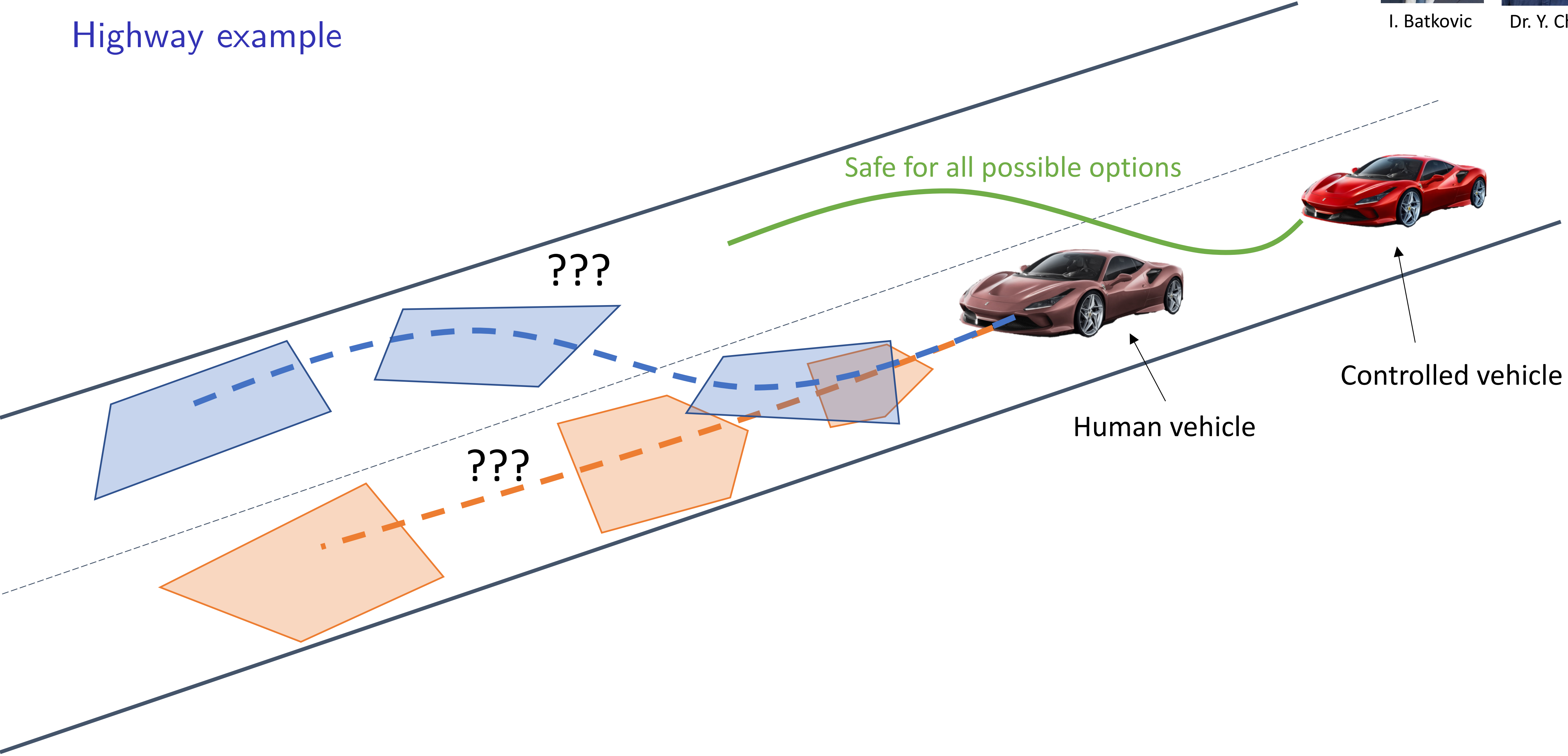


Planning in Multi-modal Uncertain Environments


I. Batkovic

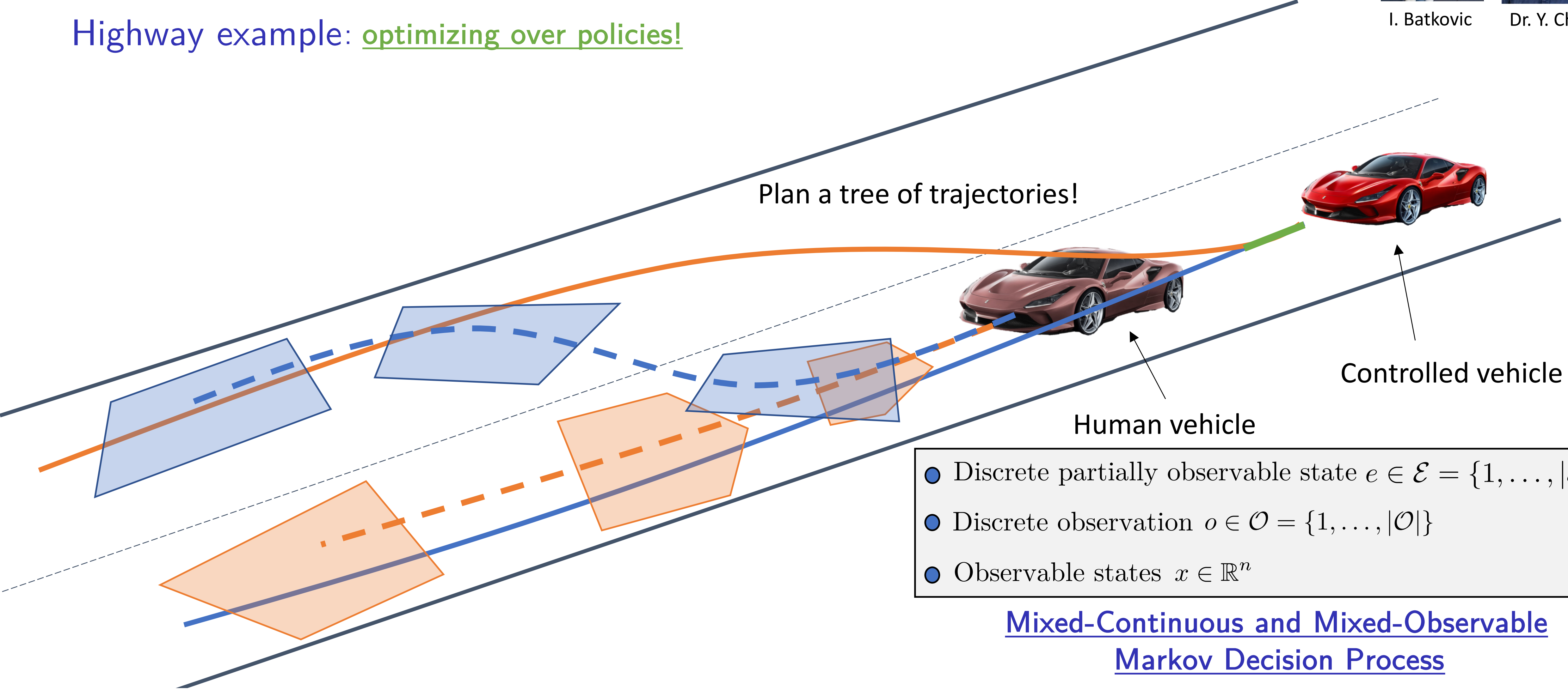

Dr. Y. Chen

Highway example



Planning in Multi-modal Uncertain Environments

Highway example: optimizing over policies!



- Discrete partially observable state $e \in \mathcal{E} = \{1, \ldots, |\mathcal{E}|\}$
- Discrete observation $o \in \mathcal{O} = \{1, \ldots, |\mathcal{O}|\}$
- Observable states $x \in \mathbb{R}^n$

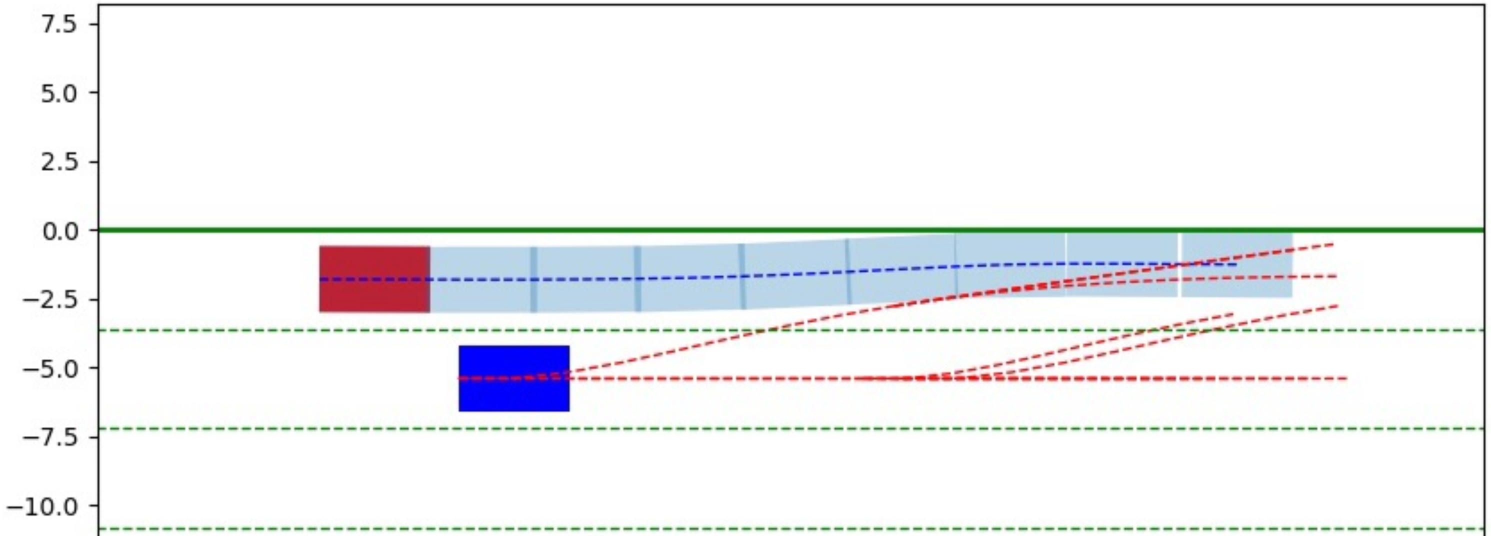
Mixed-Continuous and Mixed-Observable Markov Decision Process

I. Batkovic, U. Rosolia, M. Zanon, and P. Falcone. "A Robust Scenario MPC Approach for Uncertain Multi-modal Obstacles." IEEE Control Systems Letters 5, no. 3 (2020): 947-952.
Y. Chen, U. Rosolia, W. Ubellacker, N. Csomay-Shanklin,and A. D. Ames "Interactive multi-modal motion planning with Branch Model Predictive Control" to appear on RA-L.
U. Rosolia, Y. Chen, S. Daftry, M. Ono, Y. Yue, and A.D. Ames. "The mixed-observable constrained linear quadratic regulator problem: the exact solution and practical algorithms" arXiv:2108.12030.

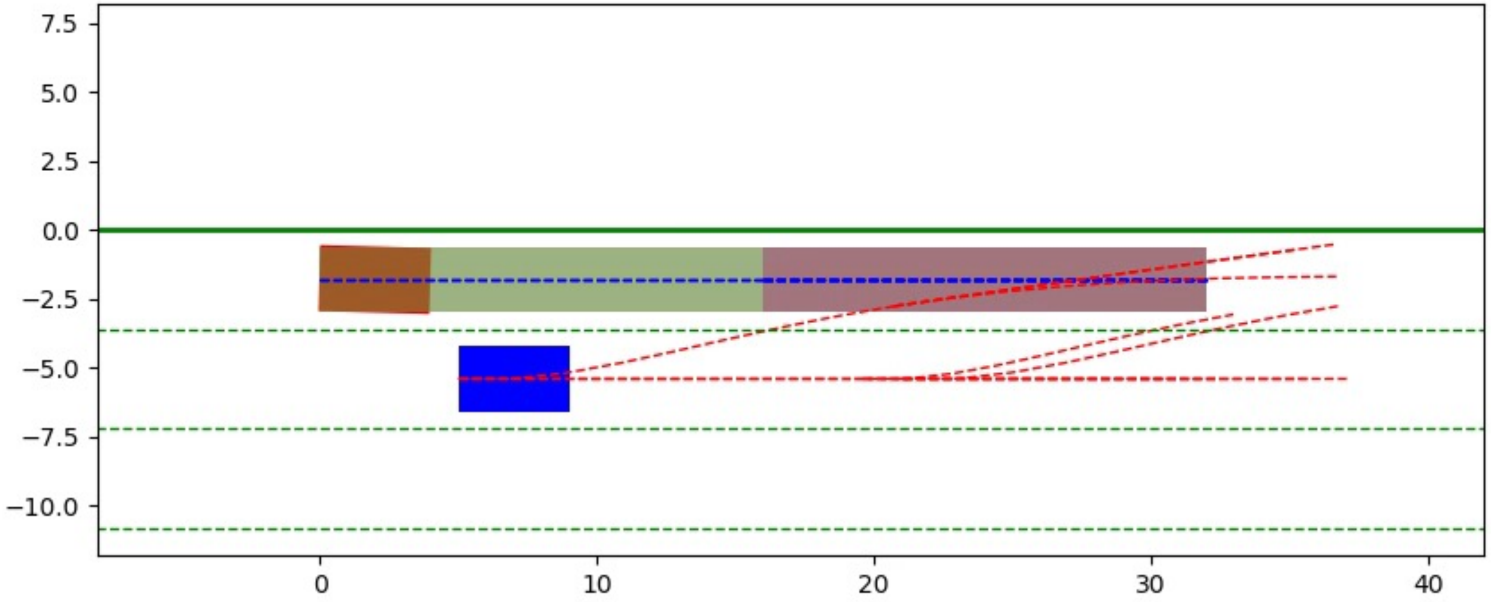
Planning in Multi-modal Uncertain Environments

Dr. Y. Chen

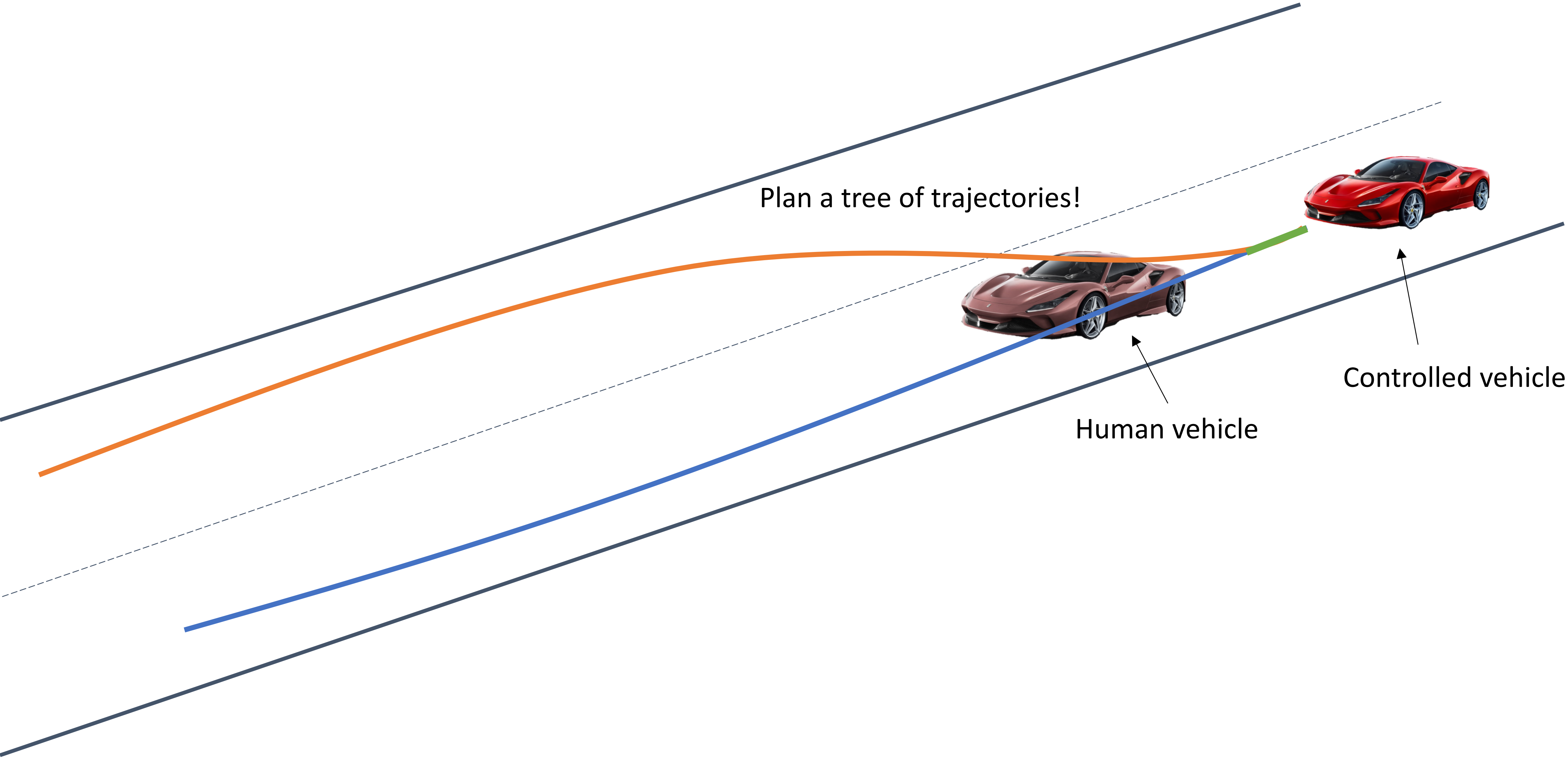
Optimizing over
open-loop actions



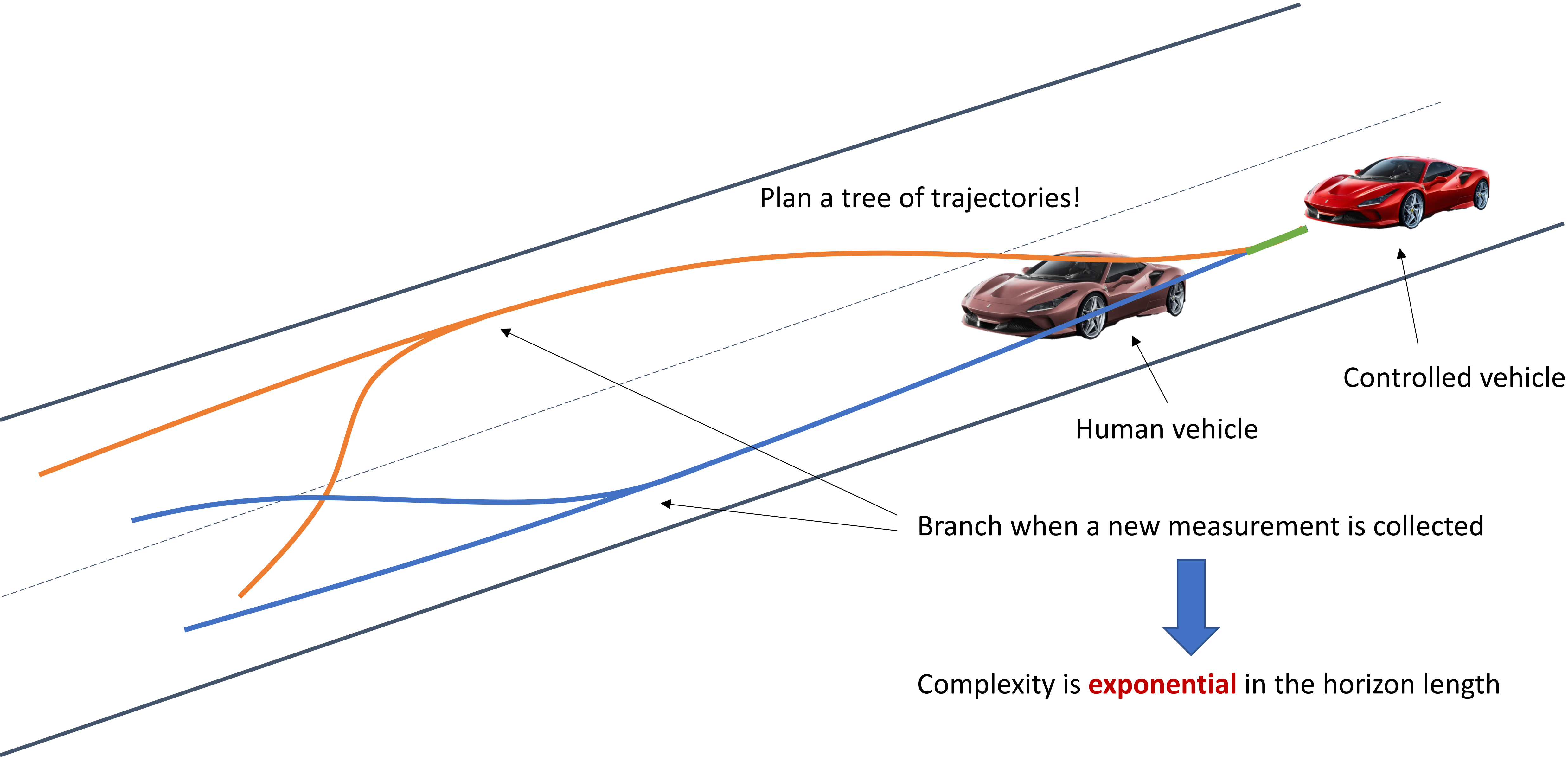
Optimizing over
closed-loop policies



How to reduce the computational complexity?

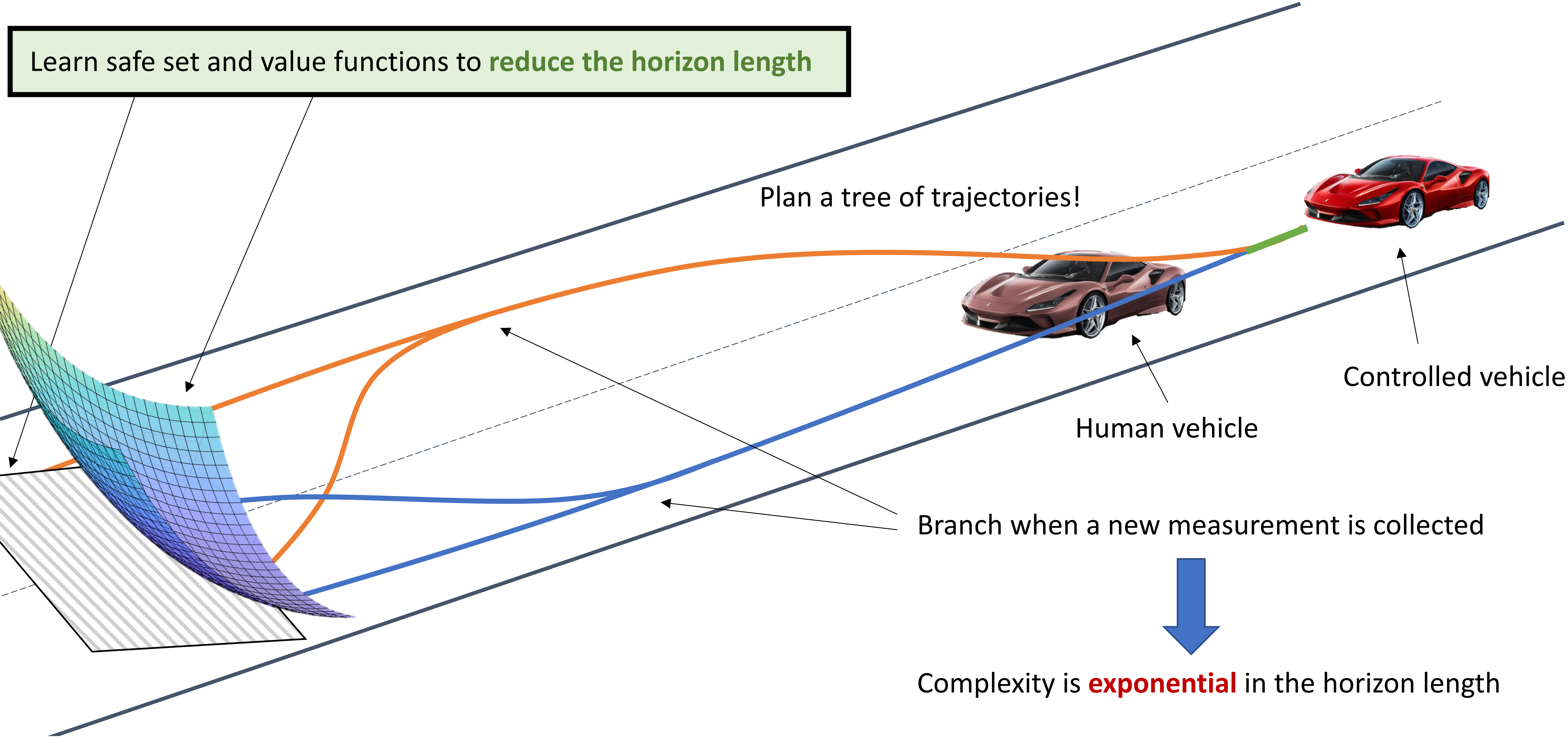


How to reduce the computational complexity?

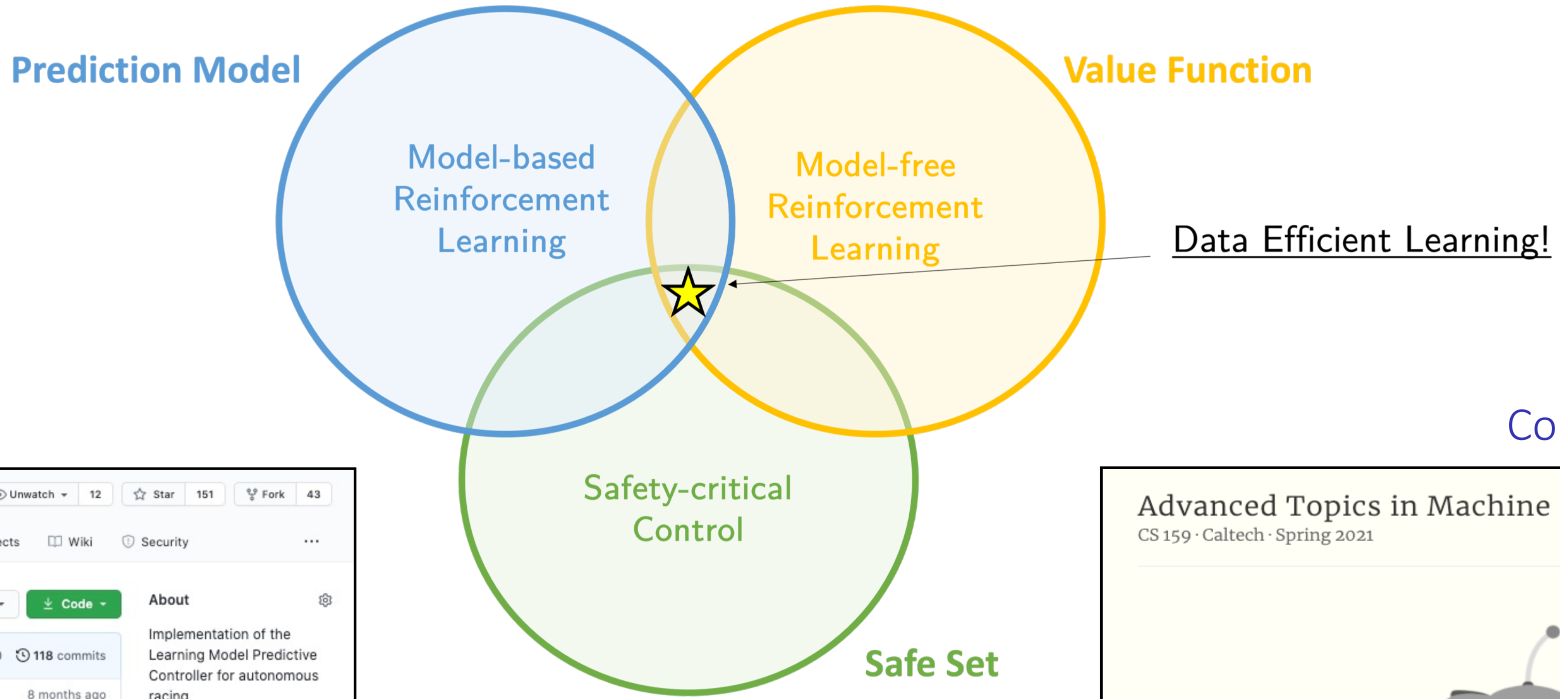


Complexity is **exponential** in the horizon length

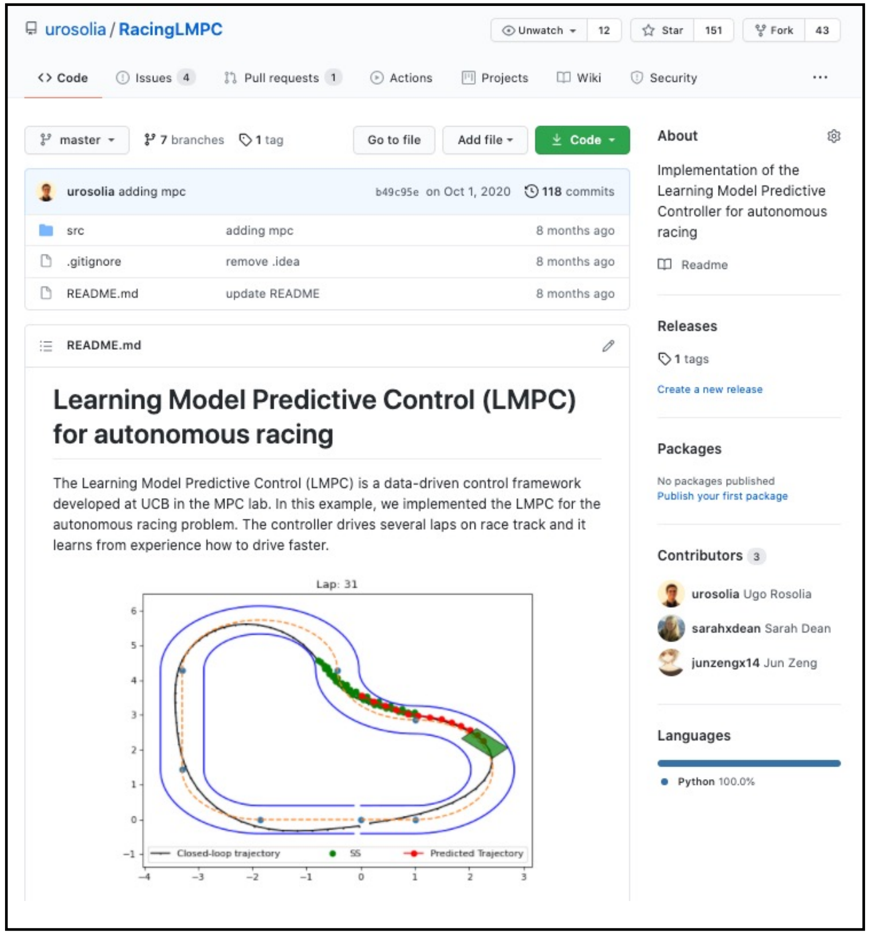
How to reduce the computational complexity?



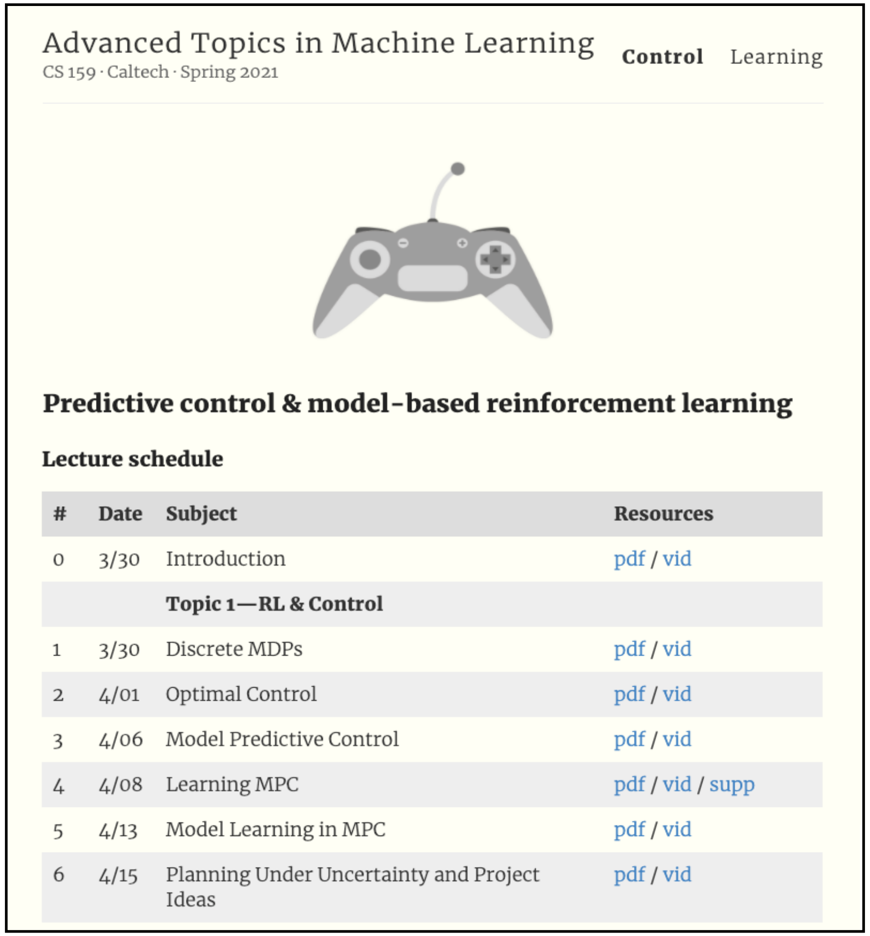
U. Rosolia, Y. Chen, S. Daftry, M. Ono, Y. Yue, and A.D. Ames. “The mixed-observable constrained linear quadratic regulator problem: the exact solution and practical algorithms” 2021, arXiv:2108.12030.



Code available online



Course material online



#	Date	Subject	Resources
0	3/30	Introduction	pdf / vid
		Topic 1—RL & Control	
1	3/30	Discrete MDPs	pdf / vid
2	4/01	Optimal Control	pdf / vid
3	4/06	Model Predictive Control	pdf / vid
4	4/08	Learning MPC	pdf / vid / supp
5	4/13	Model Learning in MPC	pdf / vid
6	4/15	Planning Under Uncertainty and Project Ideas	pdf / vid

Why multi-modal uncertainty?

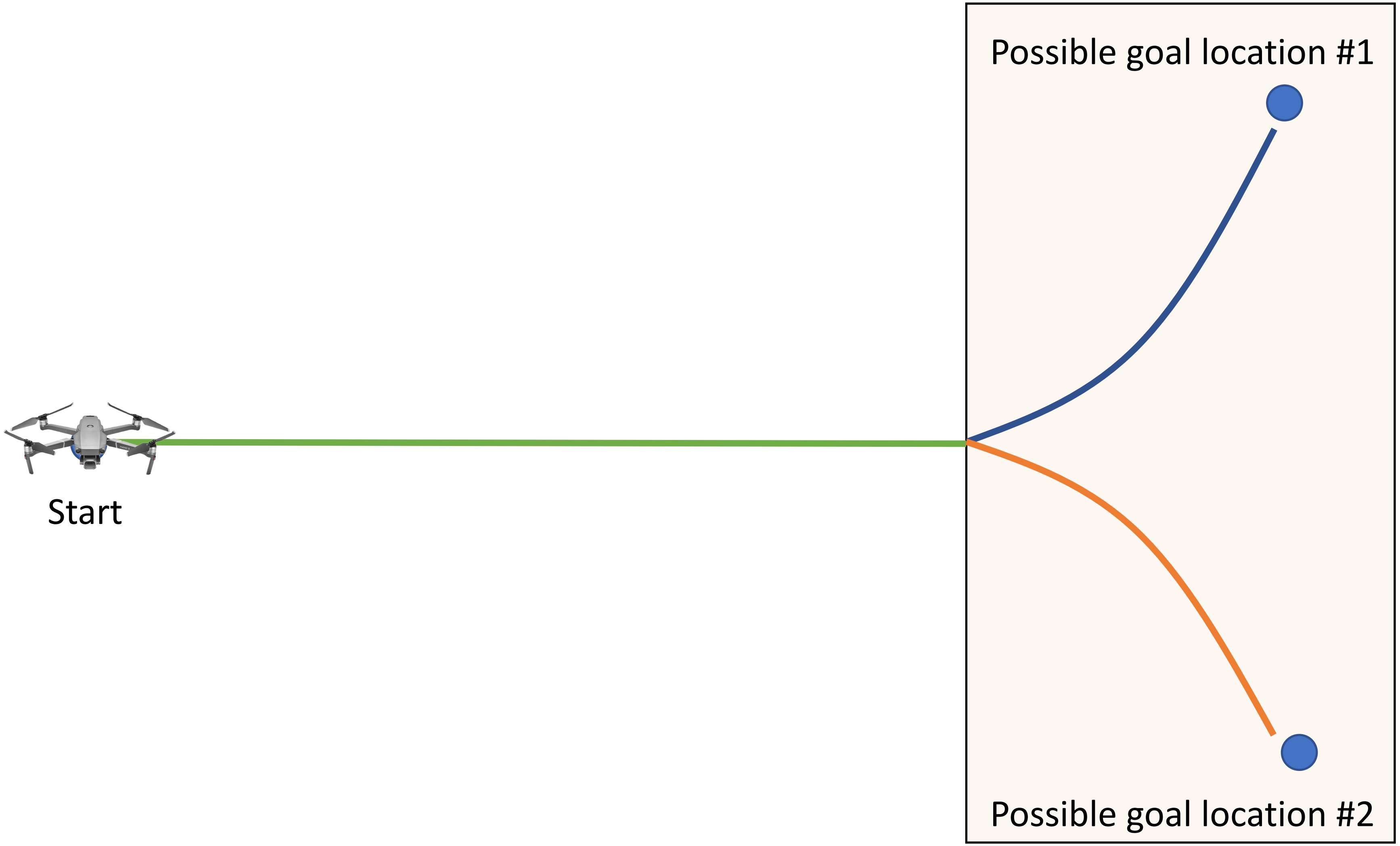
Possible goal location #1

Start

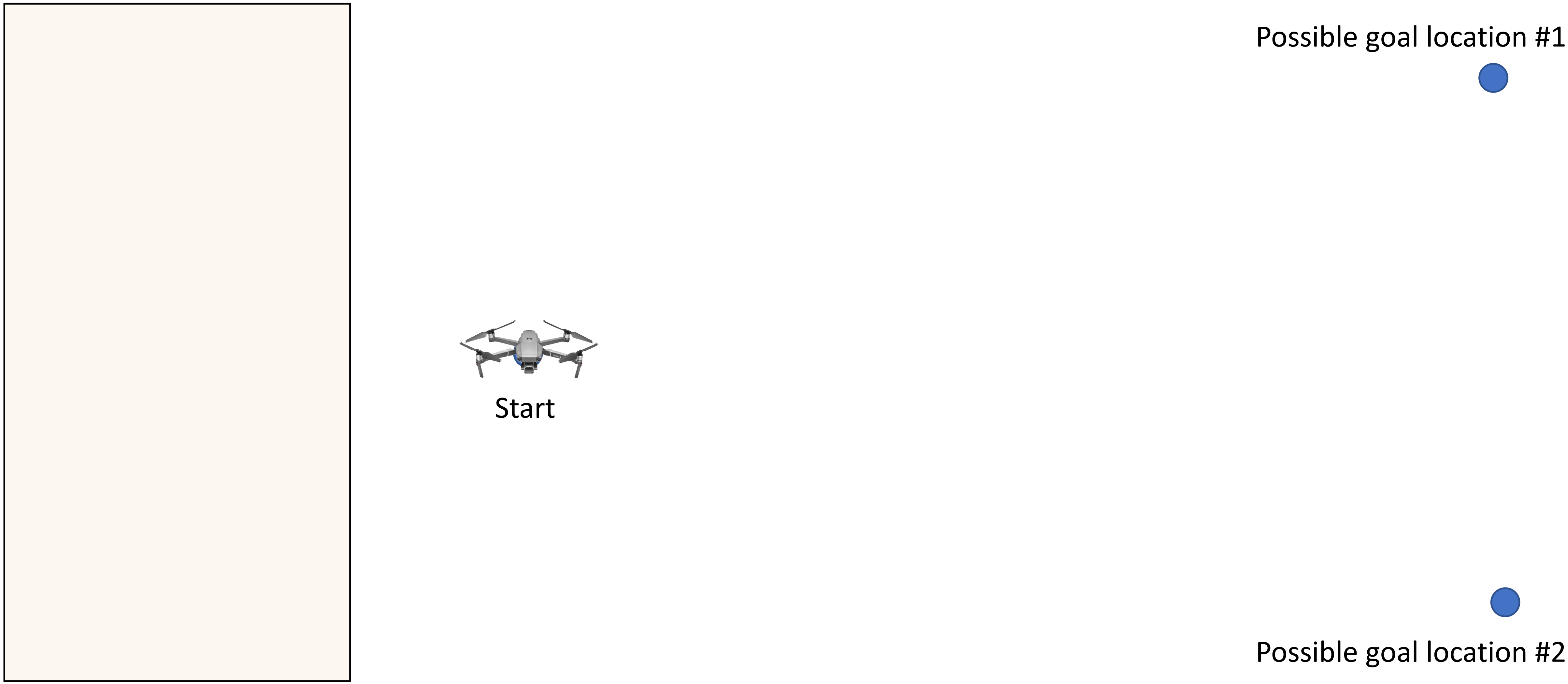
Possible goal location #2

Why multi-modal uncertainty?

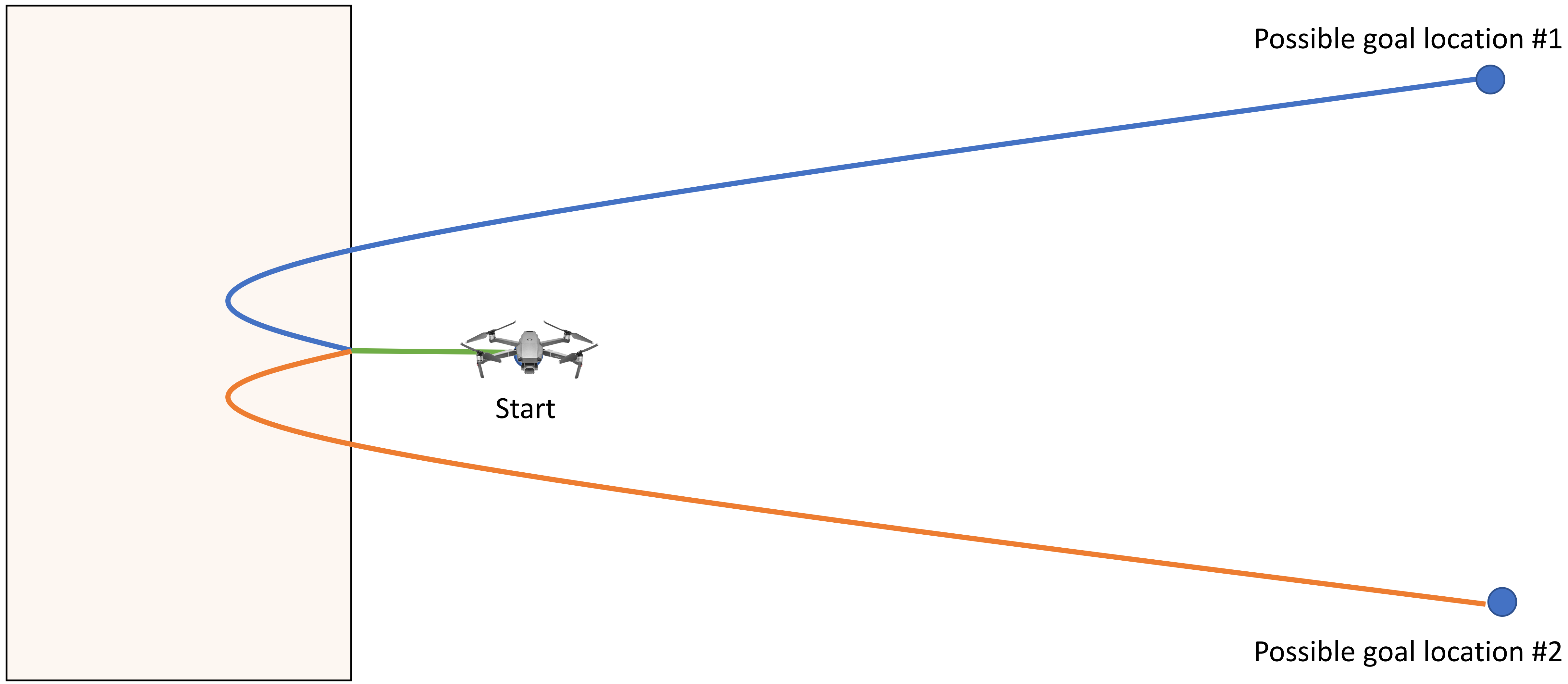
Why multi-modal uncertainty?



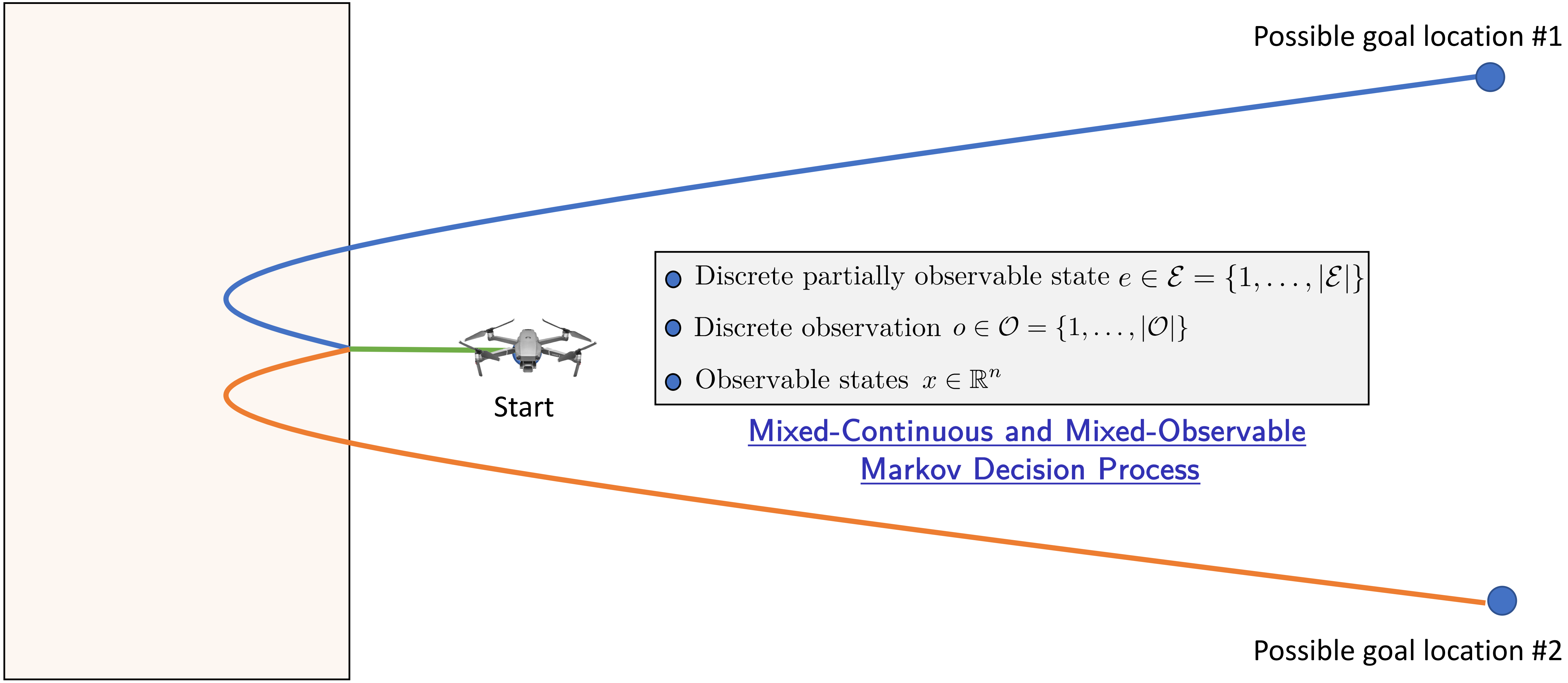
Why multi-modal uncertainty?



Why multi-modal uncertainty?



Why multi-modal uncertainty?

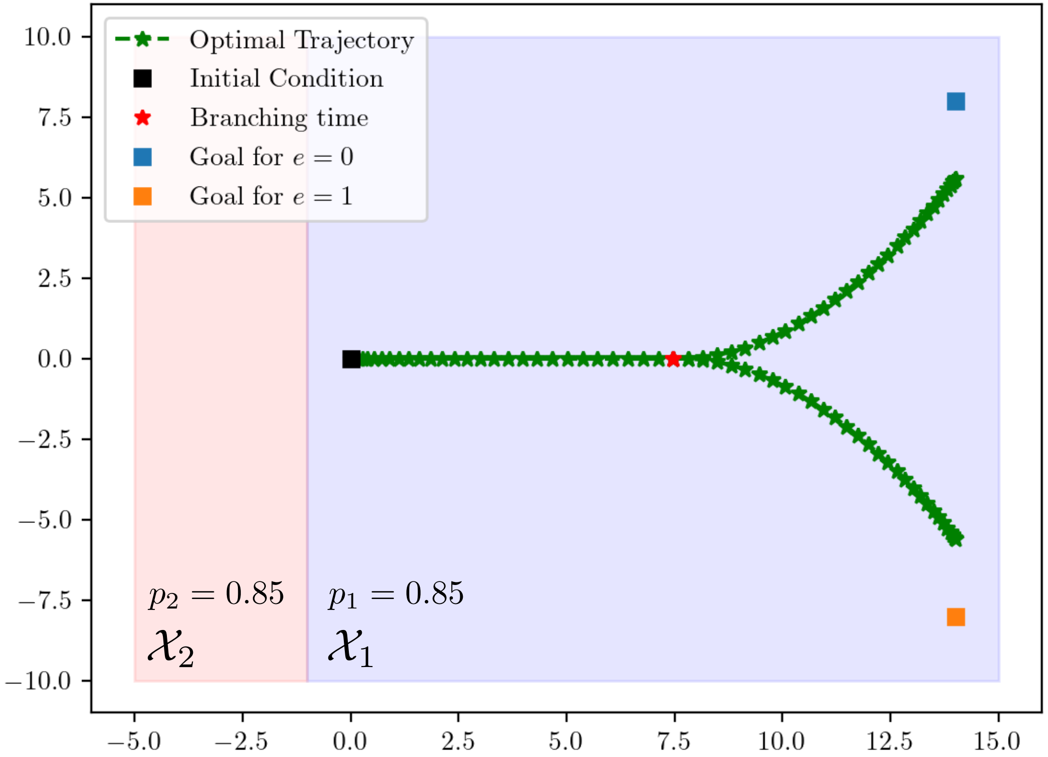


- Discrete partially observable state $e \in \mathcal{E} = \{1, \ldots, |\mathcal{E}|\}$
- Discrete observation $o \in \mathcal{O} = \{1, \ldots, |\mathcal{O}|\}$
- Observable states $x \in \mathbb{R}^n$

Mixed-Continuous and Mixed-Observable Markov Decision Process

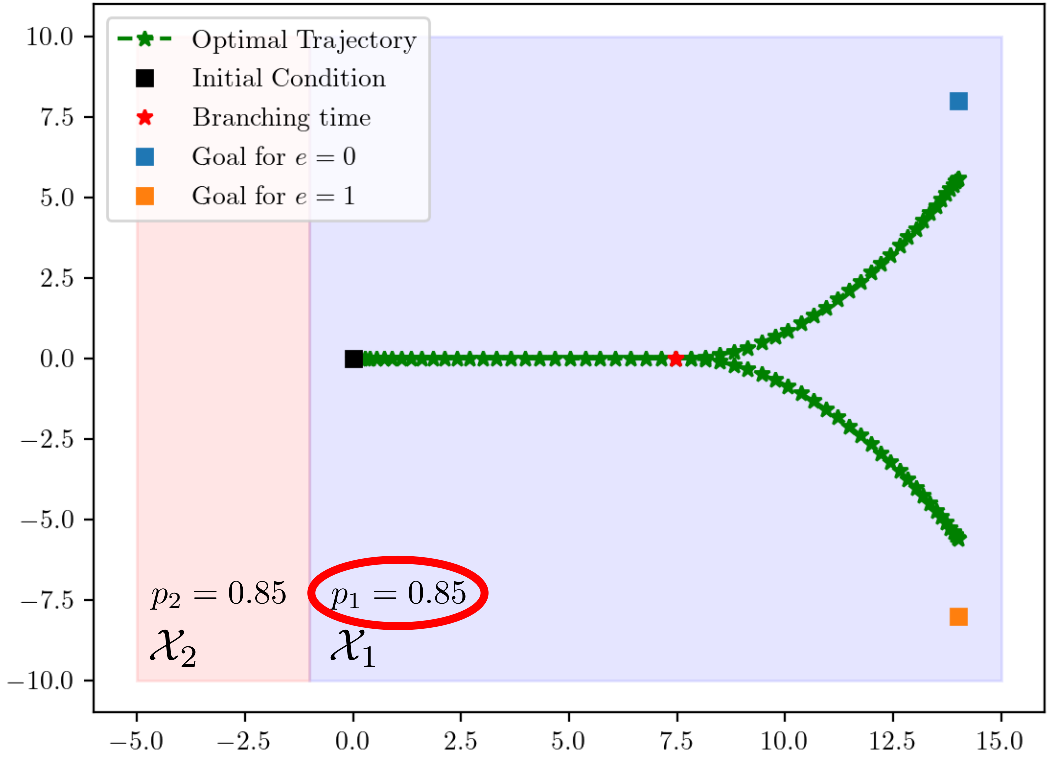
Example 2

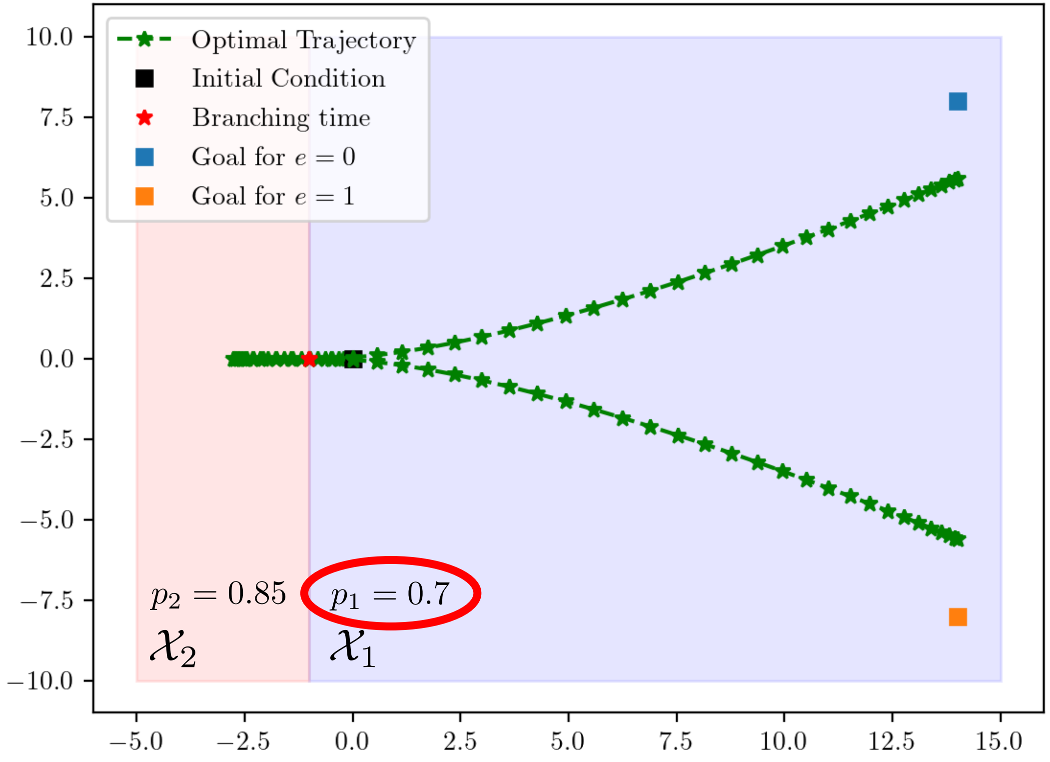
In this example $N = 60$ and $N_b = 30$



Example 2

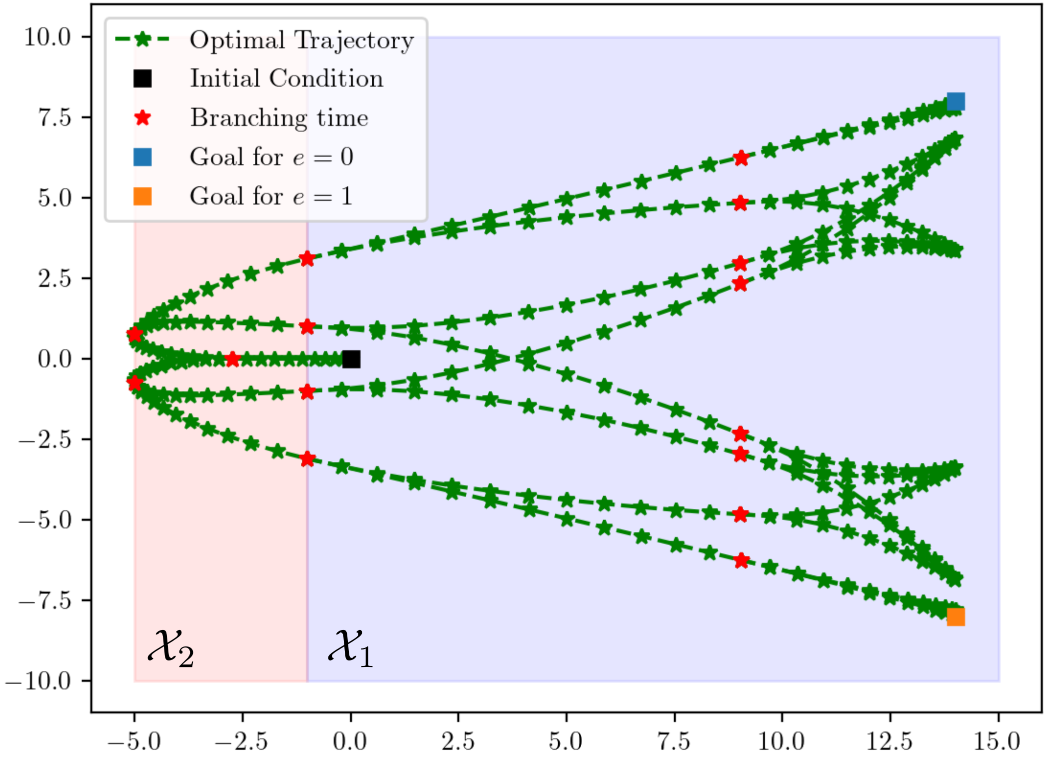
In these examples $N = 60$ and $N_b = 30$





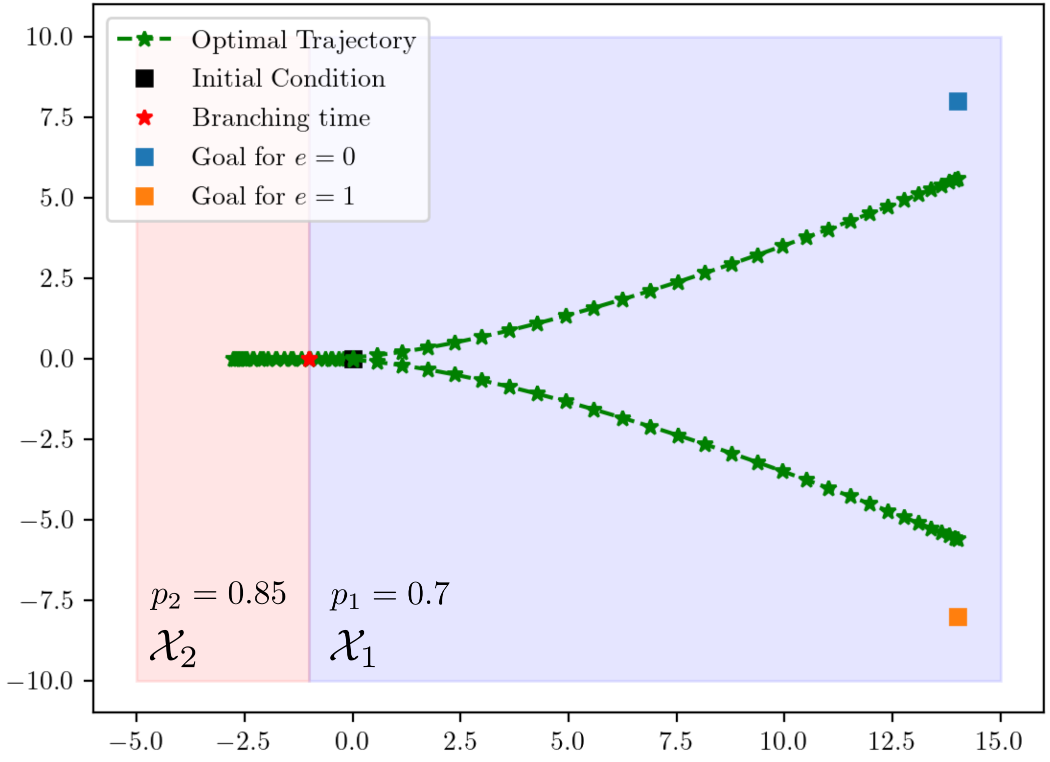
Example 2

$N = 60$ and $N_b = 12$



Optimal cost: 1237.37

$N = 60$ and $N_b = 30$



Optimal cost: 3264.31