Parallel Explicit
Tube Model Predictive Control

Kai Wang[1], Yuning Jiang[1], Juraj Oravec[2], Mario E. Villanueva[1], Boris Houska[1]

[1]ShanghaiTech University

[2]Slovak University of Technology in Bratislava

Overview

Background on Tube MPC

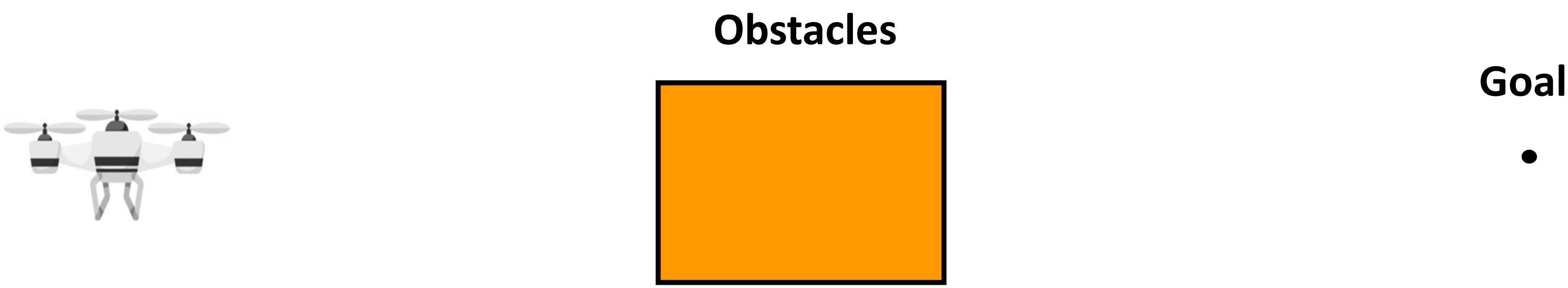
- Model Predictive Control (MPC)
- Robust MPC
- Rigid Tube MPC

Contribution

- Parallel Real-time Optimization Algorithm
- Parallel Explicit Tube MPC Controller
- Numerical Example

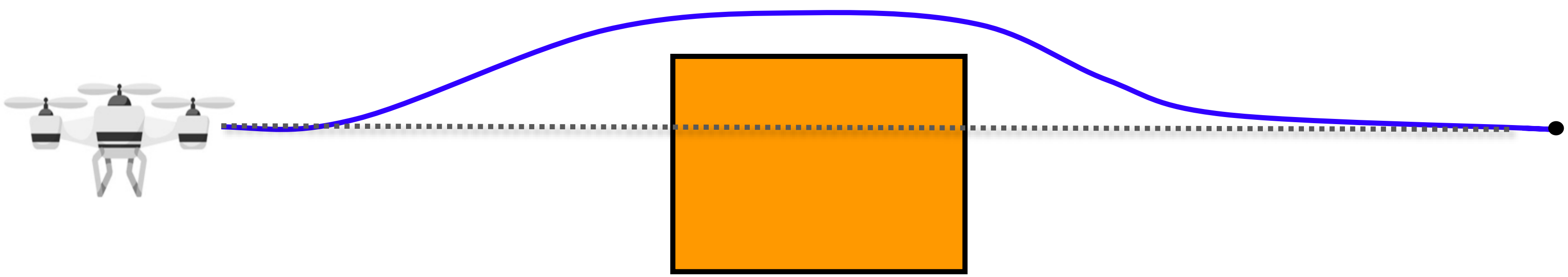
Model Predictive Control (MPC)

Uncertain Control System: $x_{k+1} = f(x_k,\ u_k,\ w_k)$



Model Predictive Control (MPC)

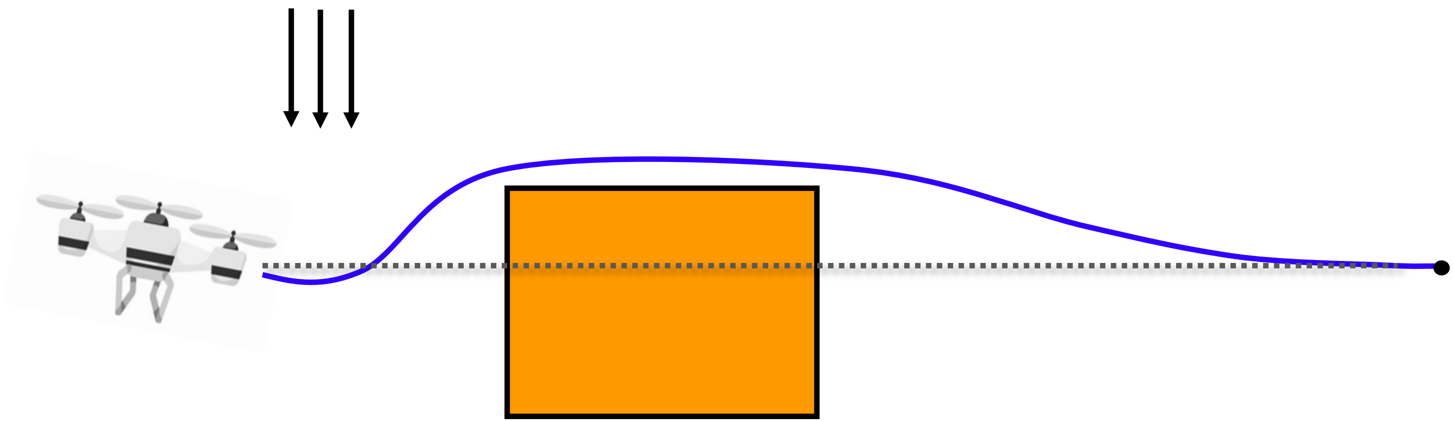
Uncertain Control System: $x_{k+1} = f(x_k,\ u_k,\ w_k)$



Certainty equivalent MPC:

- Minimize distance to the dotted line
- Subject to: System dynamics $x_{k+1} = f(x_k,\, u_k,\, \mathbf{0})$
  State and input constraints

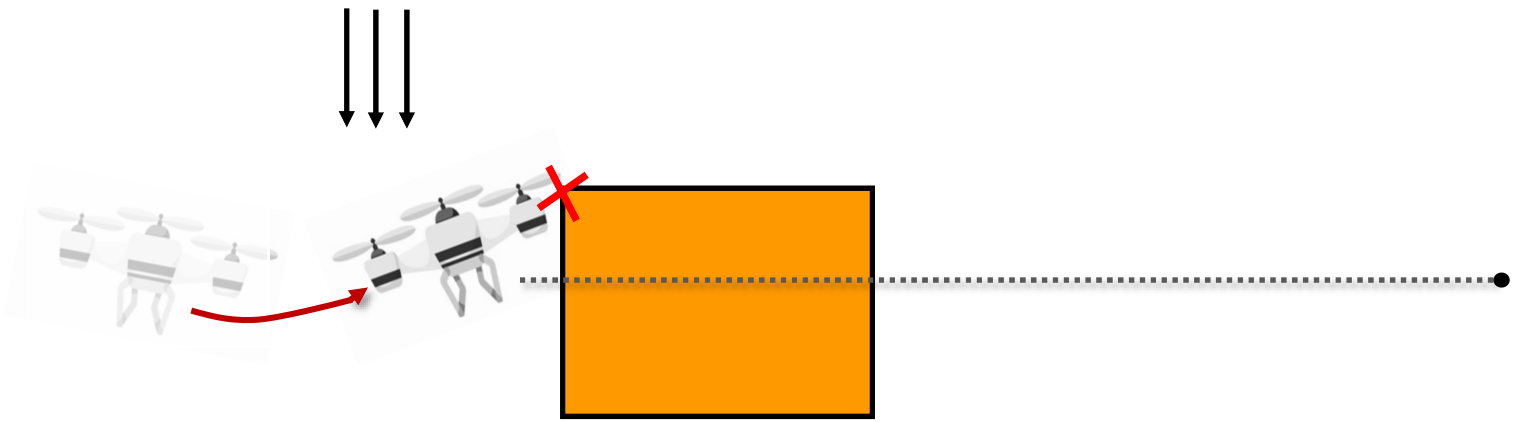
Model Predictive Control (MPC)



Repeat:

- apply optimal control law
- wait for new measurement
- re-optimize the trajectory

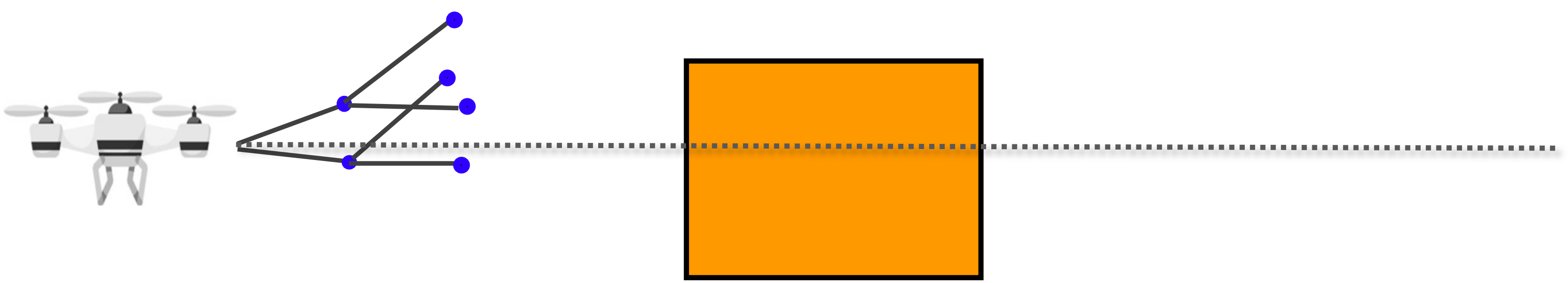
Model Predictive Control (MPC)



Problem:

- certainty equivalent prediction is optimistic
- infeasible (worst-case) scenarios possible

What is Robust MPC?



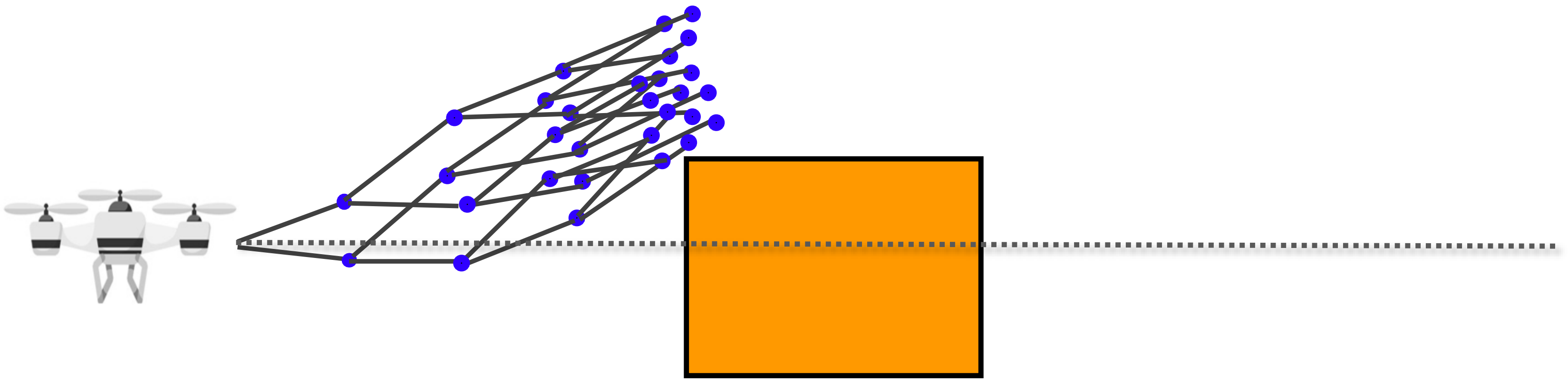
Main idea:

- take all possible uncertainty scenarios into account
- design tailored feedback policies to react to uncertainties

Limitations:

- exponential number of scenarios
- much more expensive than certainty equivalent MPC

What is Robust MPC?



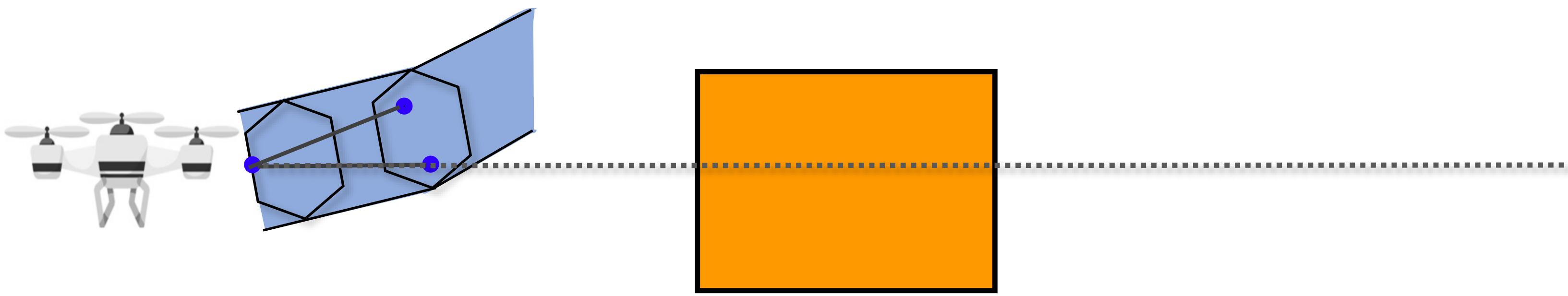
Main idea:

- take all possible uncertainty scenarios into account
- design tailored feedback policies to react to uncertainties

Limitations:

- exponential number of scenarios
- much more expensive than certainty equivalent MPC

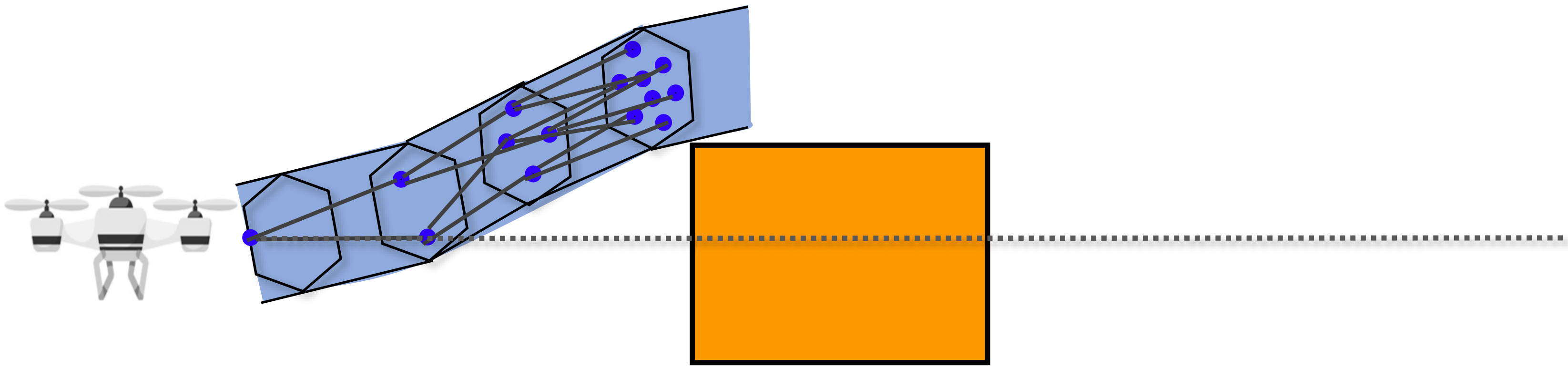
Rigid Tube MPC



Main idea:

- enclose all trajectories by a tube of constant polytopic cross-sections
- optimize the central path of the tube

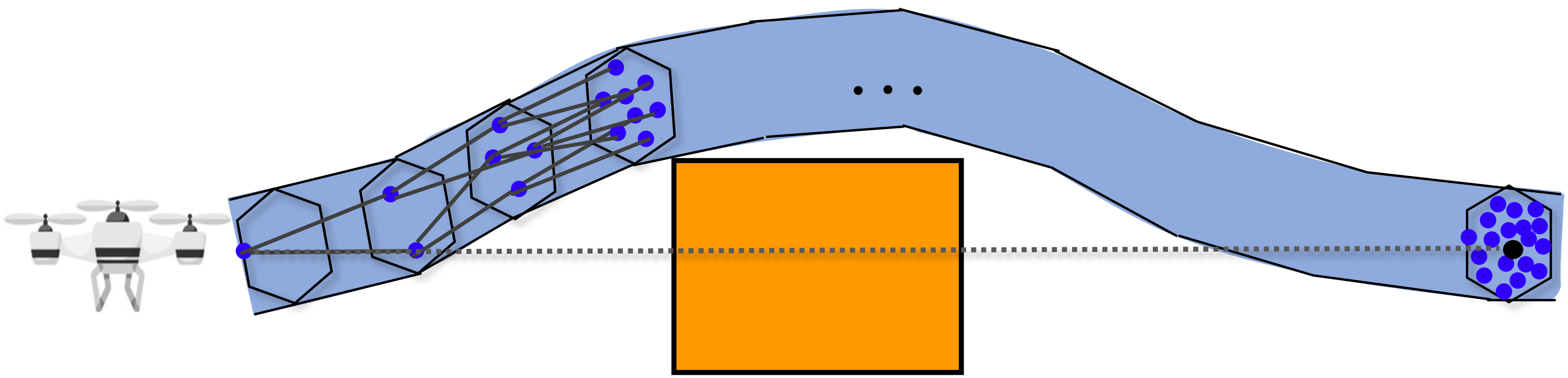
Rigid Tube MPC



Main idea:

- enclose all trajectories by a tube of constant polytopic cross-sections
- optimize the central path of the tube

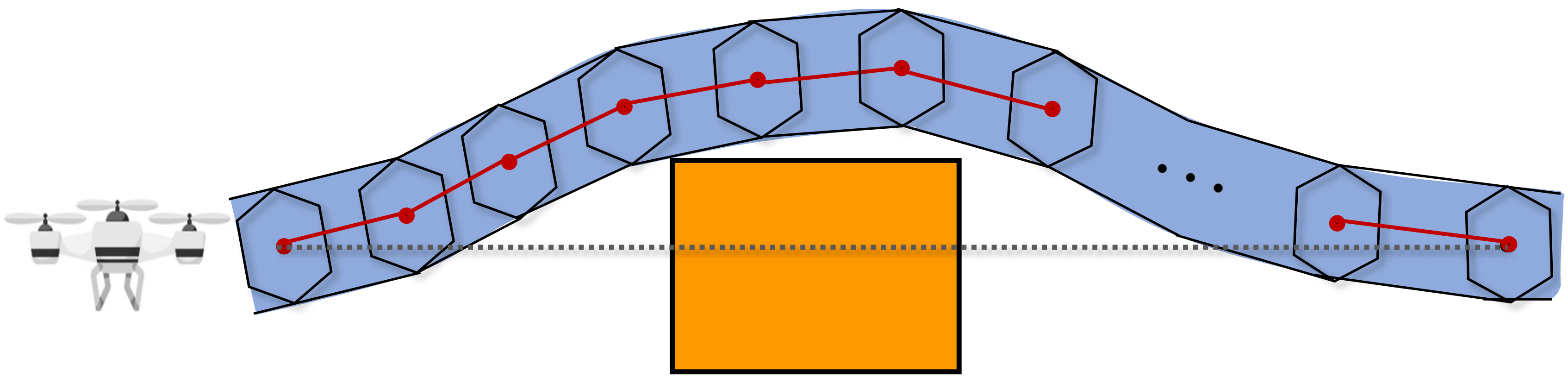
Rigid Tube MPC



Main idea:

- enclose all trajectories by a tube of constant polytopic cross-sections
- optimize the central path of the tube

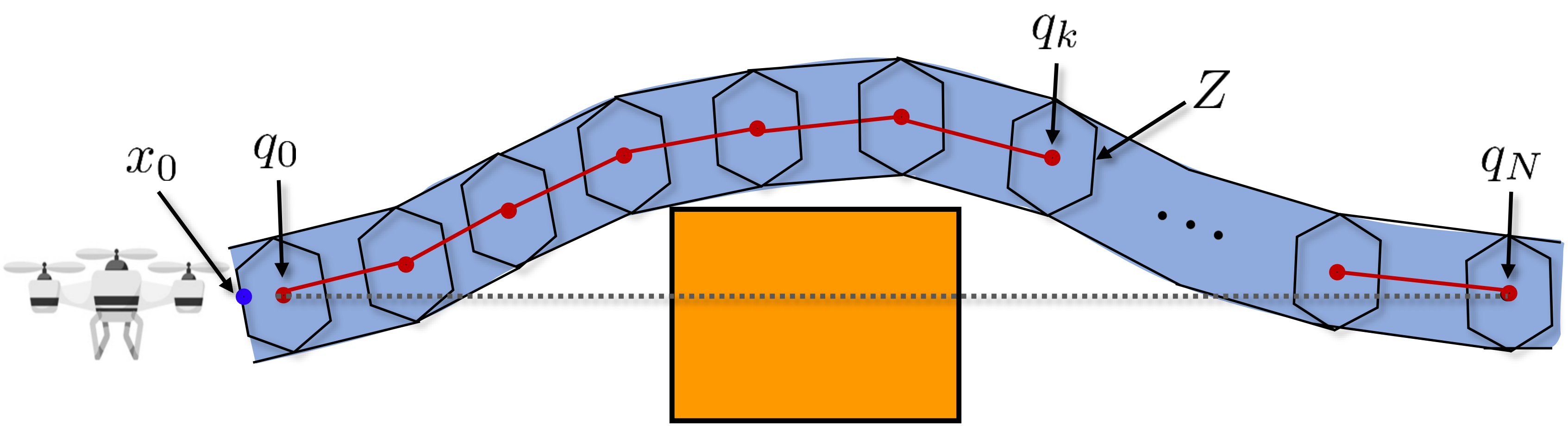
Rigid Tube MPC



Main idea:

- enclose all trajectories by a tube of constant polytopic cross-sections
- optimize the central path of the tube

Mathematical Formulation of Rigid Tube MPC



Uncertain control system:

$$x_{k+1} = Ax_k + Bu_k + w_k$$

Constraints:

$$x_k \in \mathbb{X},\ u_k \in \mathbb{U},\ w_k \in \mathbb{W}$$

Nominal system: (disturbance-free)

$q_{k+1} = Aq_k + Bv_k$ and $x_k = q_k + z_k$

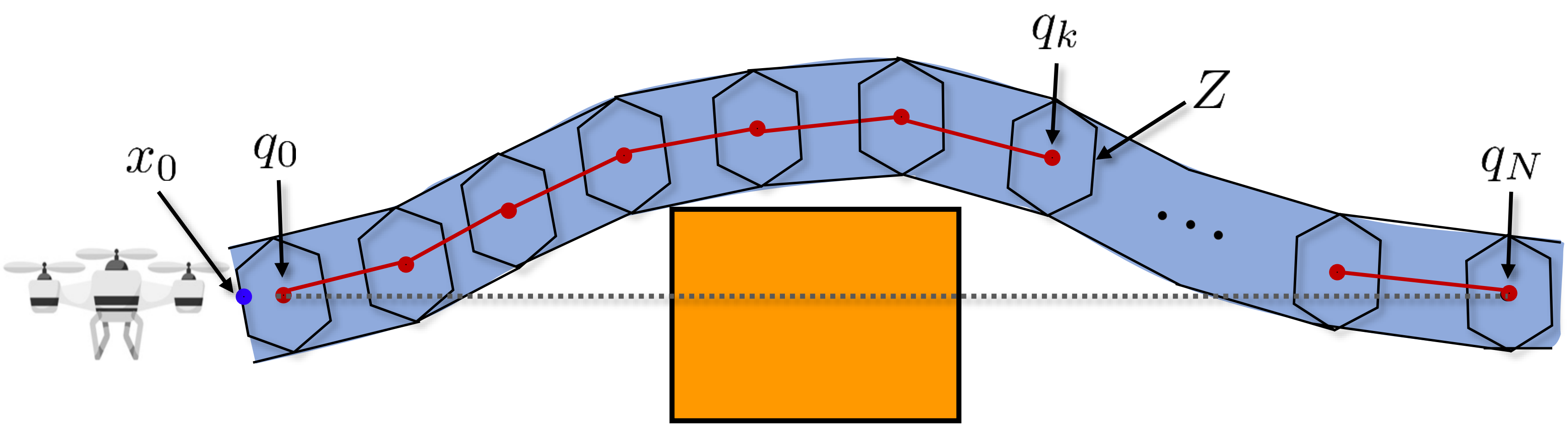
Linear feedback law:

$$\mu(k, x_k) = v_k + K(x_k - q_k)$$

Z : denotes a precomputed robust invariant set, w.r.t.,

$$z_{k+1} = (A + BK)z_k + w_k$$

Mathematical Formulation of Rigid Tube MPC



Uncertain control system:

$$x_{k+1} = Ax_k + Bu_k + w_k$$

Constraints:

$$x_k \in \mathbb{X},\ u_k \in \mathbb{U},\ w_k \in \mathbb{W}$$

Nominal system: (disturbance-free)

$q_{k+1} = Aq_k + Bv_k$ and $x_k = q_k + z_k$

Linear feedback law:

$$\mu(k, x_k) = v_k + K(x_k - q_k)$$

Z : denotes a precomputed robust invariant set, w.r.t.,

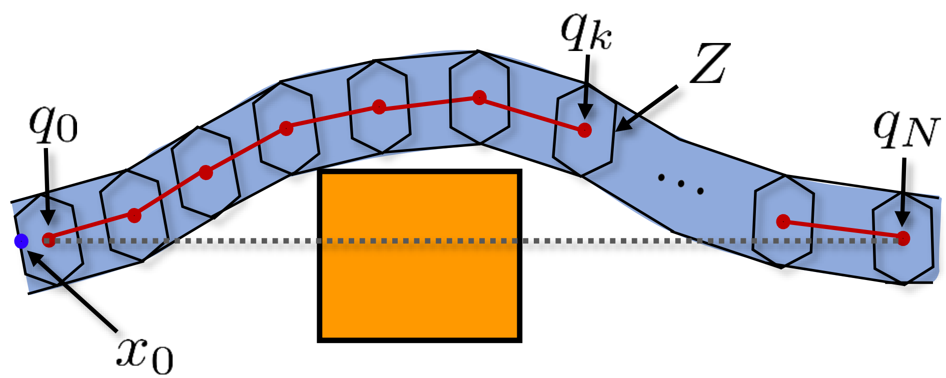
$$z_{k+1} = (A + BK)z_k + w_k$$

Mathematical Formulation of Rigid Tube MPC

Optimal Control Problem (OCP):

$$V(x_0) = \min_{q,v} \sum_{k=0}^{N-1} \ell(q_k, v_k) + m(q_N)$$

$$\text{s.t.} \begin{cases} \forall k = 0, \ldots, N-1 \\ q_{k+1} = Aq_k + Bv_k \\ q_k \in \mathbb{X} \ominus Z \,,\; q_N \in \mathbb{X}_T \\ v_k \in \mathbb{U} \ominus KZ \\ x_0 \in \{q_0\} \oplus Z \end{cases}$$



Standard assumptions:

- $(A+BK)\mathbb{X}_T \subset \mathbb{X}_T,\ \mathbb{X}_T \subset \mathbb{X} \ominus Z$ and $K\mathbb{X}_T \subset \mathbb{U} \ominus KZ$
- $m((A+BK)q) + \ell(q, Kq) \leq m(q),\ \forall q \in \mathbb{X}_T$

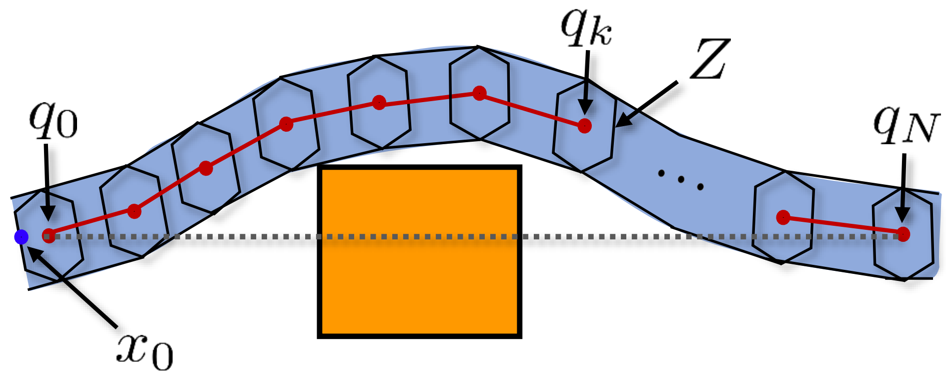
- stage cost: $\ell(q,v) = q^\intercal Q q + v^\intercal R v$
- terminal cost: $m(q) = q^\intercal P q$
- P, Q and R are symmetric positive definite
- $\mathbb{X}_T$: terminal constraint set

Mathematical Formulation of Rigid Tube MPC

Optimal Control Problem (OCP):

$$V(x_0) = \min_{q,v} \sum_{k=0}^{N-1} \ell(q_k, v_k) + m(q_N)$$

$$\text{s.t.} \quad \begin{cases} \forall k = 0, \ldots, N-1 \\ q_{k+1} = Aq_k + Bv_k \\ q_k \in \mathbb{X} \ominus Z \,,\; q_N \in \mathbb{X}_T \\ v_k \in \mathbb{U} \ominus KZ \\ x_0 \in \{q_0\} \oplus Z \end{cases}$$



Standard assumptions:

- $(A+BK)\mathbb{X}_T \subset \mathbb{X}_T$, $\mathbb{X}_T \subset \mathbb{X} \ominus Z$ and $K\mathbb{X}_T \subset \mathbb{U} \ominus KZ$
- $m((A+BK)q) + \ell(q, Kq) \leq m(q)$, $\forall q \in \mathbb{X}_T$

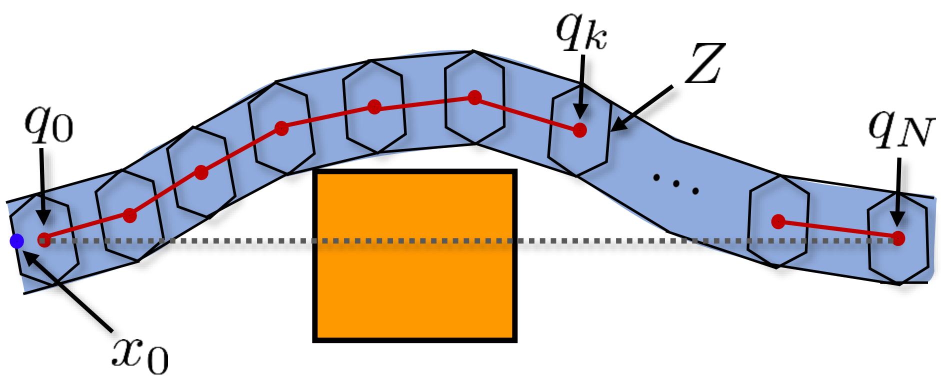
- stage cost: $\ell(q,v) = q^\intercal Q q + v^\intercal R v$
- terminal cost: $m(q) = q^\intercal P q$
- P, Q and R are symmetric positive definite
- $\mathbb{X}_T$: terminal constraint set

Mathematical Formulation of Rigid Tube MPC

Optimal Control Problem (OCP):

$$V(x_0) = \min_{q,v} \sum_{k=0}^{N-1} \ell(q_k, v_k) + m(q_N)$$

$$\text{s.t.} \quad \begin{cases} \forall k = 0, \ldots, N-1 \\ q_{k+1} = Aq_k + Bv_k \\ q_k \in \mathbb{X} \ominus Z \,,\; q_N \in \mathbb{X}_T \\ v_k \in \mathbb{U} \ominus KZ \\ x_0 \in \{q_0\} \oplus Z \end{cases}$$



Standard assumptions:

- $(A + BK)\mathbb{X}_T \subset \mathbb{X}_T$, $\mathbb{X}_T \subset \mathbb{X} \ominus Z$ and $K\mathbb{X}_T \subset \mathbb{U} \ominus KZ$
- $m((A + BK)q) + \ell(q, Kq) \leq m(q)$, $\forall q \in \mathbb{X}_T$

- stage cost: $\ell(q, v) = q^\intercal Qq + v^\intercal Rv$
- terminal cost: $m(q) = q^\intercal Pq$
- P, Q and R are symmetric positive definite
- $\mathbb{X}_T$: terminal constraint set

Overview

Background on Tube MPC

- Robust Model Predictive Control (MPC)
- Tube-Based Robust MPC
- Rigid Tube MPC

Contribution

- Parallel Real-time Optimization Algorithm
- Parallel Explicit Tube MPC Scheme
- Numerical Example

Goal: develop a real-time Tube MPC algorithm with limited memory footprint

Overview

Background on Tube MPC

- Robust Model Predictive Control (MPC)
- Tube-Based Robust MPC
- Rigid Tube MPC

Contribution

- Parallel Real-time Optimization Algorithm
- Parallel Explicit Tube MPC Scheme
- Numerical Example

Goal: develop a real-time Tube MPC algorithm with limited memory footprint

Rewrite the OCP of the Rigid Tube MPC

$$V(x_0) = \min_{q,v} \sum_{k=0}^{N-1} \ell(q_k, v_k) + m(q_N)$$

$$\text{s.t.} \quad \begin{cases} \forall k = 0, \ldots, N-1 \\ q_{k+1} = Aq_k + Bv_k \\ q_k \in \mathbb{X} \ominus Z \\ v_k \in \mathbb{U} \ominus KZ \\ x_0 \in \{q_0\} \oplus Z \\ q_N \in \mathbb{X}_T \end{cases}$$

- Stacked vectors and matrices

$$y_k = [q_k^\top \quad v_k^\top]^\top, \; y_N = q_N$$

$$C = [A \; B], \; D = [I \; 0]$$

- Constraint sets

$$\mathbb{Y}_0 = \left\{ y_0 \;\middle|\; \begin{array}{l} q_0 \in \mathbb{X} \ominus Z, \; v_0 \in \mathbb{U} \ominus KZ \\ Aq_0 + Bv_0 \in \mathbb{X} \ominus Z \\ x_0 \in \{q_0\} \oplus Z \end{array} \right\}$$

$$\mathbb{Y}_k = \left\{ y_k \;\middle|\; \begin{array}{l} q_k \in \mathbb{X} \ominus Z, \; v_k \in \mathbb{U} \ominus KZ \\ Aq_k + Bv_k \in \mathbb{X} \ominus Z \end{array} \right\}$$

$$\mathbb{Y}_N = \{y_N \mid q_N \in \mathbb{X}_T\}$$

- Cost function

$$J_k(y_k) = \ell(q_k, v_k), \; J_N(y_N) = m(q_N)$$

$$J(y) = \sum_{k=0}^{N} J_k(y_k)$$

Rewrite the OCP of the Rigid Tube MPC

$$V(x_0) = \min_{q,v} \sum_{k=0}^{N-1} \ell(q_k, v_k) + m(q_N)$$

$$\text{s.t.} \begin{cases} \forall k = 0, \ldots, N-1 \\ q_{k+1} = Aq_k + Bv_k \\ q_k \in \mathbb{X} \ominus Z \\ v_k \in \mathbb{U} \ominus KZ \\ x_0 \in \{q_0\} \oplus Z \\ q_N \in \mathbb{X}_T \end{cases}$$

- **Stacked vectors and matrices**

$$y_k = [q_k^\intercal \quad v_k^\intercal]^\intercal, \; y_N = q_N$$

$$C = [A \; B]\,, \; D = [I \; 0]$$

- **Constraint sets**

$$\mathbb{Y}_0 = \left\{ y_0 \;\middle|\; \begin{array}{l} q_0 \in \mathbb{X} \ominus Z, \; v_0 \in \mathbb{U} \ominus KZ \\ Aq_0 + Bv_0 \in \mathbb{X} \ominus Z \\ x_0 \in \{q_0\} \oplus Z \end{array} \right\}$$

$$\mathbb{Y}_k = \left\{ y_k \;\middle|\; \begin{array}{l} q_k \in \mathbb{X} \ominus Z, \; v_k \in \mathbb{U} \ominus KZ \\ Aq_k + Bv_k \in \mathbb{X} \ominus Z \end{array} \right\}$$

$$\mathbb{Y}_N = \{ y_N \mid q_N \in \mathbb{X}_T \}$$

- **Cost function**

$$J_k(y_k) = \ell(q_k, v_k)\,, \; J_N(y_N) = m(q_N)$$

$$J(y) = \sum_{k=0}^{N} J_k(y_k)$$

Rewrite the OCP of the Rigid Tube MPC

$$V(x_0) = \min_{q,v} \sum_{k=0}^{N-1} \ell(q_k, v_k) + m(q_N)$$

$$\text{s.t.} \begin{cases} \forall k = 0, \ldots, N-1 \\ q_{k+1} = Aq_k + Bv_k \\ q_k \in \mathbb{X} \ominus Z \\ v_k \in \mathbb{U} \ominus KZ \\ x_0 \in \{q_0\} \oplus Z \\ q_N \in \mathbb{X}_T \end{cases}$$

- **Stacked vectors and matrices**

$$y_k = [q_k^\intercal \quad v_k^\intercal]^\intercal, \; y_N = q_N$$

$$C = [A \; B] \, , \; D = [I \; 0]$$

- **Constraint sets**

$$\mathbb{Y}_0 = \left\{ y_0 \;\middle|\; \begin{array}{l} q_0 \in \mathbb{X} \ominus Z, \;\; v_0 \in \mathbb{U} \ominus KZ \\ Aq_0 + Bv_0 \in \mathbb{X} \ominus Z \\ x_0 \in \{q_0\} \oplus Z \end{array} \right\}$$

$$\mathbb{Y}_k = \left\{ y_k \;\middle|\; \begin{array}{l} q_k \in \mathbb{X} \ominus Z, \;\; v_k \in \mathbb{U} \ominus KZ \\ Aq_k + Bv_k \in \mathbb{X} \ominus Z \end{array} \right\}$$

$$\mathbb{Y}_N = \{ y_N \mid q_N \in \mathbb{X}_T \}$$

- **Cost function**

$$J_k(y_k) = \ell(q_k, v_k) \, , \; J_N(y_N) = m(q_N)$$

$$J(y) = \sum_{k=0}^{N} J_k(y_k)$$

Rewrite the OCP of the Rigid Tube MPC

$$V(x_0) = \min_{q,v} \sum_{k=0}^{N-1} \ell(q_k, v_k) + m(q_N)$$

$$\text{s.t.} \begin{cases} \forall k = 0, \dots, N-1 \\ q_{k+1} = Aq_k + Bv_k \\ q_k \in \mathbb{X} \ominus Z \\ v_k \in \mathbb{U} \ominus KZ \\ x_0 \in \{q_0\} \oplus Z \\ q_N \in \mathbb{X}_T \end{cases}$$

- **Stacked vectors and matrices**

$$y_k = [q_k^\top \quad v_k^\top]^\top, \; y_N = q_N$$

$$C = [A \; B] \, , \; D = [I \; 0]$$

- **Constraint sets**

$$\mathbb{Y}_0 = \left\{ y_0 \;\middle|\; \begin{array}{l} q_0 \in \mathbb{X} \ominus Z, \; v_0 \in \mathbb{U} \ominus KZ \\ Aq_0 + Bv_0 \in \mathbb{X} \ominus Z \\ x_0 \in \{q_0\} \oplus Z \end{array} \right\}$$

$$\mathbb{Y}_k = \left\{ y_k \;\middle|\; \begin{array}{l} q_k \in \mathbb{X} \ominus Z, \; v_k \in \mathbb{U} \ominus KZ \\ Aq_k + Bv_k \in \mathbb{X} \ominus Z \end{array} \right\}$$

$$\mathbb{Y}_N = \{ y_N \mid q_N \in \mathbb{X}_T \}$$

- **Cost function**

$$J_k(y_k) = \ell(q_k, v_k) \, , \; J_N(y_N) = m(q_N)$$

$$J(y) = \sum_{k=0}^{N} J_k(y_k)$$

Rewrite the OCP of the Rigid Tube MPC

$$V(x_0) = \min_{q,v} \sum_{k=0}^{N-1} \ell(q_k, v_k) + m(q_N)$$

$$\text{s.t.} \begin{cases} \forall k = 0, \dots, N-1 \\ q_{k+1} = Aq_k + Bv_k \\ q_k \in \mathbb{X} \ominus Z \\ v_k \in \mathbb{U} \ominus KZ \\ x_0 \in \{q_0\} \oplus Z \\ q_N \in \mathbb{X}_T \end{cases}$$

equivalent $\Updownarrow$

☆

$$V(x_0) = \min_{y} J(y)$$

$$\text{s.t.} \begin{cases} \forall k \in \{0, \dots, N-2,\} \\ Dy_{k+1} - Cy_k = 0 \quad | \; \lambda_{k+1} \\ y_N - Cy_{N-1} = 0 \quad | \; \lambda_N \\ y_k \in \mathbb{Y}_k, y_{N-1} \in \mathbb{Y}_{N-1}, y_N \in \mathbb{Y}_N \end{cases}$$

- **Stacked vectors and matrices**

$$y_k = [q_k^\mathsf{T} \quad v_k^\mathsf{T}]^\mathsf{T}, \; y_N = q_N$$

$$C = [A \; B], \; D = [I \; 0]$$

- **Constraint sets**

$$\mathbb{Y}_0 = \left\{ y_0 \;\middle|\; \begin{array}{l} q_0 \in \mathbb{X} \ominus Z, \; v_0 \in \mathbb{U} \ominus KZ \\ Aq_0 + Bv_0 \in \mathbb{X} \ominus Z \\ x_0 \in \{q_0\} \oplus Z \end{array} \right\}$$

$$\mathbb{Y}_k = \left\{ y_k \;\middle|\; \begin{array}{l} q_k \in \mathbb{X} \ominus Z, \; v_k \in \mathbb{U} \ominus KZ \\ Aq_k + Bv_k \in \mathbb{X} \ominus Z \end{array} \right\}$$

$$\mathbb{Y}_N = \{ y_N \mid q_N \in \mathbb{X}_T \}$$

- **Cost function**

$$J_k(y_k) = \ell(q_k, v_k), \; J_N(y_N) = m(q_N)$$

$$J(y) = \sum_{k=0}^{N} J_k(y_k)$$

Parallel Real-time Optimization Algorithm (Coupled QP)

$$V(x_0) = \min_y \sum_{k=0}^{N} J_k(y_k) \quad \star$$

$$\text{s.t.} \begin{cases} \forall k \in \{0,\ldots,N-2,\} \\ Dy_{k+1} - Cy_k = 0 \quad \mid \lambda_{k+1} \\ y_N - Cy_{N-1} = 0 \quad \mid \lambda_N \\ y_k \in \mathbb{Y}_k, y_{N-1} \in \mathbb{Y}_{N-1} \\ y_N \in \mathbb{Y}_N \end{cases}$$

Unconstrained auxiliary optimization problem:

$$V(x_0) = \min_y \sum_{k=0}^{N} J_k(y_k - y_k^{\text{ref}}) \quad ①$$

$$\text{s.t.} \begin{cases} \forall k \in \{0,\ldots,N-2\} \\ Dy_{k+1} - Cy_k = 0 \quad \mid \delta_{k+1} \\ y_N - Cy_{N-1} = 0 \quad \mid \delta_N \end{cases}$$

Main idea:

- problem ① approximates ☆ without needing inequality constraints
- for $y^{\text{ref}} = y^\star$, problems ① and ☆ are equivalent
- problem ① can be solved efficiently using a sparse linear algebra solver

Parallel Real-time Optimization Algorithm (Coupled QP)

$$V(x_0) = \min_y \sum_{k=0}^{N} J_k(y_k) \qquad ☆$$

$$\text{s.t.} \begin{cases} \forall k \in \{0,\dots,N-2,\} \\ D y_{k+1} - C y_k = 0 & \mid \lambda_{k+1} \\ y_N - C y_{N-1} = 0 & \mid \lambda_N \\ y_k \in \mathbb{Y}_k, y_{N-1} \in \mathbb{Y}_{N-1} \\ y_N \in \mathbb{Y}_N \end{cases}$$

Unconstrained auxiliary optimization problem:

$$V(x_0) = \min_y \sum_{k=0}^{N} J_k(y_k - y_k^{\text{ref}}) \qquad ①$$

$$\text{s.t.} \begin{cases} \forall k \in \{0,\dots,N-2\} \\ D y_{k+1} - C y_k = 0 & \mid \delta_{k+1} \\ y_N - C y_{N-1} = 0 & \mid \delta_N \end{cases}$$

Main idea:

- problem ① approximates ☆ without needing inequality constraints
- for $y^{\text{ref}} = y^\star$, problems ① and ☆ are equivalent
- problem ① can be solved efficiently using a sparse linear algebra solver

Parallel Real-time Optimization Algorithm (Decoupled QPs)

$$V(x_0) = \min_{y} \sum_{k=0}^{N} J_k(y_k) \qquad \star$$

$$\text{s.t.} \begin{cases} \forall k \in \{0, \ldots, N-2, \} \\ D y_{k+1} - C y_k = 0 \quad \mid \lambda_{k+1} \\ y_N - C y_{N-1} = 0 \quad \mid \lambda_N \\ y_k \in \mathbb{Y}_k, y_{N-1} \in \mathbb{Y}_{N-1} \\ y_N \in \mathbb{Y}_N \end{cases}$$

Augmented Lagrangian optimization problem:

$$\xi^j = \underset{\xi \in \mathbb{Y}}{\operatorname{argmin}} \; J(\xi) + (G^\intercal \lambda^j)^\intercal \xi + (\xi - y^j)^\intercal H (\xi - y^j) \qquad ②$$

- y^j and λ^j current approximations, and
- $G = \begin{pmatrix} -C & D & & & 0 \\ & -C & D & & \\ & & \ddots & \ddots & \\ 0 & & & -C & I \end{pmatrix}$, $H = \nabla^2 J(y)$

Notice that ② has a separable structure ：

$$\xi_0^j = \underset{\xi \in \mathbb{Y}_0}{\operatorname{argmin}} \; J_0(\xi) - (C^\intercal \lambda_1^j)^\intercal \xi + J_0(\xi - y_0^j)$$

$$\xi_k^j = \underset{\xi \in \mathbb{Y}_k}{\operatorname{argmin}} \; J_k(\xi) + (D^\intercal \lambda_k^j - C^\intercal \lambda_{k+1}^j)^\intercal \xi + J_k(\xi - y_k^j)$$

$$\xi_N^j = \underset{\xi \in \mathbb{Y}_N}{\operatorname{argmin}} \; J_N(\xi) + (\lambda_N^j)^\intercal \xi + J_N(\xi - y_N^j)$$

strictly convex quadratic programs

Parallel Real-time Optimization Algorithm (Decoupled QPs)

$$V(x_0) = \min_{y} \sum_{k=0}^{N} J_k(y_k) \quad \star$$

$$\text{s.t.} \quad \begin{cases} \forall k \in \{0, \dots, N-2,\} \\ D y_{k+1} - C y_k = 0 & \mid \lambda_{k+1} \\ y_N - C y_{N-1} = 0 & \mid \lambda_N \\ y_k \in \mathbb{Y}_k, y_{N-1} \in \mathbb{Y}_{N-1} \\ y_N \in \mathbb{Y}_N \end{cases}$$

Augmented Lagrangian optimization problem:

$$\xi^j = \underset{\xi \in \mathbb{Y}}{\operatorname{argmin}} \ J(\xi) + (G^\intercal \lambda^j)^\intercal \xi + (\xi - y^j)^\intercal H (\xi - y^j) \quad ②$$

- y^j and λ^j current approximations, and
- $G = \begin{pmatrix} -C & D & & & 0 \\ & -C & D & & \\ & & \ddots & \ddots & \\ 0 & & & -C & I \end{pmatrix}$, $H = \nabla^2 J(y)$

Notice that ② has a separable structure ：

$$\xi_0^j = \underset{\xi \in \mathbb{Y}_0}{\operatorname{argmin}} \ J_0(\xi) - (C^\intercal \lambda_1^j)^\intercal \xi + J_0(\xi - y_0^j)$$

$$\xi_k^j = \underset{\xi \in \mathbb{Y}_k}{\operatorname{argmin}} \ J_k(\xi) + (D^\intercal \lambda_k^j - C^\intercal \lambda_{k+1}^j)^\intercal \xi + J_k(\xi - y_k^j)$$

$$\xi_N^j = \underset{\xi \in \mathbb{Y}_N}{\operatorname{argmin}} \ J_N(\xi) + (\lambda_N^j)^\intercal \xi + J_N(\xi - y_N^j)$$

strictly convex quadratic programs

Parallel Real-time Optimization Algorithm (Decoupled QPs)

$$\xi_0^j = \underset{\xi \in \mathbb{Y}_0}{\operatorname{argmin}} \;\; J_0(\xi) - (C^\intercal \lambda_1^j)^\intercal \xi + J_0(\xi - y_0^j)$$

$$\xi_k^j = \underset{\xi \in \mathbb{Y}_k}{\operatorname{argmin}} \;\; J_k(\xi) + (D^\intercal \lambda_k^j - C^\intercal \lambda_{k+1}^j)^\intercal \xi + J_k(\xi - y_k^j)$$

$$\xi_N^j = \underset{\xi \in \mathbb{Y}_N}{\operatorname{argmin}} \;\; J_N(\xi) + (\lambda_N^j)^\intercal \xi + J_N(\xi - y_N^j)$$

Parallel Real-time Optimization Algorithm (Decoupled QPs)

$$\xi_0^j = \underset{\xi \in \mathbb{Y}_0}{\operatorname{argmin}} \; J_0(\xi) - (C^\intercal \lambda_1^j)^\intercal \xi + J_0(\xi - y_0^j)$$

$$\xi_k^j = \underset{\xi \in \mathbb{Y}_k}{\operatorname{argmin}} \; J_k(\xi) + (D^\intercal \lambda_k^j - C^\intercal \lambda_{k+1}^j)^\intercal \xi + J_k(\xi - y_k^j)$$

$$\xi_N^j = \underset{\xi \in \mathbb{Y}_N}{\operatorname{argmin}} \; J_N(\xi) + (\lambda_N^j)^\intercal \xi + J_N(\xi - y_N^j)$$

using multi-parametric quadratic programming

pre-compute three solution maps

$$\xi_0^j(\lambda_1^j, y_0^j, x_0) = \underset{\xi \in \mathbb{Y}_0}{\operatorname{argmin}} \; J_0(\xi) - (C^\intercal \lambda_1^j)^\intercal \xi + J_0(\xi - y_0^j), \; (x_0 \text{ is hidden in } \mathbb{Y}_0)$$

$$\xi_k^j(\lambda_k^j, \lambda_{k+1}^j, y_k^j) = \underset{\xi \in \mathbb{Y}_k}{\operatorname{argmin}} \; J_k(\xi) + (D^\intercal \lambda_k^j - C^\intercal \lambda_{k+1}^j)^\intercal \xi + J_k(\xi - y_k^j)$$

$$\xi_N^j(\lambda_N^j, y_N^j) = \underset{\xi \in \mathbb{Y}_N}{\operatorname{argmin}} \; J_N(\xi) + (\lambda_N^j)^\intercal \xi + J_N(\xi - y_N^j)$$

The memory requirements do not depend on the prediction horizon N

Parallel Real-time Optimization Algorithm (Decoupled QPs)

$$\xi_0^j = \operatorname*{argmin}_{\xi \in \mathbb{Y}_0} \; J_0(\xi) - (C^\intercal \lambda_1^j)^\intercal \xi + J_0(\xi - y_0^j)$$

$$\xi_k^j = \operatorname*{argmin}_{\xi \in \mathbb{Y}_k} \; J_k(\xi) + (D^\intercal \lambda_k^j - C^\intercal \lambda_{k+1}^j)^\intercal \xi + J_k(\xi - y_k^j)$$

$$\xi_N^j = \operatorname*{argmin}_{\xi \in \mathbb{Y}_N} \; J_N(\xi) + (\lambda_N^j)^\intercal \xi + J_N(\xi - y_N^j)$$

using multi-parametric quadratic programming

pre-compute three solution maps

$$\xi_0^j(\lambda_1^j, y_0^j, x_0) = \operatorname*{argmin}_{\xi \in \mathbb{Y}_0} \; J_0(\xi) - (C^\intercal \lambda_1^j)^\intercal \xi + J_0(\xi - y_0^j), \;\; (x_0 \text{ is hidden in } \mathbb{Y}_0)$$

$$\xi_k^j(\lambda_k^j, \lambda_{k+1}^j, y_k^j) = \operatorname*{argmin}_{\xi \in \mathbb{Y}_k} \; J_k(\xi) + (D^\intercal \lambda_k^j - C^\intercal \lambda_{k+1}^j)^\intercal \xi + J_k(\xi - y_k^j)$$

$$\xi_N^j(\lambda_N^j, y_N^j) = \operatorname*{argmin}_{\xi \in \mathbb{Y}_N} \; J_N(\xi) + (\lambda_N^j)^\intercal \xi + J_N(\xi - y_N^j)$$

The memory requirements do not depend on the prediction horizon N

Parallel Real-time Optimization Algorithm

Optimization Algorithm:

For $j = 1 \to m$ (with initial guesses y^1 and λ^1)

Step1: input $y^j = [y_0^j, \dots, y_N^j]$, $\lambda^j = [\lambda_1^j, \dots, \lambda_N^j]$

Step2: compute $\xi^j = (\xi_0^j, \xi_1^j, \dots, \xi_N^j)$ using decoupled QPs

Step3: set $y^{\mathrm{ref}} = 2\xi^j - y^j$

Step4: update next iterate using coupled QP:

$$y^{j+1} = \underset{y}{\mathrm{argmin}} \sum_{k=0}^{N} J_k(y_k - y_k^{\mathrm{ref}})$$

$$\text{s.t.} \quad \begin{cases} \forall k \in \{0, \dots, N-2\} \\ Dy_{k+1} - Cy_k = 0 \quad \mid \delta_{k+1} \\ y_N - Cy_{N-1} = 0 \quad \mid \delta_N \end{cases}$$

$$\lambda^{j+1} = \lambda^j + \delta^j$$

End

Parallel Real-time Optimization Algorithm

Optimization Algorithm:

For $j = 1 \to m$ (with initial guesses y^1 and λ^1)

Step1: input $y^j = [y_0^j, \ldots, y_N^j]$, $\lambda^j = [\lambda_1^j, \ldots, \lambda_N^j]$

Step2: compute $\xi^j = (\xi_0^j, \xi_1^j, \ldots, \xi_N^j)$ using decoupled QPs

Step3: set $y^{\mathrm{ref}} = 2\xi^j - y^j$

Step4: update next iterate using coupled QP:

$$y^{j+1} = \underset{y}{\operatorname{argmin}} \sum_{k=0}^{N} J_k(y_k - y_k^{\mathrm{ref}})$$

$$\text{s.t.} \quad \begin{cases} \forall k \in \{0, \ldots, N-2\} \\ D y_{k+1} - C y_k = 0 \quad | \ \delta_{k+1} \\ y_N - C y_{N-1} = 0 \quad | \ \delta_N \end{cases}$$

$$\lambda^{j+1} = \lambda^j + \delta^j$$

End

Parallel Real-time Optimization Algorithm

Optimization Algorithm:

For $j = 1 \to m$ (with initial guesses y^1 and λ^1)

Step1: input $y^j = [y_0^j, \ldots, y_N^j]$, $\lambda^j = [\lambda_1^j, \ldots, \lambda_N^j]$

Step2: compute $\xi^j = (\xi_0^j, \xi_1^j, \ldots, \xi_N^j)$ using decoupled QPs

Step3: set $y^{\mathrm{ref}} = 2\xi^j - y^j$

Step4: update next iterate using coupled QP:

$$y^{j+1} = \underset{y}{\operatorname{argmin}} \sum_{k=0}^{N} J_k(y_k - y_k^{\mathrm{ref}})$$

$$\text{s.t.} \quad \begin{cases} \forall k \in \{0, \ldots, N-2\} \\ D y_{k+1} - C y_k = 0 \quad \mid \delta_{k+1} \\ y_N - C y_{N-1} = 0 \quad \mid \delta_N \end{cases}$$

$$\lambda^{j+1} = \lambda^j + \delta^j$$

End

Parallel Real-time Optimization Algorithm

Optimization Algorithm:

For $j = 1 \to m$ (with initial guesses y^1 and λ^1)

Step1: input $y^j = [y_0^j, \ldots, y_N^j]$, $\lambda^j = [\lambda_1^j, \ldots, \lambda_N^j]$

Step2: compute $\xi^j = (\xi_0^j, \xi_1^j, \ldots, \xi_N^j)$ using decoupled QPs

Step3: set $y^{\mathrm{ref}} = 2\xi^j - y^j$

Step4: update next iterate using coupled QP:

$$y^{j+1} = \underset{y}{\operatorname{argmin}} \sum_{k=0}^{N} J_k(y_k - y_k^{\mathrm{ref}})$$

$$\text{s.t.} \quad \begin{cases} \forall k \in \{0, \ldots, N-2\} \\ D y_{k+1} - C y_k = 0 \quad \mid \delta_{k+1} \\ y_N - C y_{N-1} = 0 \quad \mid \delta_N \end{cases}$$

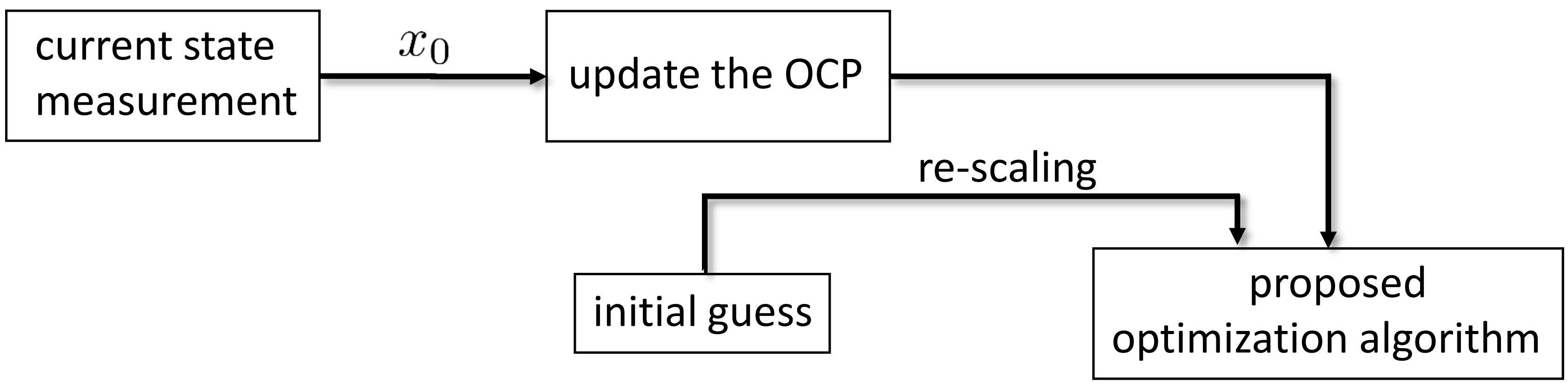
$$\lambda^{j+1} = \lambda^j + \delta^j$$

End

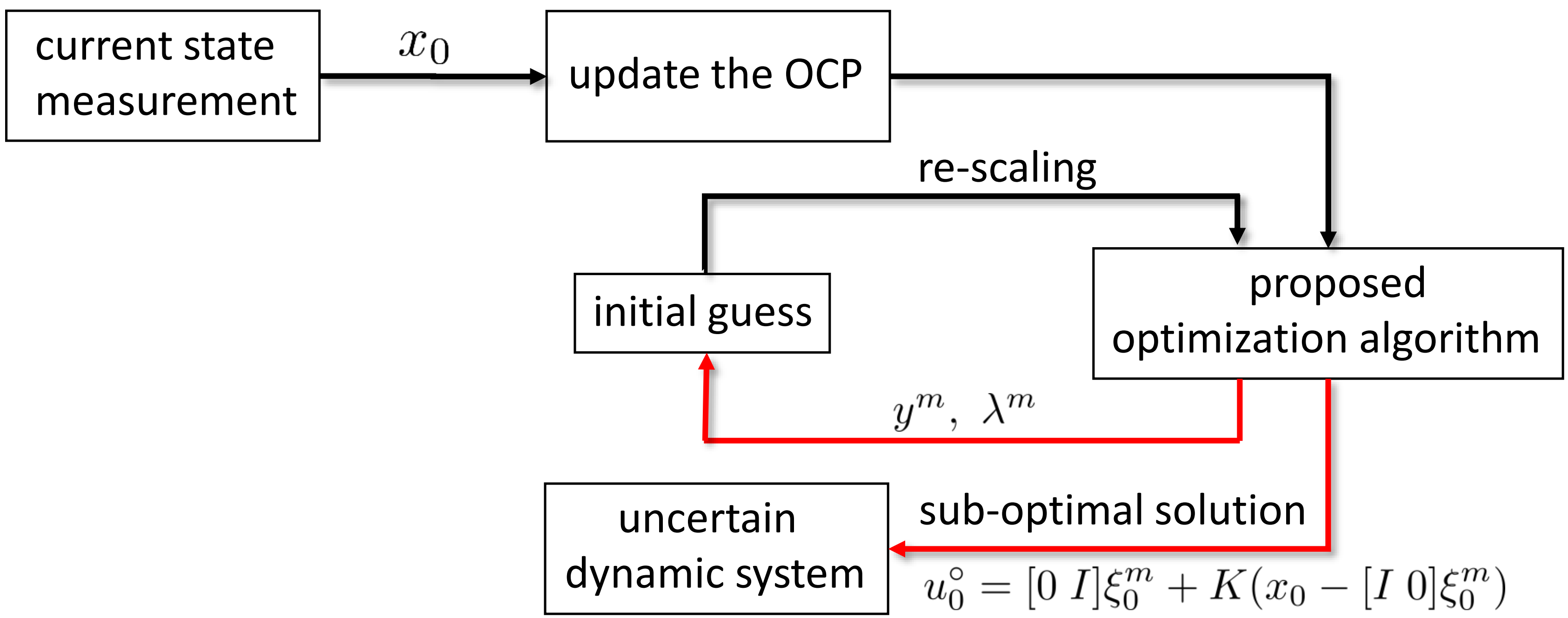
The algorithm converges linearly: $(0 < \kappa < 1)$

$$J(y^{j+1} - y^\star) + J^\star(\lambda^{j+1} - \lambda^\star) \le \kappa \left(J(y^j - y^\star) + J^\star(\lambda^j - \lambda^\star) \right)$$

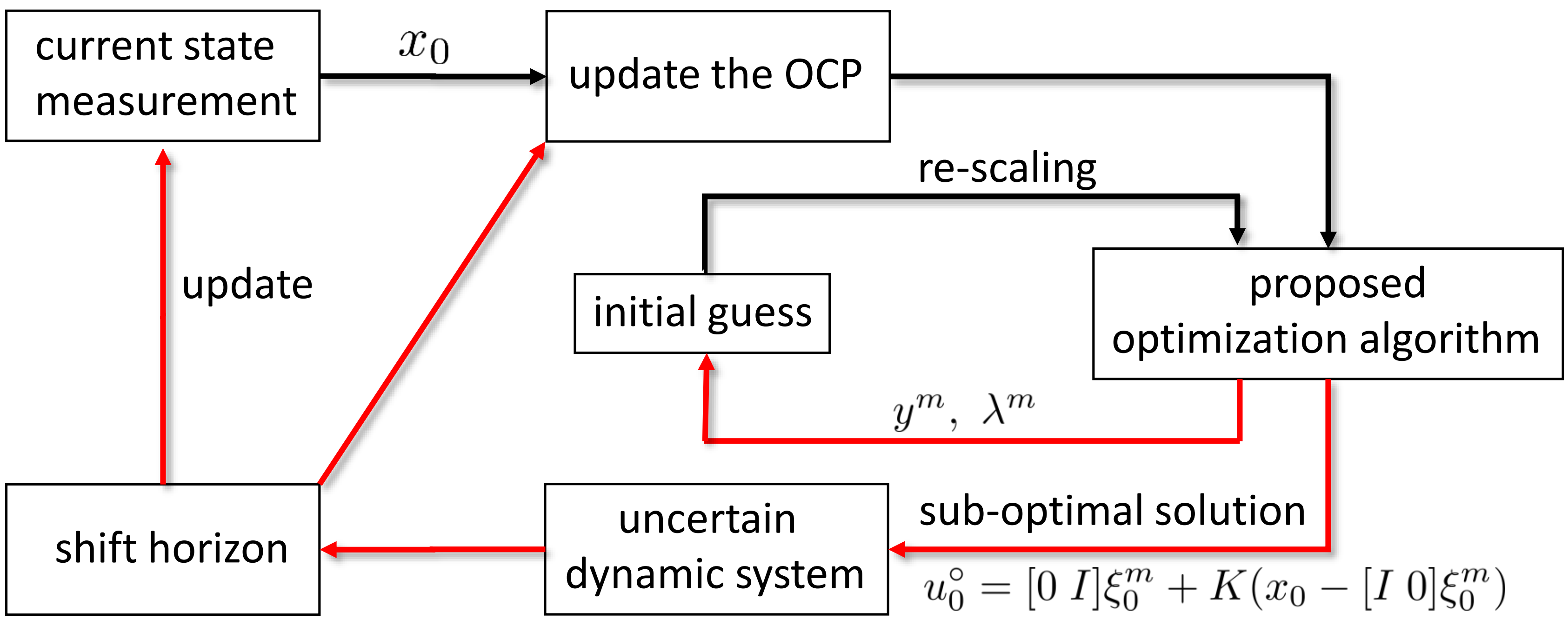
Parallel Explicit Tube MPC Controller



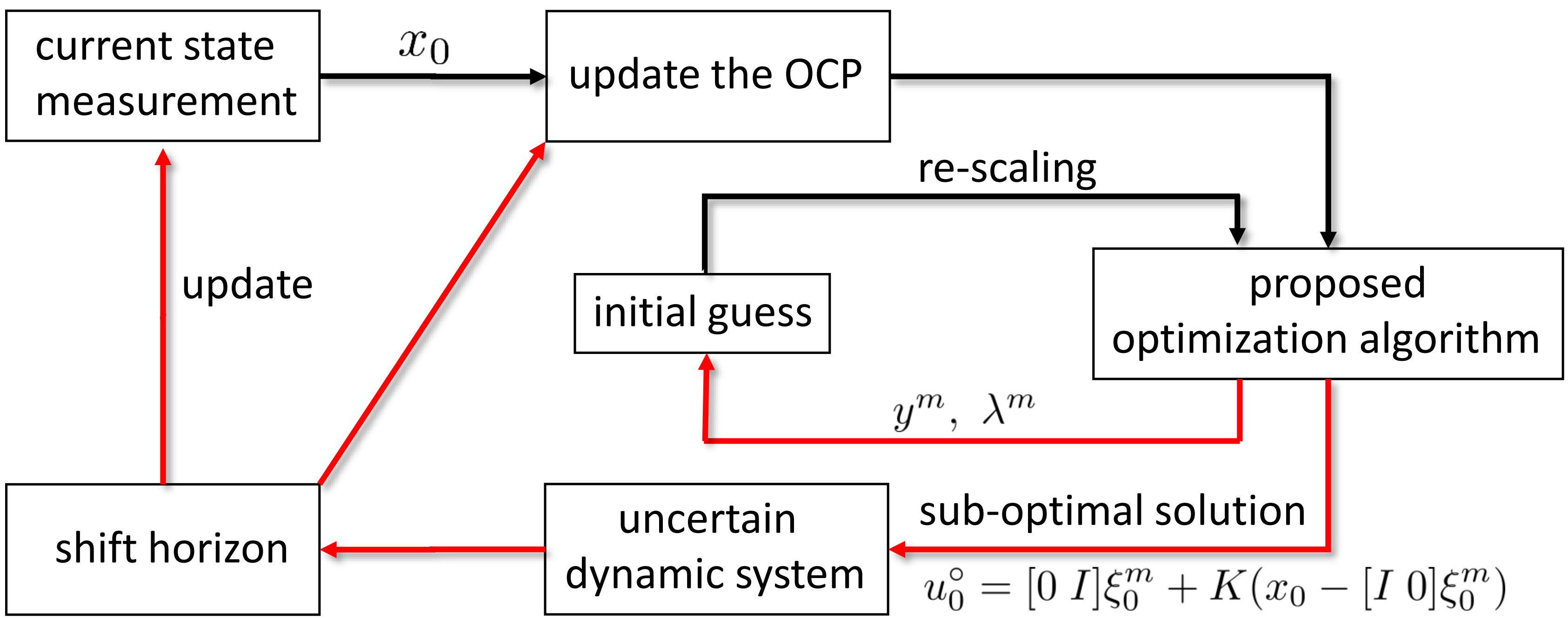
Parallel Explicit Tube MPC Controller



Parallel Explicit Tube MPC Controller



Parallel Explicit Tube MPC Controller (Recursive Feasibility)



Recursive Feasibility

we assume that the state constant set $\mathbb{X}$ to be robust control invariant, the subop-timal solution preserves the recursive feasibility

Parallel Explicit Tube MPC Controller (Stability)

Theorem 1

If the number of inner loops in the proposed optimization algorithm satisfies

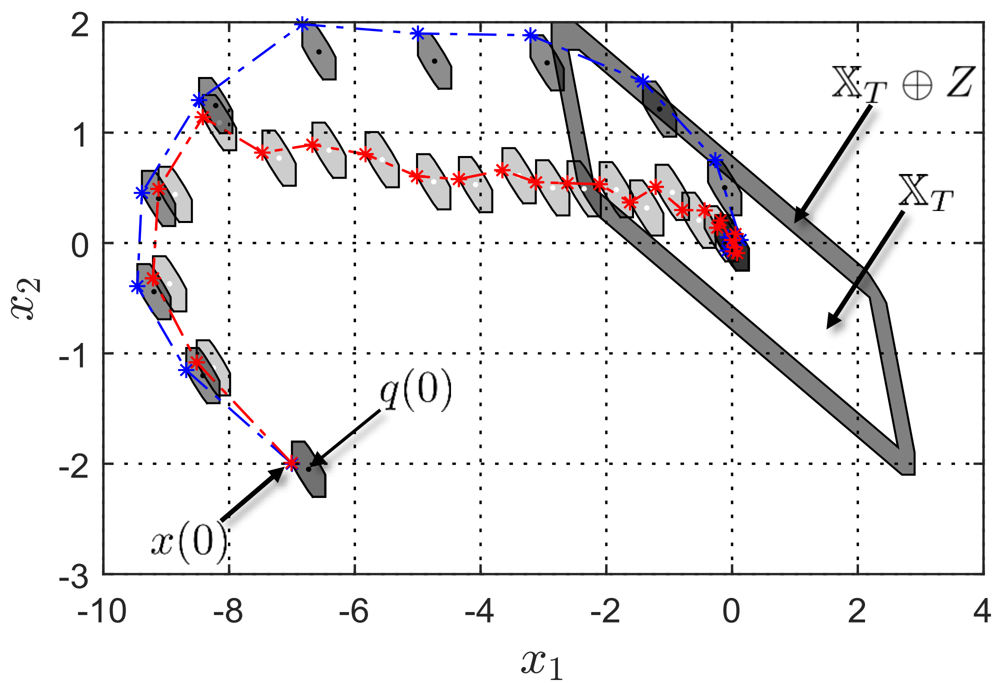
$$m \geq 2\frac{\log\left(\eta\sqrt{\sigma}+\frac{\tau\sigma}{2}\right)}{\log(1/\kappa)} \ ,$$

the proposed scheme yields an **asymptotically stable** closed-loop controller.

- $0 < \kappa < 1$: linear convergence rate
- $\eta, \tau > 0$: satisfy the inequality $|V(x_0^+) - V(x_1^\star)| \leq \eta \|x_0^+ - x_1^\star\|_Q + \frac{\tau}{2}\|x_0^+ - x_1^\star\|_Q^2$
- $\sigma > 0$: satisfies the inequality $\|x_0^+ - x_1^\star\|_Q^2 \leq \sigma\kappa^m J_0(y_0^\star)$

Numerical Example

- Double integrator example: $A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \; B = \begin{pmatrix} 0.5 \\ 1 \end{pmatrix}$
- Disturbance set: $\mathbb{W} = \{w \mid \|w\|_\infty \leqslant 0.1\}$
- State and input constraint: $\mathbb{X} = \{x \mid [0\ 1]x \leq 2\}, \quad \mathbb{U} = \{u \mid |u| \leq 1\}$
- Matrices: $Q = I,\ R = 0.1,\ K = -(0.62,\ 1.27),\ P = \begin{pmatrix} 2.06 & 0.60 \\ 0.60 & 1.40 \end{pmatrix}$



Numerical Example (Memory Requirements)

Parallel Explicit Tube MPC

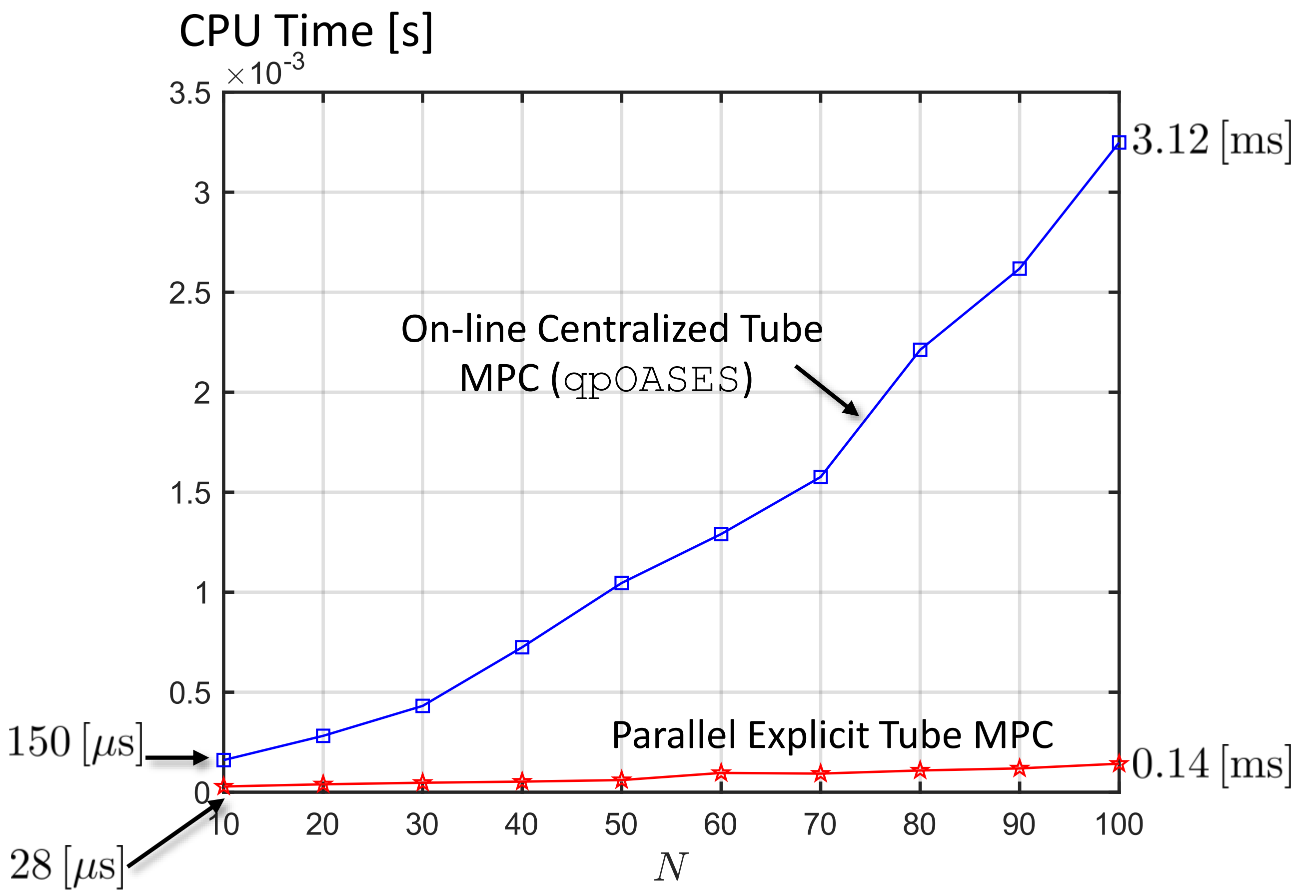
N	number of regions	memory [KB]
10	456	36
20	456	36
30	456	36
50	456	36
$\vdots$	456	36

VS

Explicit Tube MPC

N	number of regions	memory [KB]
10	1648	174
20	5312	1028
30	11050	3108
50	25160	11500
70	42700	27066

Numerical Example (Runtime Performance)



Summary

Conclusions:

- Derived a real-time and parallelizable Tube MPC scheme
- Maintained recursive feasibility and stability
- Reduced the storage of Explicit MPC by orders of magnitude

Future work:

- Implementation on the embedded applications
- Extension to nonlinear systems

Summary

Conclusions:

- Derived a real-time and parallelizable Tube MPC scheme
- Maintained recursive feasibility and stability
- Reduced the storage of Explicit MPC by orders of magnitude

Future work:

- Implementation on the embedded applications
- Extension to nonlinear systems

Summary

Conclusions:

- Derived a real-time and parallelizable Tube MPC scheme
- Maintained recursive feasibility and stability
- Reduced the storage of Explicit MPC by orders of magnitude

Future work:

- Implementation on the embedded applications
- Extension to nonlinear systems

Contact: `wangkai1@shanghaitech.edu.cn`

Thank you! Questions?