CS159 Lecture 4: Learning MPC

Ugo Rosolia

Caltech

Spring 2021

Today's Class: Learning Model Predictive Control (LMPC)

Goal

Design a policy iteration algorithm:

Today's Class: Learning Model Predictive Control (LMPC)

Goal

Design a policy iteration algorithm:

- Discuss requirements for terminal components.

Today's Class: Learning Model Predictive Control (LMPC)

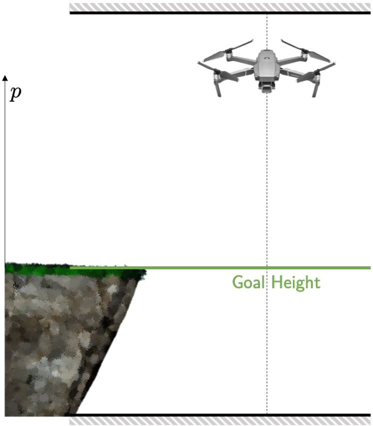
Goal

Design a policy iteration algorithm:

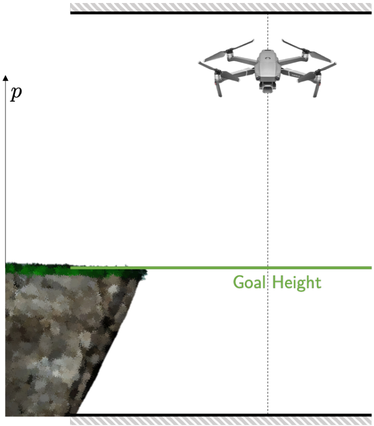
- Discuss requirements for terminal components.
- Learning MPC: Construct terminal components from data.

Table of Contents

Recap of Lecture #3



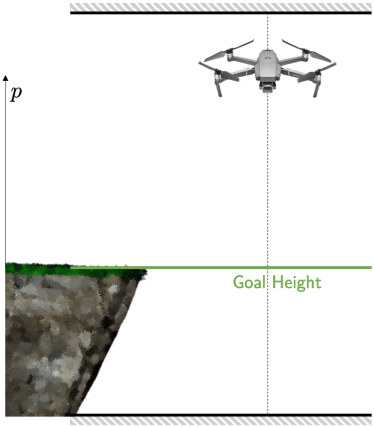
Recap of Lecture #3



- State

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

Recap of Lecture #3

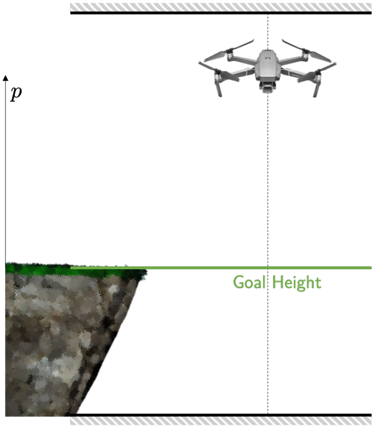


- State

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

- Input $u = a =$ acceleration

Recap of Lecture #3



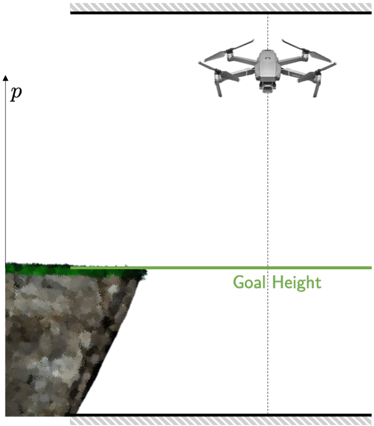
- State

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

- Input $u = a =$ acceleration
- Dynamics

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

Recap of Lecture #3



- State

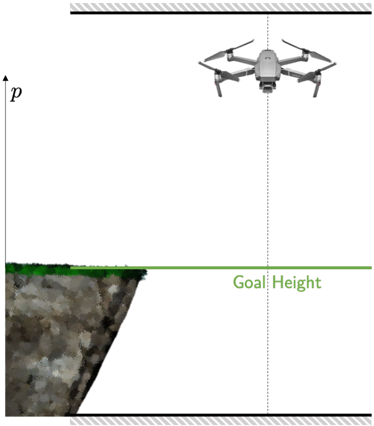
$$x = \begin{bmatrix} p \\ v \end{bmatrix} = \begin{bmatrix} \text{position} \\ \text{velocity} \end{bmatrix}$$

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

Recap of Lecture #3



- State

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

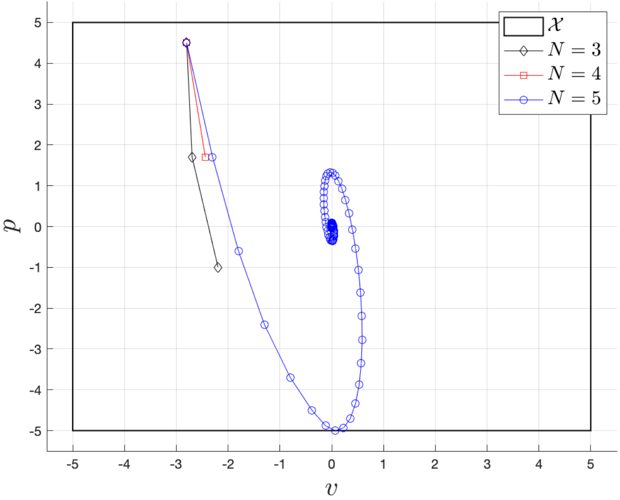
- Input $u = a = \text{acceleration}$
- Dynamics

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

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

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

Recap of Lecture #3



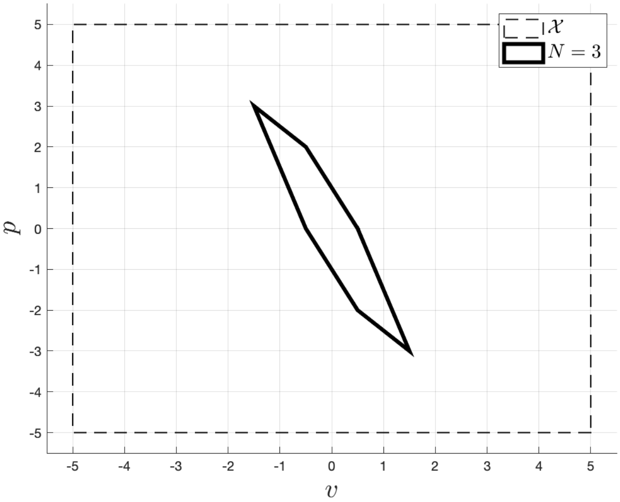
The MPC problem is not feasible at time step $t = 3$ when $N = 3$.
The MPC problem is not feasible at time step $t = 1$ when $N = 4$.

Recap of Lecture #3

The solution was to set $\mathcal{X}_F = \{0\}$.

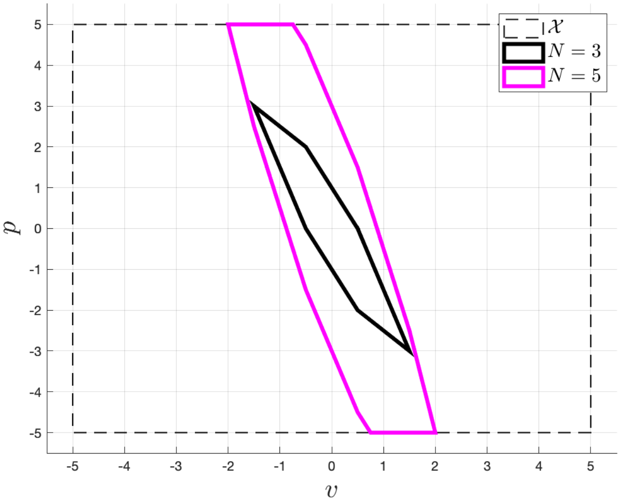
Recap of Lecture #3

The solution was to set $\mathcal{X}_F = \{0\}$.



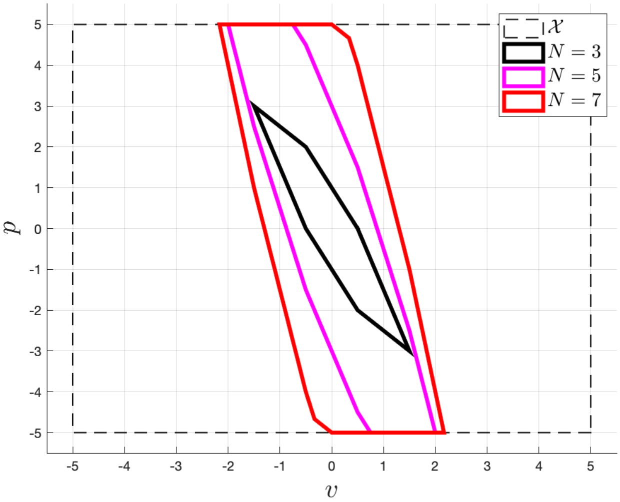
Recap of Lecture #3

The solution was to set $\mathcal{X}_F = \{0\}$.



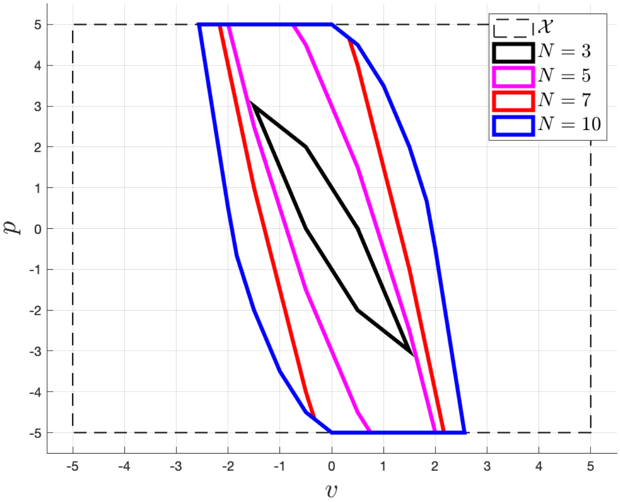
Recap of Lecture #3

The solution was to set $\mathcal{X}_F = \{0\}$.



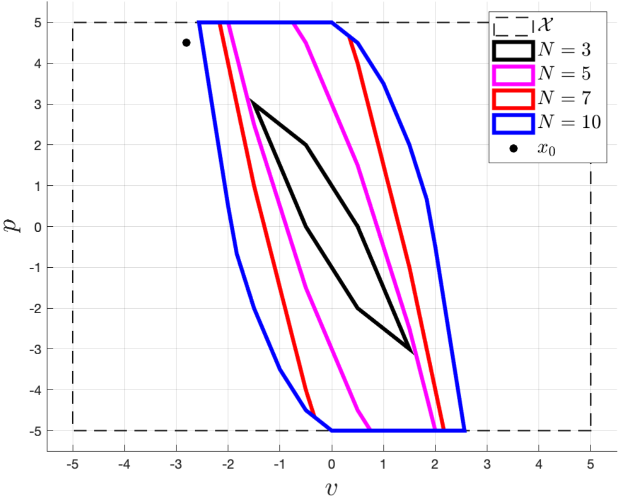
Recap of Lecture #3

The solution was to set $\mathcal{X}_F = \{0\}$.



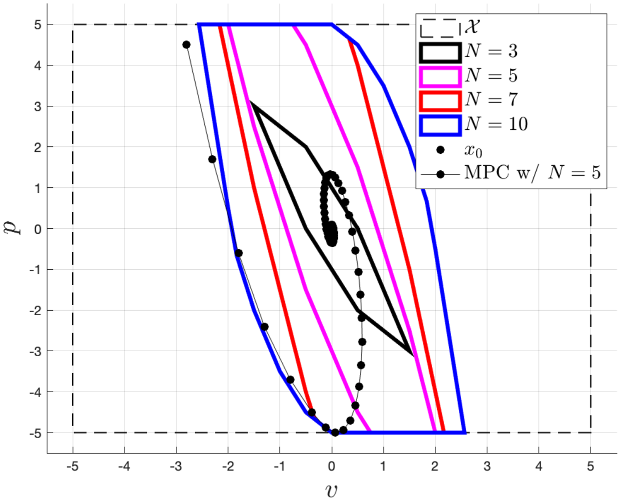
Have we solved the problem?

The solution was to set $\mathcal{X}_F = \{0\}$.

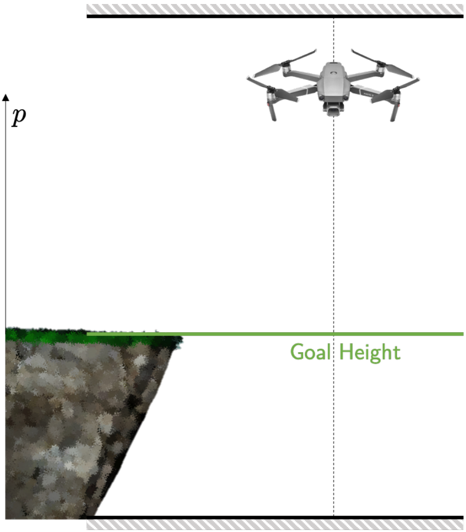


Have we solved the problem?

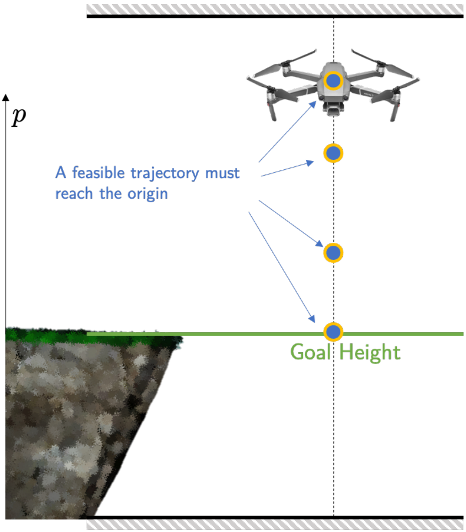
The solution was to set $\mathcal{X}_F = \{0\}$.



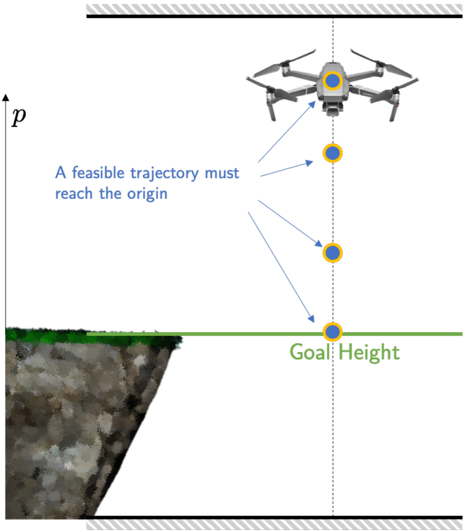
Drone Regulation Problem



Drone Regulation Problem

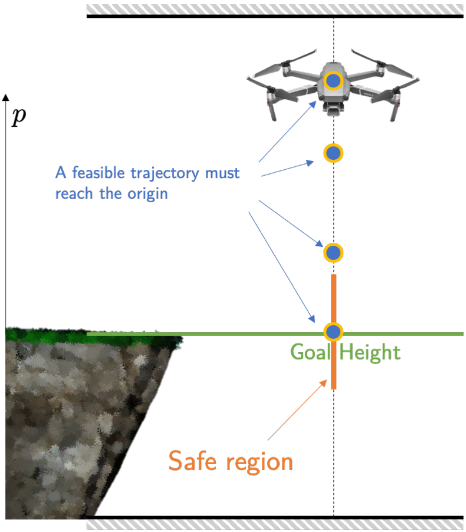


Drone Regulation Problem



Can we use as terminal constraint set a safe set?

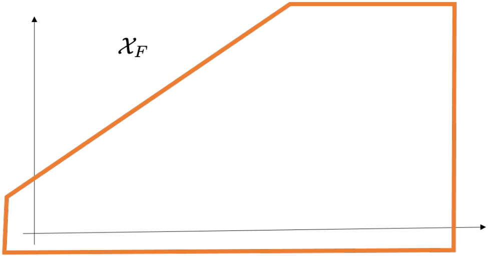
Drone Regulation Problem



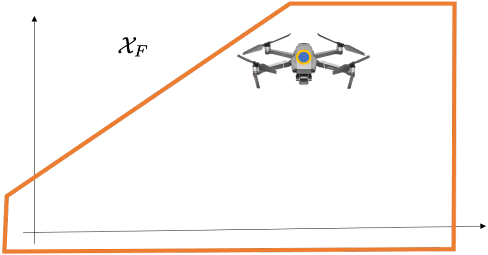
Can we use as terminal constraint set a safe set?

Table of Contents

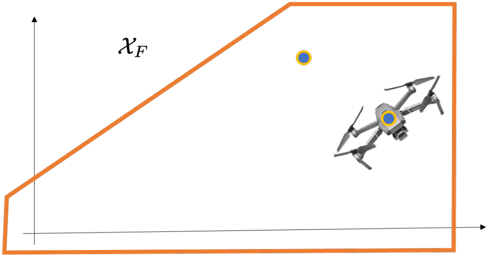
Safe Sets



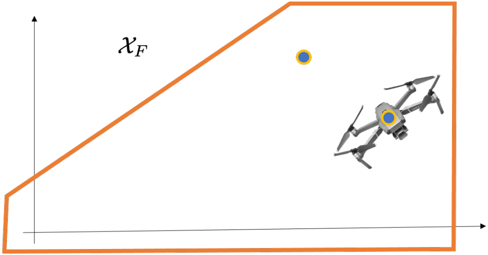
Safe Sets



Safe Sets



Safe Sets



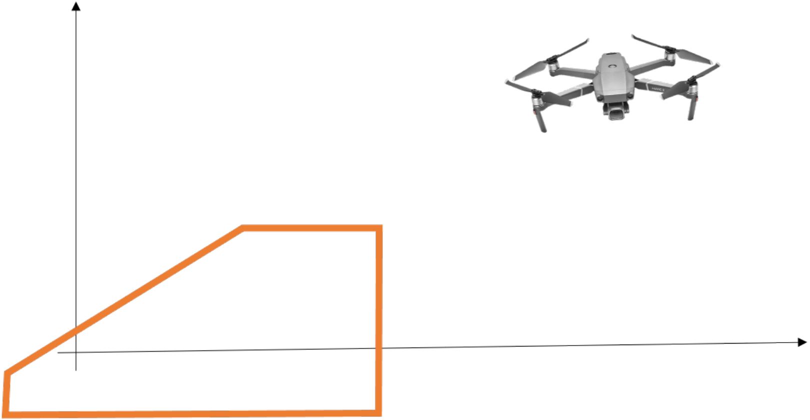
Control Invariant

A set $\mathcal{X}_F$ is control invariant for a system $x_{k+1} = f(x_k, u_k)$, if

$$\forall x \in \mathcal{X}_F,\ \exists u \in \mathcal{U} \text{ such that } f(x, u) \in \mathcal{X}_F.$$

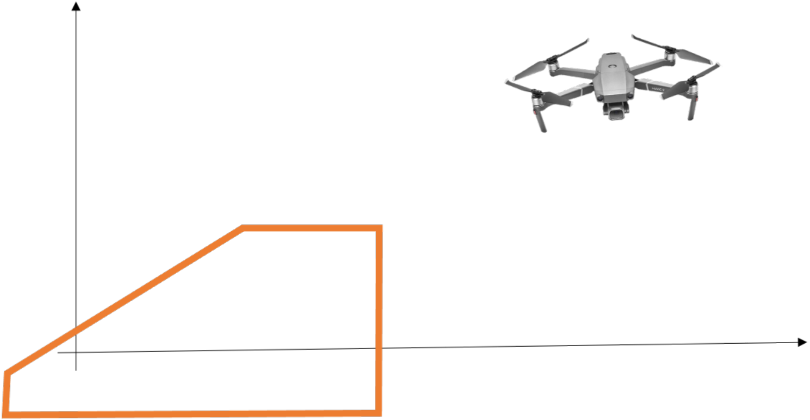
Recursive Feasibility

Let the terminal set $\mathcal{X}_F$ be a control invariant.



Recursive Feasibility

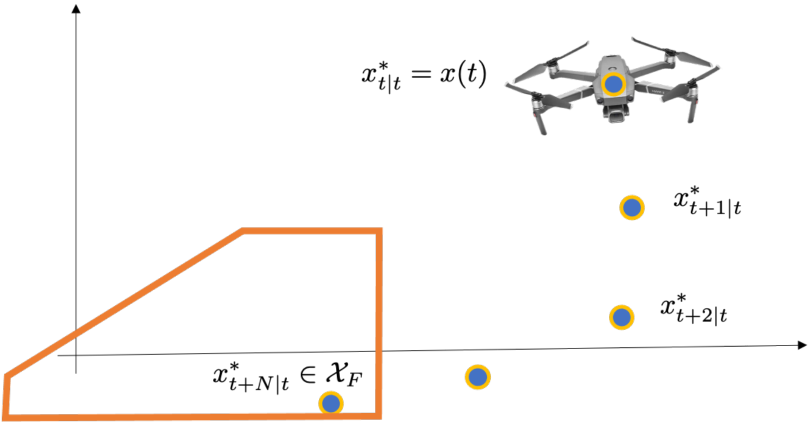
Let the terminal set $\mathcal{X}_F$ be a control invariant.



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

Recursive Feasibility

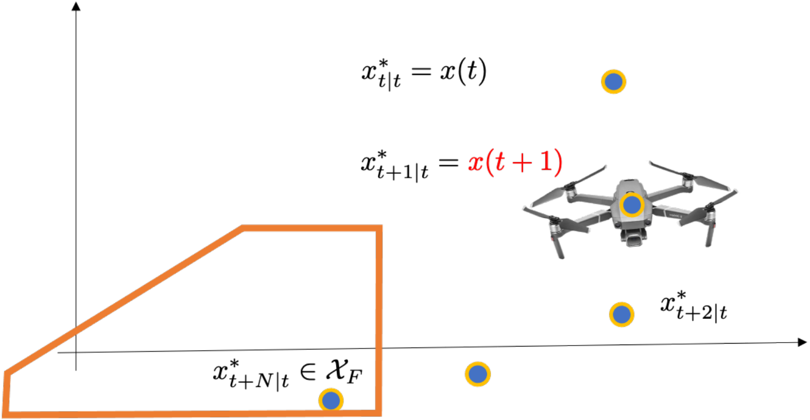
Let the terminal set $\mathcal{X}_F$ be a control invariant.



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

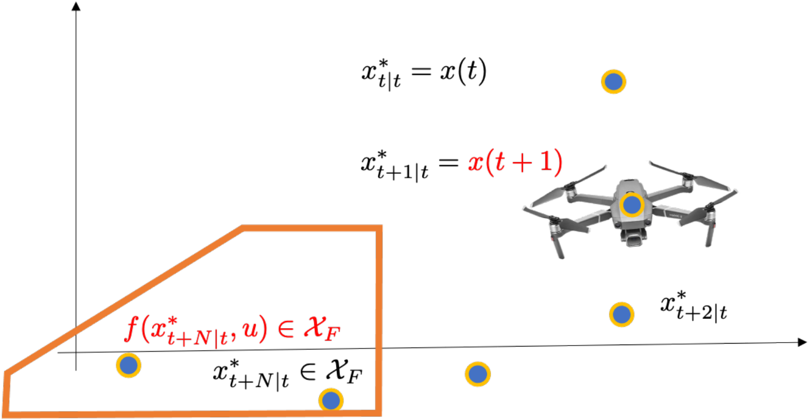
Let the terminal set $\mathcal{X}_F$ be a control invariant.



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

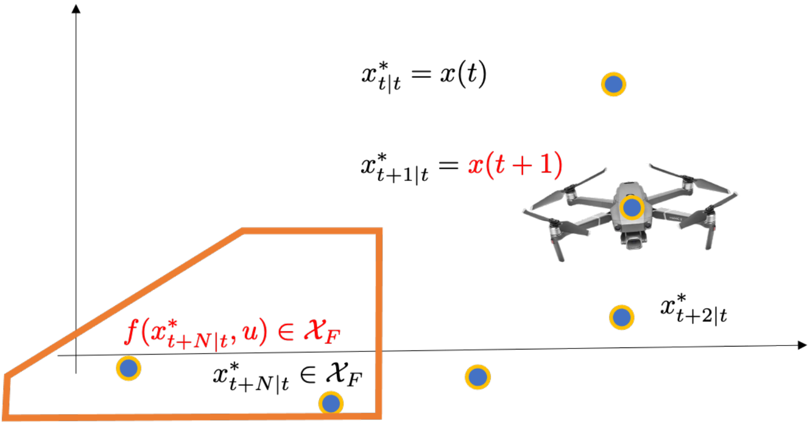
Recursive Feasibility

Let the terminal set $\mathcal{X}_F$ be a control invariant.



Recursive Feasibility

Let the terminal set $\mathcal{X}_F$ be a control invariant.

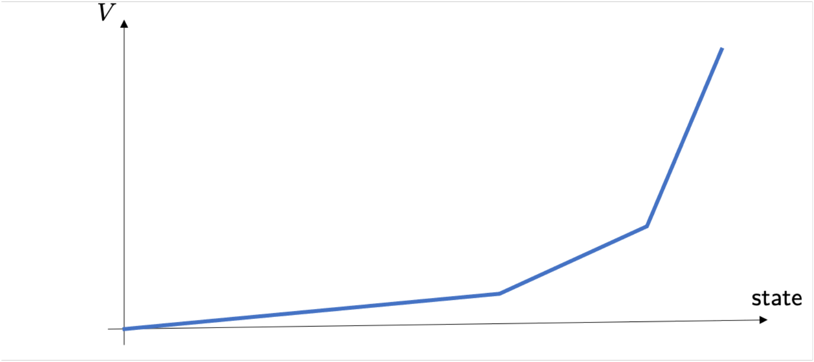


At $t+1$, the state sequence $\{x^*_{t+1|t}, \dots, x^*_{t+N|t}, f(x^*_{t+N|t}, u)\}$ and input sequence $\{u^*_{t+1|t}, \dots, u^*_{t+N-1|t}, u\}$ are feasible.

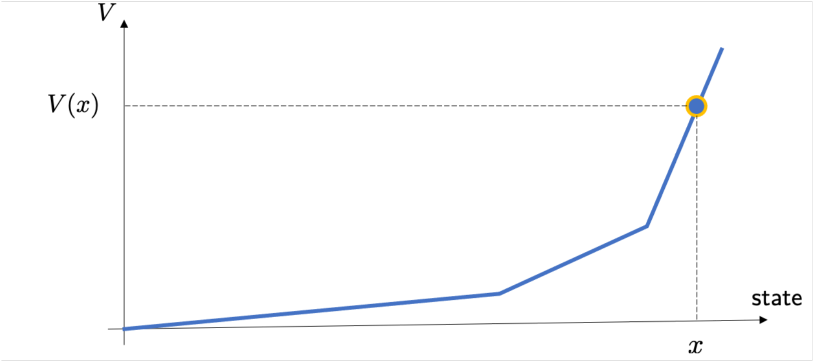
Table of Contents

Value Function Approximation

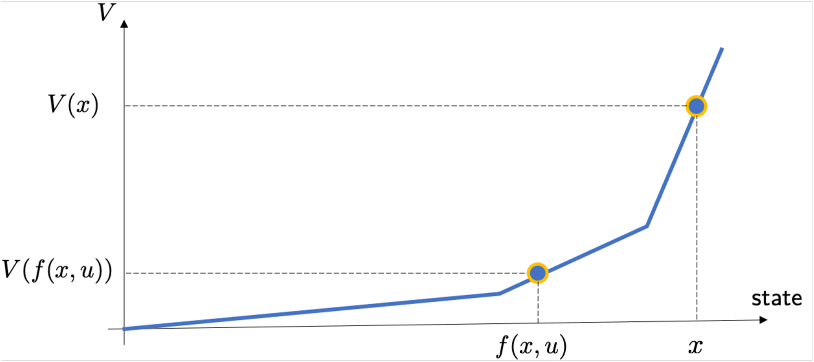
Value Function Approximation



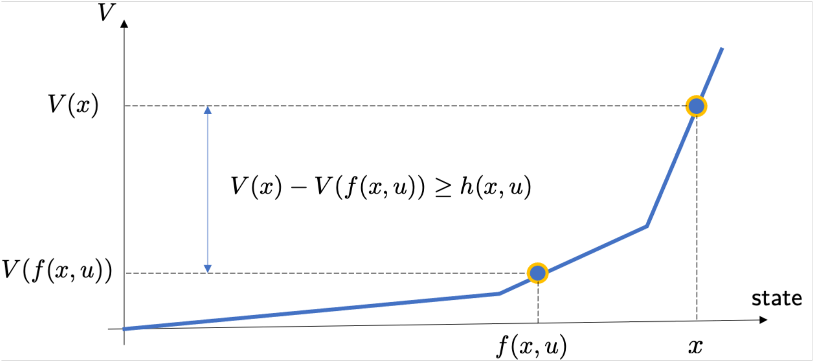
Value Function Approximation



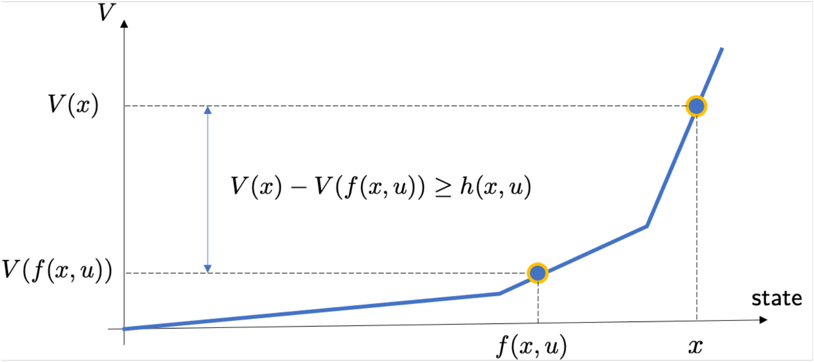
Value Function Approximation



Value Function Approximation



Value Function Approximation



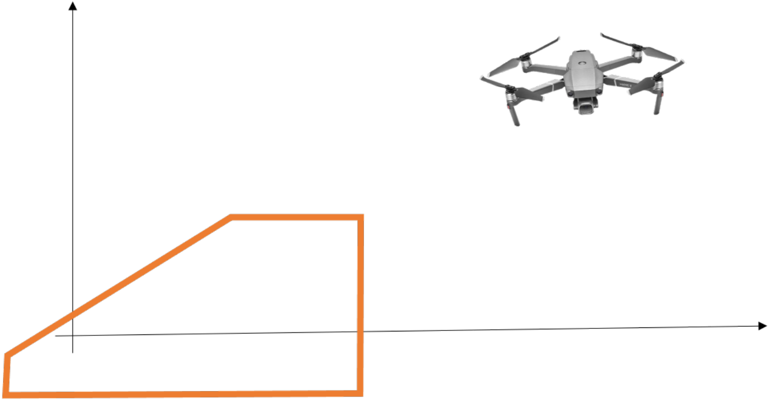
Control Lyapunov Function

A function $V : \mathcal{X}_F \to \mathbb{R}$ is control Lyapunov function for the control invariant set $\mathcal{X}_F$, if $\forall x \in \mathcal{X}_F$

$$\exists u \in \mathcal{U} \text{ such that } V(x) \geq h(x,u) + V(f(x,u)) \text{ and } f(x,u) \in \mathcal{X}_F.$$

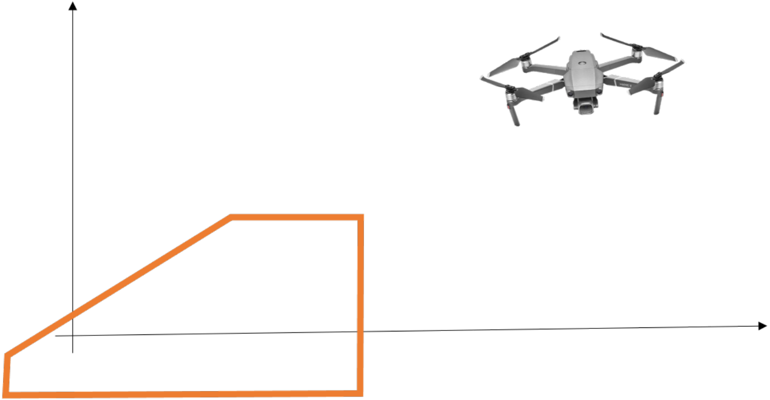
Stability

Assume that $V(x) \geq h(x, u) + V(f(x, u))$, $h(x, u) > 0, \forall x \neq x_g$, $h(x_g, 0) = 0$ and $x_g = f(x_g, 0)$.



Stability

Assume that $V(x) \geq h(x,u) + V(f(x,u))$, $h(x,u) > 0, \forall x \neq x_g$, $h(x_g, 0) = 0$ and $x_g = f(x_g, 0)$.

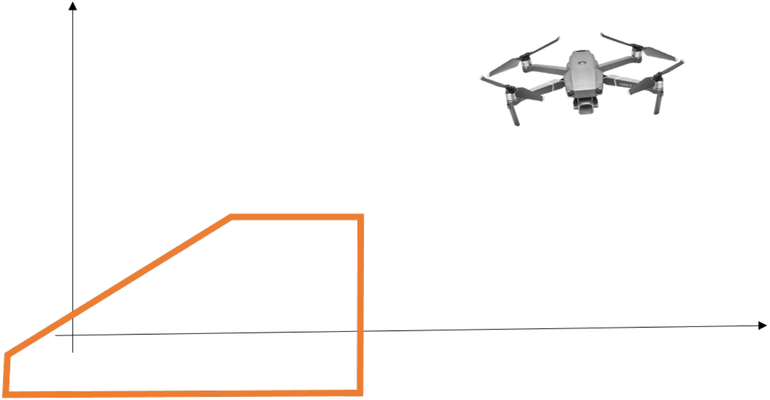


Key idea: show that $\lim_{t \to \infty} J_t^*(x_t) = 0$

Important: A formal proof of stability is done using Lyapunov theory. In the following, we only show that $J_t^*(x_t) = 0$.

Stability

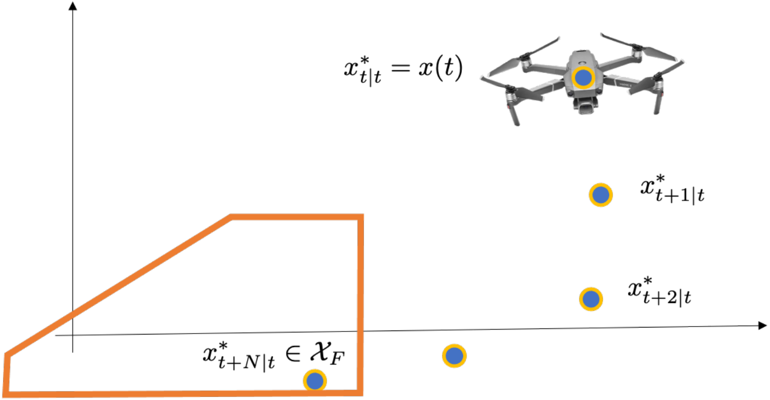
Assume that $V(x) \geq h(x,u) + V(f(x,u))$, $h(x,u) > 0, \forall x \neq x_g$, $h(x_g, 0) = 0$ and $x_g = f(x_g, 0)$.



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

Stability

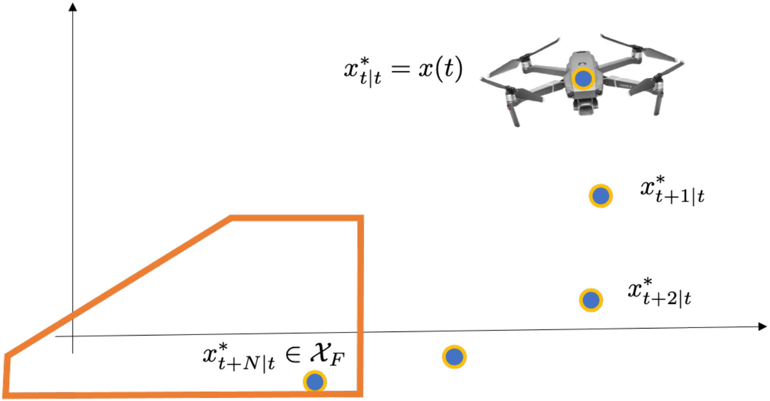
Assume that $V(x) \geq h(x,u) + V(f(x,u))$, $h(x,u) > 0, \forall x \neq x_g$, $h(x_g, 0) = 0$ and $x_g = f(x_g, 0)$.



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.

Stability

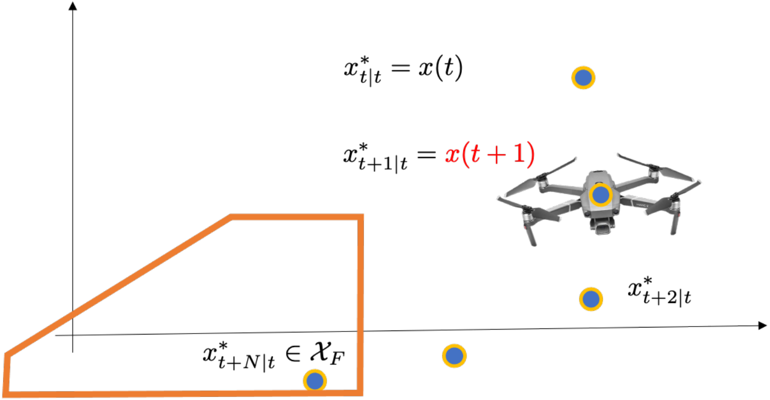
Assume that $V(x) \geq h(x,u) + V(f(x,u))$, $h(x,u) > 0, \forall x \neq x_g$, $h(x_g, 0) = 0$ and $x_g = f(x_g, 0)$.



The cost is $J^*_t(x_t) = \sum_{k=t}^{t+N-1} h(x^*_{k|t}, u^*_{k,t}) + V(x^*_{t+N|t})$

Stability

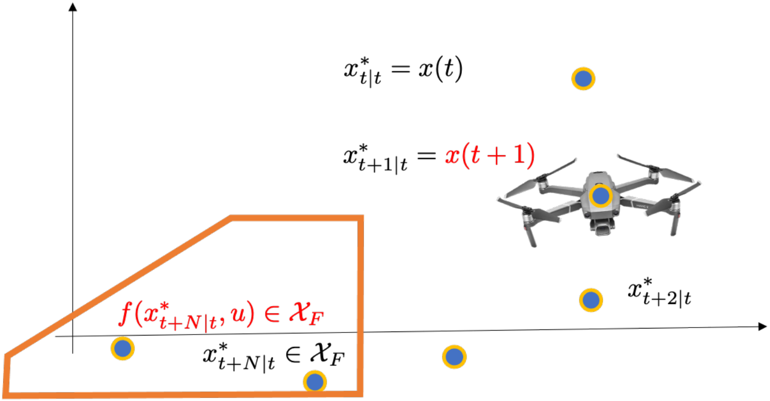
Assume that $V(x) \geq h(x,u) + V(f(x,u))$, $h(x,u) > 0, \forall x \neq x_g$, $h(x_g, 0) = 0$ and $x_g = f(x_g, 0)$.



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

Stability

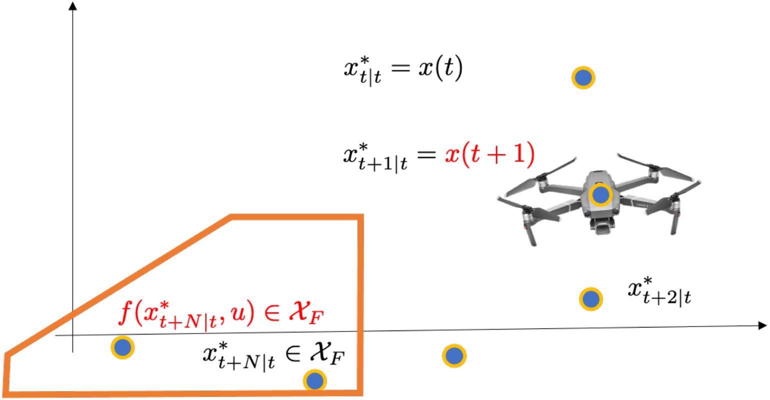
Assume that $V(x) \geq h(x,u) + V(f(x,u))$, $h(x,u) > 0, \forall x \neq x_g$, $h(x_g, 0) = 0$ and $x_g = f(x_g, 0)$.



At $t+1$, the state sequence $\{x^*_{t+1|t}, \ldots, x^*_{t+N|t}, f(x^*_{t+N|t}, u)\}$ and input sequence $\{u^*_{t+1|t}, \ldots, u^*_{t+N-1|t}, u\}$ are feasible.

Stability

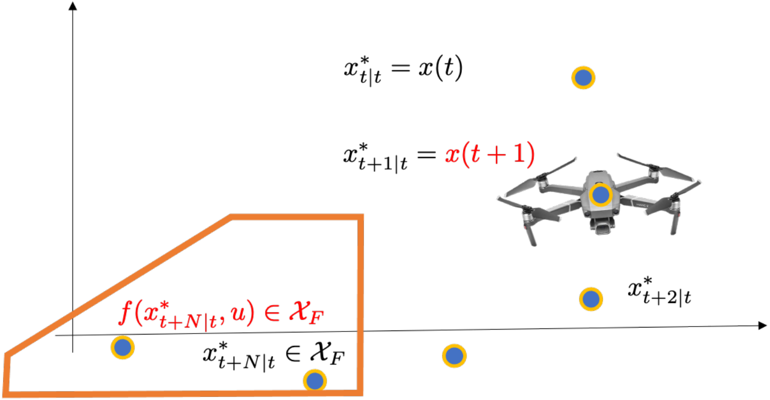
Assume that $V(x) \geq h(x,u) + V(f(x,u))$, $h(x,u) > 0, \forall x \neq x_g$, $h(x_g, 0) = 0$ and $x_g = f(x_g, 0)$.



Notice that $J_t^*(x_t) = \sum_{k=t}^{t+N-1} h(x^*_{k|t}, u^*_{k,t}) + V(x^*_{t+N|t})$

Stability

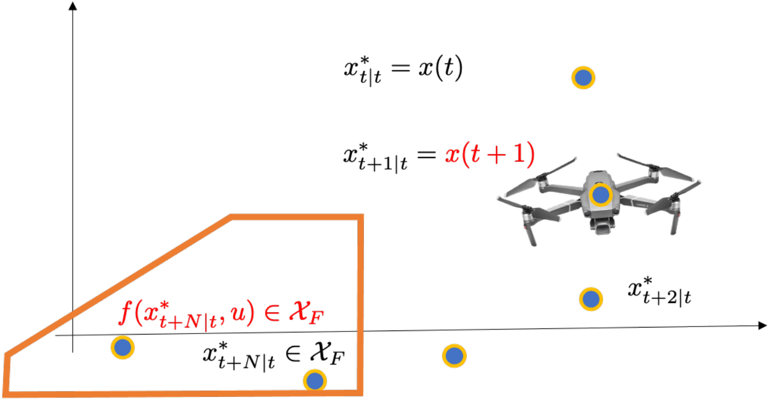
Assume that $V(x) \geq h(x,u) + V(f(x,u))$, $h(x,u) > 0, \forall x \neq x_g$, $h(x_g, 0) = 0$ and $x_g = f(x_g, 0)$.



Notice that $J^*_t(x_t) = \sum_{k=t}^{t+N-1} h(x^*_{k|t}, u^*_{k,t}) + V(x^*_{t+N|t}) \geq h(x^*_{t|t}, u^*_{t,t}) + \sum_{k=t+1}^{t+N-1} h(x^*_{k|t}, u^*_{k,t}) + h(x^*_{t+N|t}, u) + V(f(x^*_{t+N|t}, u))$

Stability

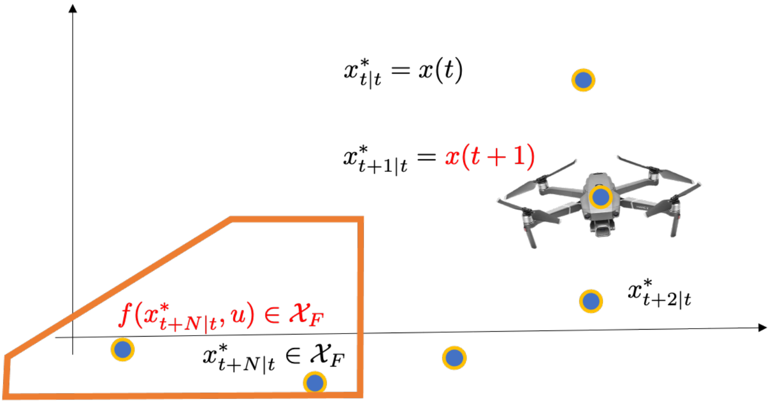
Assume that $V(x) \geq h(x,u) + V(f(x,u))$, $h(x,u) > 0, \forall x \neq x_g$, $h(x_g, 0) = 0$ and $x_g = f(x_g, 0)$.



At $t+1$, $J^*_t(x_t) \geq h(x^*_{t|t}, u^*_{t,t}) + J^*_{t+1}(x_{t+1})$.

Stability

Assume that $V(x) \geq h(x,u) + V(f(x,u))$, $h(x,u) > 0, \forall x \neq x_g$, $h(x_g, 0) = 0$ and $x_g = f(x_g, 0)$.



At $t+1$, $J^*_t(x_t) \geq h(x^*_{t|t}, u^*_{t,t}) + J^*_{t+1}(x_{t+1})$.
Therefore, $J^*_{t+1}(x_{t+1}) < J^*_t(x_t)$ for all $x \neq x_g$.

Summary

A solution: We have shown when the terminal set $\mathcal{X}_F$ is a control invariant and the terminal cost $V(x)$ is an approximation to the value function:

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

Summary

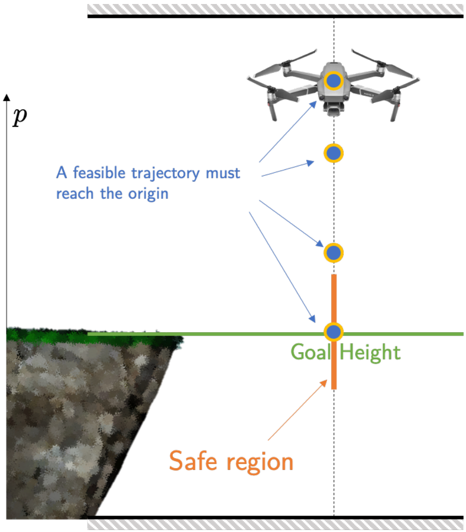
A solution: We have shown when the terminal set $\mathcal{X}_F$ is a control invariant and the terminal cost $V(x)$ is an approximation to the value function:

- 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: Computing the terminal components is computationally expensive, even for deterministic linear constrained dynamical systems.

Table of Contents

Drone Regulation Problem



Can we use as terminal constraint set a safe set?

Drone Regulation Problem

Consider the following finite time optimal control problem:

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

where $h(x, u) = x^\top Qx + u^\top Ru$.

Design Rules

Assumption:

Design Rules

Assumption:

1. We are given an control invariant $\mathcal{O}_\infty = \{x \in \mathbb{R}^n \mid F_f x \leq b_f\}$ for the LQR policy $\pi^{\mathrm{LQR}}(x)$, i.e.,

$$\forall x \in \mathcal{O}_\infty \text{ we have that } Kx \in \mathcal{U}, (A + BK)x \in \mathcal{O}_\infty.$$

Design Rules

Assumption:

1. We are given an control invariant $\mathcal{O}_\infty = \{x \in \mathbb{R}^n \mid F_f x \leq b_f\}$ for the LQR policy $\pi^{\mathrm{LQR}}(x)$, i.e.,

$$\forall x \in \mathcal{O}_\infty \text{ we have that } Kx \in \mathcal{U}, (A + BK)x \in \mathcal{O}_\infty.$$

2. We are given the matrix P which can be used to compute the value function associated with the LQR policy $\pi^{\mathrm{LQR}}(x)$, i.e.,

$$V(x) = x^\top P x$$

(P is computed solving the discrete time Riccati equation)

Design Rules

Assumption:

1. We are given an control invariant $\mathcal{O}_\infty = \{x \in \mathbb{R}^n \mid F_f x \leq b_f\}$ for the LQR policy $\pi^{\mathrm{LQR}}(x)$, i.e.,

$$\forall x \in \mathcal{O}_\infty \text{ we have that } Kx \in \mathcal{U}, (A+BK)x \in \mathcal{O}_\infty.$$

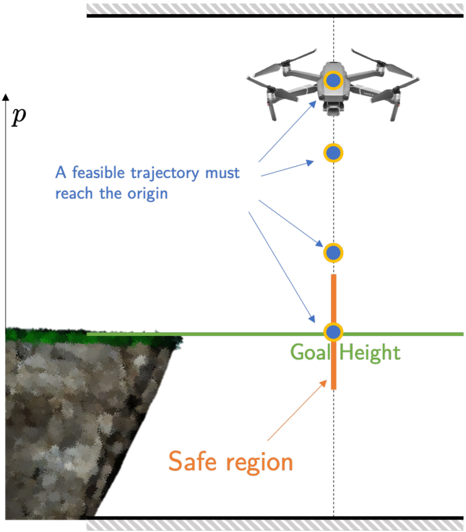
2. We are given the matrix P which can be used to compute the value function associated with the LQR policy $\pi^{\mathrm{LQR}}(x)$, i.e.,

$$V(x) = x^\top P x$$

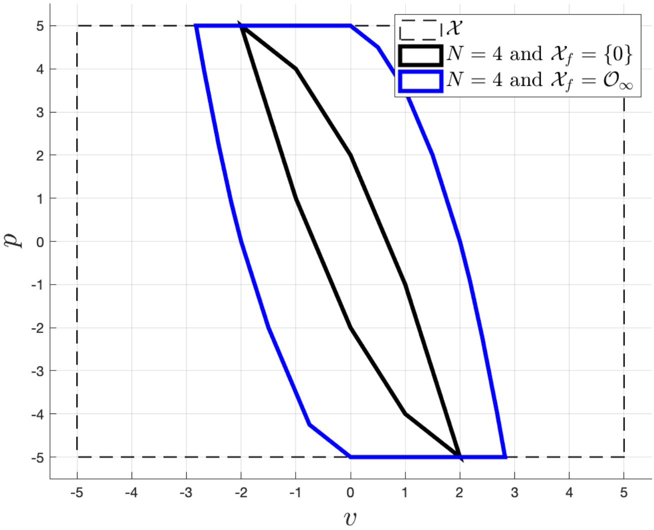
(P is computed solving the discrete time Riccati equation)

Result: Using $\mathcal{O}_\infty$ as terminal constraint and $V(x)$ as terminal cost guarantees recursive feasibility and stability.

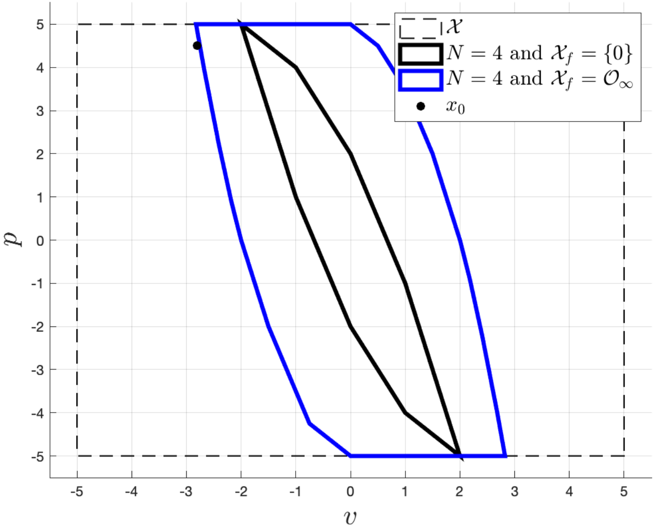
Drone Regulation Problem



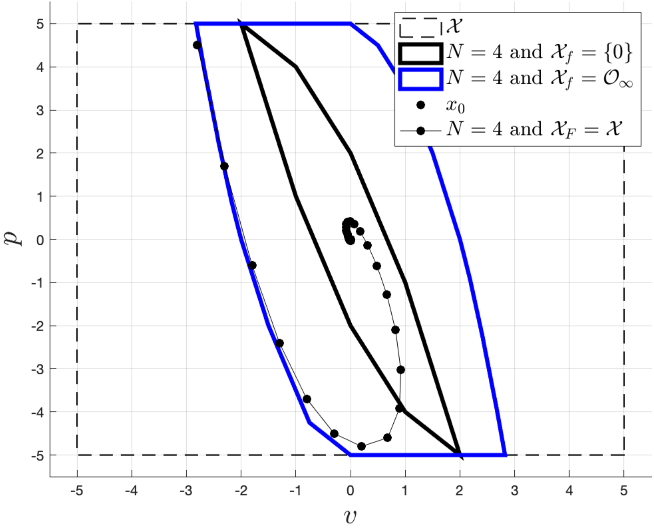
Drone Regulation Problem – Region of Attraction



Drone Regulation Problem – Region of Attraction



Drone Regulation Problem – Region of Attraction



The MPC is designed setting $Q_F = 10^4$.

Table of Contents

Estimating Terminal Components from Data

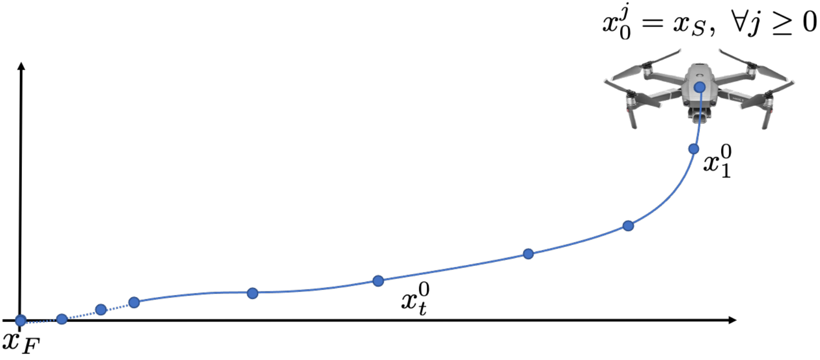
In several applications robots are doing the same or similar tasks. Can we learn **safe regions** and **value function approximations** from data?

Iterative Tasks

Iteratively drive the drone to a goal state x_F from an initial state x_S.

Iterative Tasks

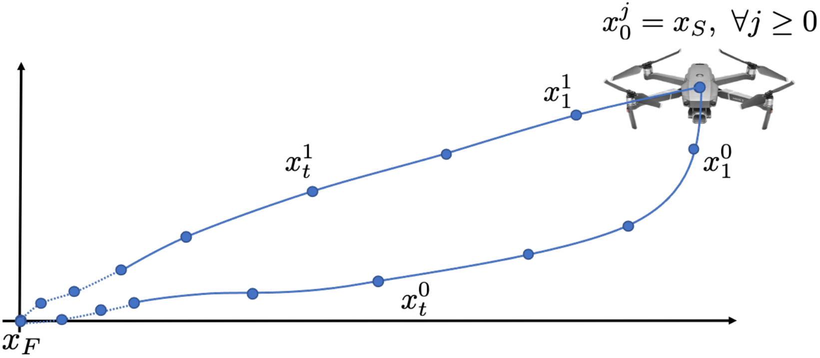
Iteratively drive the drone to a goal state x_F from an initial state x_S.



Roll-out = one execution of the control task.

Iterative Tasks

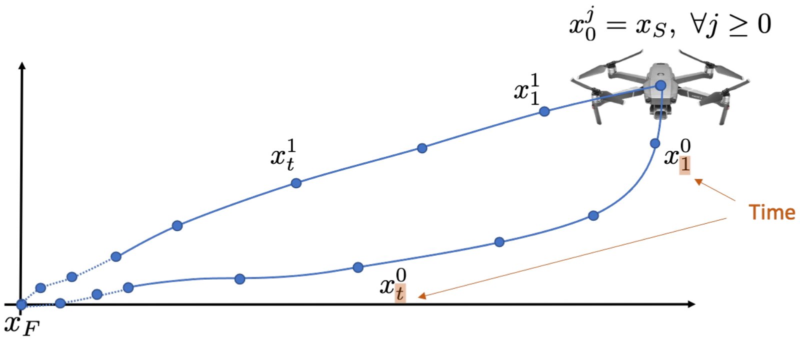
Iteratively drive the drone to a goal state x_F from an initial state x_S.



Roll-out = one execution of the control task.

Iterative Tasks

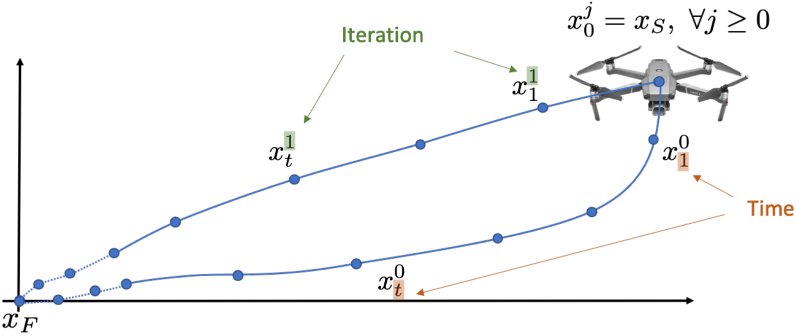
Iteratively drive the drone to a goal state x_F from an initial state x_S.



Roll-out = one execution of the control task.

Iterative Tasks

Iteratively drive the drone to a goal state x_F from an initial state x_S.

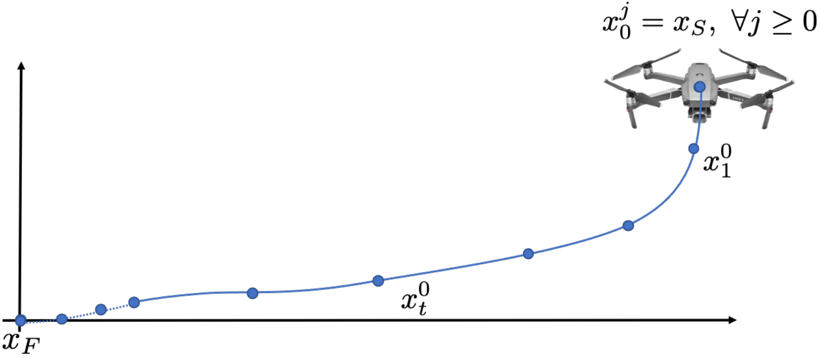


Roll-out = one execution of the control task.

Table of Contents

Safe Set

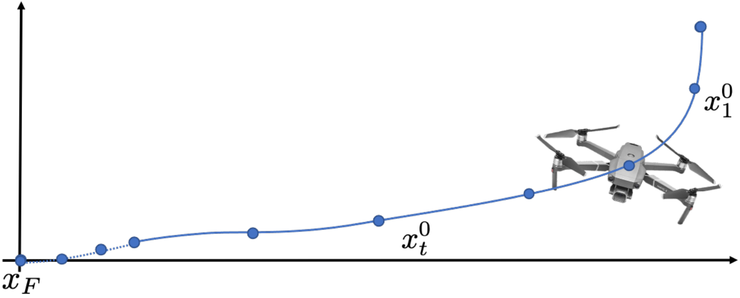
Assume a demonstration is given.



Safe Set

Assume a demonstration is given.

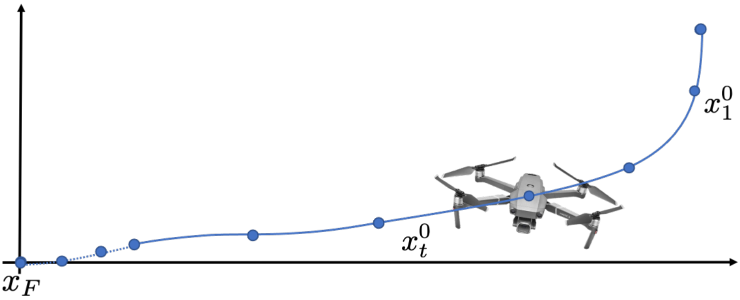
$$x_0^j = x_S, \ \forall j \geq 0$$



Safe Set

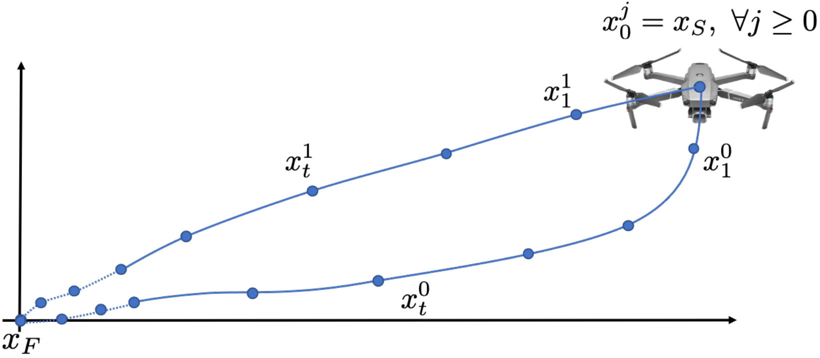
Assume a demonstration is given.

$$x_0^j = x_S, \ \forall j \geq 0$$



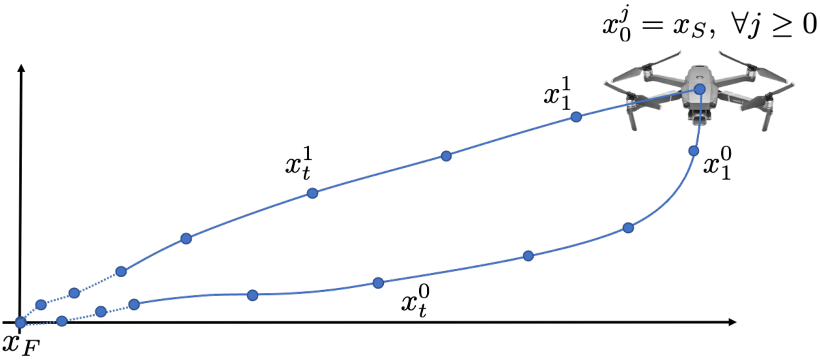
Safe Set

Assume j demonstrations are given.



Safe Set

Assume j demonstrations are given.



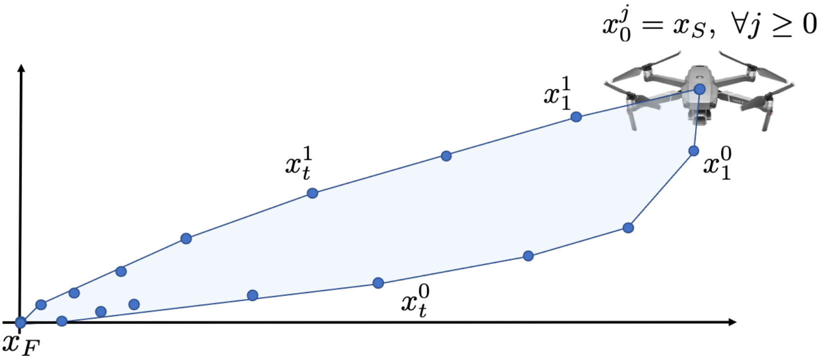
Safe Set for j roll-outs

Define the sampled safe set as

$$\mathcal{SS}^j = \text{set of stored data} = \cup_{i=0}^{j} \cup_{t=0}^{\infty} x_t^j$$

Safe Set

Assume j demonstrations are given.



Convex Safe Set for j roll-outs

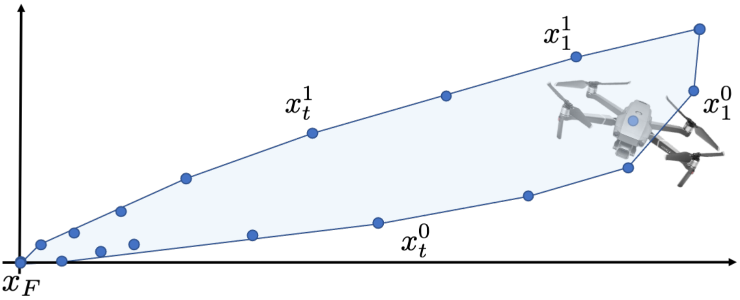
Define the sampled safe set as

$$\mathcal{CS}^j = \mathrm{conv}(\text{set of stored data}) = \mathrm{conv}(\cup_{i=0}^{j} \cup_{t=0}^{\infty} x_t^j)$$

Safe Set

Assume j demonstrations are given.

$$x_0^j = x_S, \ \forall j \geq 0$$



Convex Safe Set for j roll-outs

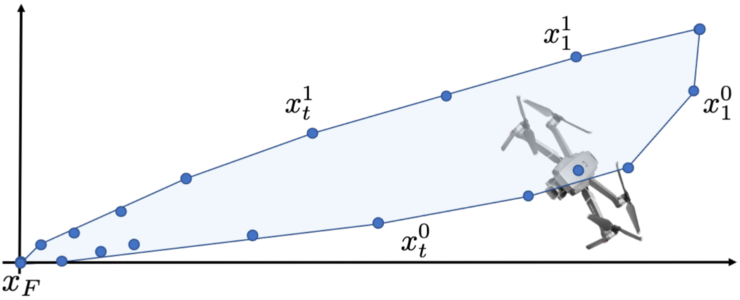
Define the sampled safe set as

$$\mathcal{CS}^j = \mathrm{conv}(\text{set of stored data}) = \mathrm{conv}(\cup_{i=0}^{j} \cup_{t=0}^{\infty} x_t^j)$$

Safe Set

Assume j demonstrations are given.

$$x_0^j = x_S, \ \forall j \geq 0$$



Concex Safe Set for j roll-outs

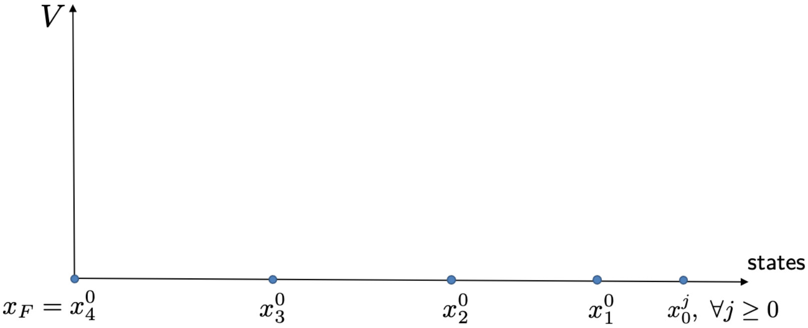
Define the sampled safe set as

$$\mathcal{CS}^j = \text{conv}(\text{set of stored data}) = \text{conv}(\cup_{i=0}^{j} \cup_{t=0}^{\infty} x_t^j)$$

Table of Contents

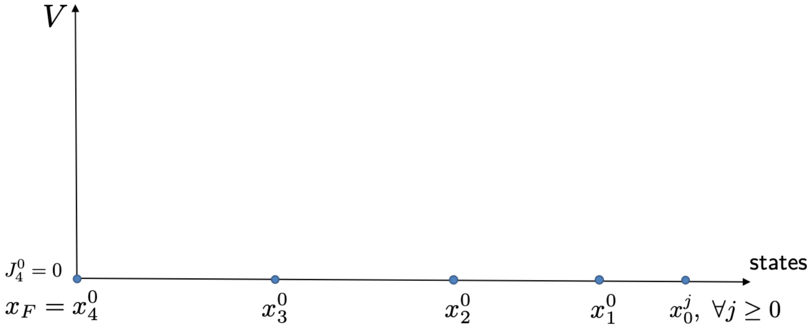
Value Function Approximation

Assume a demonstration is given.



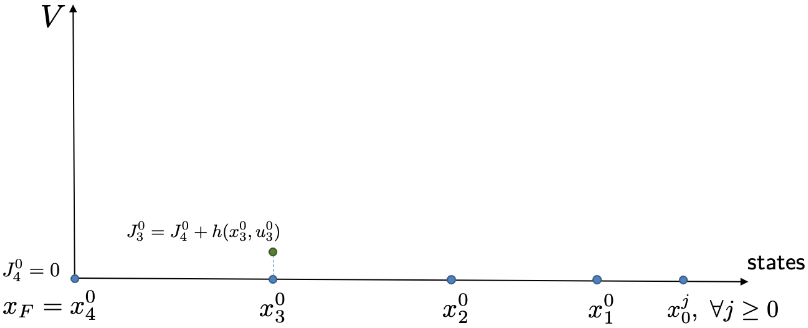
Value Function Approximation

Assume a demonstration is given.



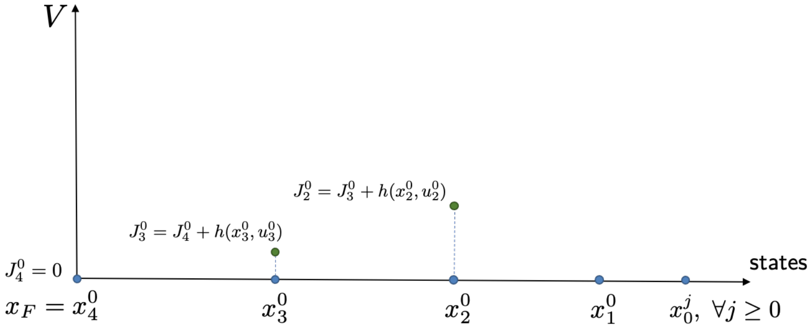
Value Function Approximation

Assume a demonstration is given.



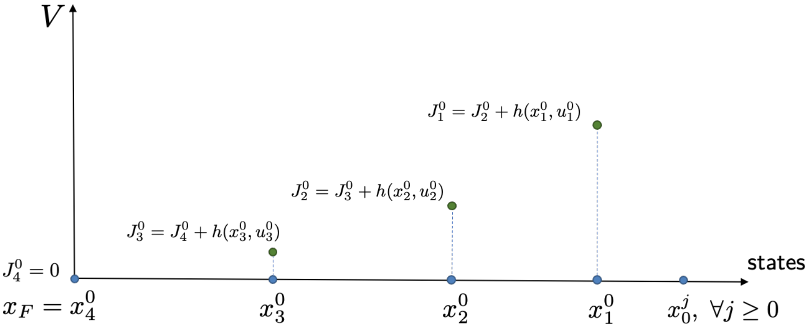
Value Function Approximation

Assume a demonstration is given.



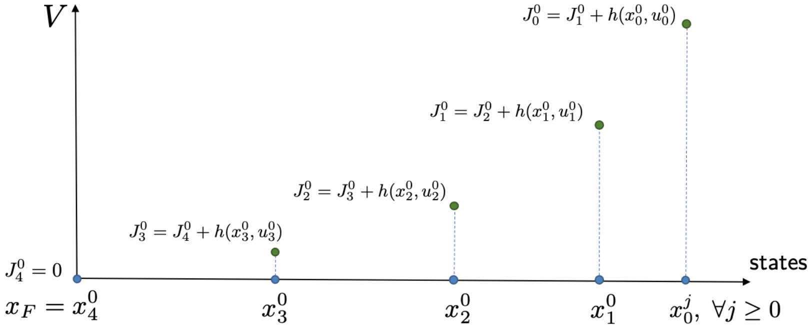
Value Function Approximation

Assume a demonstration is given.



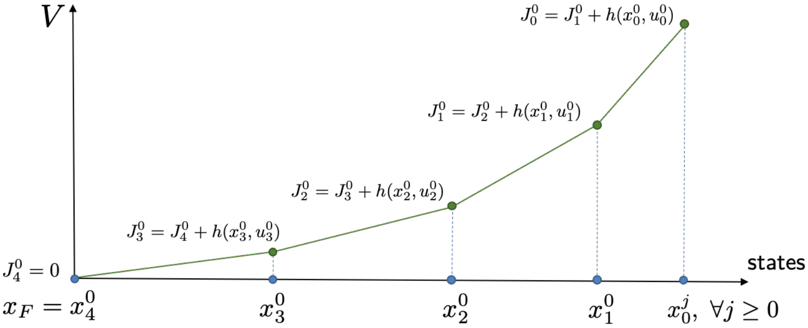
Value Function Approximation

Assume a demonstration is given.



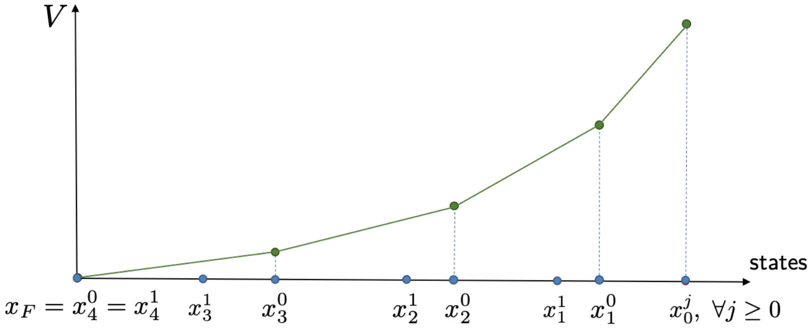
Value Function Approximation

Assume j demonstrations are given.



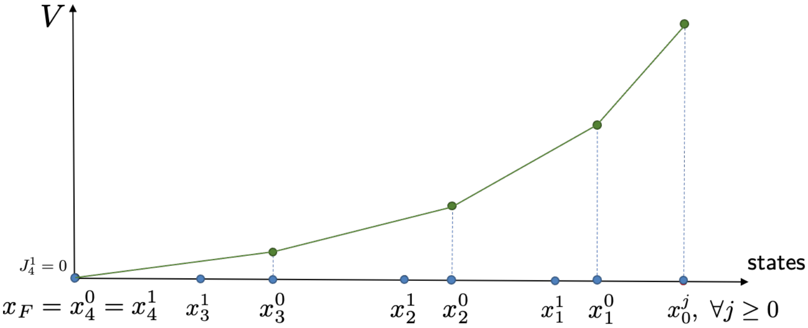
Value Function Approximation

Assume j demonstrations are given.



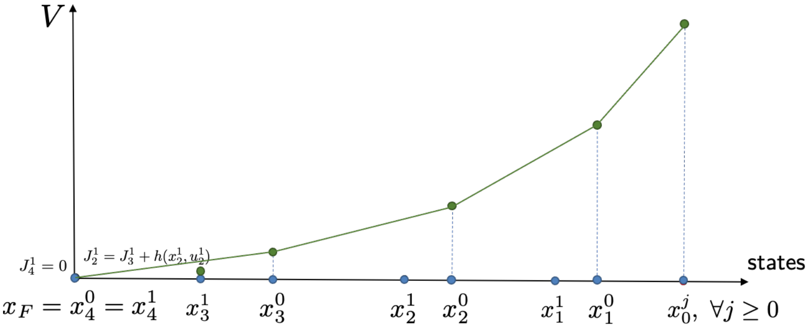
Value Function Approximation

Assume j demonstrations are given.



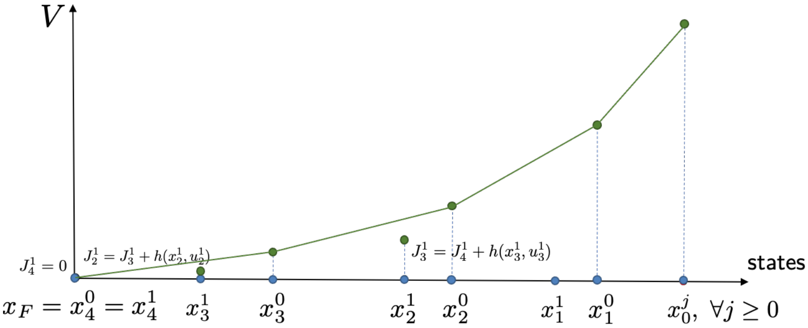
Value Function Approximation

Assume j demonstrations are given.



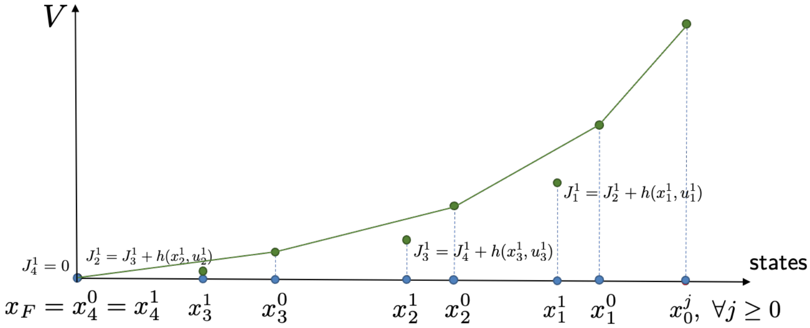
Value Function Approximation

Assume j demonstrations are given.



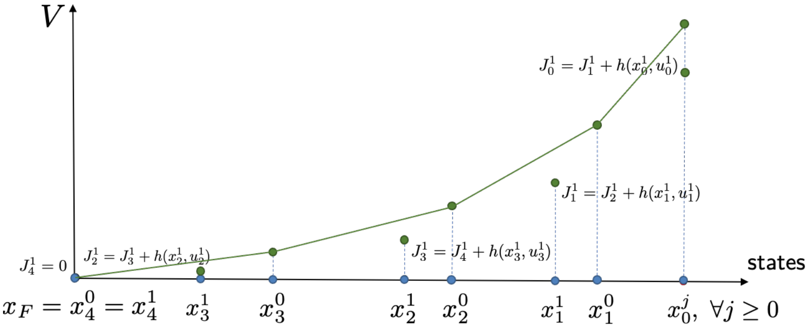
Value Function Approximation

Assume j demonstrations are given.



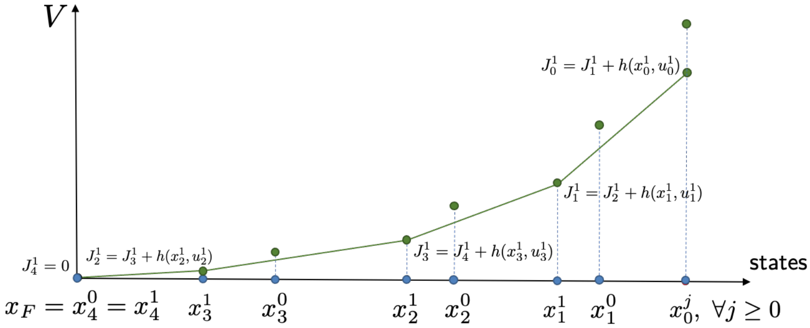
Value Function Approximation

Assume j demonstrations are given.



Value Function Approximation

Assume j demonstrations are given.



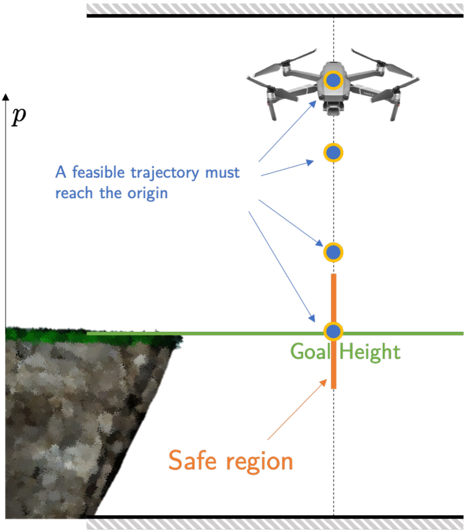
Value Function Approximation for j roll-outs

$$V^j(\textcolor{red}{x}) = \min_{\lambda_t^i \geq 0} \quad \sum_{i=0}^{j} \sum_{t=0}^{\infty} J_t^i \lambda_t^i$$

$$\text{subject to} \quad \sum_{i=0}^{j} \sum_{t=0}^{\infty} x_t^i \lambda_t^i = \textcolor{red}{x}, \sum_{i=0}^{j} \sum_{t=0}^{\infty} \lambda_t^i = 1$$

Table of Contents

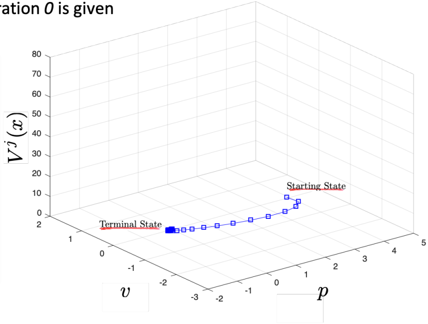
Drone Regulation Problem – A policy iteration strategy



LMPC – A policy iteration strategy

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

→ **Approximation Procedure**



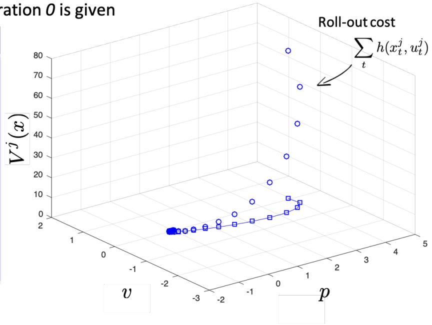
LMPC – A policy iteration strategy

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

Approximation Procedure

Step 0: Set iteration counter *j=0*

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



LMPC – A policy iteration strategy

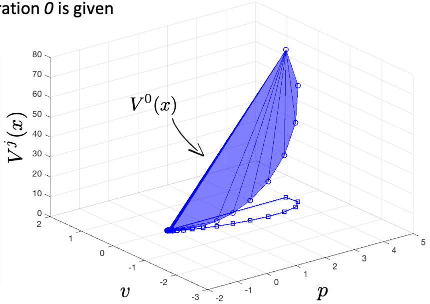
Assumption: A first feasible trajectory at iteration *0* is given

Approximation Procedure

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



LMPC – A policy iteration strategy

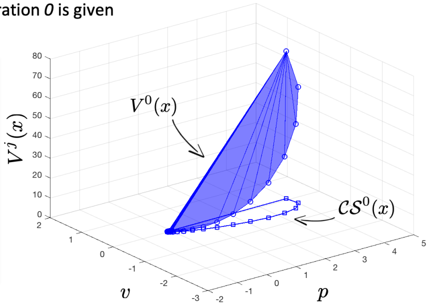
Assumption: A first feasible trajectory at iteration *0* is given

Approximation Procedure

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



LMPC – A policy iteration strategy

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

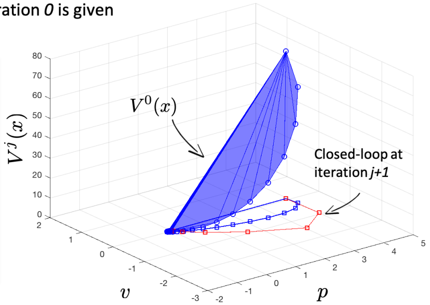
Approximation Procedure

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



LMPC – A policy iteration strategy

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

Approximation Procedure

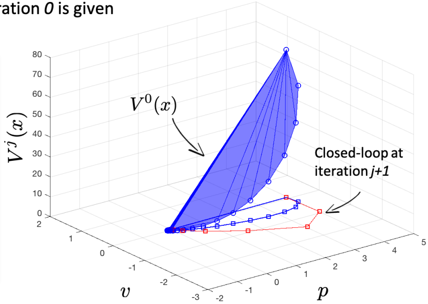
Step 0: Set iteration counter j=0

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

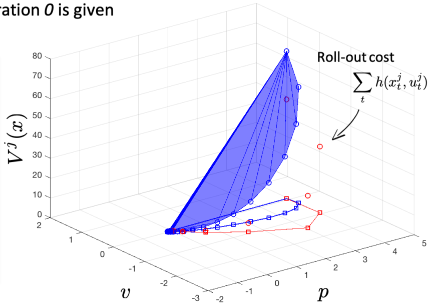


LMPC – A policy iteration strategy

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

Approximation Procedure

- 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



LMPC – A policy iteration strategy

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

Approximation Procedure

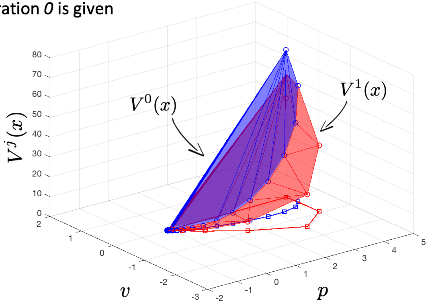
Step 0: Set iteration counter j=*0*

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



LMPC – A policy iteration strategy

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

Approximation Procedure

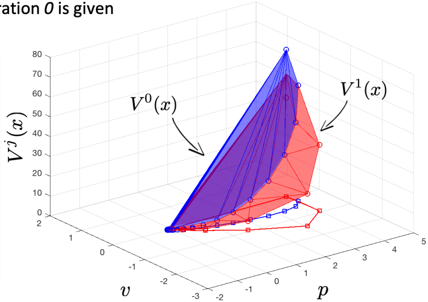
Step 0: Set iteration counter $j=0$

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 value function approximation is defined over a subset of the state space.
The LMPC policy is used to enlarge the region of which the value function approximation is defined.

LMPC – A policy iteration strategy

Algorithm Steps:

1. Set $j = 0$. Select a policy π^j that can complete the task from x_S, run the closed-loop system and store the closed-loop trajectory $\mathbf{x}^j = [x_0^j, x_1^j, \dots]$.

LMPC – A policy iteration strategy

Algorithm Steps:

1. Set $j = 0$. Select a policy π^j that can complete the task from x_S, run the closed-loop system and store the closed-loop trajectory $\mathbf{x}^j = [x_0^j, x_1^j, \ldots]$.
2. **(Policy Evaluation).** Compute the value function approximation V^j and the convex safe set $\mathcal{CS}^j$ using data stored up to iteration j.

LMPC – A policy iteration strategy

Algorithm Steps:

1. Set $j = 0$. Select a policy π^j that can complete the task from x_S, run the closed-loop system and store the closed-loop trajectory $\mathbf{x}^j = [x_0^j, x_1^j, \ldots]$.
2. **(Policy Evaluation).** Compute the value function approximation V^j and the convex safe set $\mathcal{CS}^j$ using data stored up to iteration j.
3. **(Policy Improvement).** Compute the LMPC policy π^{j+1} using the convex safe set $\mathcal{CS}^j$ as terminal constraint and the value function approximation V^j as terminal cost.

LMPC – A policy iteration strategy

Algorithm Steps:

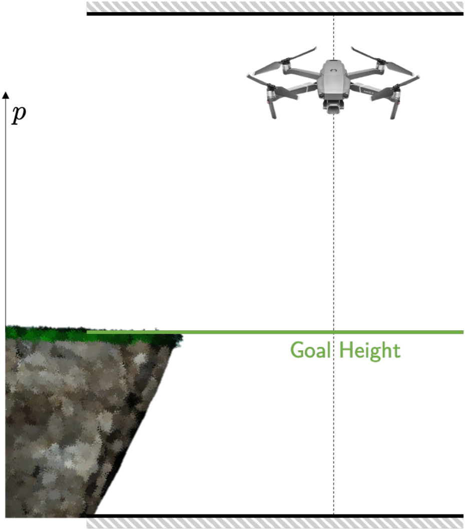
1. Set $j = 0$. Select a policy π^j that can complete the task from x_S, run the closed-loop system and store the closed-loop trajectory $\mathbf{x}^j = [x_0^j, x_1^j, \dots]$.
2. **(Policy Evaluation).** Compute the value function approximation V^j and the convex safe set $\mathcal{CS}^j$ using data stored up to iteration j.
3. **(Policy Improvement).** Compute the LMPC policy π^{j+1} using the convex safe set $\mathcal{CS}^j$ as terminal constraint and the value function approximation V^j as terminal cost.
4. Run the closed-loop system from x_S and store the closed-loop trajectory $\mathbf{x}^{j+1} = [x_0^{j+1}, x_1^{j+1}, \dots]$

LMPC – A policy iteration strategy

Algorithm Steps:

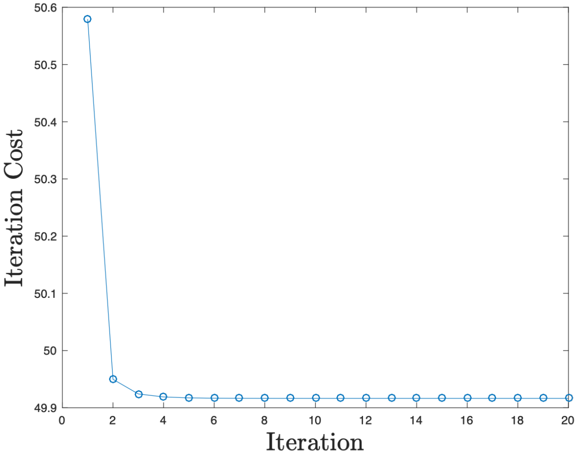
1. Set $j = 0$. Select a policy π^j that can complete the task from x_S, run the closed-loop system and store the closed-loop trajectory $\mathbf{x}^j = [x_0^j, x_1^j, \ldots]$.
2. **(Policy Evaluation).** Compute the value function approximation V^j and the convex safe set $\mathcal{CS}^j$ using data stored up to iteration j.
3. **(Policy Improvement).** Compute the LMPC policy π^{j+1} using the convex safe set $\mathcal{CS}^j$ as terminal constraint and the value function approximation V^j as terminal cost.
4. Run the closed-loop system from x_S and store the closed-loop trajectory $\mathbf{x}^{j+1} = [x_0^{j+1}, x_1^{j+1}, \ldots]$
5. If $\mathbf{x}^{j+1} = \mathbf{x}^j$ stop, $\pi^{\mathrm{LMPC}} = \pi^{j+1}$. Otherwise, set $j = j + 1$ and go to Step 2.

Drone Regulation Problem

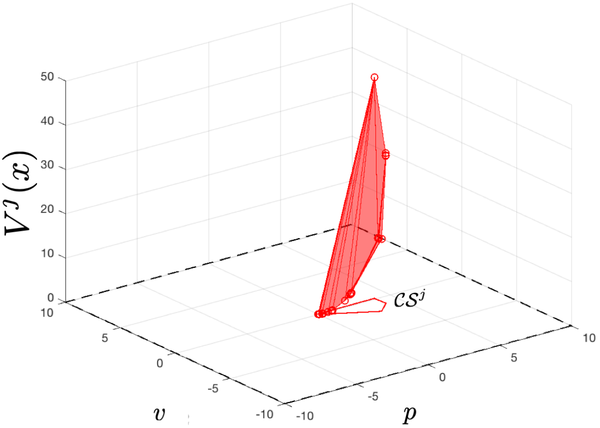


Drone Regulation Problem – Iteration Cost

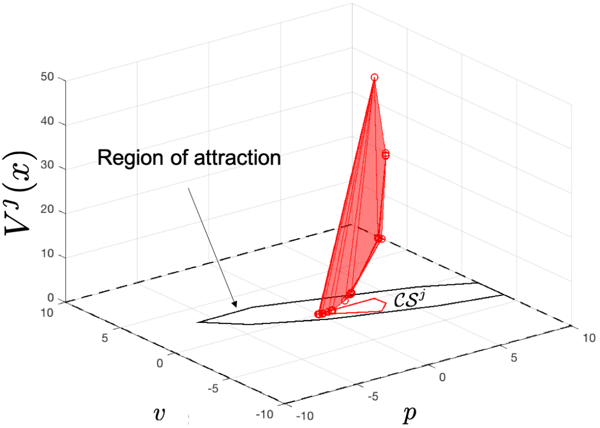
Iteration cost = cost of the roll-out = $\sum_{t=0}^{\infty} h(x_t^j, u_t^j)$



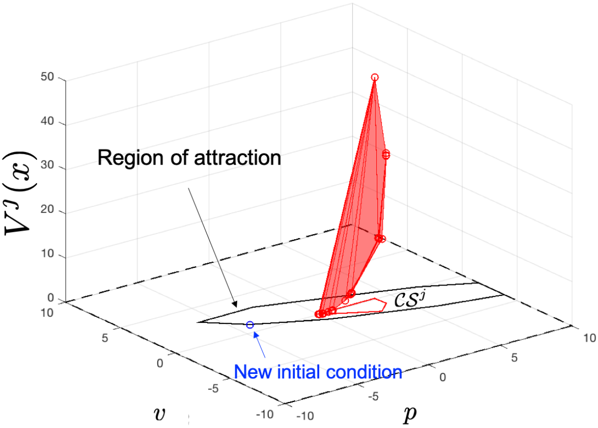
Drone Regulation Problem – Region of Attraction



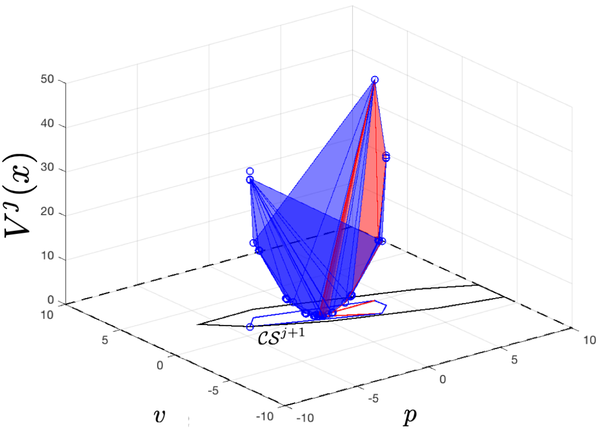
Drone Regulation Problem – Region of Attraction



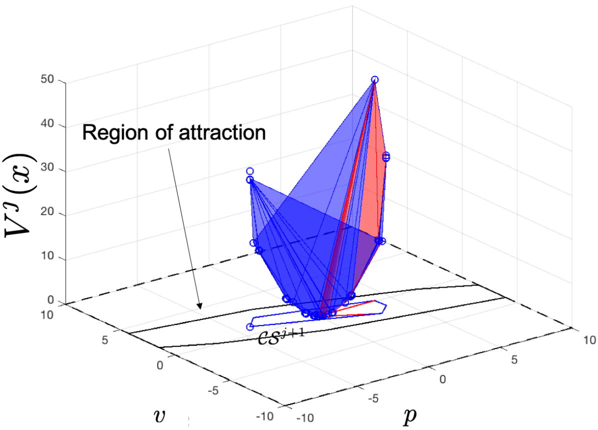
Drone Regulation Problem – Region of Attraction



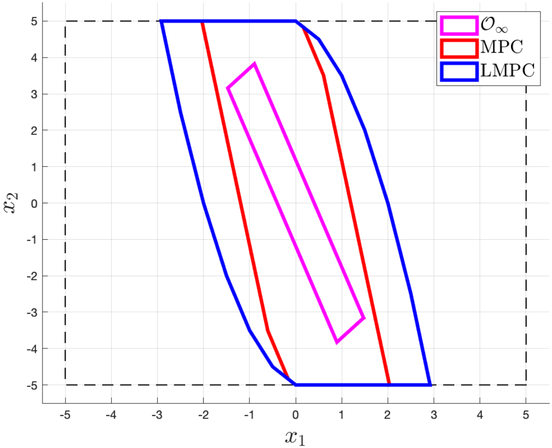
Drone Regulation Problem – Region of Attraction



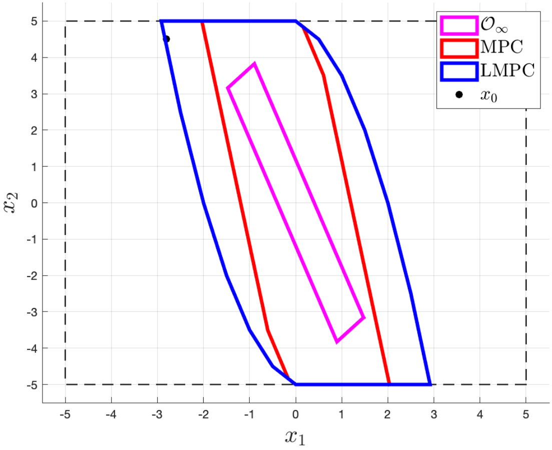
Drone Regulation Problem – Region of Attraction



Drone Regulation Problem – Region of Attraction



Drone Regulation Problem – Region of Attraction



LMPC: Properties

Theorem

Let $\mathbf{x}^j = [x_0^j, x_1^j, \ldots]$ be the closed-loop trajectory from the starting state x_S at iteration j. Consider sequence $\{\mathbf{x}^j\}$ of closed-loop trajectories and assume that for $c < \infty$ we have that

$$\mathbf{x}^c = \mathbf{x}^{c+1}$$

Then we have that

- At each iteration state and input constraints are satisfied.
- The closed-loop cost $J_{0\rightarrow\infty}^j(x_S)$ is non-increasing, i.e.,

$$J_{0\rightarrow\infty}^{j+1}(x_S) = \sum_{t=0}^{\infty} h(x_t^{j+1}, u_t^{j+1}) \leq \sum_{t=0}^{\infty} h(x_t^j, u_t^j) = J_{0\rightarrow\infty}^j(x_S)$$

- $\mathbf{x}^c = \mathbf{x}^*$, under mild conditions (LICQ holds at each time t)