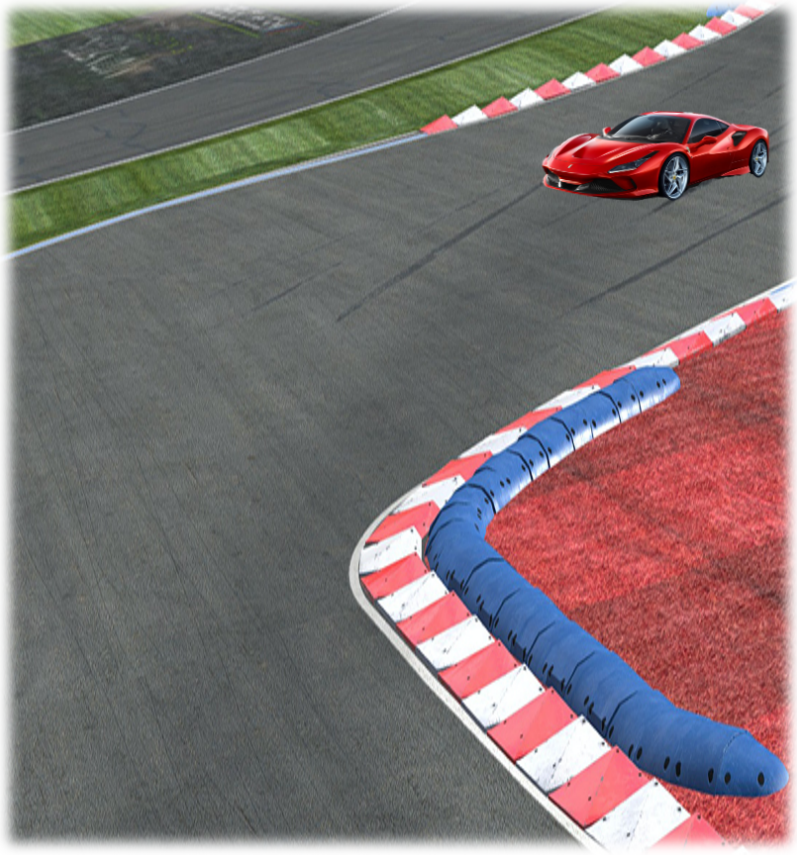
CS159 Lecture 3: Model Predictive Control

Ugo Rosolia

Caltech

Spring 2021

Today's Class: Model Predictive Control



Compute a control policy mapping continuous states to continuous control actions

$$\pi : \mathbb{R}^n \to \mathbb{R}^d.$$

Table of Contents

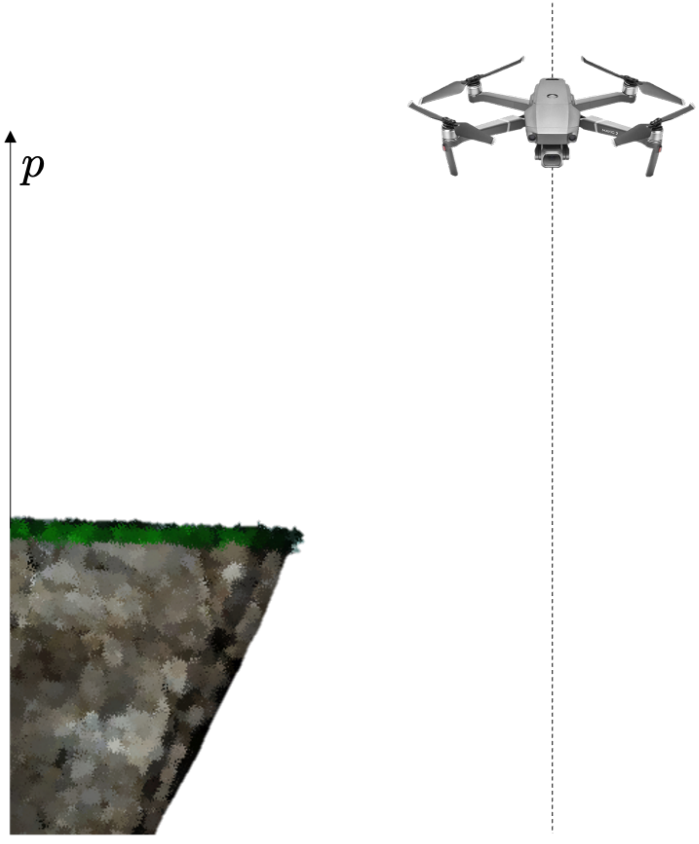
Problem Formulation

Model Predictive Control
- History of MPC
- Numerical Example – The Drone Regulation Problem

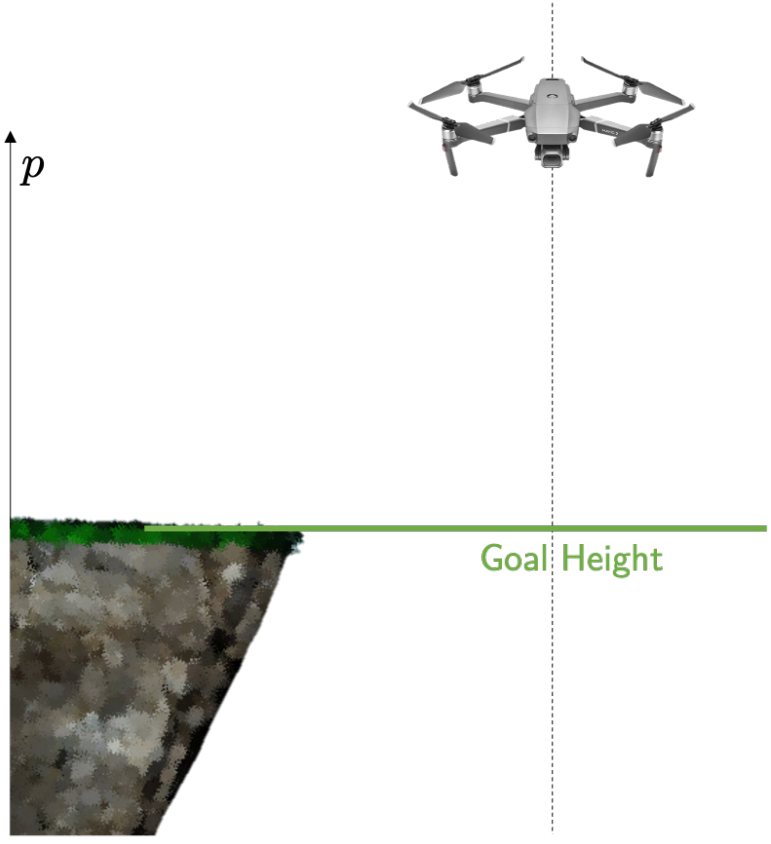
MPC Closed-loop Properties
- Notation
- Terminal Components
- Constraints Satisfaction
- Stability
- Nonlinear Systems

Problem Formulation

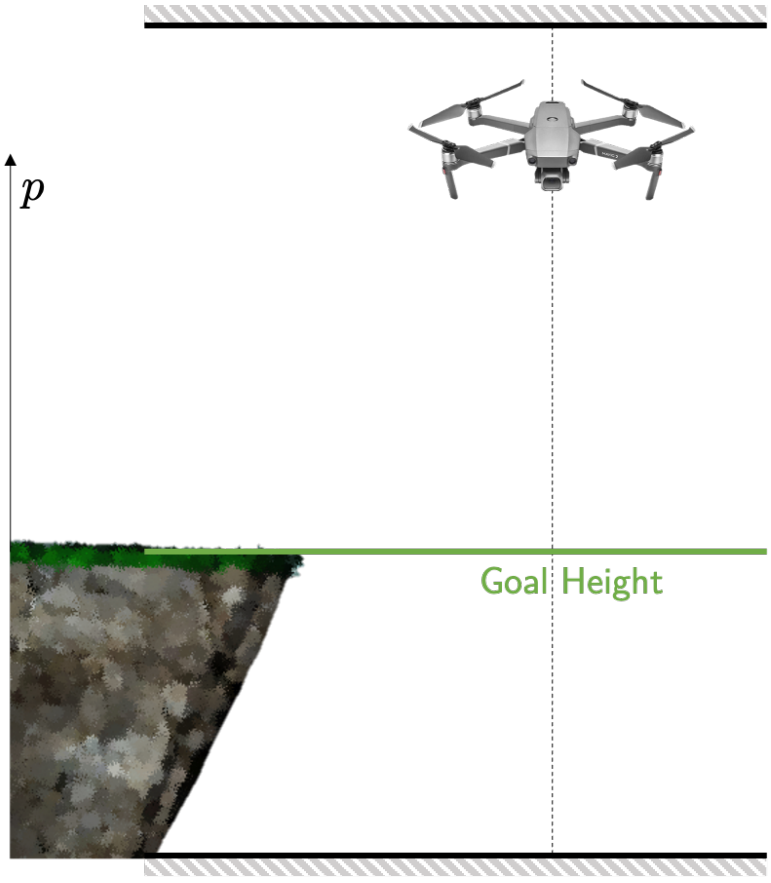
Problem Formulation



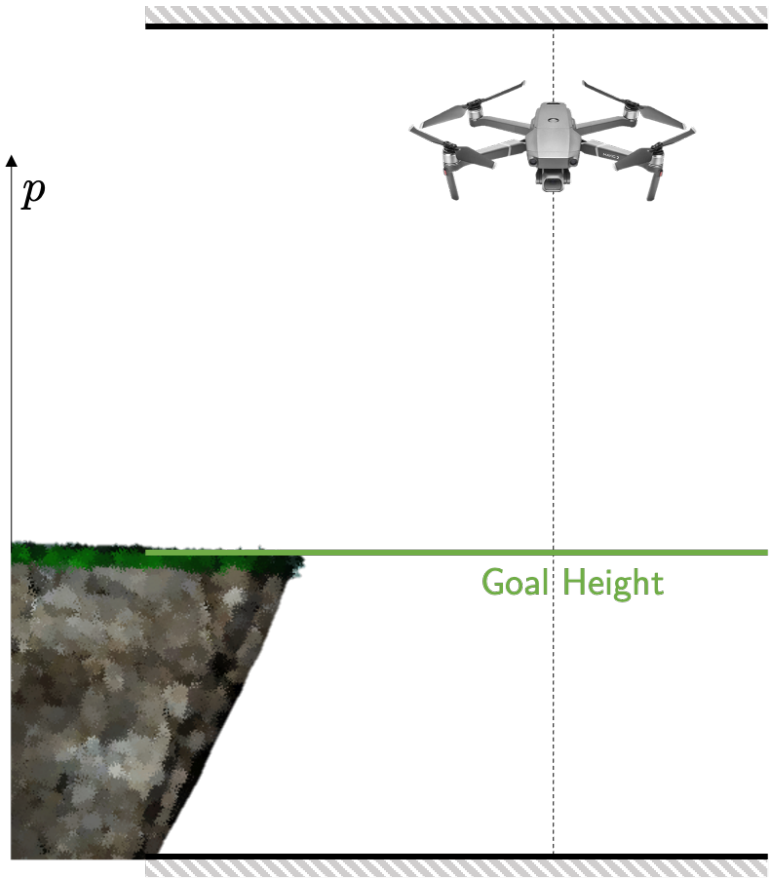
Problem Formulation



Problem Formulation



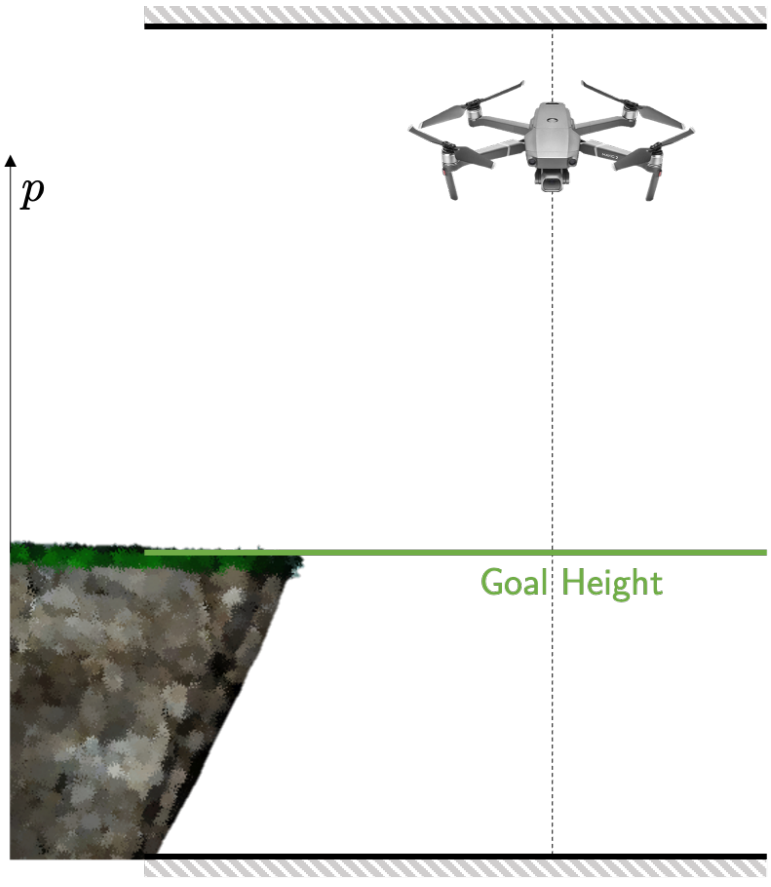
Problem Formulation



- State

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

Problem Formulation

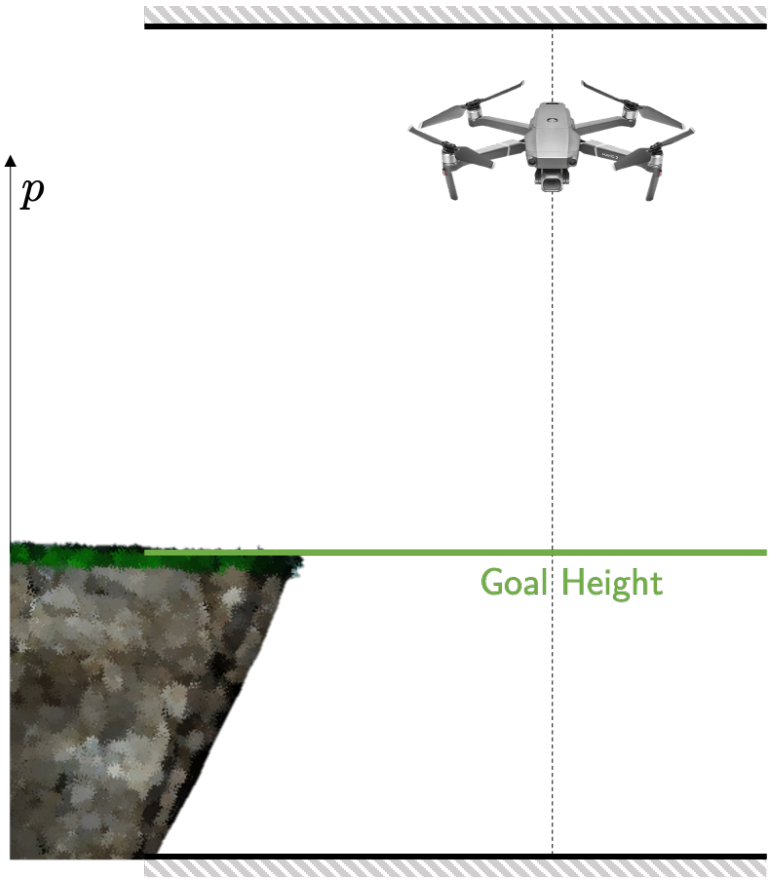


- State

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

- Input $u = a =$ acceleration

Problem Formulation



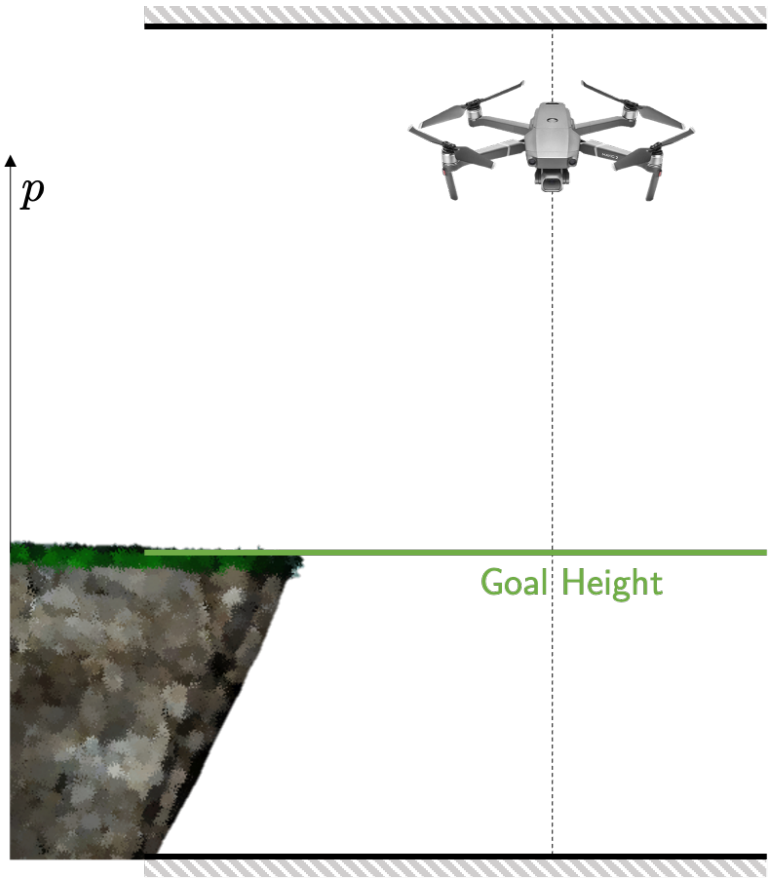
- State

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

- Input $u = a =$ acceleration
- Dynamics

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

Problem Formulation



- State

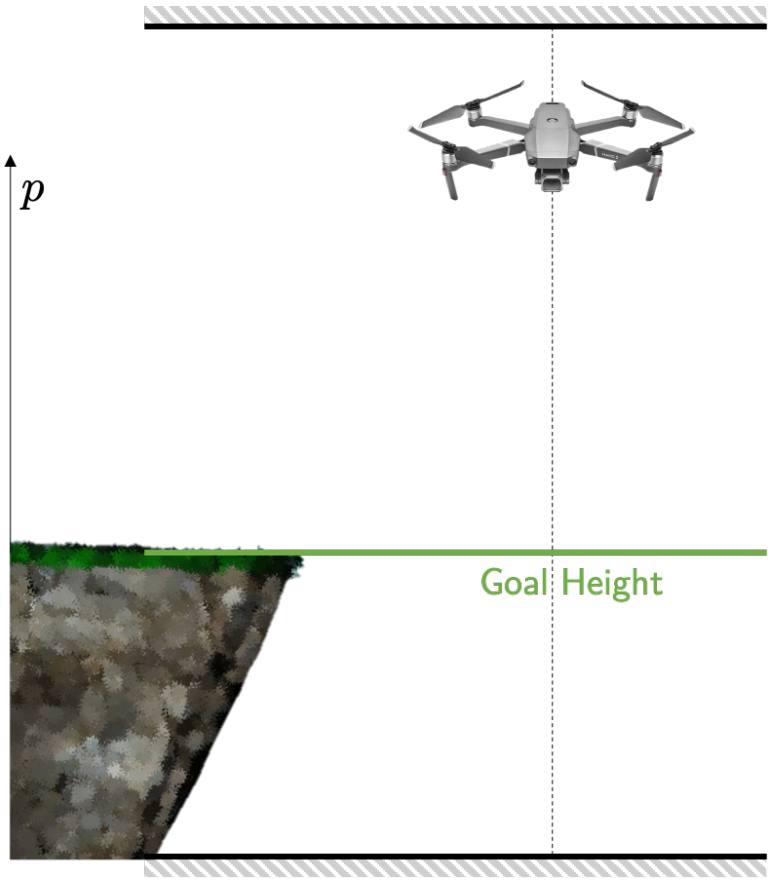
$$x = \begin{bmatrix} p \\ v \end{bmatrix} = \begin{bmatrix} \text{position} \\ \text{velocity} \end{bmatrix}$$

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

Problem Formulation



- 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

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

Problem Formulation

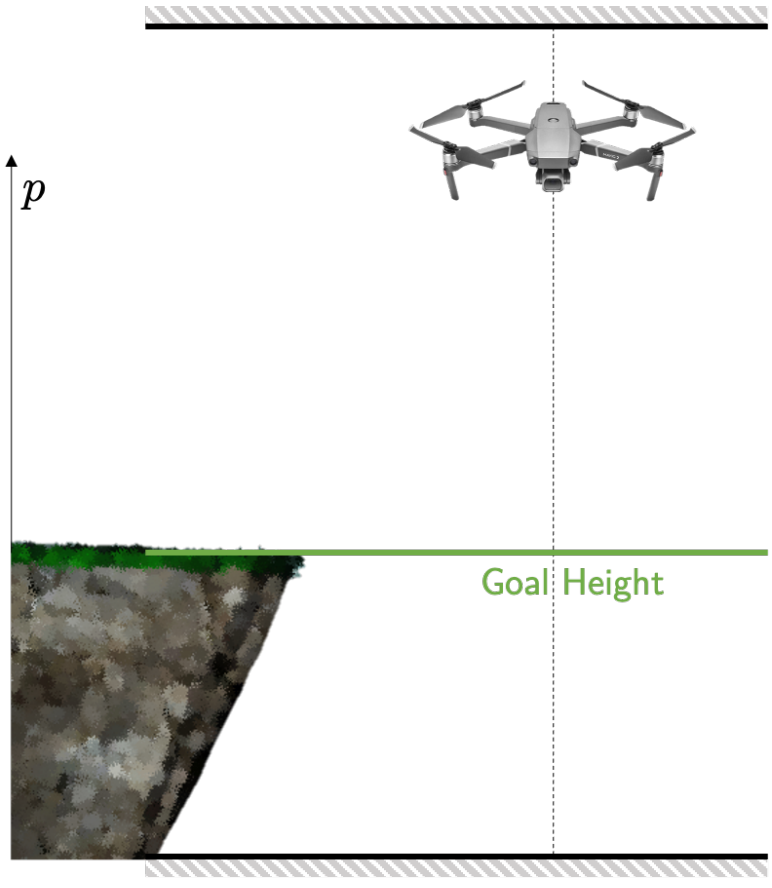
Our goal is to find a control policy which solves the following infinite time optimal control problem:

$$
\begin{aligned}
\pi^* = \arg\min_{\pi} \quad & \sum_{k=0}^{\infty} (x_k^\top Q x_k + u_k^\top R u_k) \\
\text{subject to} \quad & x_{k+1} = A x_k + B u_k, \ \forall k \in \{0, 1, \ldots\}, \\
& x_k \in \mathcal{X}, \ u_k \in \mathcal{U}, \forall k \in \{0, 1, \ldots\}, \\
& u_k = \pi(x_k), \forall k \in \{0, 1, \ldots\}, \\
& x_0 = x(0)
\end{aligned}
\tag{1}
$$

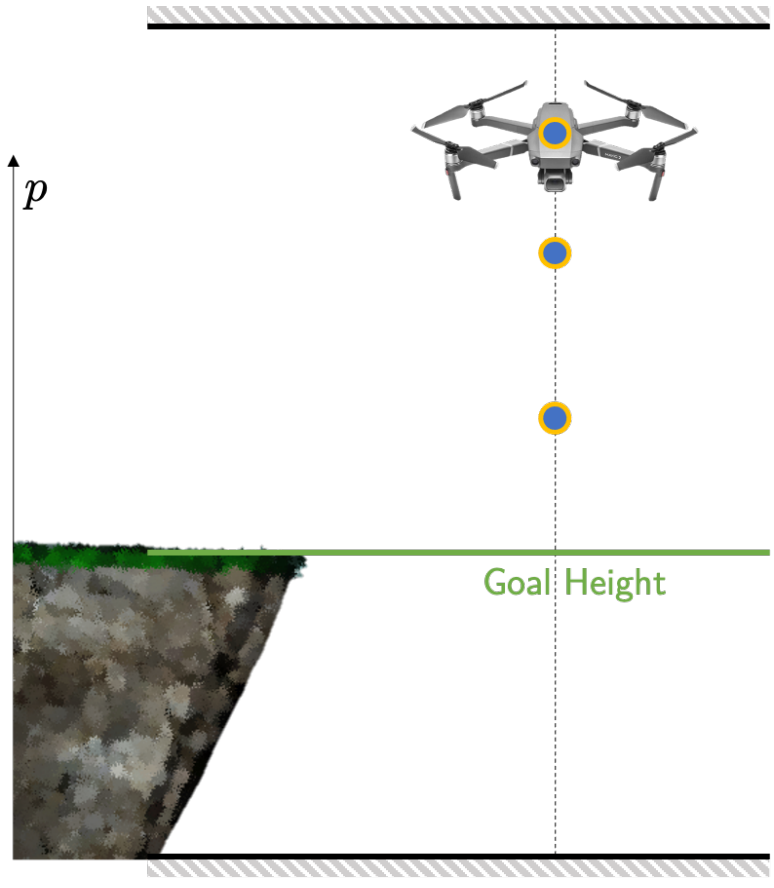
where

- $\pi : \mathbb{R}^n \to \mathbb{R}^d$ is a control policy.
- $\mathcal{X}$ and $\mathcal{U}$ are constraint sets.

Receding Horizon Control

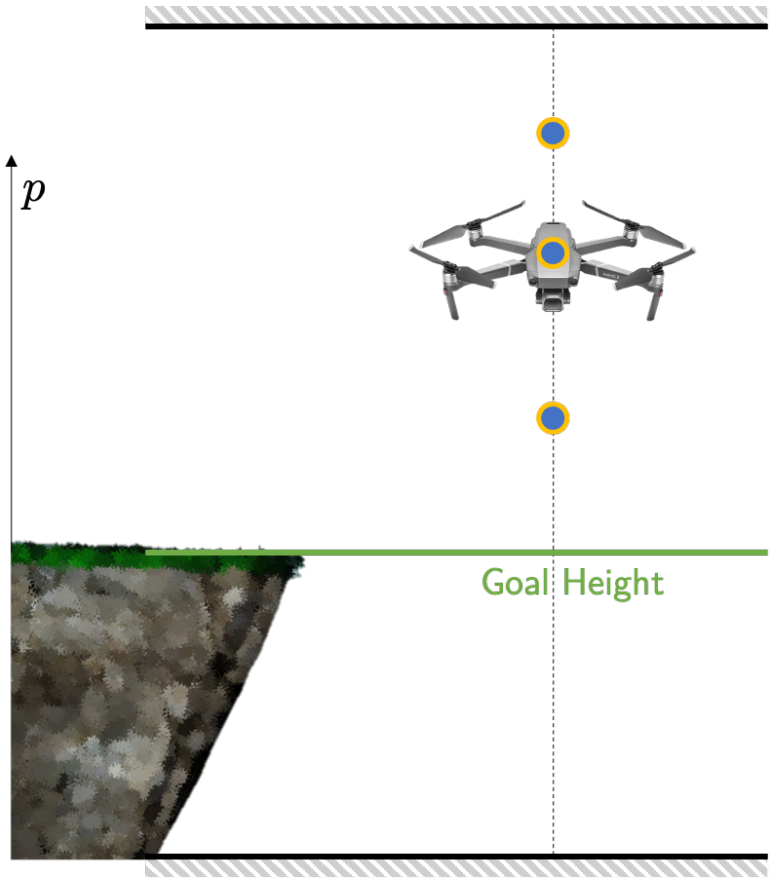


Receding Horizon Control



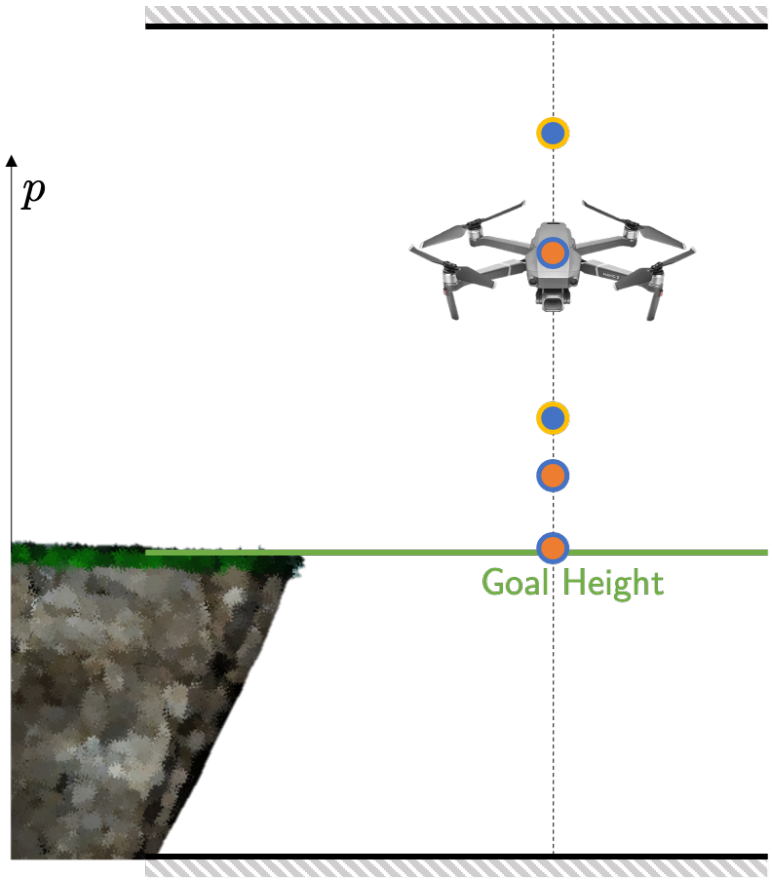
- Plan $\{u_t^*, \ldots, u_{t+N-1}^*\}$ using the batch approach.

Receding Horizon Control



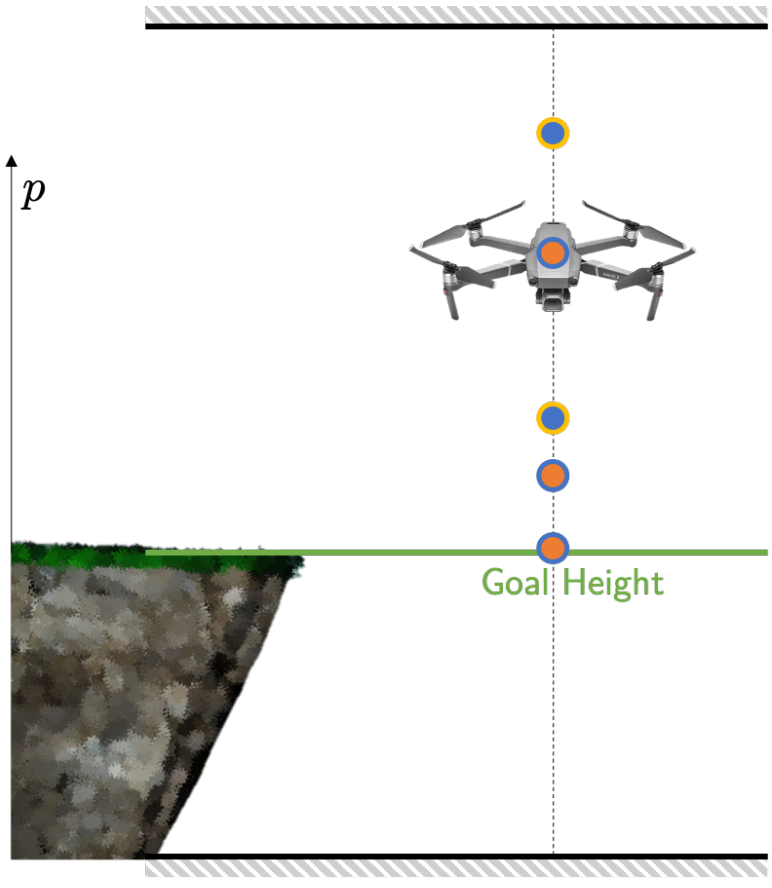
- Plan $\{u_t^*, \ldots, u_{t+N-1}^*\}$ using the batch approach.

Receding Horizon Control



- Plan $\{u_t^*, \ldots, u_{t+N-1}^*\}$ using the batch approach.

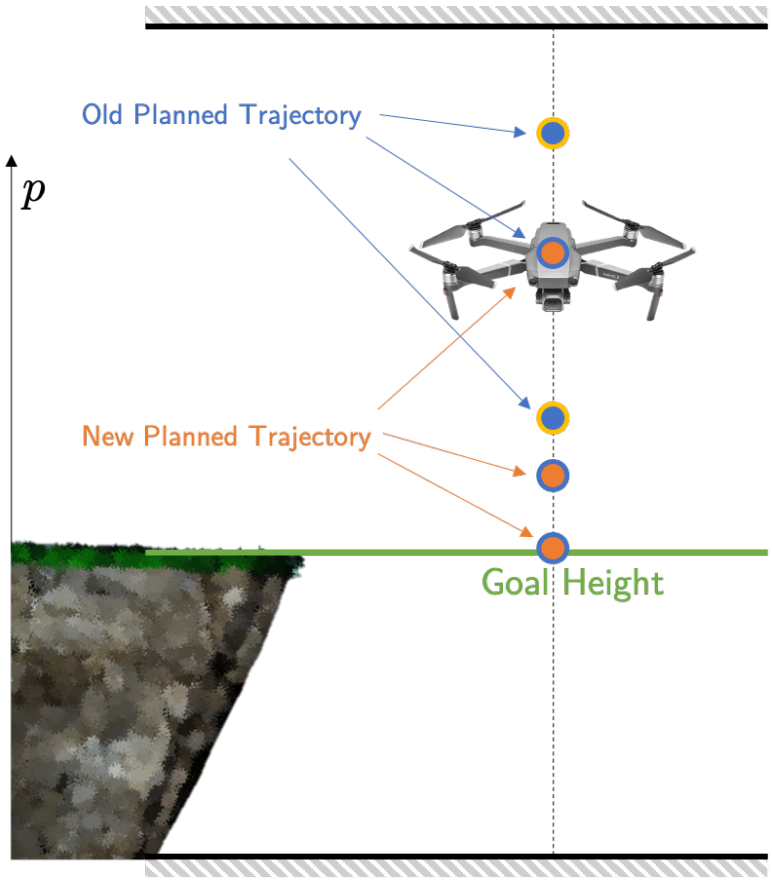
Receding Horizon Control



- Plan $\{u_t^*, \ldots, u_{t+N-1}^*\}$ using the batch approach.
- The MPC policy

$$\pi^{\text{mpc}}(x(t)) = u_t^*.$$

Receding Horizon Control



- Plan $\{u_t^*, \ldots, u_{t+N-1}^*\}$ using the batch approach.
- The MPC policy

$$\pi^{\text{mpc}}(x(t)) = u_t^*.$$

Table of Contents

Problem Formulation

Model Predictive Control

History of MPC

Numerical Example – The Drone Regulation Problem

MPC Closed-loop Properties

Notation

Terminal Components

Constraints Satisfaction

Stability

Nonlinear Systems

History of MPC*

- **A. I. Propoi, 1963**, "Use of linear programming methods for synthesizing sampled-data automatic systems", *Automation and Remote Control*.
- Mid 1990s: extensive theoretical effort devoted to provide conditions for guaranteeing feasibility and closed-loop stability
- 2000s: development of tractable robust MPC approaches; nonlinear and hybrid MPC;
- 2010s: stochastic MPC; distributed large-scale MPC; economic MPC
- late 2010s: safe integration with learning-based approaches; distributionally robust MPC; fast solvers for nonlinear systems

*Slide from Berkeley ME231. Original slide set by F. Borrelli, M. Morari, C. Jones

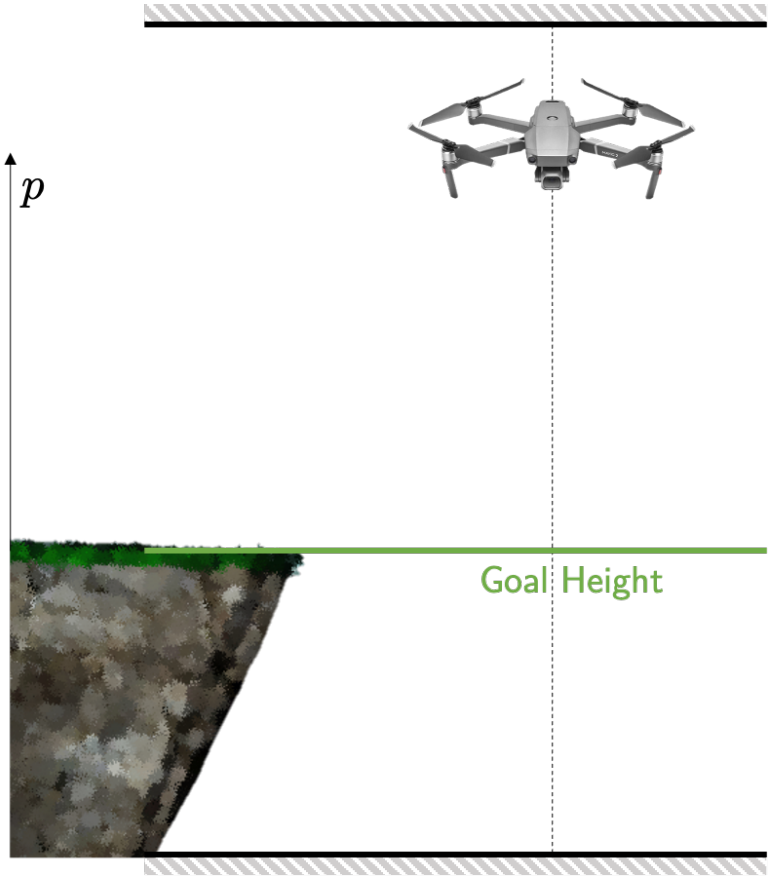
Table of Contents

Problem Formulation

Model Predictive Control
- History of MPC
- Numerical Example – The Drone Regulation Problem

MPC Closed-loop Properties
- Notation
- Terminal Components
- Constraints Satisfaction
- Stability
- Nonlinear Systems

Numerical 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

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

Numerical Example – MPC vs LQR

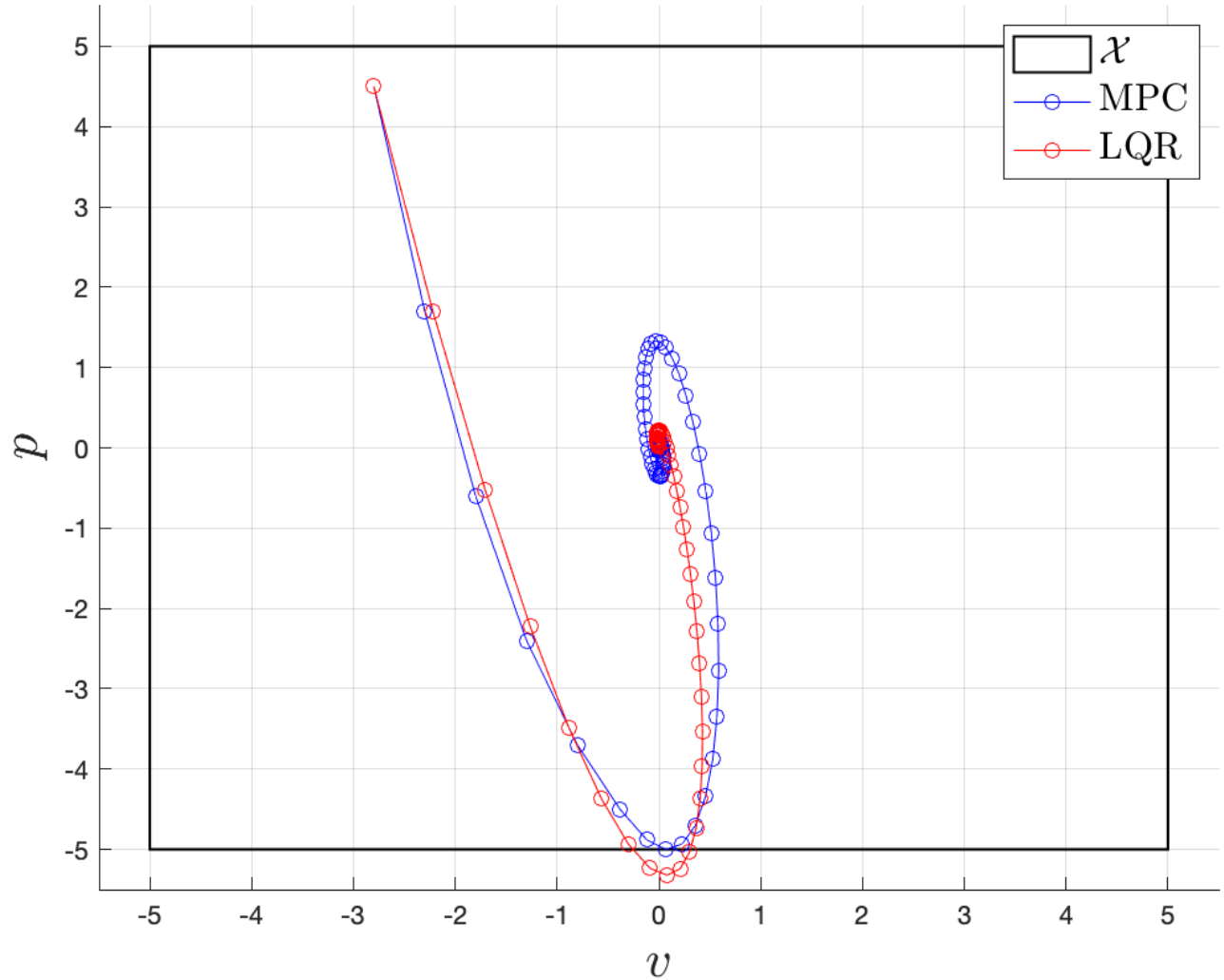
Model Predictive Control

$$\begin{aligned} \min_{u_t,\ldots,u_{t+N-1}} \quad & \sum_{k=0}^{N-1} (x_k^\top Q x_k + u_k^\top R u_k) + x_{t+N}^\top Q_F x_{t+N} \\ \text{subject to} \quad & x_{k+1} = A x_k + B u_k \\ & x_k \in \mathcal{X}, u_k \in \mathcal{U} \\ & x_t = x(0), x_N \in \mathcal{X}_F \end{aligned}$$

Linear Quadratic Regulator

$$\begin{aligned} \min_{u_t,u_{t+1},\ldots} \quad & \sum_{k=0}^{\infty} (x_k^\top Q x_k + u_k^\top R u_k) \\ \text{subject to} \quad & x_{k+1} = A x_k + B u_k \\ & x_t = x(0) \end{aligned}$$

Numerical Example – MPC vs LQR



Tuning $Q = I$, $Q_F = 10I$ and $R = 10^4$

MPC Features

Pros

- Can handle
  - linear models
  - nonlinear models
  - any cost
  - constraints
  - delays
  - disturbances

Cons

- Computationally demanding (Need to solve an FTOCP)

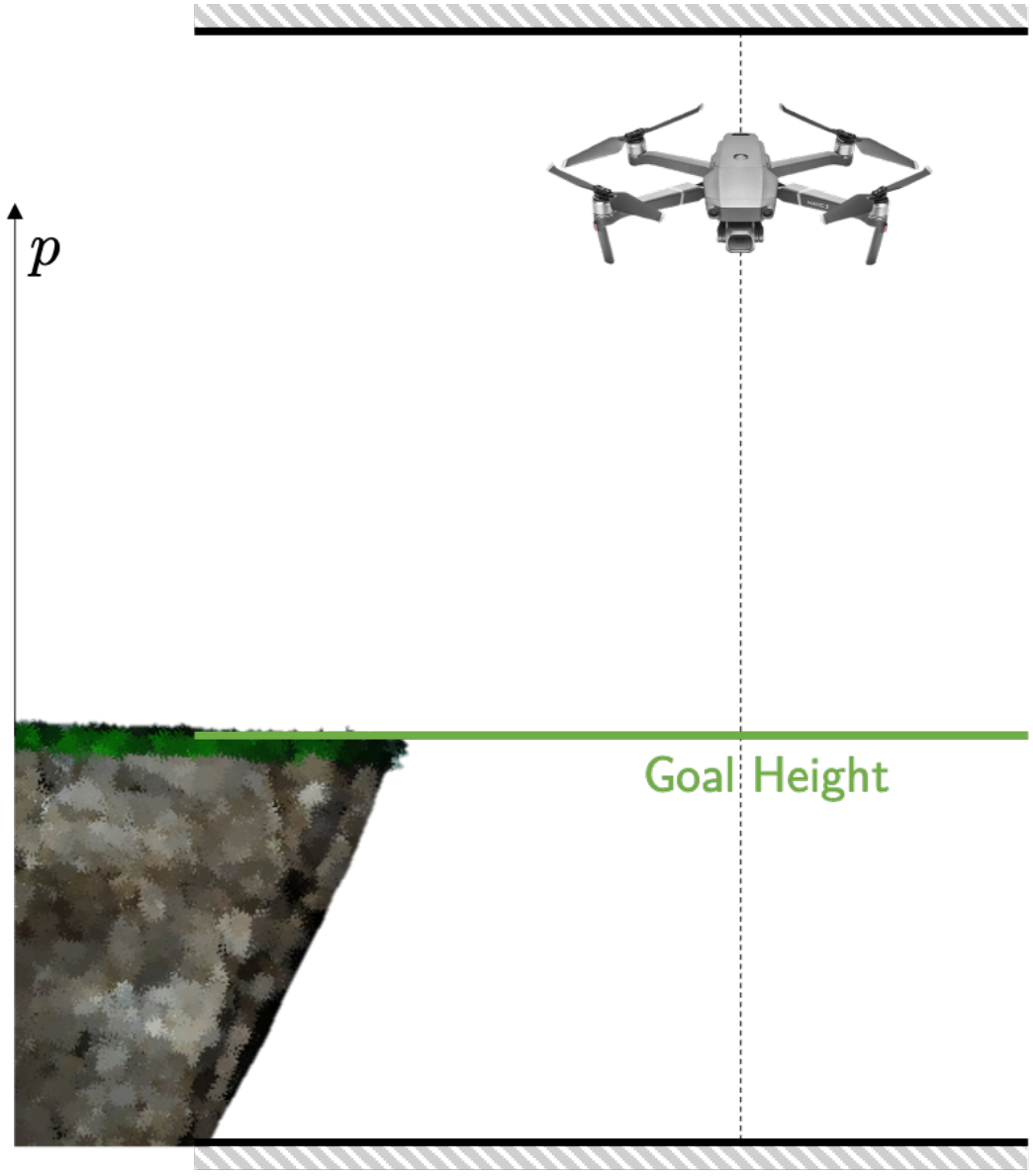
Drone Regulation Problem

Model Predictive Control

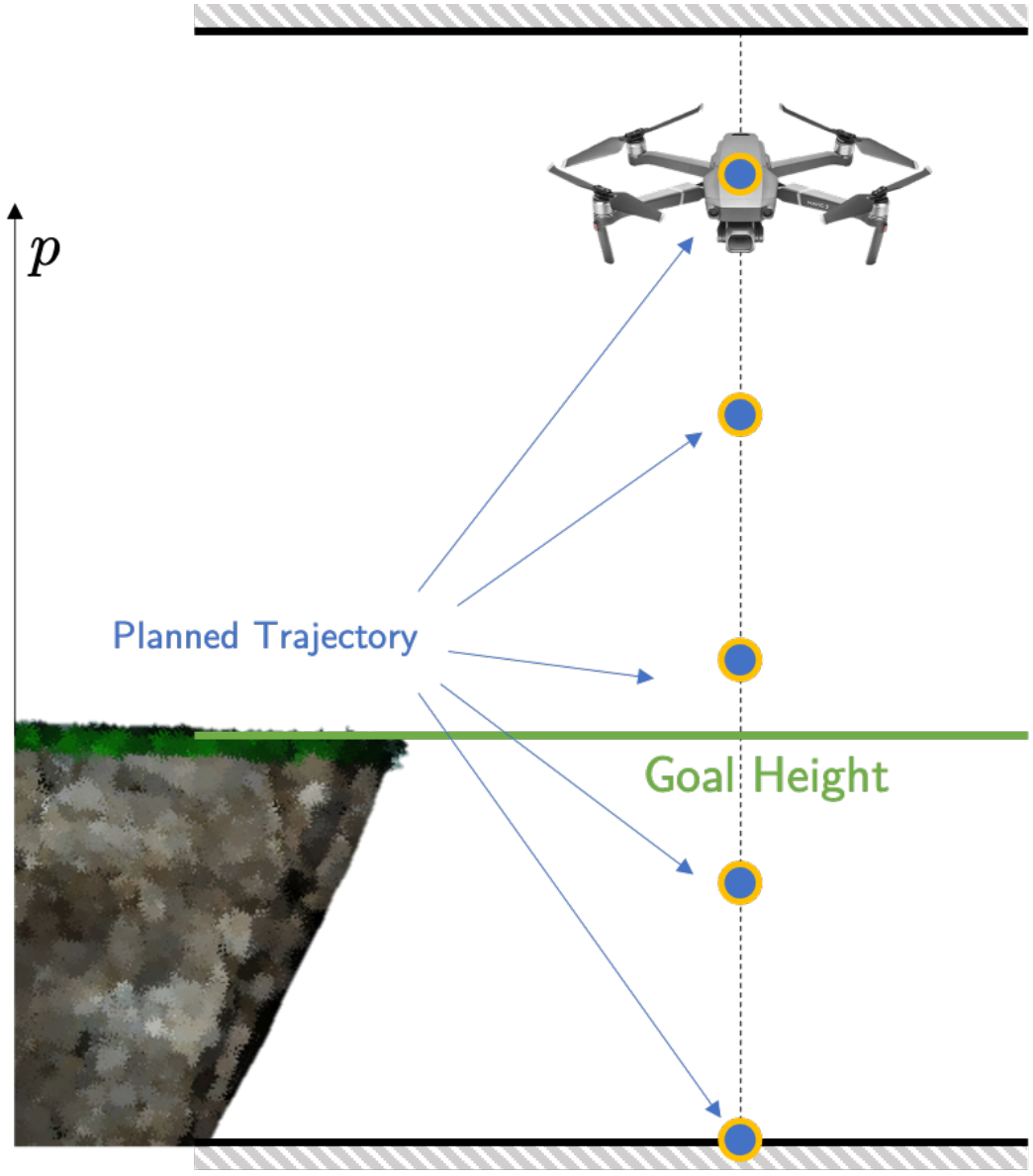
$$\begin{aligned}
\min_{u_t,\ldots,u_{t+N-1}} \quad & \sum_{k=0}^{N-1} (x_k^\top Q x_k + u_k^\top R u_k) + x_{t+N}^\top Q_F x_{t+N} \\
\text{subject to} \quad & x_{k+1} = A x_k + B u_k \\
& x_k \in \mathcal{X}, u_k \in \mathcal{U} \\
& x_t = x(0), x_N \in \mathcal{X}_F
\end{aligned}$$

- Longer horizon $\rightarrow$ more variables $\rightarrow$ the computational cost increases

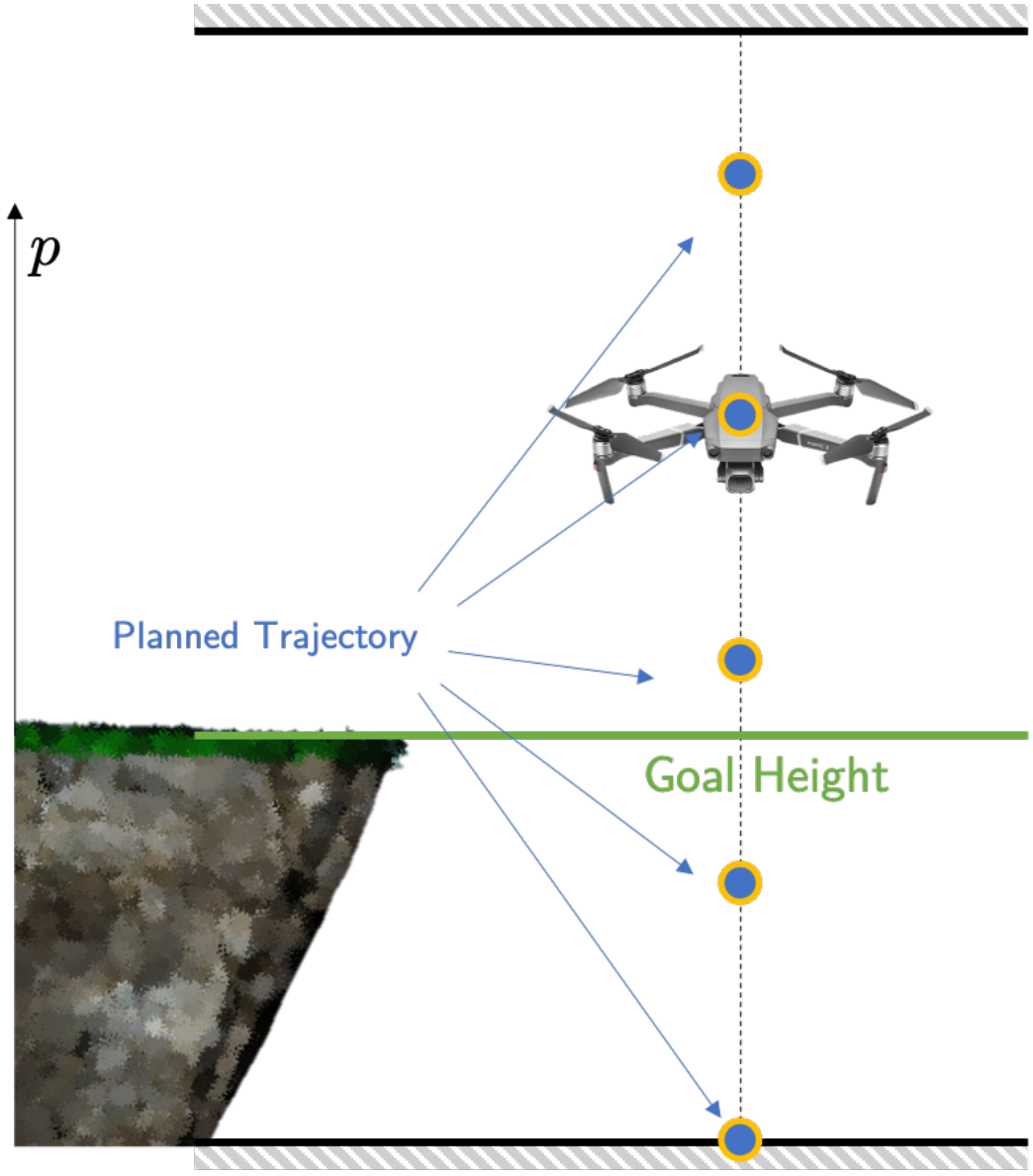
Drone Regulation Problem



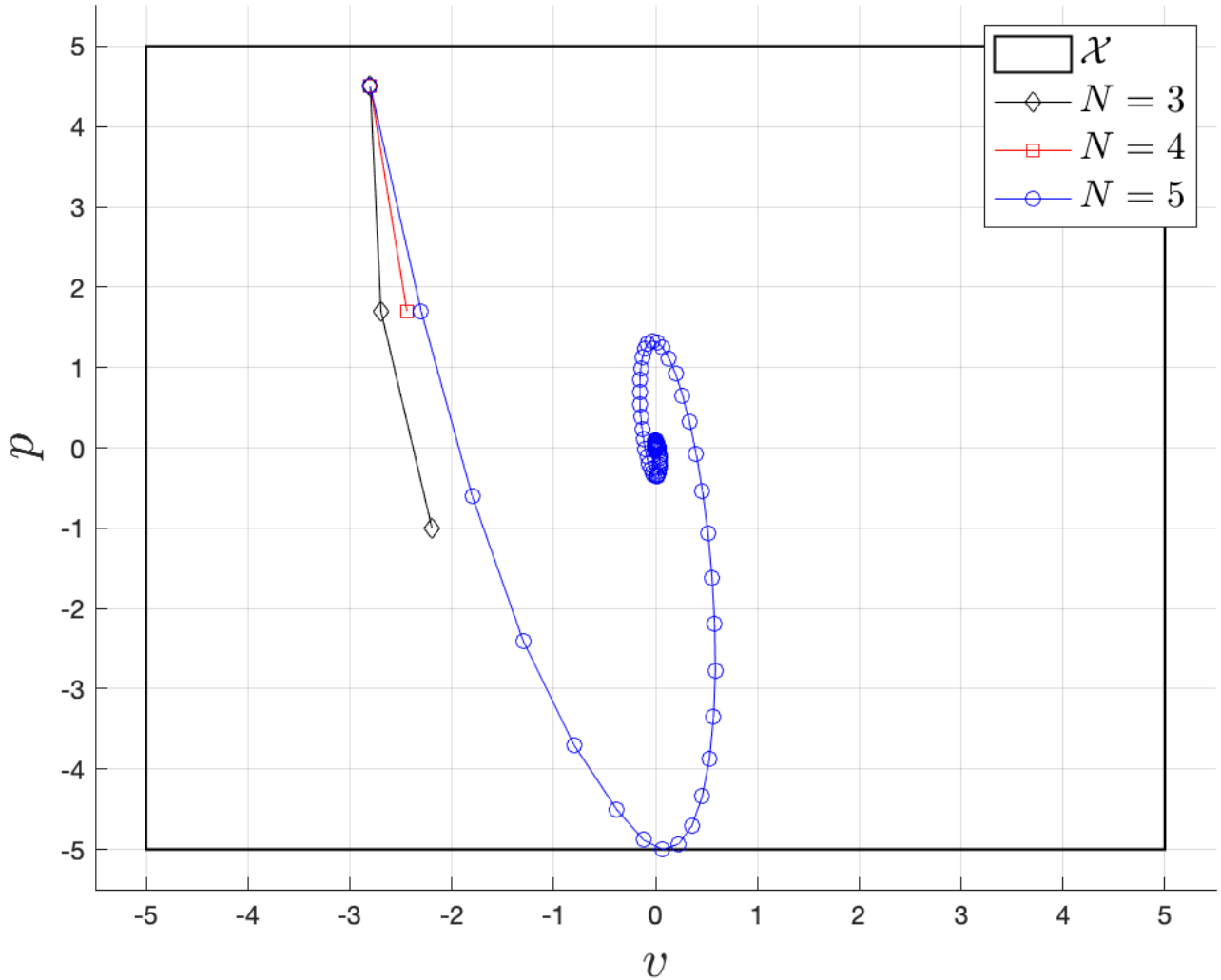
Drone Regulation Problem



Drone Regulation Problem

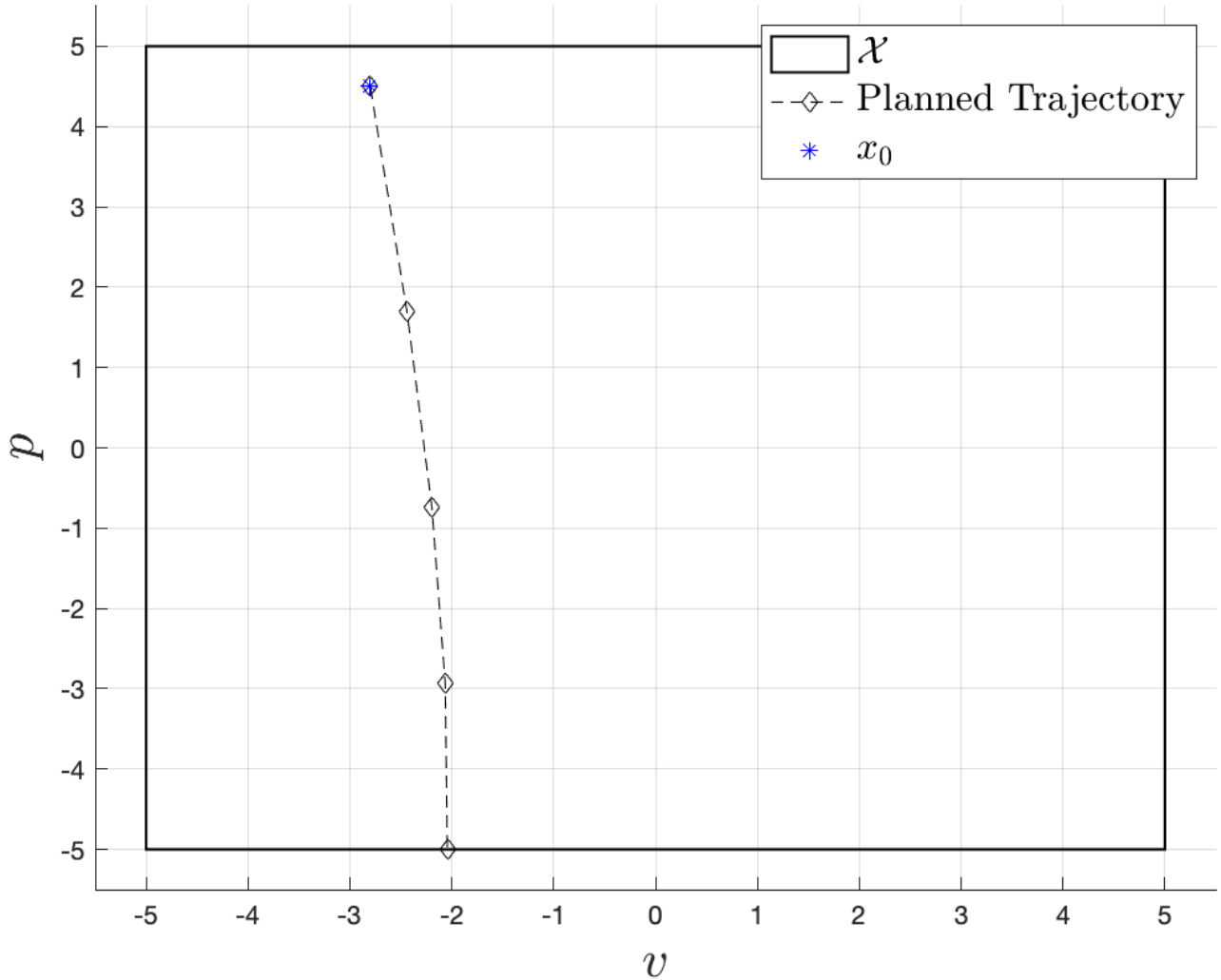


Drone Regulation Problem



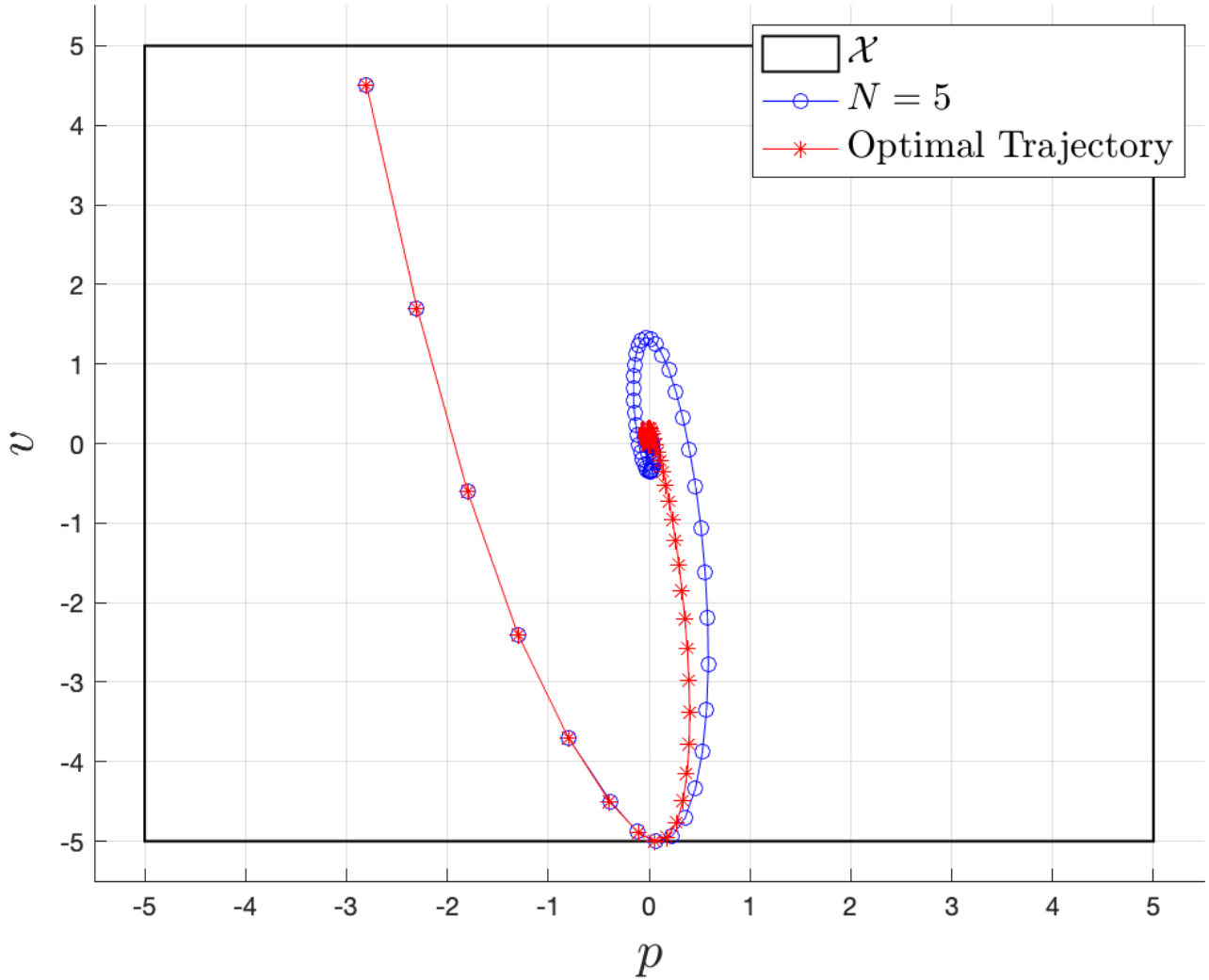
The MPC problem is not feasible at time step $t = 3$ when $N = 3$.
The MPC problem is not feasible at time step $t = 1$ when $N = 4$.

Drone Regulation Problem



Tuning $Q = I$, $Q_F = 10I$ and $R = 10^4$.
Planned trajectory at time $t = 0$ for $N = 4$.

Drone Regulation Problem



The closed-loop trajectory is not guaranteed to be optimal.

Summary

The MPC may take "short-sighted" control actions that lead to the following issues:

- **Feasibility**. After some steps the MPC problems is not feasible (Infeasibility occurs without disturbances and model mismatch!)
- **Stability**. The closed-loop system may not reach the desired goal state!
- **Optimality**. The controller may not be optimal!

Why do we care about feasibility in model-based RL?

Why do we care about feasibility in model-based RL?

Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models

Kurtland Chua **Roberto Calandra** **Rowan McAllister** **Sergey Levine**
Berkeley Artificial Intelligence Research
University of California, Berkeley
{kchua, roberto.calandra, rmcallister, svlevine}@berkeley.edu

Abstract

Model-based reinforcement learning (RL) algorithms can attain excellent sample efficiency, but often lag behind the best model-free algorithms in terms of asymptotic performance. This is especially true with high-capacity parametric function approximators, such as deep networks. In this paper, we study how to bridge this gap, by employing uncertainty-aware dynamics models. We propose a new algorithm called probabilistic ensembles with trajectory sampling (PETS) that combines uncertainty-aware deep network dynamics models with sampling-based uncertainty propagation. Our comparison to state-of-the-art model-based and model-free deep RL algorithms shows that our approach matches the asymptotic performance of model-free algorithms on several challenging benchmark tasks, while requiring significantly fewer samples (e.g., 8 and 125 times fewer samples than Soft Actor Critic and Proximal Policy Optimization respectively on the half-cheetah task).

K. Chua, R. Calandra, R. McAllister and S. Levine. "Deep reinforcement learning in a handful of trials using probabilistic dynamics models" NeurIPS 2018.

Why do we care about feasibility in model-based RL?

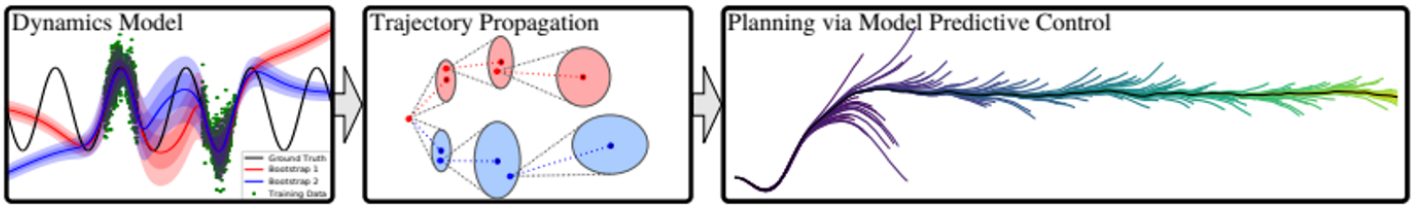


Figure 1: Our method (PE-TS): **Model**: Our probabilistic ensemble (PE) dynamics model is shown as an ensemble of two bootstraps (bootstrap disagreement far from data captures epistemic uncertainty: our subjective uncertainty due to a lack of data), each a probabilistic neural network that captures aleatoric uncertainty (inherent variance of the observed data). **Propagation**: Our trajectory sampling (TS) propagation technique uses our dynamics model to re-sample each particle (with associated bootstrap) according to its probabilistic prediction at each point in time, up until horizon T. **Planning**: At each time step, our MPC algorithm computes an optimal action sequence, applies the first action in the sequence, and repeats until the task-horizon.

Key Idea: Learn a dynamical model using an ensemble of DNN and solve a FTOCP via the batch approach.

K. Chua, R. Calandra, R. McAllister and S. Levine. "Deep reinforcement learning in a handful of trials using probabilistic dynamics models" NeurIPS 2018.

Why do we care about feasibility in model-based RL?

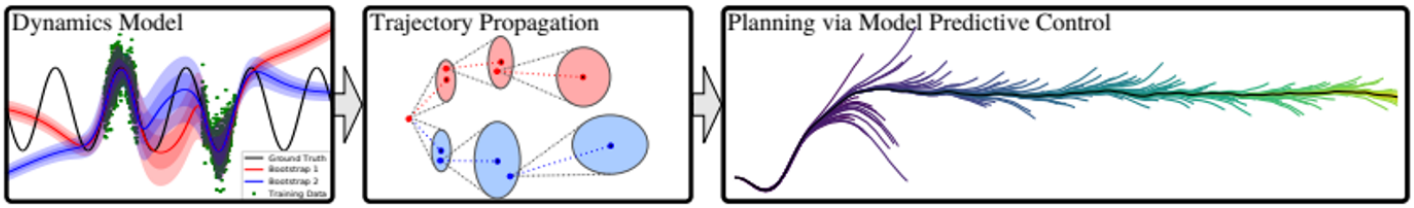


Figure 1: Our method (PE-TS): **Model**: Our probabilistic ensemble (PE) dynamics model is shown as an ensemble of two bootstraps (bootstrap disagreement far from data captures epistemic uncertainty: our subjective uncertainty due to a lack of data), each a probabilistic neural network that captures aleatoric uncertainty (inherent variance of the observed data). **Propagation**: Our trajectory sampling (TS) propagation technique uses our dynamics model to re-sample each particle (with associated bootstrap) according to its probabilistic prediction at each point in time, up until horizon T. **Planning**: At each time step, our MPC algorithm computes an optimal action sequence, applies the first action in the sequence, and repeats until the task-horizon.

Key Idea: Learn a dynamical model using an ensemble of DNN and solve a FTOCP via the batch approach.

Key Message: Recursive feasibility issues will be a problem also in this more complicated set-up.

K. Chua, R. Calandra, R. McAllister and S. Levine. "Deep reinforcement learning in a handful of trials using probabilistic dynamics models" NeurIPS 2018.

Why do we care about feasibility in model-based RL?

Safety Augmented Value Estimation From Demonstrations (SAVED): Safe Deep Model-Based RL for Sparse Cost Robotic Tasks

Brijen Thananjeyan, Ashwin Balakrishna, Ugo Rosolia, Felix Li, Rowan McAllister, Joseph E. Gonzalez, Sergey Levine, Francesco Borrelli, and Ken Goldberg

***Abstract*—Reinforcement learning (RL) for robotics is challenging due to the difficulty in hand-engineering a dense cost function, which can lead to unintended behavior, and dynamical uncertainty, which makes exploration and constraint satisfaction challenging. We address these issues with a new model-based reinforcement learning algorithm, Safety Augmented Value Estimation from Demonstrations (SAVED), which uses supervision that only identifies task completion and a modest set of suboptimal demonstrations to constrain exploration and learn efficiently while handling complex constraints. We then compare SAVED with 3 state-of-the-art model-based and model-free RL algorithms on 6 standard simulation benchmarks involving navigation and manipulation and a physical knot-tying task on the da Vinci surgical robot. Results suggest that SAVED outperforms prior methods in terms of success rate, constraint satisfaction, and sample efficiency, making it feasible to safely learn a control policy directly on a real robot in less than an hour. For tasks on the robot, baselines succeed less than 5% of the time while SAVED has a success rate of over 75% in the first 50 training iterations. Code and supplementary material is available at https://tinyurl.com/saved-rl.**

***Index Terms*—Reinforcement learning, imitation learning, optimal control.**

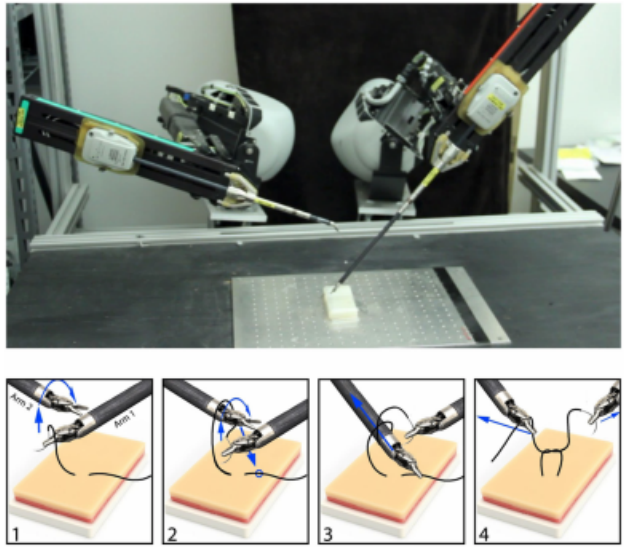
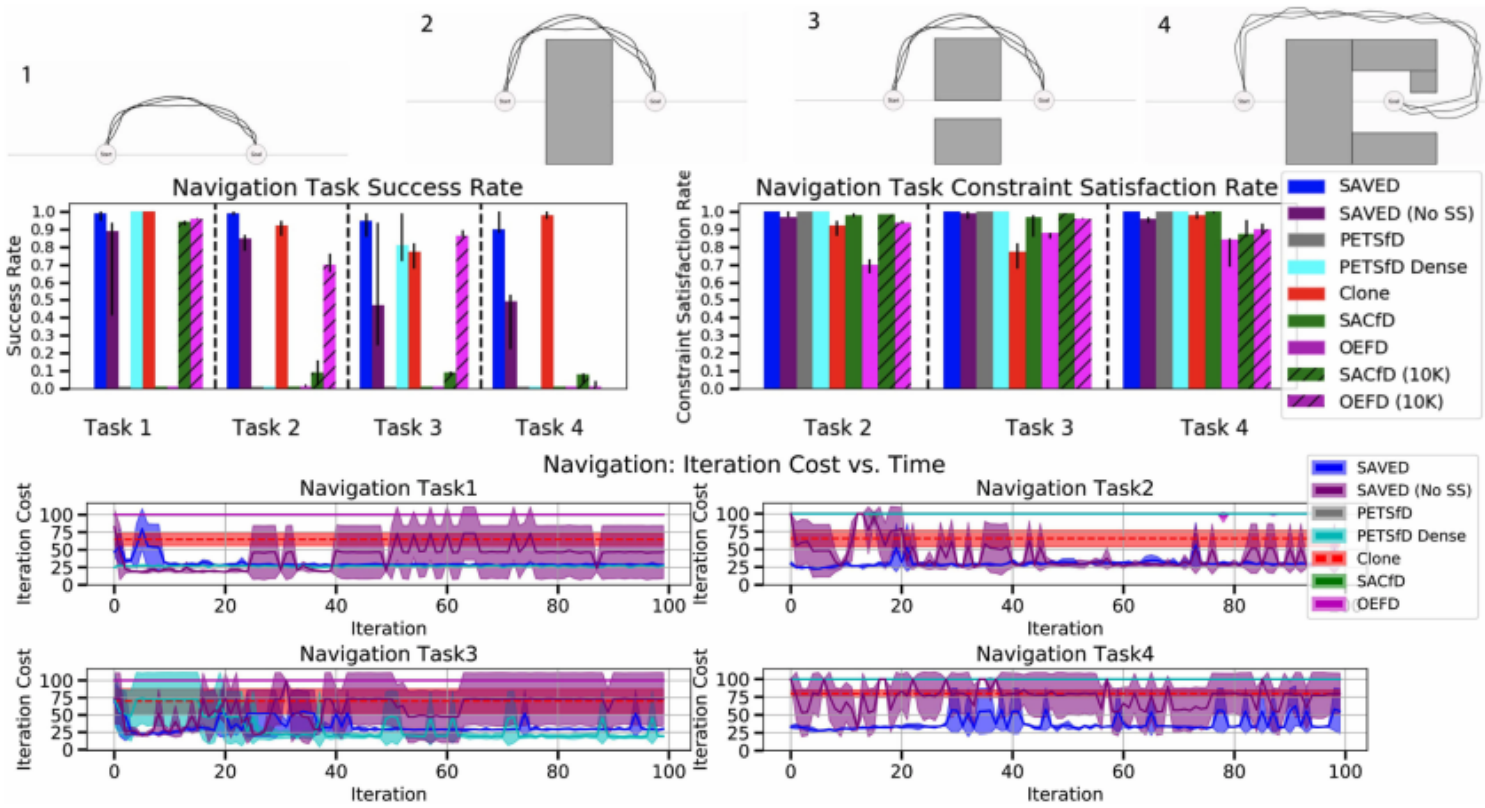


Fig. 1. SAVED is able to safely learn maneuvers on the da Vinci surgical

B. Thananjeyan*, A. Balakrishna*, U. Rosolia, F. Li, R. McAllister, J. E. Gonzalez, S. Levine, F. Borrelli, and K. Goldberg. "Safety augmented value estimation from demonstrations (saved): Safe deep model-based rl for sparse cost robotic tasks." IEEE Robotics and Automation Letters 5 (2020).

Why do we care about feasibility in model-based RL?



Key Message: Receding horizon planning may lead to safety constraints violation!

B. Thananjeyan*, A. Balakrishna*, U. Rosolia, F. Li, R. McAllister, J. E. Gonzalez, S. Levine, F. Borrelli, and K. Goldberg. "Safety augmented value estimation from demonstrations (saved): Safe deep model-based rl for sparse cost robotic tasks." IEEE Robotics and Automation Letters 5 (2020).

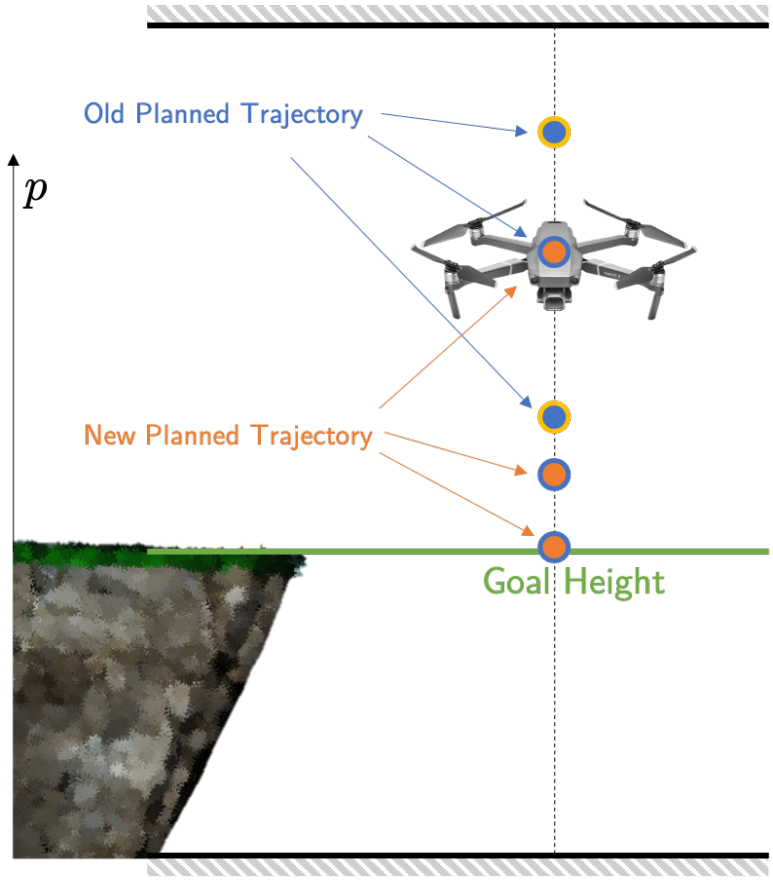
Table of Contents

Problem Formulation

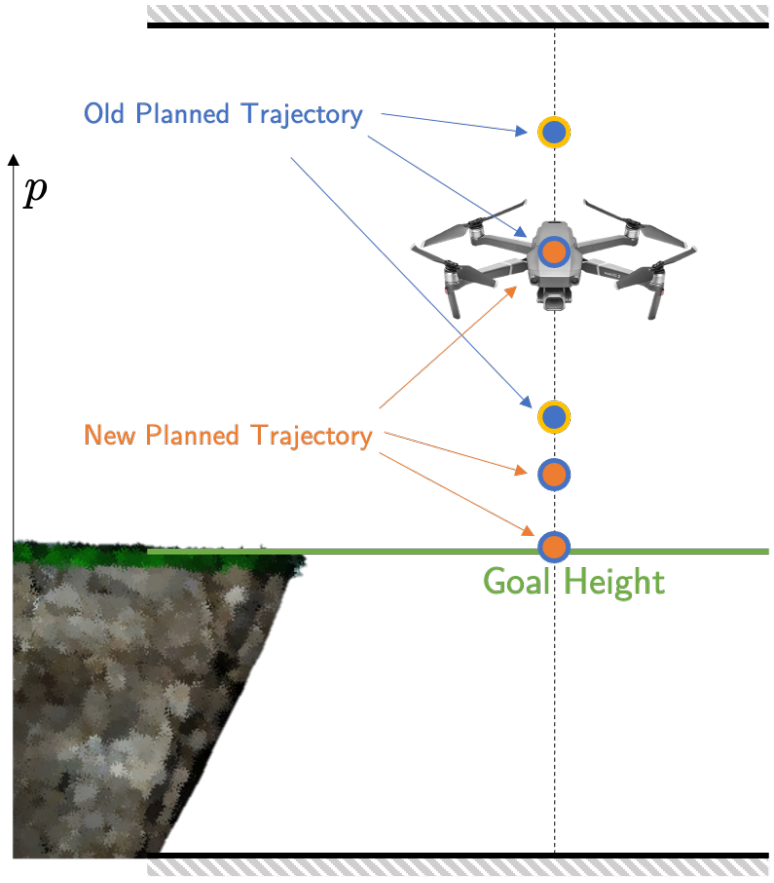
Model Predictive Control
- History of MPC
- Numerical Example – The Drone Regulation Problem

MPC Closed-loop Properties
- Notation
- Terminal Components
- Constraints Satisfaction
- Stability
- Nonlinear Systems

Notation

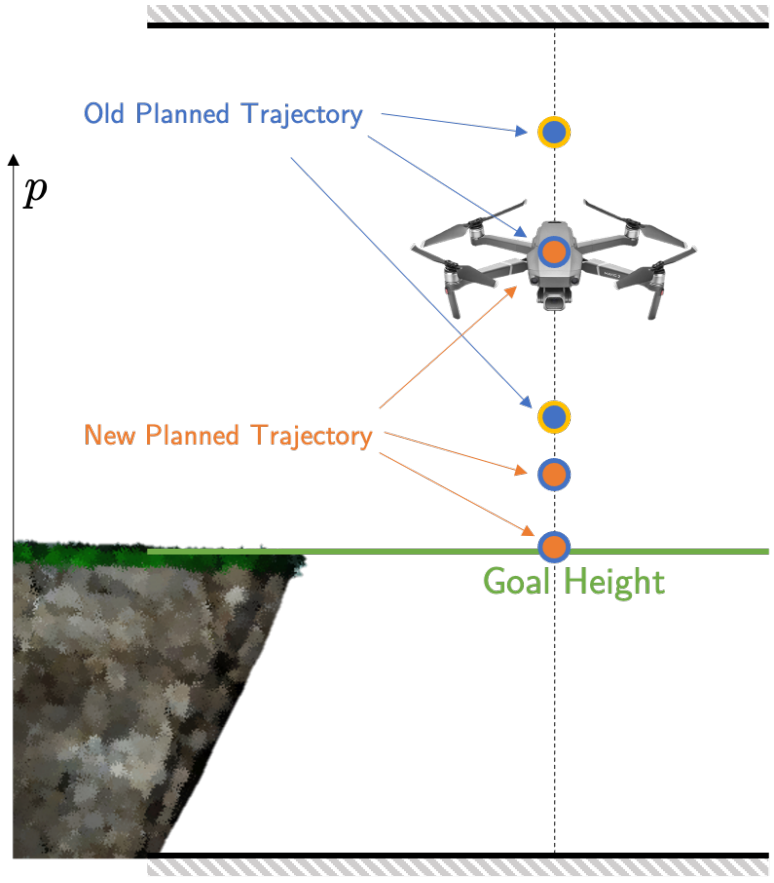

Old Planned Trajectory
p
New Planned Trajectory
Goal Height

Notation



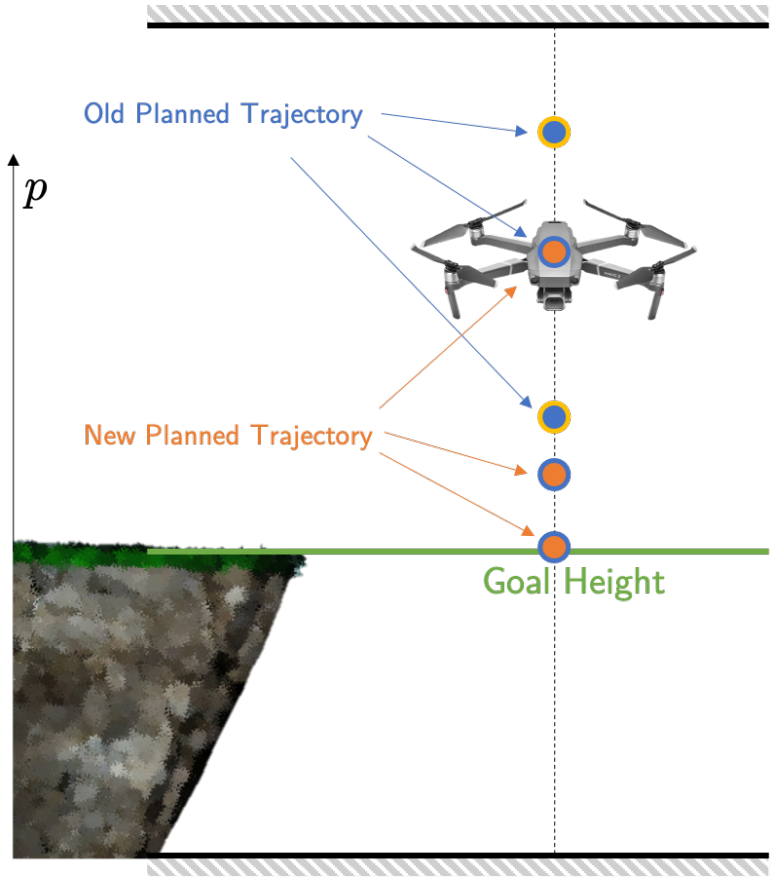
- Need to distinguish between the optimal plan at time t and the optimal plan at time $t+1$.

Notation



- Need to distinguish between the optimal plan at time t and the optimal plan at time $t+1$.
- $u^*_{k|t}$ is the input that we will apply at time k, if we follow the plan computed at time t.

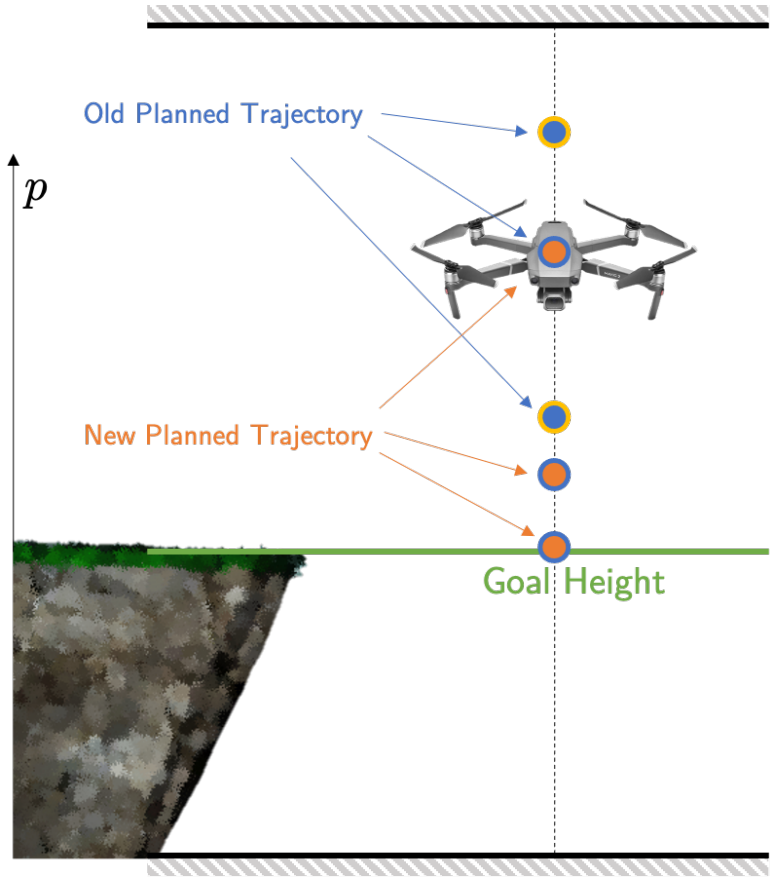
Notation



- Need to distinguish between the optimal plan at time t and the optimal plan at time $t+1$.
- $u^*_{k|t}$ is the input that we will apply at time k, if we follow the plan computed at time t.
- At time t, we plan

$$\{u^*_{t|t}, u^*_{t+1|t}, \ldots, u^*_{t+N-1|t}\}$$

Notation



- Need to distinguish between the optimal plan at time t and the optimal plan at time $t+1$.
- $u^*_{k|t}$ is the input that we will apply at time k, if we follow the plan computed at time t.
- At time t, we plan

$$\{u^*_{t|t}, u^*_{t+1|t}, \ldots, u^*_{t+N-1|t}\}$$

- At time $t+1$, we plan

$$\{u^*_{t+1|t+1}, u^*_{t+2|t+1}, \ldots, u^*_{t+N|t+1}\}$$

Table of Contents

Problem Formulation

Model Predictive Control
- History of MPC
- Numerical Example – The Drone Regulation Problem

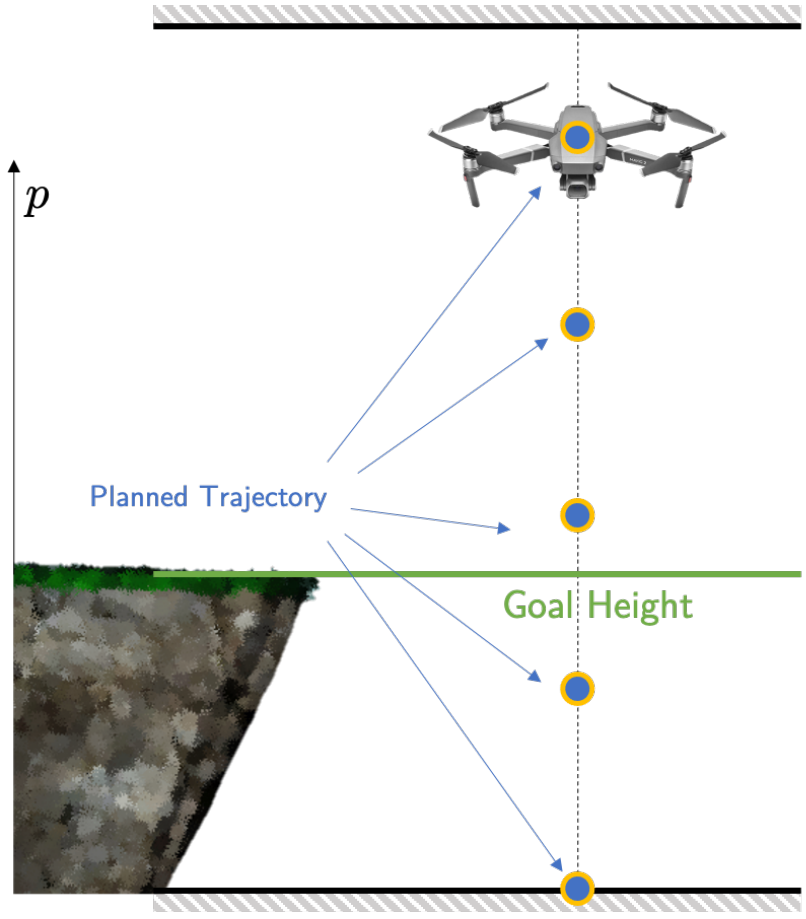
MPC Closed-loop Properties
- Notation
- Terminal Components
- Constraints Satisfaction
- Stability
- Nonlinear Systems

Constraint Satisfaction and Stability

$$
\begin{aligned}
J_t^*(x(0)) = \min_{u_{t|t},\ldots,u_{t+N-1|t}} \quad & \sum_{k=0}^{N-1} h(x_{k|t}, u_{k|t}) + V(x_{t+N|t}) \\
\text{subject to} \quad & x_{k+1|t} = Ax_{k|t} + Bu_{k|t}, \forall k \in \{t, \ldots, t+N-1\} \\
& x_{k|t} \in \mathcal{X}, u_{k|t} \in \mathcal{U}, \forall k \in \{t, \ldots, t+N-1\} \\
& x_{t|t} = x(0), x_N \in \mathcal{X}_F
\end{aligned}
$$

Solution: The terminal cost $V(x_{t+T|t})$ and terminal constraint $\mathcal{X}_F$, often referred to as *terminal components*, should approximate the tail of cost and constraints beyond the prediction horizon.

Drone Regulation Problem



Constraint Satisfaction and Stability – Solution

Recursive feasibility

If at time t the MPC problem is feasible $\rightarrow$ the MPC problem is feasible at time $t+1$.

Constraint Satisfaction and Stability – Solution

Recursive feasibility

If at time t the MPC problem is feasible $\rightarrow$ the MPC problem is feasible at time $t+1$.

Stability

We to guarantee that the closed-loop is stable. Informally, we want to guarantee that

$$\lim_{t \to \infty} ||x_t|| = 0.$$

Table of Contents

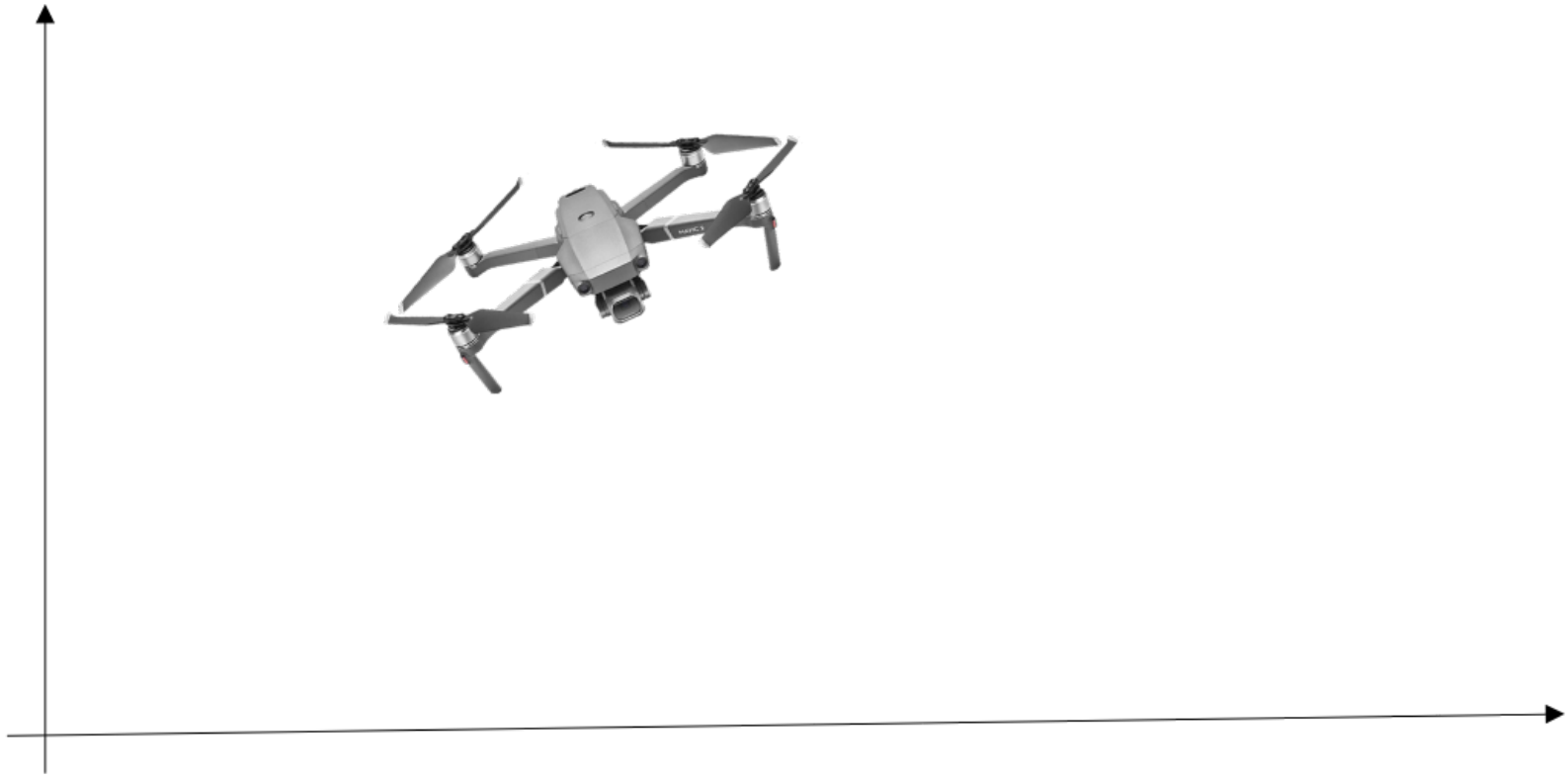
Problem Formulation

Model Predictive Control
- History of MPC
- Numerical Example – The Drone Regulation Problem

MPC Closed-loop Properties
- Notation
- Terminal Components
- Constraints Satisfaction
- Stability
- Nonlinear Systems

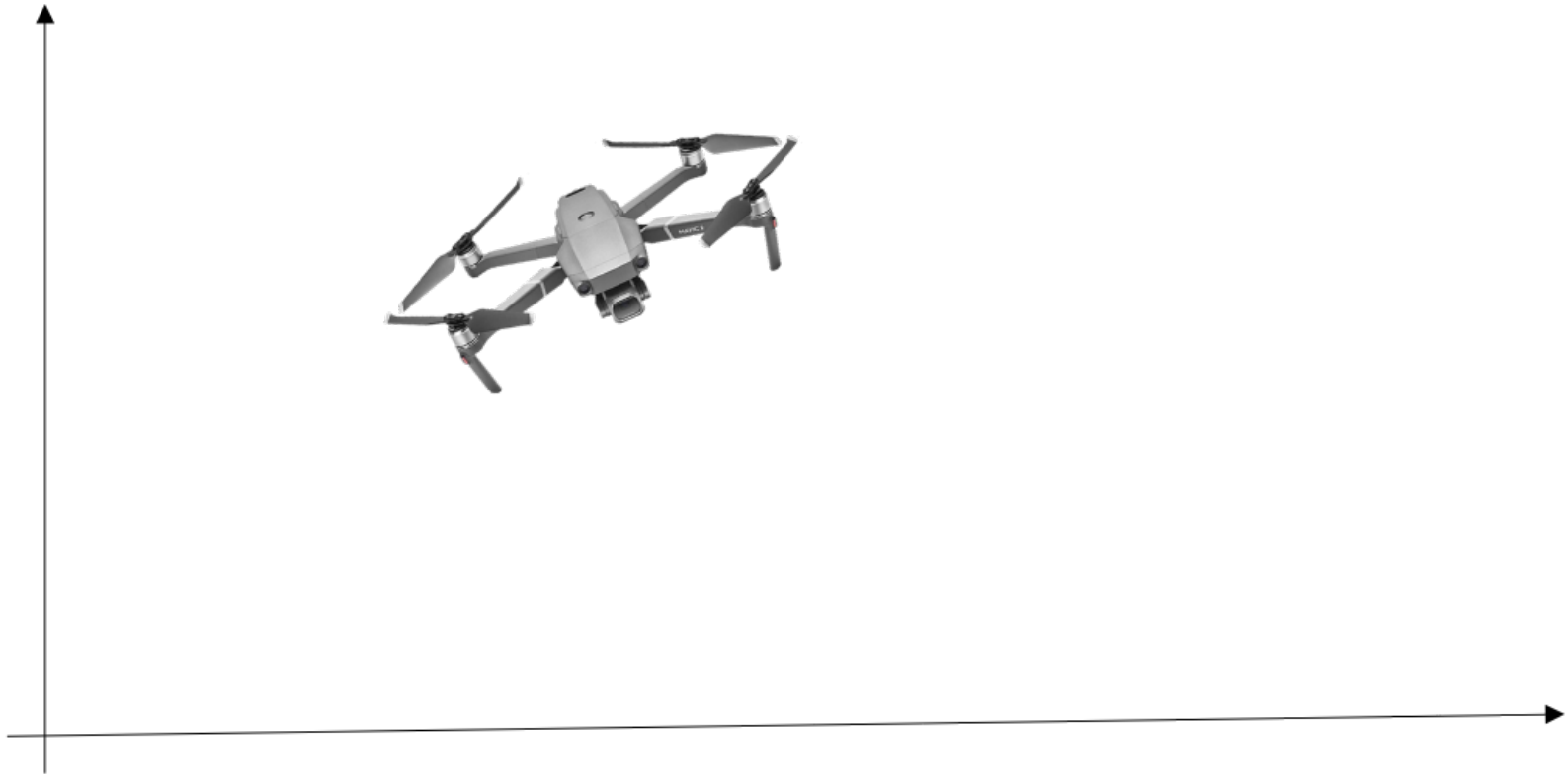
Recursive Feasibility

Consider the terminal set $\mathcal{X}_F = x_g = \{0\}$, which is an unforced equilibrium point for the system (i.e., $x_g = Ax_g$).



Recursive Feasibility

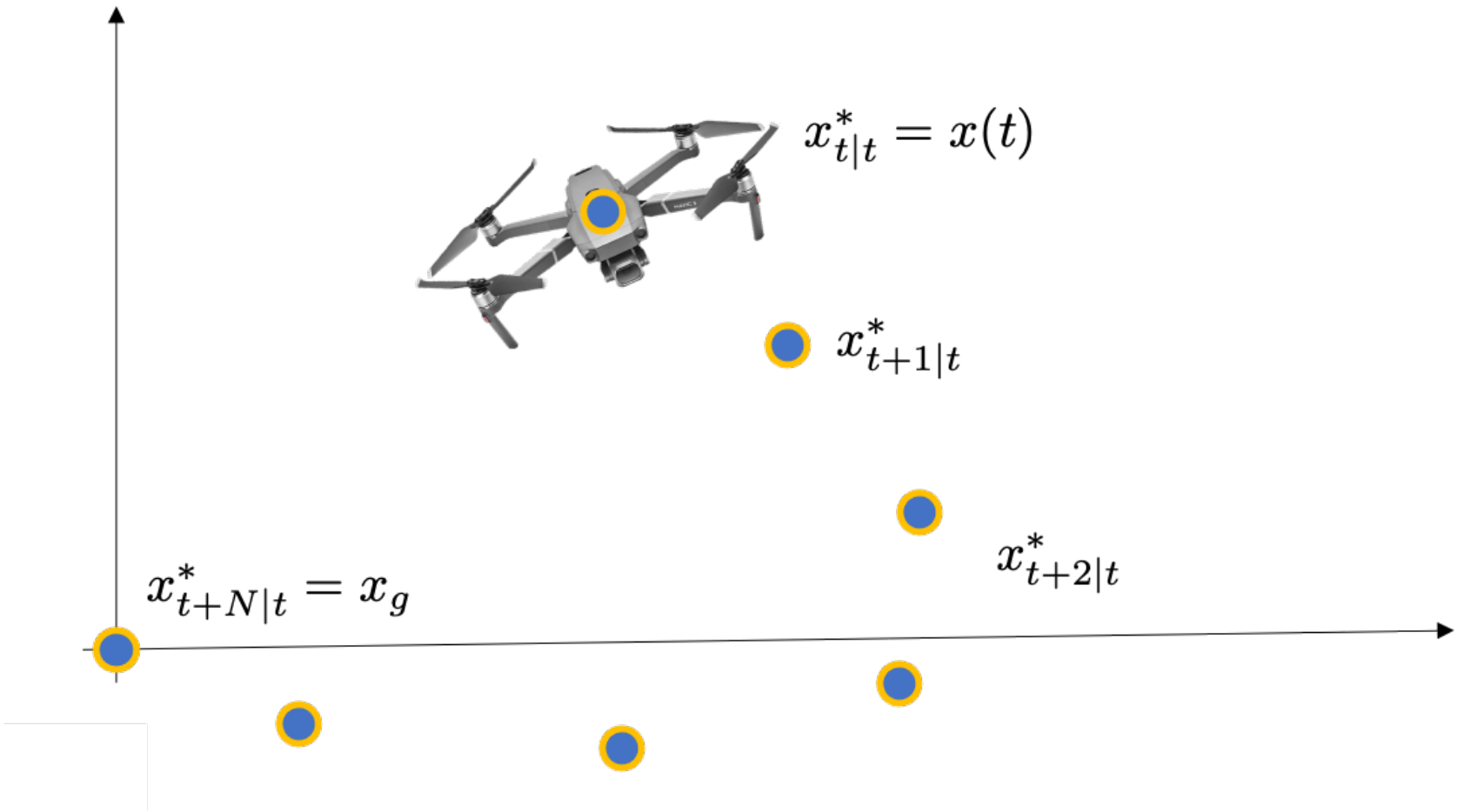
Consider the terminal set $\mathcal{X}_F = x_g = \{0\}$, which is an unforced equilibrium point for the system (i.e., $x_g = Ax_g$).



Assume that at time $t = 0$ the MPC problem is feasible.

Recursive Feasibility

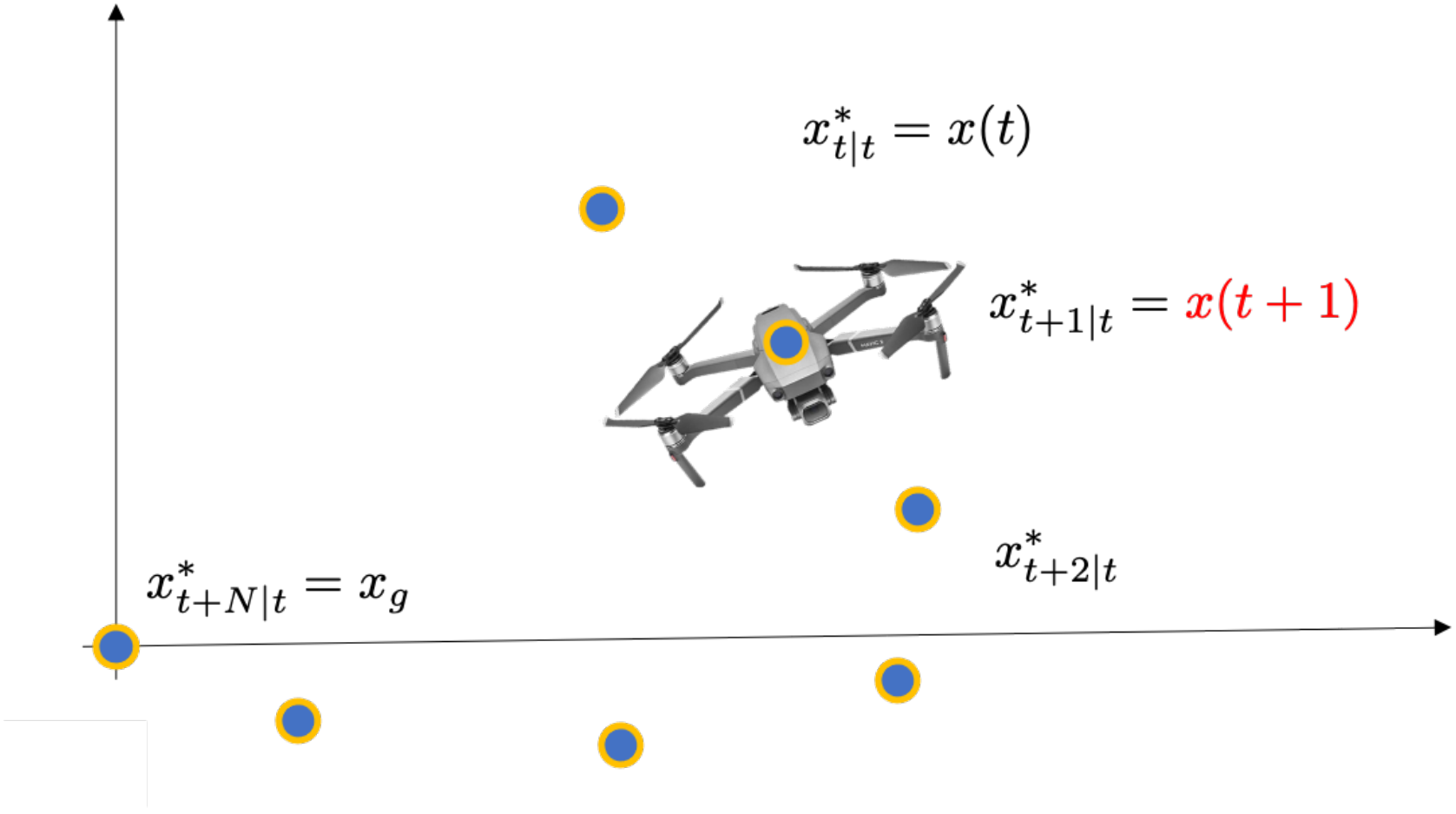
Consider the terminal set $\mathcal{X}_F = x_g = \{0\}$, which is an unforced equilibrium point for the system (i.e., $x_g = Ax_g$).



Let $\{x^*_{t|t}, \ldots, x^*_{t+N|t}\}$ and $\{u^*_{t|t}, \ldots, u^*_{t+N-1|t}\}$ be the optimal state-input sequences.

Recursive Feasibility

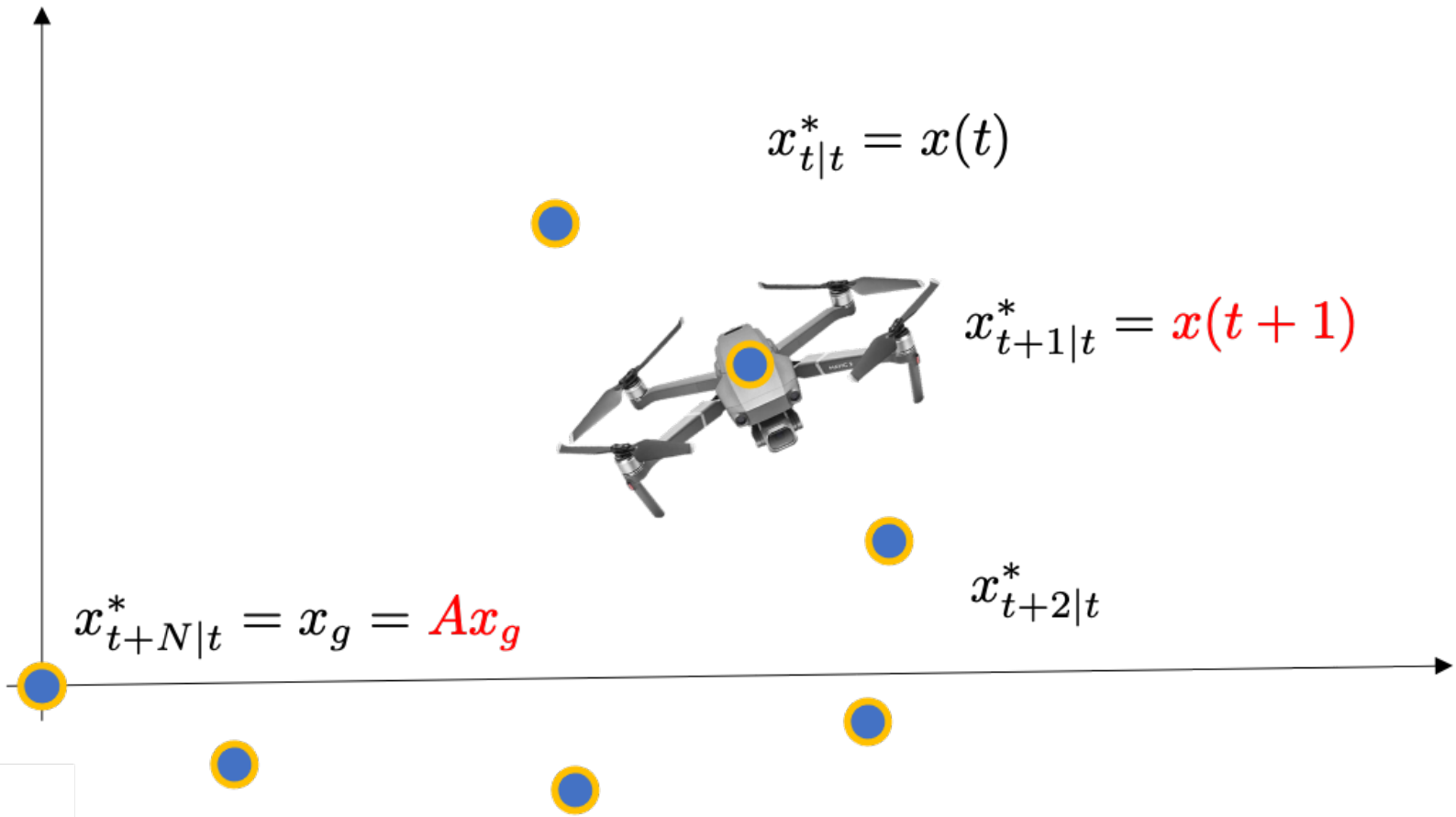
Consider the terminal set $\mathcal{X}_F = x_g = \{0\}$, which is an unforced equilibrium point for the system (i.e., $x_g = Ax_g$).



Apply $u^*_{t|t}$ to the system.

Recursive Feasibility

Consider the terminal set $\mathcal{X}_F = x_g = \{0\}$, which is an unforced equilibrium point for the system (i.e., $x_g = Ax_g$).



Then $\{x^*_{t+1|t}, \ldots, x^*_{t+N|t}, x_g\}$ and $\{u^*_{t+1|t}, \ldots, u^*_{t+N-1|t}, 0\}$ are feasible state-input sequences for the FTOCP at $t+1$.

Table of Contents

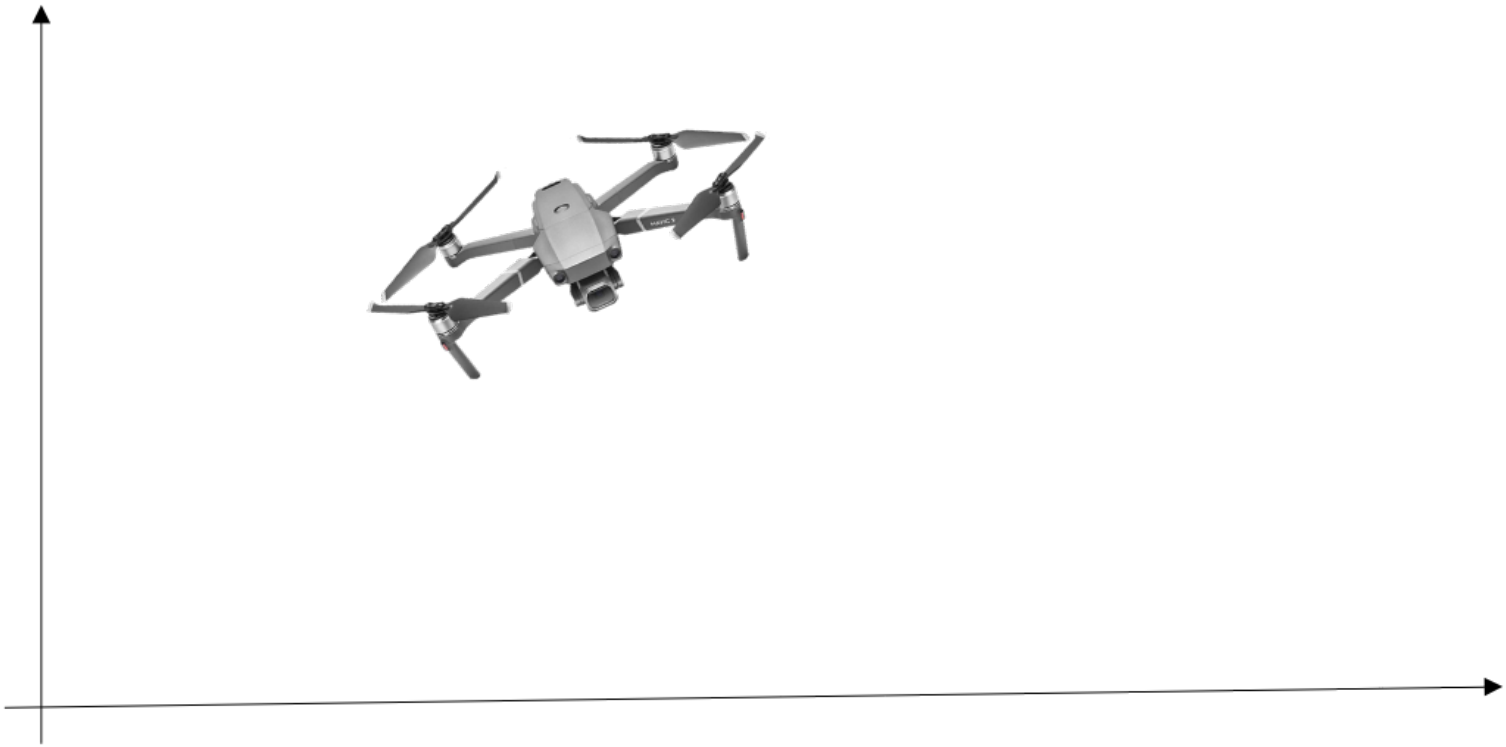
Problem Formulation

Model Predictive Control
- History of MPC
- Numerical Example – The Drone Regulation Problem

MPC Closed-loop Properties
- Notation
- Terminal Components
- Constraints Satisfaction
- Stability
- Nonlinear Systems

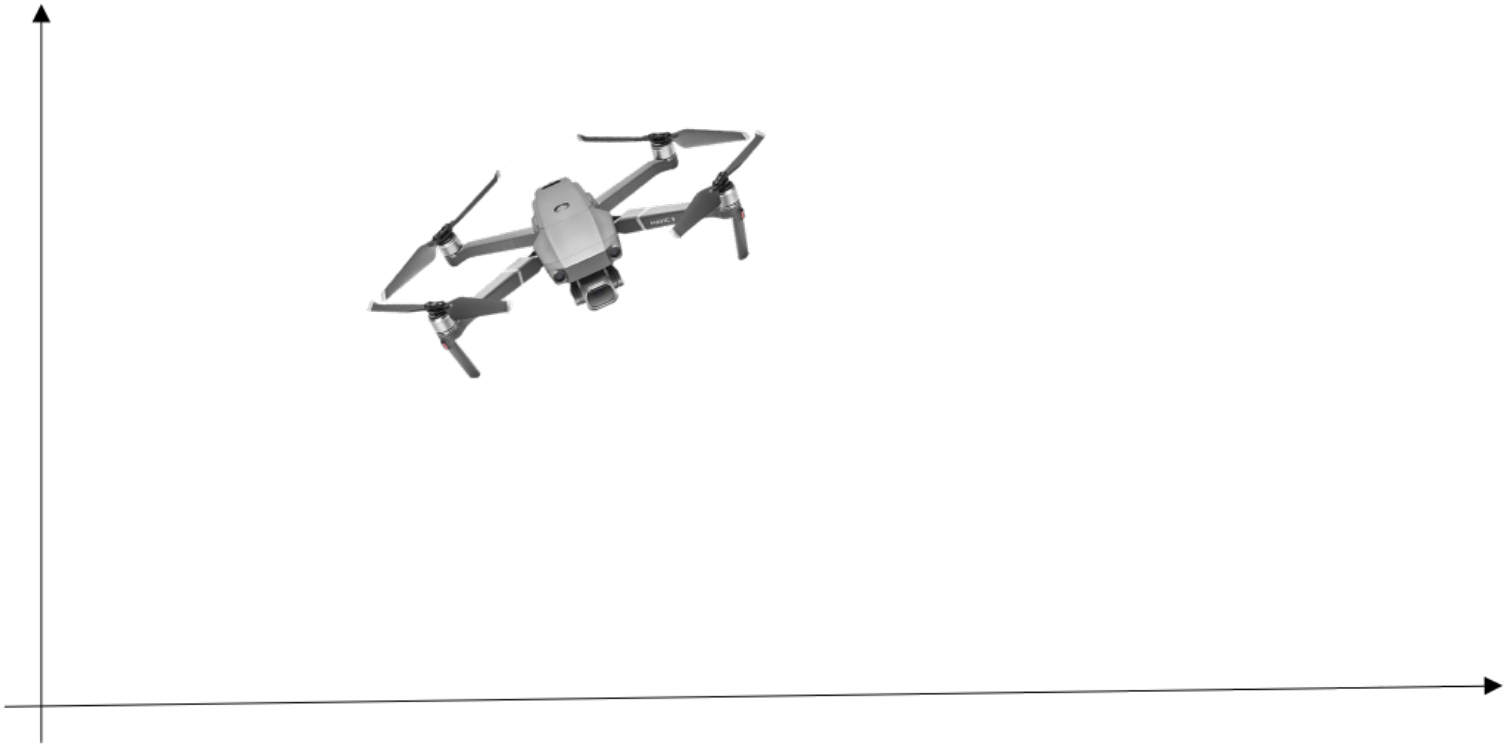
Stability

Consider the terminal cost $V(x) = x^\top Q_F x$.
Recall that $h(x, u) = x^\top Q x + u^\top R u$.



Stability

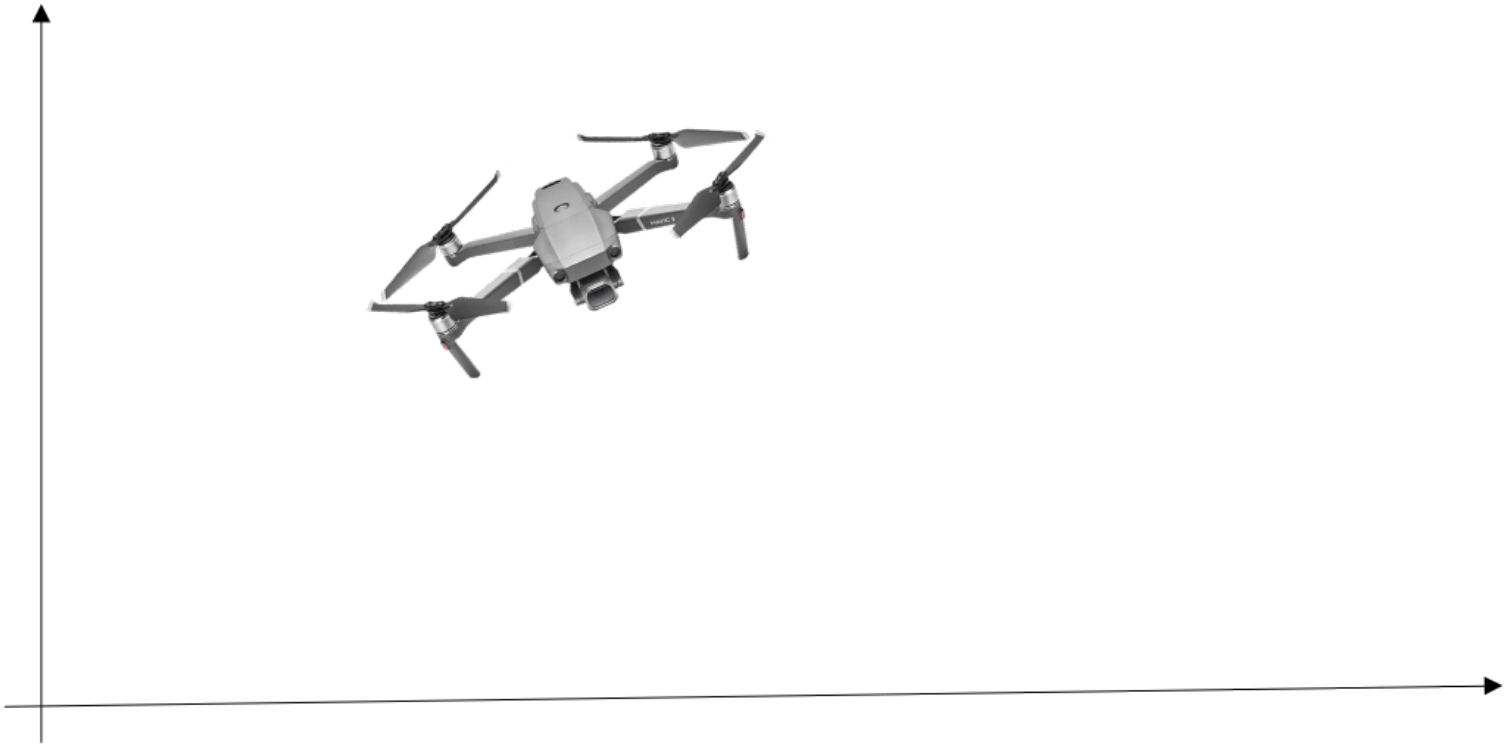
Consider the terminal cost $V(x) = x^\top Q_F x$.
Recall that $h(x, u) = x^\top Q x + u^\top R u$.



Goal: show that $\lim_{t\to\infty} J_t^*(x(t)) = 0$.

Stability

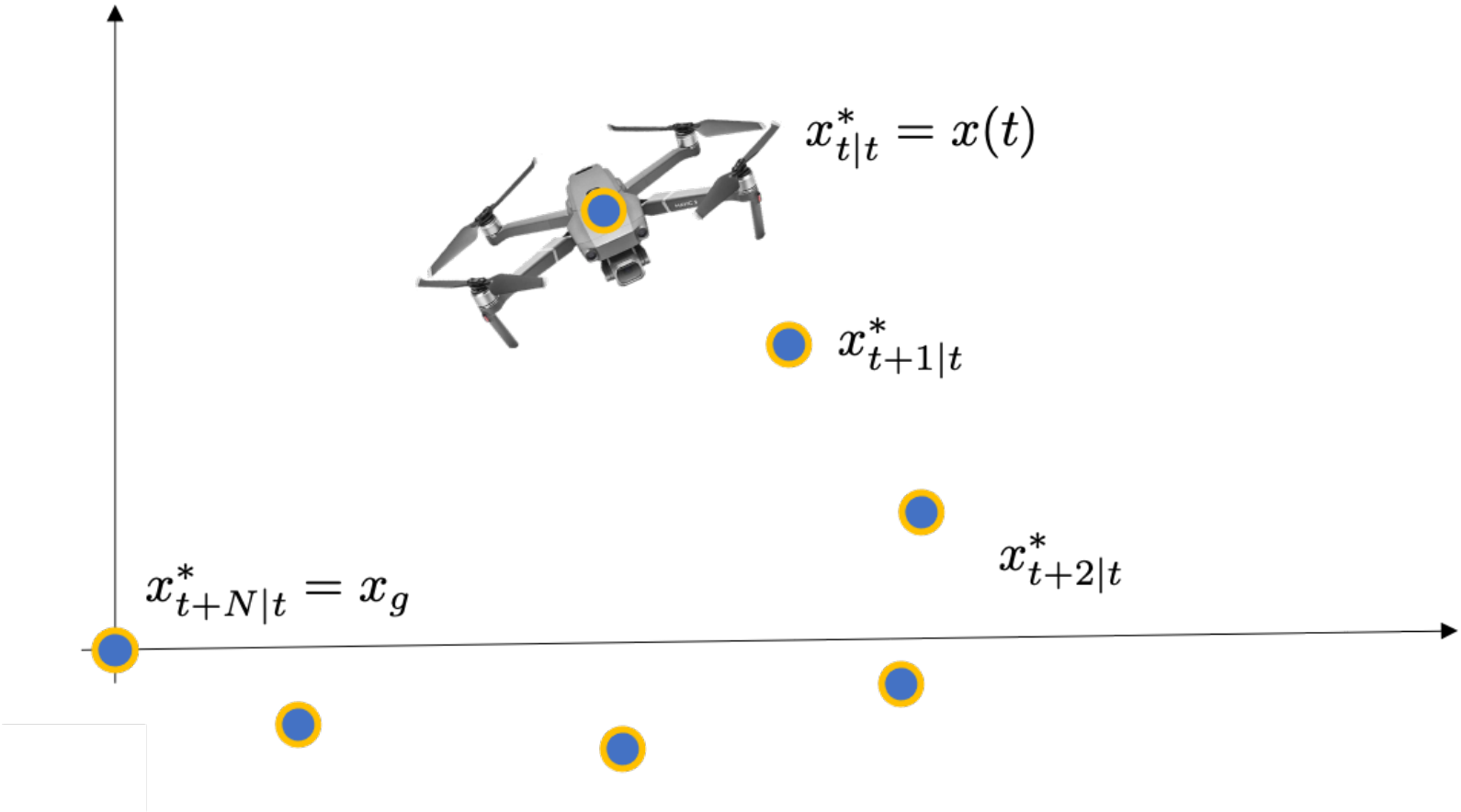
Consider the terminal cost $V(x) = x^\top Q_F x$.
Recall that $h(x, u) = x^\top Q x + u^\top R u$.



Assume that at time $t = 0$ the MPC problem is feasible.

Stability

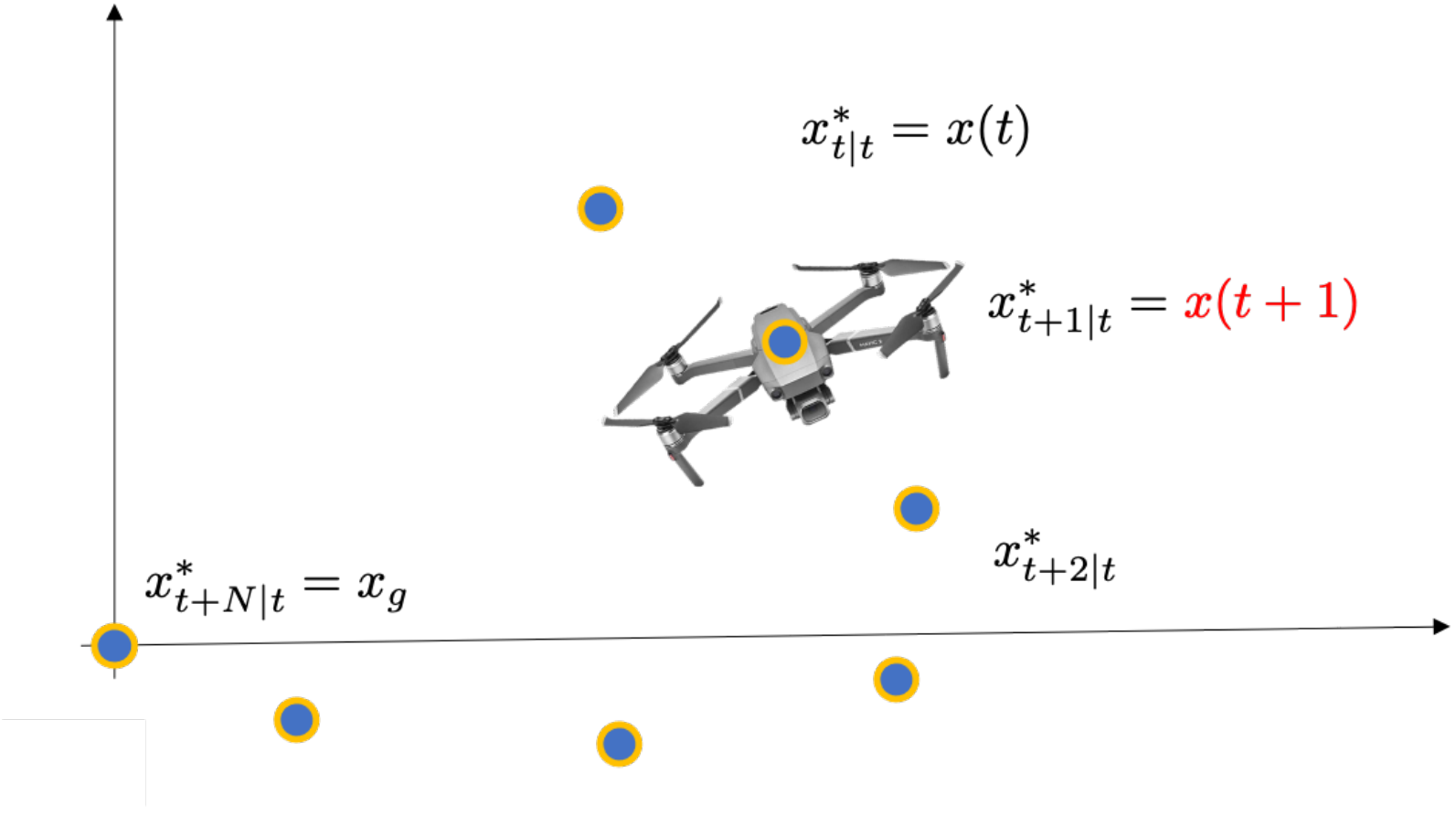
Consider the terminal cost $V(x) = x^\top Q_F x$.
Recall that $h(x, u) = x^\top Q x + u^\top R u$.



Let $J^*_t(x(t)) = \sum_{k=t}^{t+N-1} h(x^*_{k|t}, u^*_{k|t}) + V(x^*_{t+N|t})$ be the optimal cost.

Stability

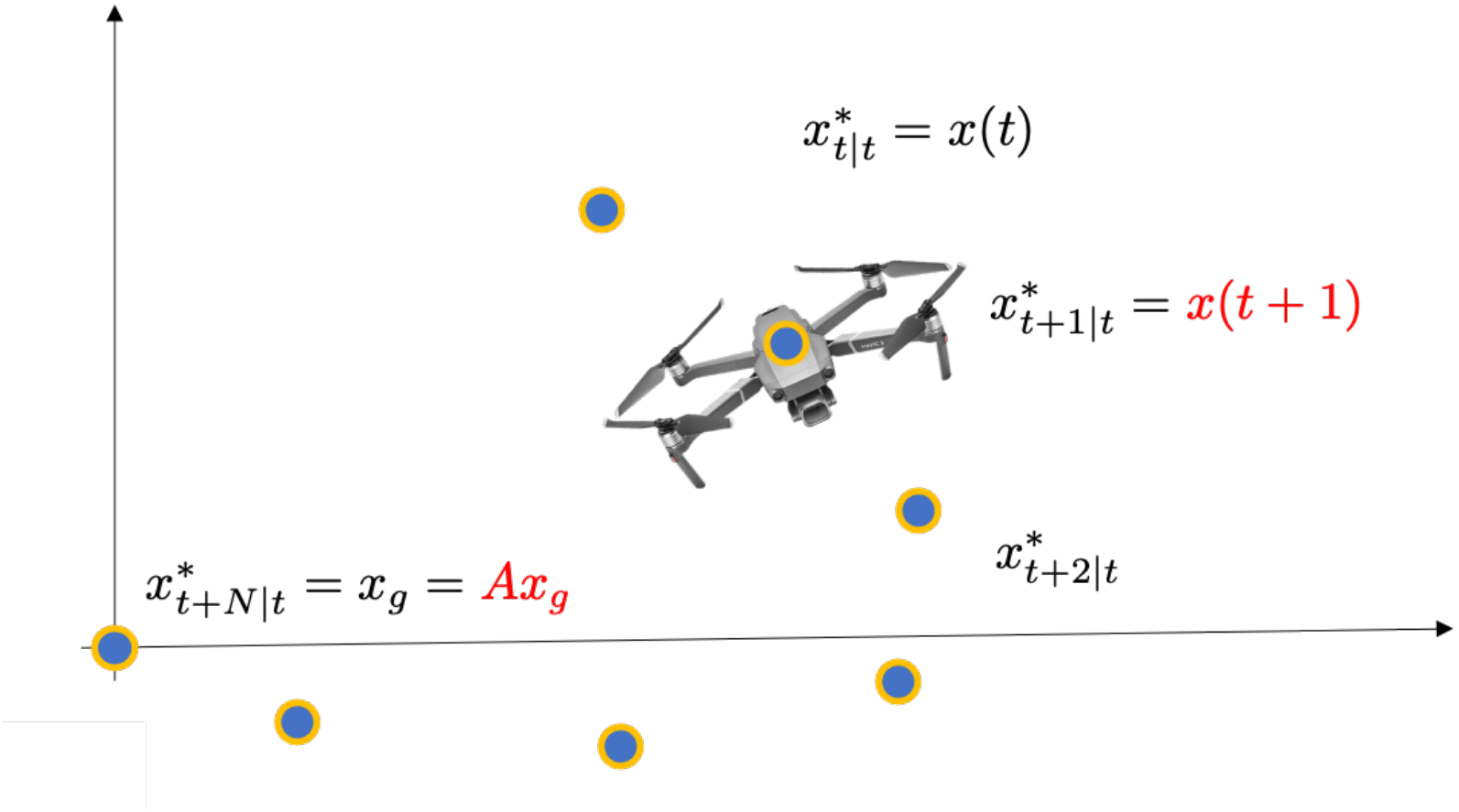
Consider the terminal cost $V(x) = x^\top Q_F x$.
Recall that $h(x, u) = x^\top Q x + u^\top R u$.



Apply $u^*_{t|t}$ to the system.

Stability

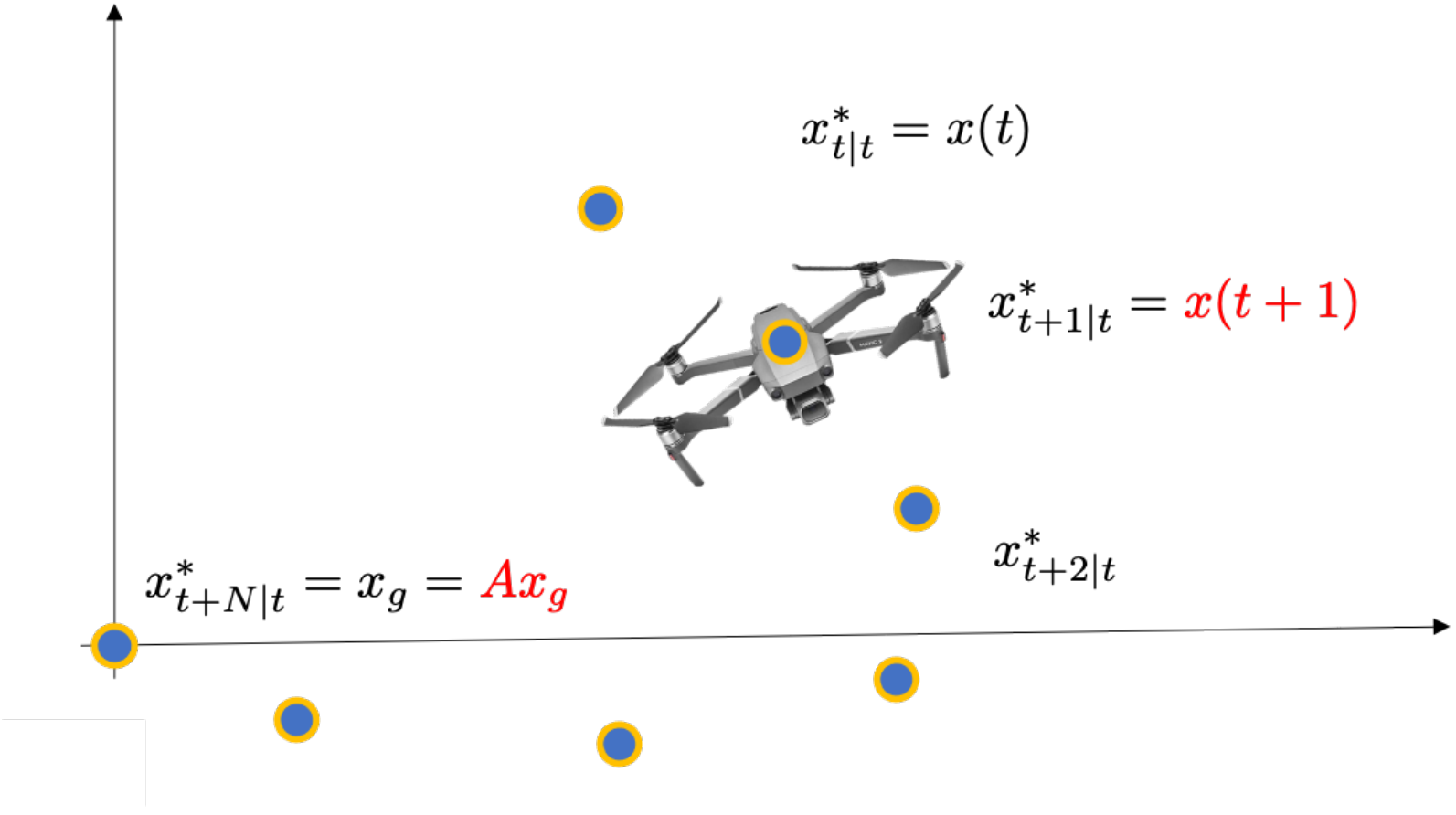
Consider the terminal cost $V(x) = x^\top Q_F x$.
Recall that $h(x, u) = x^\top Q x + u^\top R u$.



Then $\{x^*_{t+1|t}, \ldots, x^*_{t+N|t}, x_g\}$ and $\{u^*_{t+1|t}, \ldots, u^*_{t+N-1|t}, 0\}$ are feasible state-input sequences for the FTOCP at $t+1$.

Stability

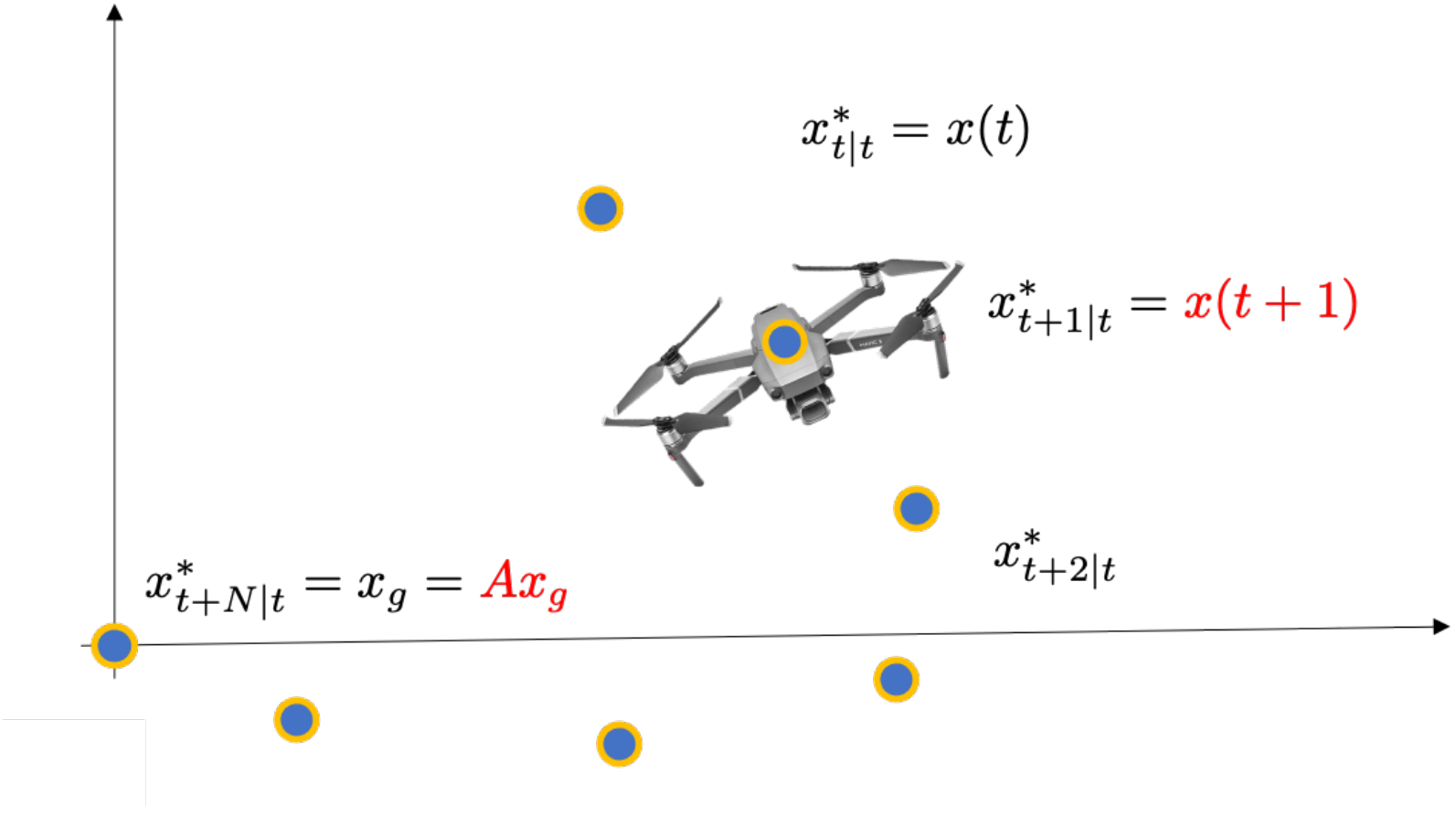
Consider the terminal cost $V(x) = x^\top Q_F x$.
Recall that $h(x, u) = x^\top Q x + u^\top R u$.



The cost is $J^*_t(x(t)) - h(x^*_{t|t}, u^*_{t|t})$.

Stability

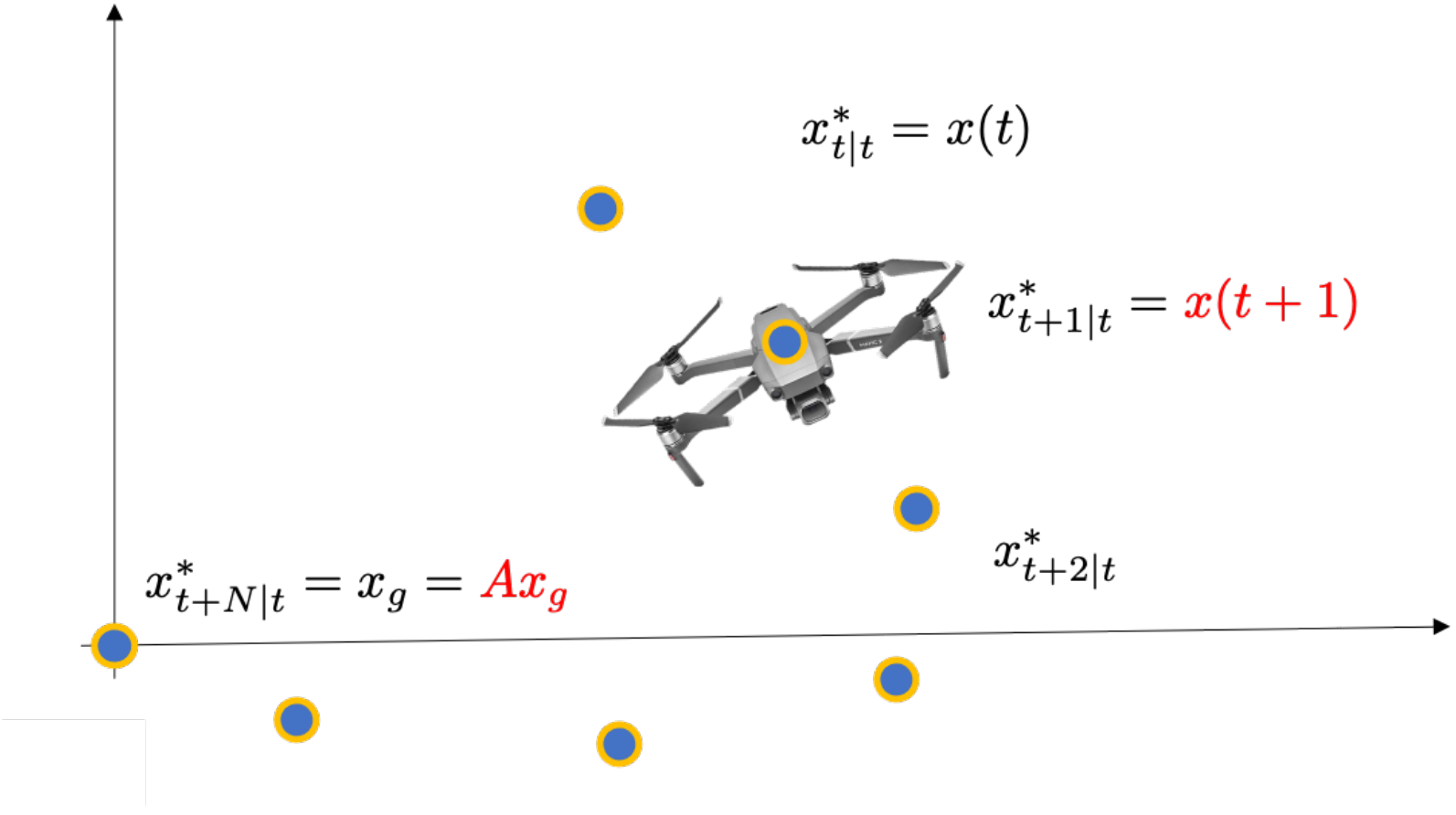
Consider the terminal cost $V(x) = x^\top Q_F x$.
Recall that $h(x, u) = x^\top Q x + u^\top R u$.



Then $J^*_{t+1}(x(t+1)) \leq J^*_t(x(t)) - h(x^*_{t|t}, u^*_{t|t})$.

Stability

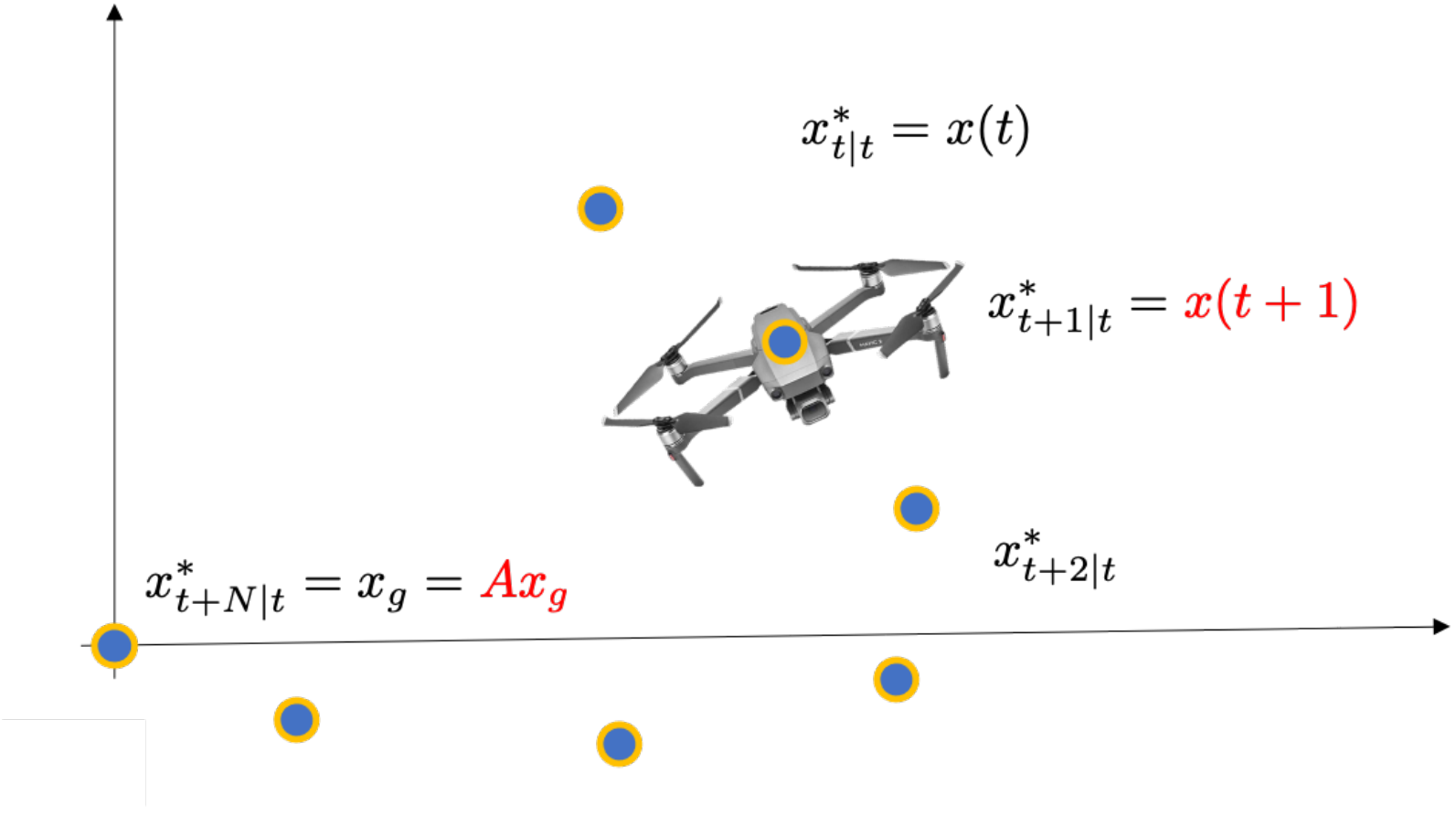
Consider the terminal cost $V(x) = x^\top Q_F x$.
Recall that $h(x, u) = x^\top Q x + u^\top R u$.



Thus $J^*_{t+1}(x(t+1)) < J^*_t(x(t))$ for all $x(t) \neq x_g$.

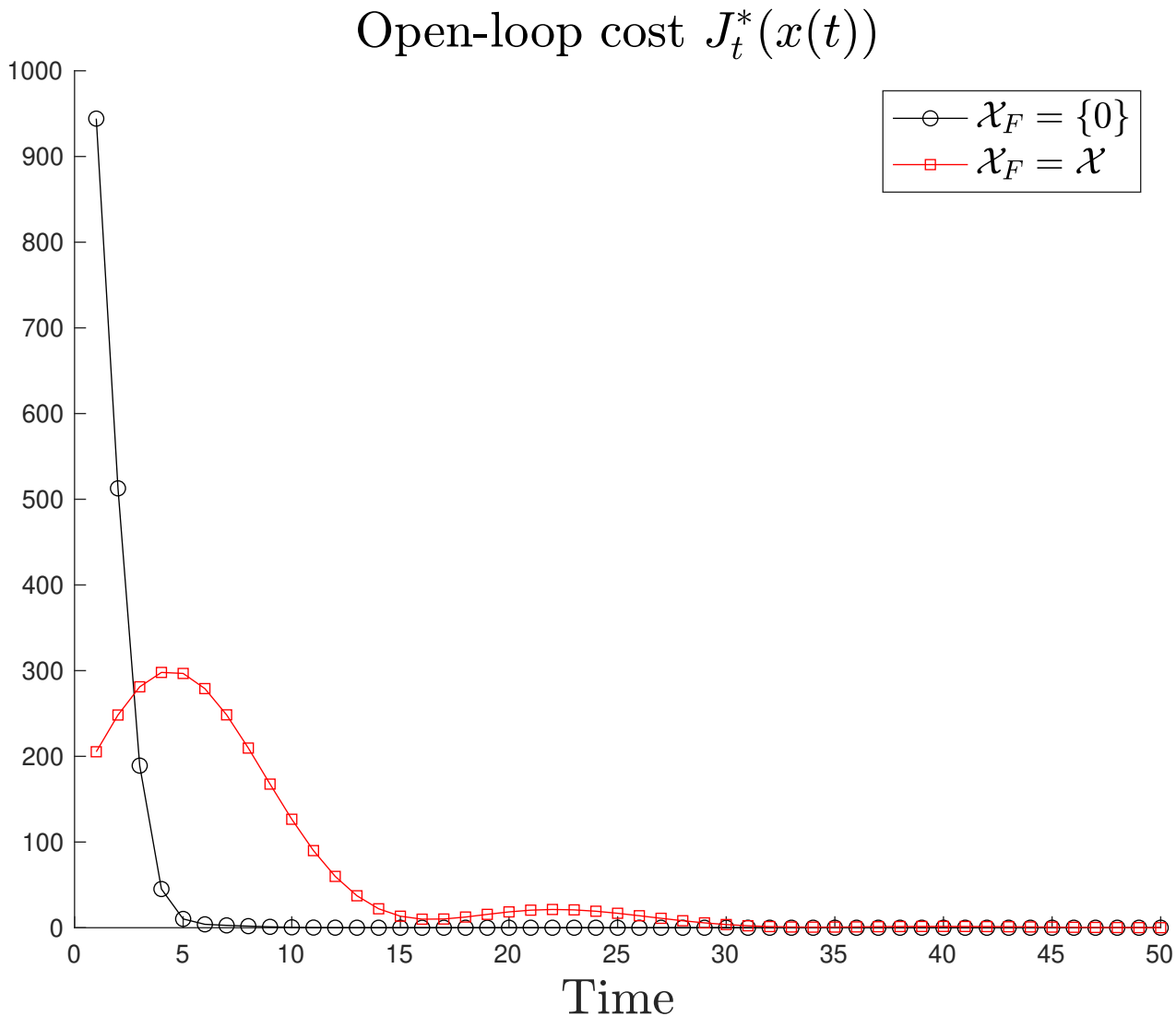
Stability

Consider the terminal cost $V(x) = x^\top Q_F x$.
Recall that $h(x, u) = x^\top Q x + u^\top R u$.



Therefore, $\lim_{t\to\infty} J^*_t(x(t)) = 0$.

Stability


Open-loop cost $J_t^*(x(t))$
$\mathcal{X}_F = \{0\}$
$\mathcal{X}_F = \mathcal{X}$
1000
900
800
700
600
500
400
300
200
100
0
0
5
10
15
20
25
30
35
40
45
50
Time

Summary of Safety and Stability Properties

Key Message: When the MPC terminal components are not designed correctly, the closed-loop system may violate safety constraints and convergence to the goal state/set is not guaranteed

A solution: We have shown that for the terminal set $\mathcal{X}_F = \{0\}$ and the terminal cost $V(x) = 0$.

- The MPC problem is feasible at all times
- The closed-loop system is stable as for the positive definite open-loop cost we have $J^*_{t+1}(x(t+1)) < J^*_t(x(t)), \forall x(t) \notin \mathcal{X}_F$

Main drawback: The terminal constraint set affects the region of attraction of the controller.

Summary of Safety and Stability Properties

Key Message: When the MPC terminal components are not designed correctly, the closed-loop system may violate safety constraints and convergence to the goal state/set is not guaranteed

A solution: We have shown that for the terminal set $\mathcal{X}_F = \{0\}$ and the terminal cost $V(x) = 0$.

- The MPC problem is feasible at all times
- The closed-loop system is stable as for the positive definite open-loop cost we have $J^*_{t+1}(x(t+1)) < J^*_t(x(t)), \forall x(t) \notin \mathcal{X}_F$

Main drawback: The terminal constraint set affects the region of attraction of the controller.

Summary of Safety and Stability Properties

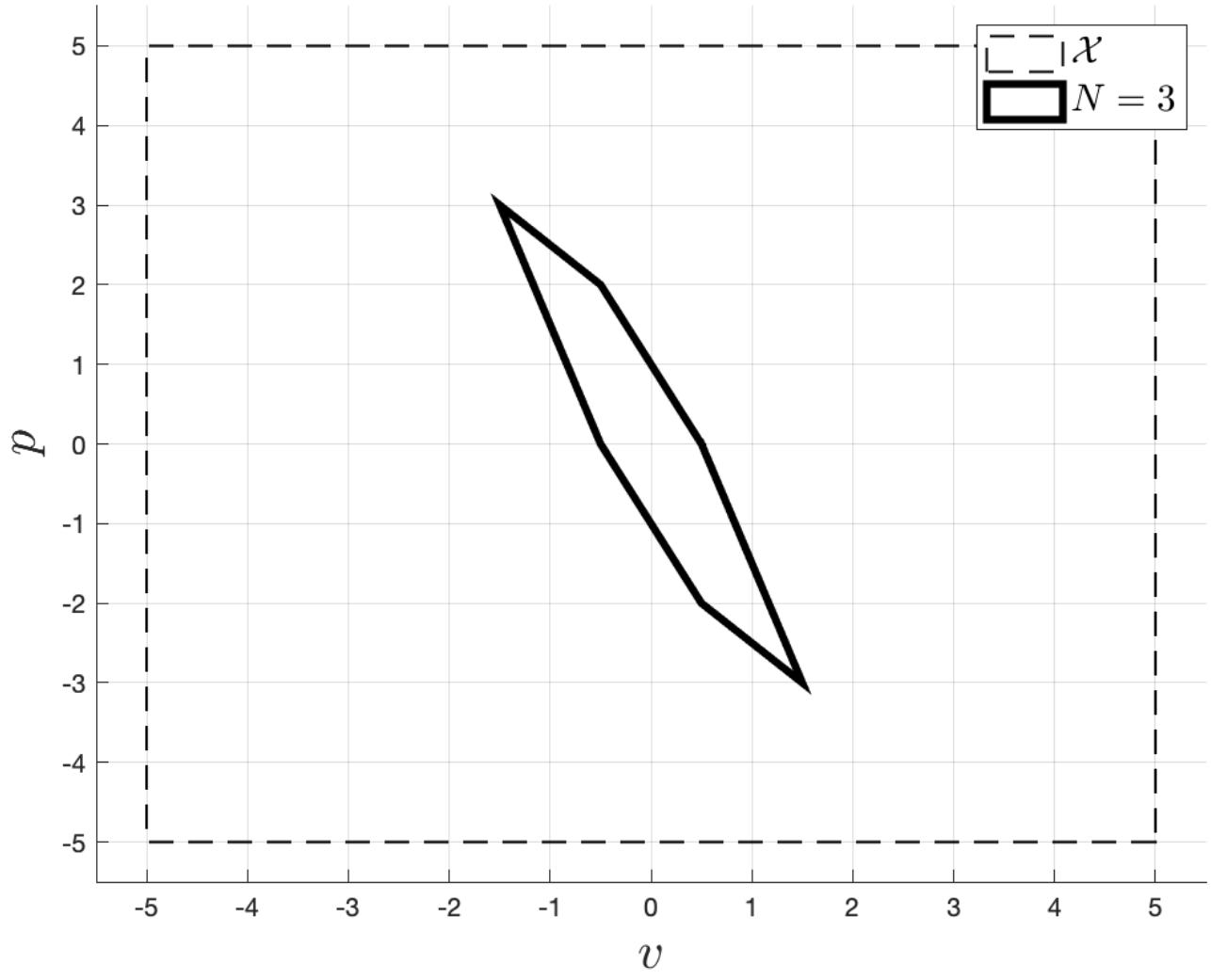
Key Message: When the MPC terminal components are not designed correctly, the closed-loop system may violate safety constraints and convergence to the goal state/set is not guaranteed

A solution: We have shown that for the terminal set $\mathcal{X}_F = \{0\}$ and the terminal cost $V(x) = 0$.

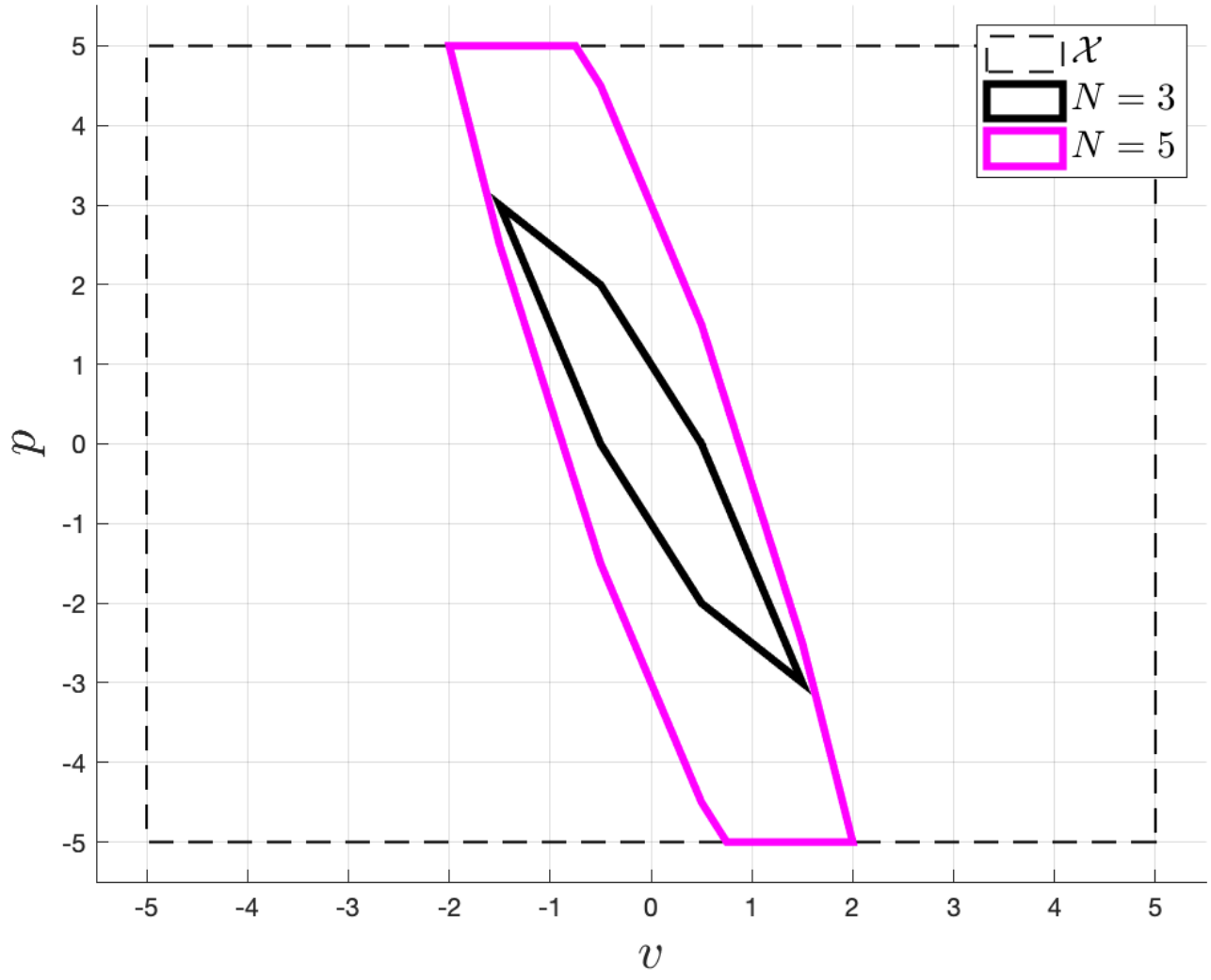
- The MPC problem is feasible at all times
- The closed-loop system is stable as for the positive definite open-loop cost we have $J^*_{t+1}(x(t+1)) < J^*_t(x(t)), \forall x(t) \notin \mathcal{X}_F$

Main drawback: The terminal constraint set affects the region of attraction of the controller.

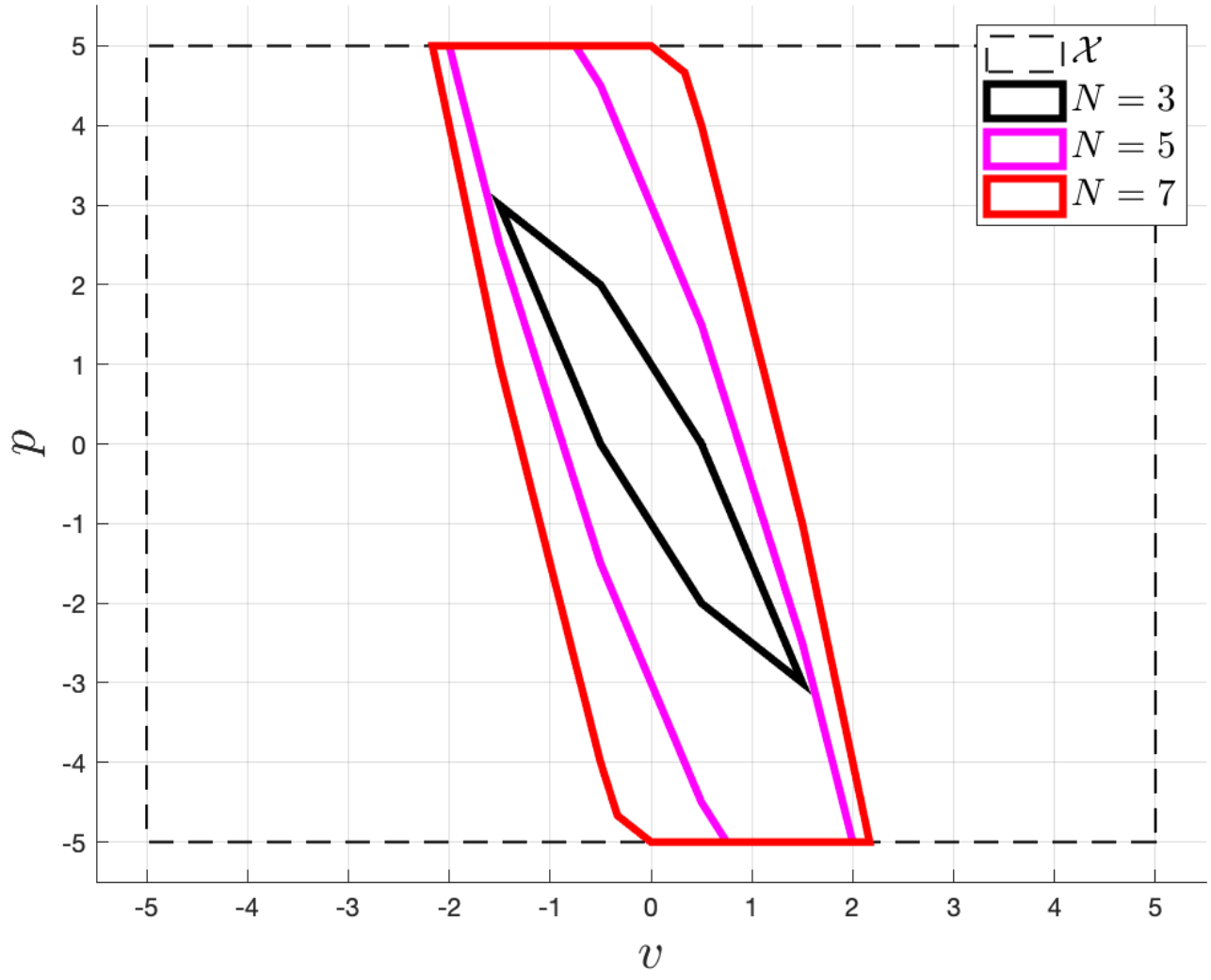
Region of Attraction



Region of Attraction



Region of Attraction



Region of Attraction

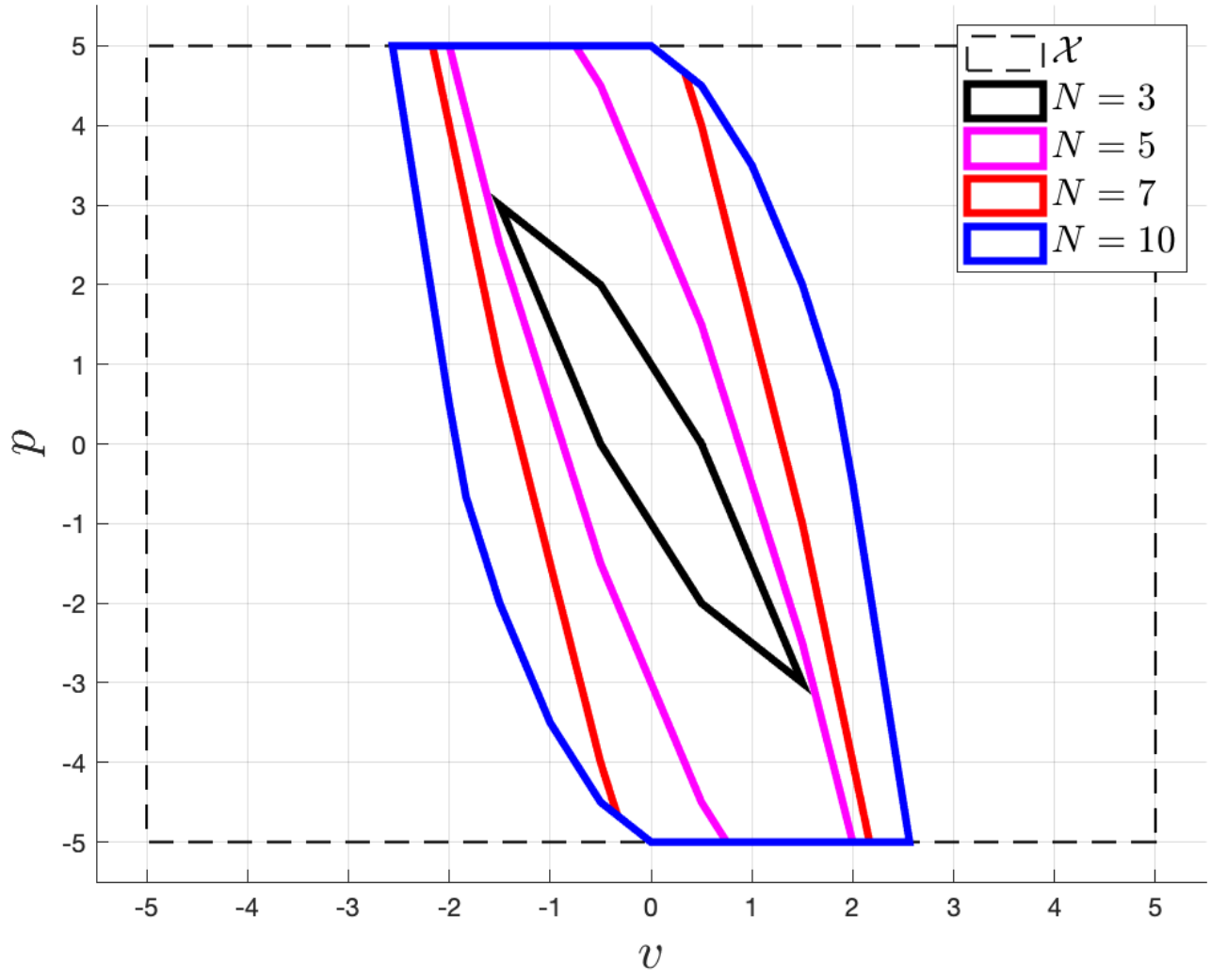


Table of Contents

Problem Formulation

Model Predictive Control
- History of MPC
- Numerical Example – The Drone Regulation Problem

MPC Closed-loop Properties
- Notation
- Terminal Components
- Constraints Satisfaction
- Stability
- Nonlinear Systems

Nonlinear MPC

Extension to Nonlinear MPC

Consider the nonlinear system dynamics: $x(t+1) = f(x(t), u(t))$

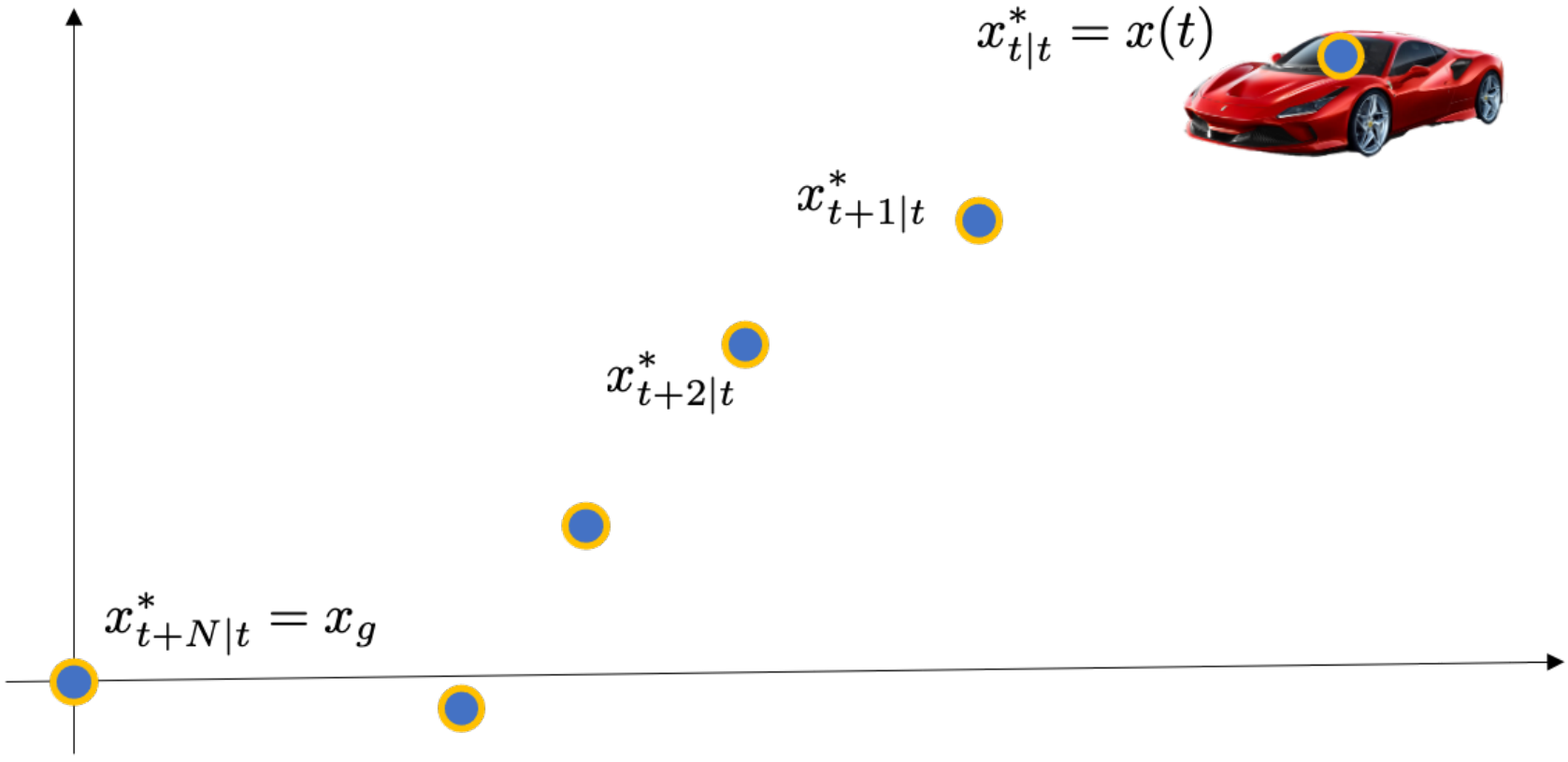
$$
\begin{aligned}
J_t^*(x(0)) = \min_{u_{t|t},\ldots,u_{t+N-1|t}} \quad & \sum_{k=0}^{T-1} h(x_{k|t}, u_{k|t}) + V(x_{t+T|t}) \\
\text{subject to} \quad & x_{k+1|t} = f(x_{k|t}, u_{k|t}), \forall k \in \{t, \ldots, t+N-1\} \\
& x_{k|t} \in \mathcal{X}, u_{k|t} \in \mathcal{U}, \forall k \in \{t, \ldots, t+N-1\} \\
& x_{t|t} = x(0), x_N \in \mathcal{X}_F
\end{aligned}
$$

- Our derivations did not rely on linearity!!!

Nonlinear MPC

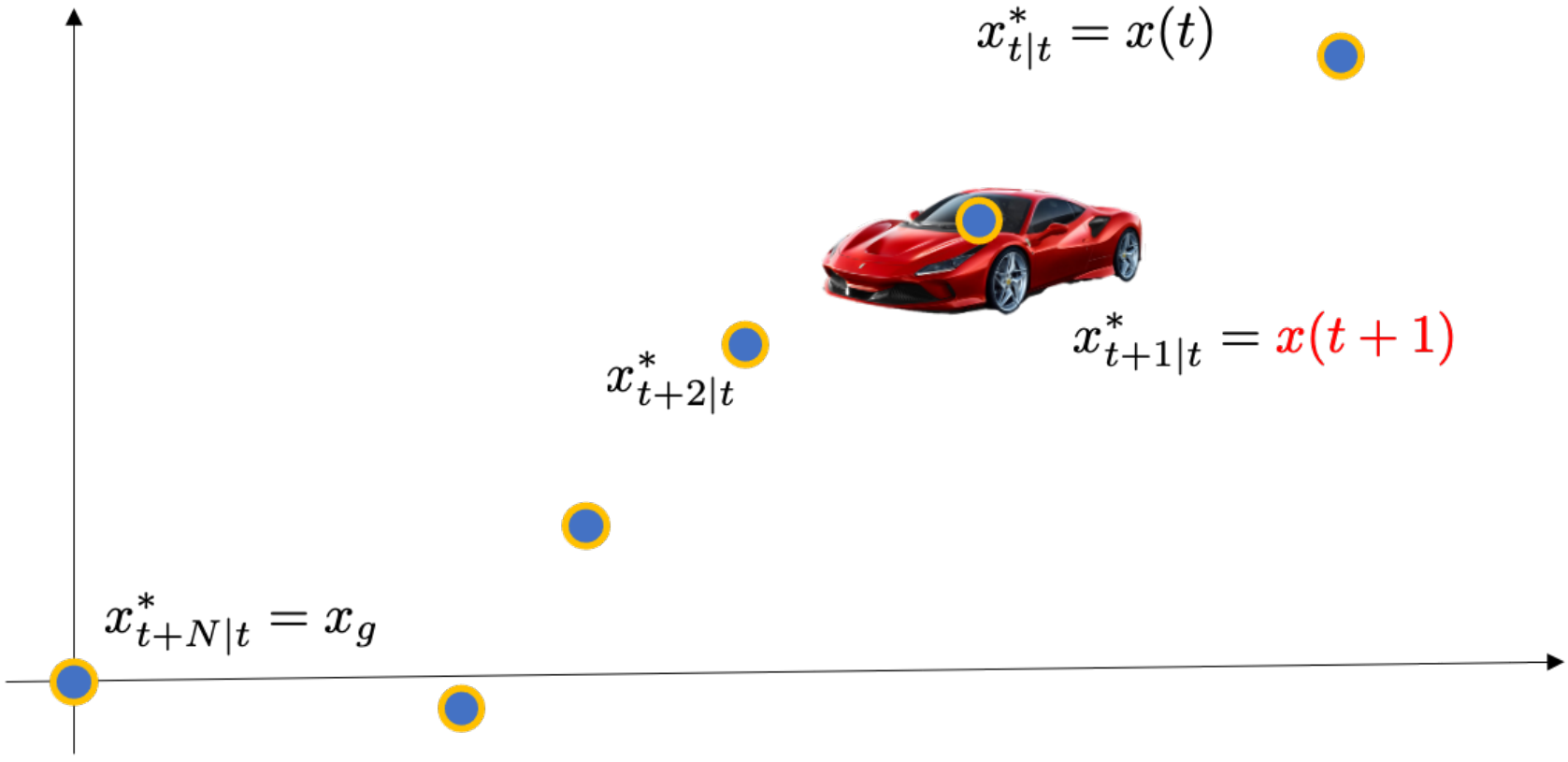
Assume that at time $t = 0$ the MPC problem is feasible.

Nonlinear MPC



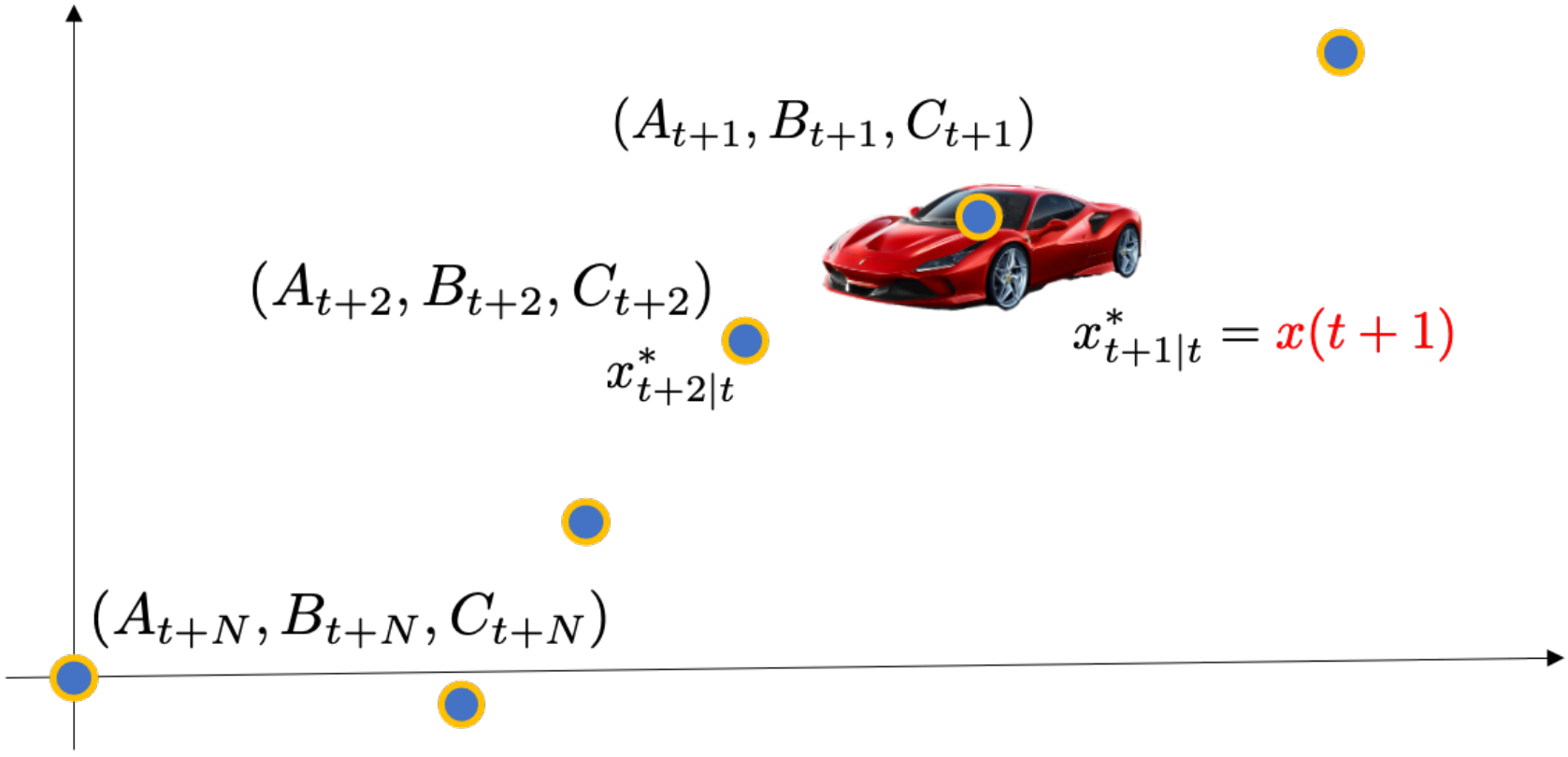
Apply $u^*_{t|t}$.

Nonlinear MPC



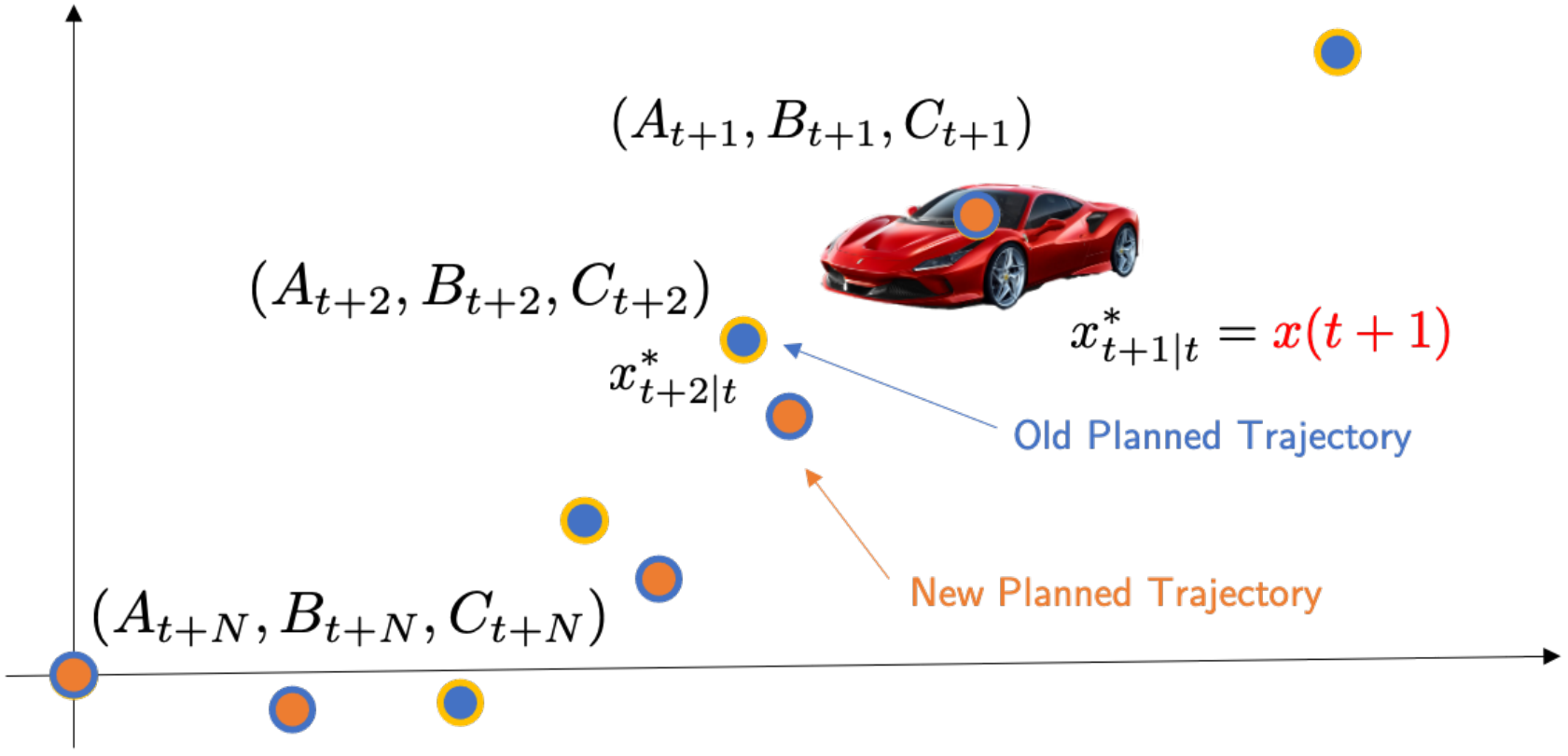
Then $\{x^*_{t+1|t}, \ldots, x^*_{t+N|t}, x_g\}$ and $\{u^*_{t+1|t}, \ldots, u^*_{t+N-1|t}, 0\}$ are feasible state-input sequences for the FTOCP at $t+1$.

Nonlinear MPC



We can linearize around the feasible trajectory! (e.g., $A_k = A(x^*_{k|t}, x^*_{k|t})$, $B_k = B(x^*_{k|t}, x^*_{k|t})$ and $C_k = C(x^*_{k|t}, x^*_{k|t})$).

Nonlinear MPC



What is next?

Safety and Convergence:

In the next lectures we will see how to

- Compute the terminal components to guarantee safety and convergence (hard computationally)
- Learn the terminal components form data