深蓝学院
shenlanxueyuan.com

Model Predictive Control

Lecture 6

主讲人 Jialin Ji

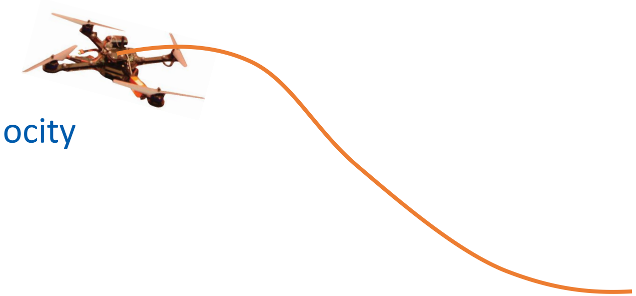
D. Eng. Candidate in Robotics
Zhejiang University

Outline

- Reactive Control & Optimal Control → strengths and weakness
- What is Model Predictive Control (MPC)? Why MPC?
- Brief history of MPC & research using MPC in robotics
- MPC design → pipeline and parameters choices
- Linear MPC → quadratic programming & MPC and LQR
- Explicit MPC
- Linear Time-Varying MPC and Nonlinear MPC
- Tube MPC
- Hybrid MPC
- Model Predictive Contouring Control (MPCC)

Preliminaries

The Control Task: trajectory following for example



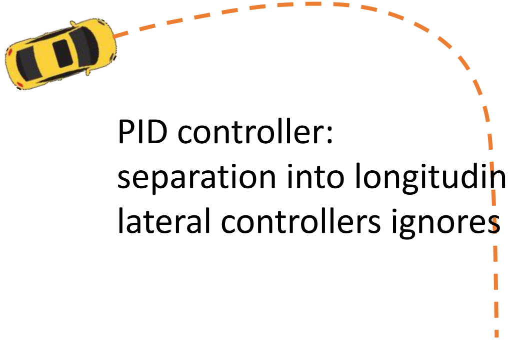
$$\boldsymbol{e}_p = \boldsymbol{p} - \boldsymbol{p}_{des}, \quad \boldsymbol{e}_v = \boldsymbol{v} - \boldsymbol{v}_{des}$$ errors on position and velocity

$$\boldsymbol{F}_{des} = -K_p \boldsymbol{e}_p - K_v \boldsymbol{e}_v + mg\boldsymbol{z}_w + m\boldsymbol{a}_{des}$$ control gains

$$\boldsymbol{z}_{b,des} = \frac{\boldsymbol{F}_{des}}{||\boldsymbol{F}_{des}||}, \quad \boldsymbol{R}_{des}\boldsymbol{e}_3 = \boldsymbol{z}_{b,des}$$ desired thrust and attitude

Limitations of Reactive Control

- Non-trivial for more complex systems
- Control gains must be tuned manually
- No handling of coupled dynamics and constraints
- Ignores future decisions



PID controller:
separation into longitudinal and lateral controllers ignores coupling

Mellinger, Daniel, and Vijay Kumar. "Minimum snap trajectory generation and control for quadrotors." *2011 IEEE international conference on robotics and automation*. IEEE, 2011.

Preliminaries

Optimal Control Procedure

- Dynamic model $$\boldsymbol{x}_{k+1} = f(\underset{\text{state}}{\boldsymbol{x}_k}, \underset{\text{input}}{\boldsymbol{u}_k}), \quad \underset{\text{initial condition}}{\boldsymbol{x}_0}$$

- Objective function $$\min_{U_N} \sum_{k=0}^{N-1} \underbrace{q(\boldsymbol{x}_k, \boldsymbol{u}_k)}_{\text{stage cost}} + \underbrace{p(\boldsymbol{x}_N)}_{\text{terminal cost}}$$

- Optimal solution vector $$\boldsymbol{z}^* = [\boldsymbol{u}_0', \ldots, \boldsymbol{u}_{N-1}']'$$ optimizer

Difficulties of Open-Loop Optimal Control

- The dynamic model is usually inaccurate. Model errors accumulate over time.
- Long task-horizons make the problem intractable.
- The system may be affected by external disturbances.

Preliminaries

Model Predictive Control (MPC)

Use a dynamical **model** of the process to **predict** its future evolution and choose the “best” **control** action.

- **Feedback** of the measurement information

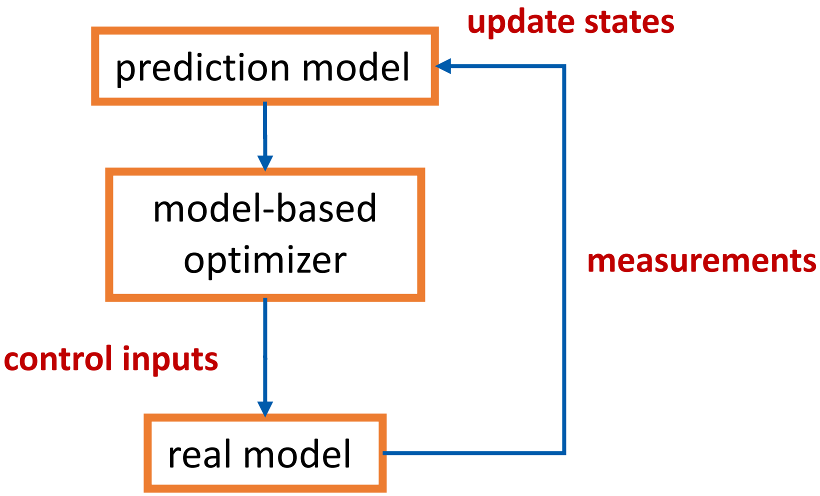
  → Start from the estimated current state

- **Optimize** the best control sequence

  → Find controls for limited preview into the future

- **Receding horizon** framework

  → Apply only the first input, then re-plan



Borrelli, Francesco, Alberto Bemporad, and Manfred Morari. Predictive control for linear and hybrid systems. Cambridge University Press, 2017.

Preliminaries

Model Predictive Control (MPC)

Use a dynamical **model** of the process to **predict** its future evolution and choose the "best" **control** action.

- MPC Formulation (reference tracking for e.g.)

$$\min_{u_0,u_1,\ldots,u_N} \sum_{k}^{N} ||\boldsymbol{y}_k - \boldsymbol{r}(t)||^2 + \rho\Delta\boldsymbol{u}_k^2$$

s. t.

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k)$$
$$\boldsymbol{y}_k = g(\boldsymbol{x}_k)$$

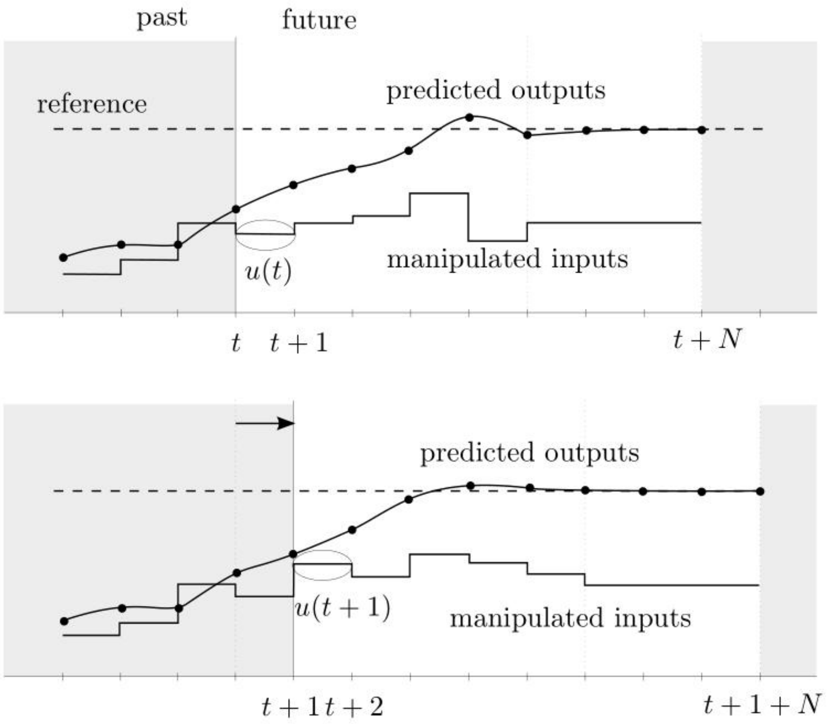
prediction model

$$\boldsymbol{x}_{min} \le \boldsymbol{x}_k \le \boldsymbol{x}_{max}$$
$$\boldsymbol{u}_{min} \le \boldsymbol{u}_k \le \boldsymbol{u}_{max}$$

constraints

$$\boldsymbol{x}_0 = \boldsymbol{x}(t)$$

state feedback



Borrelli, Francesco, Alberto Bemporad, and Manfred Morari. Predictive control for linear and hybrid systems. Cambridge University Press, 2017.

Preliminaries

Model Predictive Control (MPC)

Other names:

- Open Loop Optimal Feedback
- Reactive Scheduling
- Receding Horizon Control

Advantages

- Accounts for errors
- Reduces problem size (solver is usually warm-started with the previous solution)

Preliminaries

Brief History of MPC

- Process control → linear MPC (some nonlinear too) — 1970-2000
- Automotive control → explicit, hybrid MPC — 2001-2010
- Aerospace systems and UAVs → linear time-varying MPC — >2005
- Information and Communication Technologies (ICT) (wireless nets, cloud) → distributed/decentralized MPC — >2005
- Energy, finance, automotive, water → stochastic MPC — >2010
- Industrial production → embedded optimization solvers for MPC — >2010
- Machine learning → data-driven MPC — today

Preliminaries

Research of MPC in Robotics

Nonlinear MPC for Quadrotor Fault-Tolerant Control

Whole-body MPC and online gait sequence generation for wheeled-legged robots

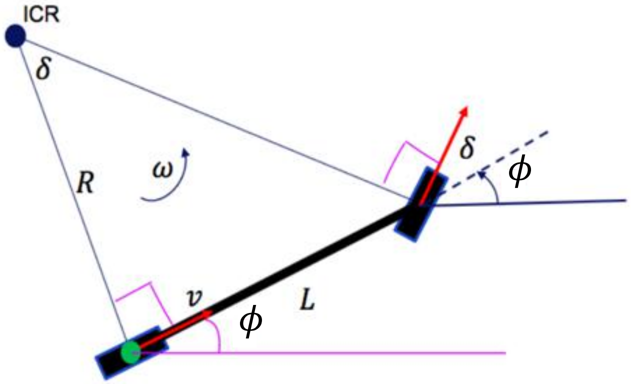
Nan, Fang, et al. "Nonlinear MPC for Quadrotor Fault-Tolerant Control." arXiv preprint arXiv:2109.12886 (2021).

Bjelonic, Marko, et al. "Whole-body mpc and online gait sequence generation for wheeled-legged robots." 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020.

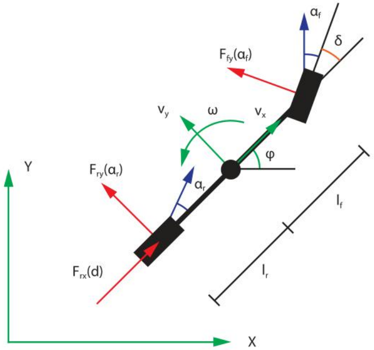
MPC Design

Design Parameters of MPC

- Prediction model → trade-off in choice of model family:



$$\dot{x} = v\cos(\phi)$$
$$\dot{y} = v\sin(\phi)$$
$$\dot{\phi} = \frac{v}{L}\tan(\delta)$$
$$\dot{v} = a$$



$$\dot{x} = v_x\cos(\varphi) - v_y\sin(\varphi)$$
$$\dot{y} = v_x\sin(\varphi) + v_y\cos(\varphi)$$
$$\dot{\varphi} = \omega$$
$$\dot{v}_x = \frac{1}{m}\left(F_{r,x} - F_{f,y}\sin(\delta) + m v_y \omega\right)$$
$$\dot{v}_y = \frac{1}{m}\left(F_{r,y} + F_{f,y}\cos(\delta) - m v_x \omega\right)$$
$$\dot{\omega} = \frac{1}{I_z}\left(F_{f,y} l_f \cos(\delta) - F_{r,y} l_r\right)$$

Kinematic Bicycle Model

Dynamic Bicycle Model

Simplicity

vs.

Accuracy

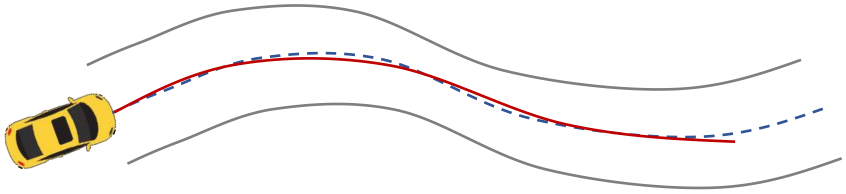
Kong, Jason, et al. "Kinematic and dynamic vehicle models for autonomous driving control design." 2015 IEEE intelligent vehicles symposium (IV). IEEE, 2015.

Liniger, Alexander, Alexander Domahidi, and Manfred Morari. "Optimization-based autonomous racing of 1: 43 scale RC cars." Optimal Control Applications and Methods 36.5 (2015): 628-647.

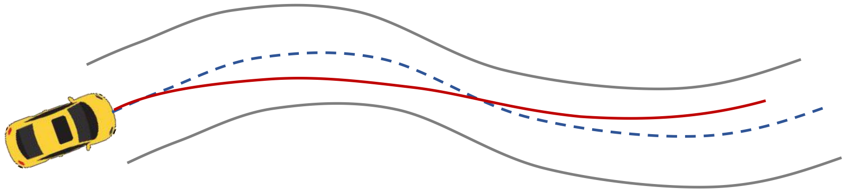
MPC Design

Design Parameters of MPC

- Prediction model → trade-off of accuracy and simplicity
- Cost function → vary in different requirements



Minimize deviation from reference

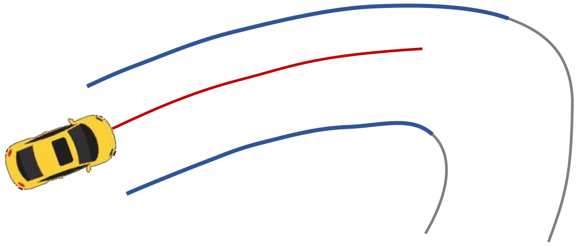


Maximize process inside track bounds

MPC Design

Design Parameters of MPC

- Prediction model → trade-off of accuracy and simplicity
- Cost function → vary in different requirements
- Prediction horizon → trade-off of computation overload and recursive feasibility



- Short prediction horizon
  - Reduced computation
  - Myopic behavior (potentially unsafe)
- Additional constraints at the end of the prediction horizon can ensure recursive feasibility.

MPC Design

Design Parameters of MPC

- Prediction model → trade-off of accuracy and simplicity
- Cost function → vary in different requirements
- Prediction horizon
- Terminal constraints
- …

→ trade-off of computation overload and recursive feasibility

Linear MPC – Unconstrained Case

- Linear prediction model: $\begin{cases} \boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k \\ \boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k \end{cases}$

- Relation between input and states: $$\boldsymbol{x}_k = \boldsymbol{A}^k\boldsymbol{x}_0 + \sum_{j=0}^{k-1} \boldsymbol{A}^j\boldsymbol{B}\boldsymbol{u}_{k-1-j}$$
- Quadratic costs:

$$J = \boldsymbol{x}_N'\boldsymbol{P}\boldsymbol{x}_N + \sum_{k=0}^{N-1} (\boldsymbol{x}_k'\boldsymbol{Q}\boldsymbol{x}_k + \boldsymbol{u}_k'\boldsymbol{R}\boldsymbol{u}_k)$$

$P = P' \succ 0$
$Q = Q' \succ 0$
$R = R' \succ 0$

Positive semi-definite

- Goal: find the best control sequence $u_{0:N-1}^*$ that minimizes J

Linear MPC – Unconstrained Case

$$J(\mathbf{z}, \mathbf{x}_0) = \mathbf{x}_0' Q \mathbf{x}_0 + \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{N-1} \\ \mathbf{x}_N \end{bmatrix}' \overbrace{\begin{bmatrix} Q & 0 & 0 & \cdots & 0 \\ 0 & Q & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & Q & 0 \\ 0 & 0 & \cdots & 0 & P \end{bmatrix}}^{\bar{Q}} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{N-1} \\ \mathbf{x}_N \end{bmatrix} + \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N-1} \end{bmatrix}' \overbrace{\begin{bmatrix} R & 0 & \cdots & 0 \\ 0 & R & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & R \end{bmatrix}}^{\bar{R}} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N-1} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_{N-1} \\ \mathbf{x}_N \end{bmatrix} = \underbrace{\begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix}}_{\bar{S}} \underbrace{\begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N-1} \end{bmatrix}}_{\mathbf{z}} + \underbrace{\begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}}_{\bar{T}} \mathbf{x}_0$$

$$J(\mathbf{z}, \mathbf{x}_0) = \left(\bar{S}\mathbf{z} + \bar{T}\mathbf{x}_0\right)' \bar{Q} \left(\bar{S}\mathbf{z} + \bar{T}\mathbf{x}_0\right) + \mathbf{z}' \bar{R} \mathbf{z} + \mathbf{x}_0' Q \mathbf{x}_0$$

$$= \frac{1}{2} \mathbf{z}' \underbrace{2\left(\bar{R} + \bar{S}' \bar{Q}\, \bar{S}\right)}_{H} \mathbf{z} + \mathbf{x}_0' \underbrace{2\bar{T}' \bar{Q}\, \bar{S}}_{F} \mathbf{z} + \frac{1}{2} \mathbf{x}_0' \underbrace{2\left(Q + \bar{T}' \bar{Q}\, \bar{T}\right)}_{Y} \mathbf{x}_0$$

Linear MPC – Unconstrained Case

$$J(\boldsymbol{z},\boldsymbol{x}_0)=\frac{1}{2}\boldsymbol{z}'\underbrace{2\left(\overline{\boldsymbol{R}}+\overline{\boldsymbol{S}}'\,\overline{\boldsymbol{Q}}\,\overline{\boldsymbol{S}}\right)}_{\boldsymbol{H}}\boldsymbol{z}+\boldsymbol{x}_0'\underbrace{2\overline{\boldsymbol{T}}'\,\overline{\boldsymbol{Q}}\,\overline{\boldsymbol{S}}}_{\boldsymbol{F}}\boldsymbol{z}+\frac{1}{2}\boldsymbol{x}_0'\underbrace{2\left(\boldsymbol{Q}+\overline{\boldsymbol{T}}'\,\overline{\boldsymbol{Q}}\,\overline{\boldsymbol{T}}\right)}_{\boldsymbol{Y}}\boldsymbol{x}_0$$

- Condensed form of MPC:

$$J(\boldsymbol{z},\boldsymbol{x}_0)=\frac{1}{2}\boldsymbol{z}'\boldsymbol{H}\boldsymbol{z}+\boldsymbol{x}_0'\boldsymbol{F}\boldsymbol{z}+\frac{1}{2}\boldsymbol{x}_0'\boldsymbol{Y}\boldsymbol{x}_0 \qquad \boldsymbol{z}=\begin{bmatrix}\boldsymbol{u}_0\\ \boldsymbol{u}_1\\ \vdots\\ \boldsymbol{u}_{N-1}\end{bmatrix}$$

The optimum is obtained by zeroing the gradient

$$\nabla_{\boldsymbol{z}}J(\boldsymbol{z},\boldsymbol{x}_0)=\boldsymbol{H}\boldsymbol{z}+\boldsymbol{F}'\boldsymbol{x}_0=0\rightarrow\boldsymbol{z}^*=-\underline{\boldsymbol{H}^{-1}}\boldsymbol{F}'\boldsymbol{x}_0$$ (“batch” solution)

unconstrained linear MPC = linear state-feedback!

Wang, Yang, and Stephen Boyd. "Fast model predictive control using online optimization." IEEE Transactions on control systems technology 18.2 (2009): 267-278.

Linear MPC – Unconstrained Case

- Non-condensed form of MPC:
  - Keep also $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ as optimization variables
  - Equality constraints

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k$$

$$\boldsymbol{z} = \begin{bmatrix} \boldsymbol{u}_0 \\ \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_{N-1} \\ \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_N \end{bmatrix}$$

More variables and constraints but very sparse

Wang, Yang, and Stephen Boyd. "Fast model predictive control using online optimization." IEEE Transactions on control systems technology 18.2 (2009): 267-278.

MPC and Linear Quadratic Regulation (LQR)

- “Batch” solution: $\boldsymbol{z}^* = -\boldsymbol{H}^{-1}\boldsymbol{F}'\boldsymbol{x}_0$
  - Analytically
  - Still requires to invert a large matrix $\boldsymbol{H}^{-1}$

- Dynamic programming LQR
  - Exploiting the sequential structure of the problem
  - Bellman’s principle of optimality A → … → B → … → C

" An optimal policy has the property that, regardless of the decisions taken to enter a particular state, the remaining decisions made for leaving that stage must constitute an optimal policy."

MPC and Linear Quadratic Regulation (LQR)

$$J = \boldsymbol{x}_N' \boldsymbol{P} \boldsymbol{x}_N + \sum_{k=0}^{N-1} (\boldsymbol{x}_k' \boldsymbol{Q} \boldsymbol{x}_k + \boldsymbol{u}_k' \boldsymbol{R} \boldsymbol{u}_k)$$

Dynamic programming LQR

- Break up the problem into smaller tail problems

$$J_N(\boldsymbol{x}_N) \triangleq \boldsymbol{x}_N' \boldsymbol{P} \boldsymbol{x}_N$$

$$\boldsymbol{u}_{N-1}(\boldsymbol{x}_{N-1}) \triangleq \arg\min_{\boldsymbol{u}_{N-1}} \overbrace{\boldsymbol{x}'_{N-1} \boldsymbol{Q} \boldsymbol{x}_{N-1} + \boldsymbol{u}'_{N-1} \boldsymbol{R} \boldsymbol{u}_{N-1}}^{\text{immediate cost}} + \overbrace{J_N^*(\boldsymbol{A}\boldsymbol{x}_{N-1} + \boldsymbol{B}\boldsymbol{u}_{N-1})}^{\text{cost-to-go}}$$

$$\triangleq \arg\min_{\boldsymbol{u}_{N-1}} \underbrace{\boldsymbol{x}'_{N-1} (\boldsymbol{A}'\boldsymbol{P}\boldsymbol{A} + \boldsymbol{Q}) \boldsymbol{x}_{N-1}}_{\text{Const.}} + \underline{\boldsymbol{u}'_{N-1} (\boldsymbol{B}'\boldsymbol{P}\boldsymbol{B} + \boldsymbol{R}) \boldsymbol{u}_{N-1} + 2\boldsymbol{x}'_{N-1} \boldsymbol{A}'\boldsymbol{P}\boldsymbol{B} \boldsymbol{u}_{N-1}}$$

- Zoroing the gradient: $2(\boldsymbol{B}'\boldsymbol{P}\boldsymbol{B} + \boldsymbol{R})\boldsymbol{u}_{N-1}^* + 2\boldsymbol{B}'\boldsymbol{P}\boldsymbol{A}\boldsymbol{x}_{N-1} = 0$

$$\boldsymbol{u}_{N-1}^*(\boldsymbol{x}_{N-1}) = \boldsymbol{K}\boldsymbol{x}_{N-1} \qquad \boldsymbol{K} = -(\boldsymbol{B}'\boldsymbol{P}\boldsymbol{B} + \boldsymbol{R})^{-1} \boldsymbol{B}'\boldsymbol{P}\boldsymbol{A}$$

$$J_{N-1}(\boldsymbol{x}_{N-1}) \triangleq \boldsymbol{x}'_{N-1} \{\underline{\boldsymbol{A}'\boldsymbol{P}\boldsymbol{A} + \boldsymbol{Q} + \boldsymbol{A}'\boldsymbol{P}\boldsymbol{B}\boldsymbol{K}}\} \boldsymbol{x}_{N-1}$$

solving the Algebraic Riccati Equation

MPC and Linear Quadratic Regulation (LQR)

Dynamic programming LQR

- Consider again the MPC cost function

$$\min_{z} \; \boldsymbol{x}_N' \boldsymbol{P} \boldsymbol{x}_N + \sum_{k=0}^{N-1} (\boldsymbol{x}_k' \boldsymbol{Q} \boldsymbol{x}_k + \boldsymbol{u}_k' \boldsymbol{R} \boldsymbol{u}_k)$$

$$\text{s.t.} \quad \boldsymbol{u}_k = \boldsymbol{K} \boldsymbol{x}_k$$

- Update matrix $\boldsymbol{P}$ and terminal gain $\boldsymbol{K}$ iteratively

$$\boldsymbol{K} = -(\boldsymbol{B}'\boldsymbol{P}\boldsymbol{B} + \boldsymbol{R})^{-1}\boldsymbol{B}'\boldsymbol{P}\boldsymbol{A}$$

$$\boldsymbol{P} = \boldsymbol{A}'\boldsymbol{P}\boldsymbol{A} + \boldsymbol{Q} + \boldsymbol{A}'\boldsymbol{P}\boldsymbol{B}\boldsymbol{K}$$

→ Linear complexity in horizon N

(unconstrained) MPC = LQR for any choice of the prediction horizon N

Constrained Linear MPC

Convex Quadratic Program (QP)

$$J(\boldsymbol{z}, \boldsymbol{x}_0) = \frac{1}{2}\boldsymbol{z}'\boldsymbol{H}\boldsymbol{z} + \boldsymbol{x}_0'\boldsymbol{F}\boldsymbol{z} + \frac{1}{2}\boldsymbol{x}_0'\boldsymbol{Y}\boldsymbol{x}_0$$

$$\min_{\boldsymbol{z}} \quad \frac{1}{2}\boldsymbol{z}'\boldsymbol{H}\boldsymbol{z} + \boldsymbol{x}_0'\boldsymbol{F}\boldsymbol{z} \qquad \text{(quadratic objective)}$$

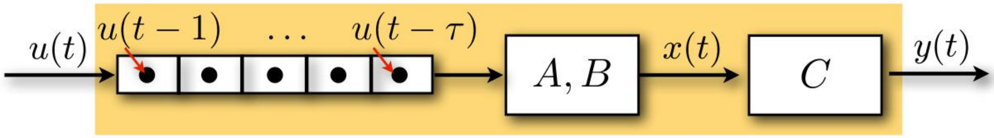
$$\text{s.t.} \quad \boldsymbol{G}\boldsymbol{z} \leq \boldsymbol{W} + \boldsymbol{S}\boldsymbol{x}_0 \qquad \text{(linear constraints)}$$

- Popular noncommercial QP solvers
  - OOQP, OSQP, qpOASES, ECOS (SOCP)
- Popular Commercial QP solvers
  - GUROBI, MOSEK (LPs, QPs, SOCPs, SDPs and MIPs …)

Linear MPC with Delays

- Linear model with delays

$$\boldsymbol{x}(t+1) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t-\tau)$$



- Delay-free model:

$$\bar{\boldsymbol{x}}_0 = \boldsymbol{x}(t+\tau) \approx \hat{\boldsymbol{x}}(t+\tau) = \boldsymbol{A}^{\tau}\boldsymbol{x}(t) + \sum_{j=0}^{\tau-1} \boldsymbol{A}^{j}\boldsymbol{B}\underbrace{\boldsymbol{u}(t-i-j)}_{\text{Past inputs}}$$

- We must have the prediction horizon $N \geq \tau$.

- $\hat{\boldsymbol{x}}(t+\tau)$ can be computed by a more complex model (numerical solution of ODE).

Explicit MPC

Can we implement constrained linear MPC **without an online QP solver**?

- Online optimization: given $\boldsymbol{x}(t)$, solve the problem at each time step t

  (the control law $\boldsymbol{u} = \boldsymbol{u}_0^*(\boldsymbol{x})$ is implicitly defined by the QP solver)

  → Quadratic Programming (QP)

- Offline optimization: solve the QP in advance for all $\boldsymbol{x}(t)$ in a given range to find the control law $\boldsymbol{u} = \boldsymbol{u}_0^*(\boldsymbol{x})$ explicity

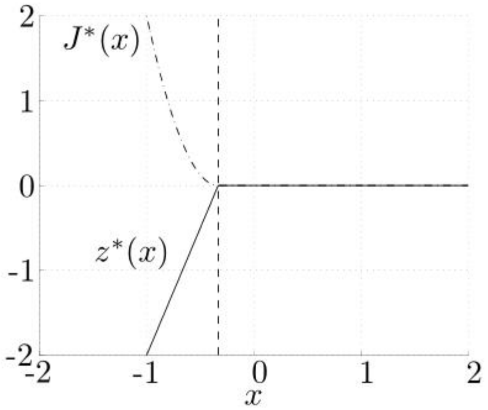
  → Multi-parametric Quadratic Programming (mpQP)

Explicit MPC

Example

$$J^*(x) = \min_z J(z,x) = \frac{1}{2} z^2$$

$$\text{s.t.} \qquad z \le 1 + 3x$$



KKT condition:

$$z + \lambda = 0$$

$$\lambda(z - 3x - 1) = 0$$

$$\lambda \ge 0$$

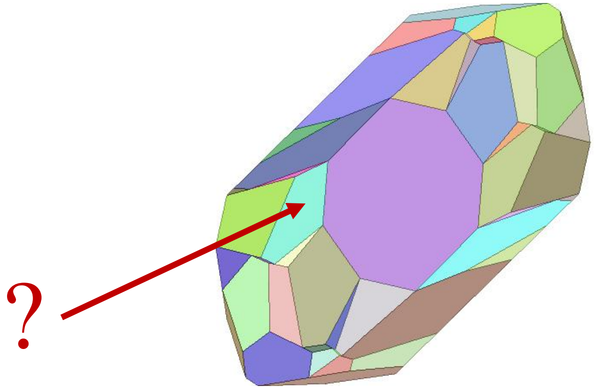
$$z - 3x - 1 \le 0$$

It can be proved that the multiparametric solution of a strictly convex QP is continuous and piecewise affine.

Borrelli, Francesco, Alberto Bemporad, and Manfred Morari. Predictive control for linear and hybrid systems. Cambridge University Press, 2017.

Explicit MPC

The number of regions is (usually) exponential with the number of possible combinations of active constraints.



Too many regions make explicit MPC less attractive, due to memory (storage of polyhedra) and throughput requirements (time to locate x_0).

Borrelli, Francesco, Alberto Bemporad, and Manfred Morari. Predictive control for linear and hybrid systems. Cambridge University Press, 2017.

Linear Time-Varying MPC and Nonlinear MPC

- Linear Time-Varying (LTV) model

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k \rightarrow \boldsymbol{x}_{k+1} = \underline{\boldsymbol{A}_k(t)}\boldsymbol{x}_k + \underline{\boldsymbol{B}_k(t)}\boldsymbol{u}_k$$

- Nonlinear model

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$$

- An LTV model can be obtained by linearizing a nonlinear model

$$\dot{\boldsymbol{x}} \approx f(\overline{\boldsymbol{x}}, \overline{\boldsymbol{u}}) + \left.\frac{\partial f}{\partial \boldsymbol{x}}\right|_{\overline{\boldsymbol{x}},\overline{\boldsymbol{u}}} (\boldsymbol{x} - \overline{\boldsymbol{x}}) + \left.\frac{\partial f}{\partial \boldsymbol{u}}\right|_{\overline{\boldsymbol{x}},\overline{\boldsymbol{u}}} (\boldsymbol{u} - \overline{\boldsymbol{u}})$$

$$\dot{\boldsymbol{x}} = \boldsymbol{A}_c\boldsymbol{x} + \boldsymbol{B}_c\boldsymbol{u} + \boldsymbol{g}_c$$

Linear Time-Varying MPC and Nonlinear MPC

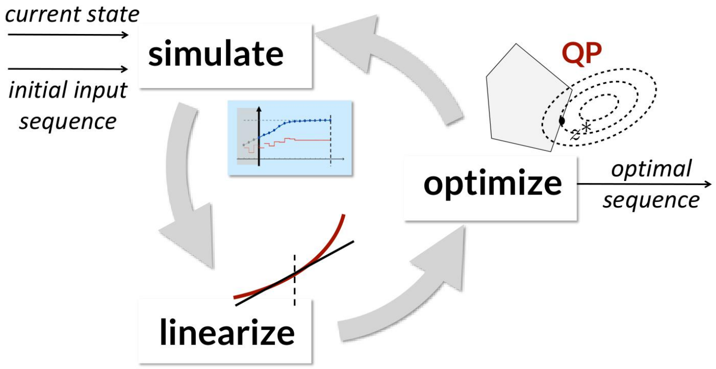
- Convert linear model to discrete-time using forward Euler method

$$\dot{\boldsymbol{x}} = \boldsymbol{A}_c\boldsymbol{x} + \boldsymbol{B}_c\boldsymbol{u} + \boldsymbol{g}_c$$

$$\frac{\boldsymbol{x}_{k+1} - \boldsymbol{x}_k}{T_s} = \boldsymbol{A}_c\boldsymbol{x}_k + \boldsymbol{B}_c\boldsymbol{u}_k + \boldsymbol{g}_c$$

$$\boldsymbol{x}_{k+1} = \underline{(\boldsymbol{I} + T_s\boldsymbol{A}_c)}\boldsymbol{x}_k + \underline{T_s\boldsymbol{B}_c}\boldsymbol{u}_k + \underline{T_s\boldsymbol{g}_c}$$

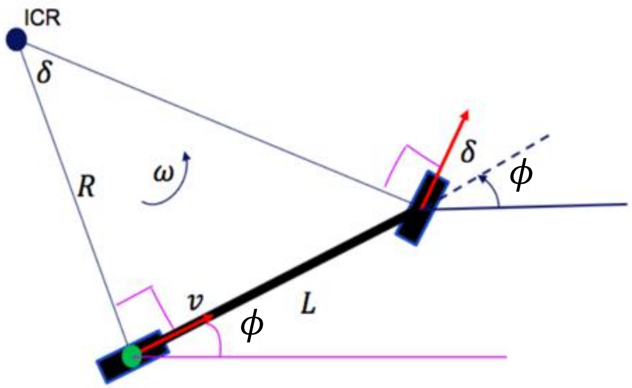
$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{g}_k$$



- Then we can solve a linear MPC online.

Linear Time-Varying MPC and Nonlinear MPC

- Example: Control longitudinal acceleration and steering angle of the vehicle simultaneously for autonomous driving of tracking a reference trajectory



$$\begin{cases} \dot{p_x} = v\cos(\phi) \\ \dot{p_y} = v\sin(\phi) \\ \dot{\phi} = \frac{v}{L}\tan(\delta) \\ \dot{v} = a \end{cases}$$

Input: $\begin{bmatrix} a \\ \delta \end{bmatrix}$

State: $\begin{bmatrix} p_x \\ p_y \\ v \\ \phi \end{bmatrix}$

(p_x, p_y) Cartesian position of rear wheel

ϕ Vehicle orientation

L Vehicle length

v velocity at rear wheel

a longitudinal acceleration

δ steering input

Linear Time-Varying MPC and Nonlinear MPC

- Linearization and discretization:

$$\begin{bmatrix}\dot{p}_x\\ \dot{p}_y\\ \dot{v}\\ \dot{\phi}\end{bmatrix} = \underbrace{\begin{pmatrix}0 & 0 & -\bar{v}\sin\bar{\phi} & \cos\bar{\phi}\\ 0 & 0 & \bar{v}\cos\bar{\phi} & \sin\bar{\phi}\\ 0 & 0 & 0 & \dfrac{\tan\bar{\delta}}{L}\\ 0 & 0 & 0 & 0\end{pmatrix}}_{\boldsymbol{A}_c} \underbrace{\begin{bmatrix}p_x\\ p_y\\ v\\ \phi\end{bmatrix}}_{\boldsymbol{x}} + \underbrace{\begin{pmatrix}0 & 0\\ 0 & 0\\ 0 & \dfrac{\bar{v}}{L}\dfrac{1}{\cos^2\bar{\delta}}\\ 1 & 0\end{pmatrix}}_{\boldsymbol{B}_c} \underbrace{\begin{bmatrix}a\\ \delta\end{bmatrix}}_{\boldsymbol{u}} + \underbrace{\begin{pmatrix}\bar{v}\bar{\phi}\sin\bar{\phi}\\ -\bar{v}\bar{\phi}\cos\bar{\phi}\\ 0\\ -\dfrac{\bar{v}}{L}\dfrac{\bar{\delta}}{\cos^2\bar{\delta}}\end{pmatrix}}_{\boldsymbol{g}_c}$$

$$\boldsymbol{x}_{k+1} = (\boldsymbol{I} + T_s\boldsymbol{A}_c)\boldsymbol{x}_k + T_s\boldsymbol{B}_c\boldsymbol{u}_k + T_s\boldsymbol{g}_c$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{g}_k$$

Linear Time-Varying MPC and Nonlinear MPC

- Stage cost to minimize:

$$(x - x_{ref})^2 + (y - y_{ref})^2 + w_v \underline{\Delta a^2} + w_\delta \underline{\Delta \delta^2}$$

→ QP formulation

- Augmented model:

$$\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{u} \end{bmatrix}_{k+1} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{u} \end{bmatrix}_k + \begin{bmatrix} \boldsymbol{B} \\ \boldsymbol{I} \end{bmatrix} \Delta \boldsymbol{u}_k \qquad z = \begin{bmatrix} \boldsymbol{u}_0 \\ \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_{N-1} \end{bmatrix} \text{ or } z = \begin{bmatrix} \Delta\boldsymbol{u}_0 \\ \Delta\boldsymbol{u}_1 \\ \vdots \\ \Delta\boldsymbol{u}_{N-1} \end{bmatrix}$$

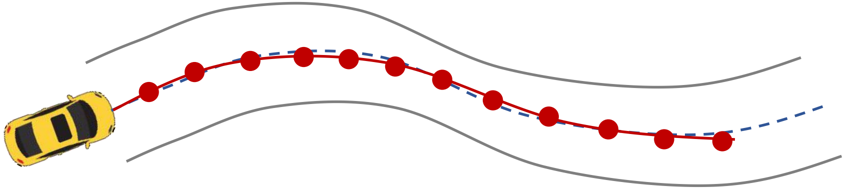
- Constraints on inputs and states:

$$a_{\min} \le a \le a_{\max}$$
$$\delta_{\min} \le \delta \le \delta_{\max}$$
$$v_{\min} \le v \le v_{\max}$$

→ Linear constrained QP formulation

Linear Time-Varying MPC and Nonlinear MPC

- Linearization at which point?



The chicken or the egg

- Some engineering techniques:
  - Artificially given according to the reference trajectory
  - Warm-start: linearize the model at the last solution $u_{k-1}^*,\ x_{k-1}^*$

Can we solve the nonlinear MPC directly?

Linear Time-Varying MPC and Nonlinear MPC

Nonlinear programming problem (NLP) and nonlinear MPC (NMPC)

- (Nonconvex) NLP is harder to solve than QP
- Convergence to a global optimum may not be guaranteed
- Some NLP methods exist
  - Sequential Quadratic Programming (SQP))
  - Interior Point Methods
- Several embedded NMPC solvers exist
  - FORCES Pro (MATLAB C code generator)

Tube MPC

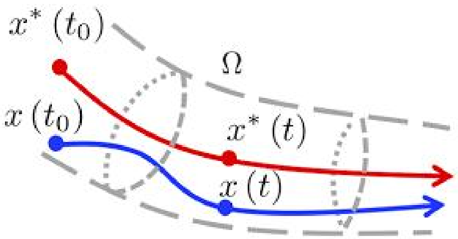
Nonlinear MPC:

- Sometimes it's hard to carry out system identification, especially for nonlinear systems
- Recursive feasibility and stability cannot be guaranteed for complex systems

Tube MPC:

- Use an independent nominal model of the system, and use a feedback controller to ensure the actual state converges to the nominal state.

an ancillary feedback controller is designed to keep the actual state within an invariant "tube" around a nominal trajectory computed neglecting disturbances.



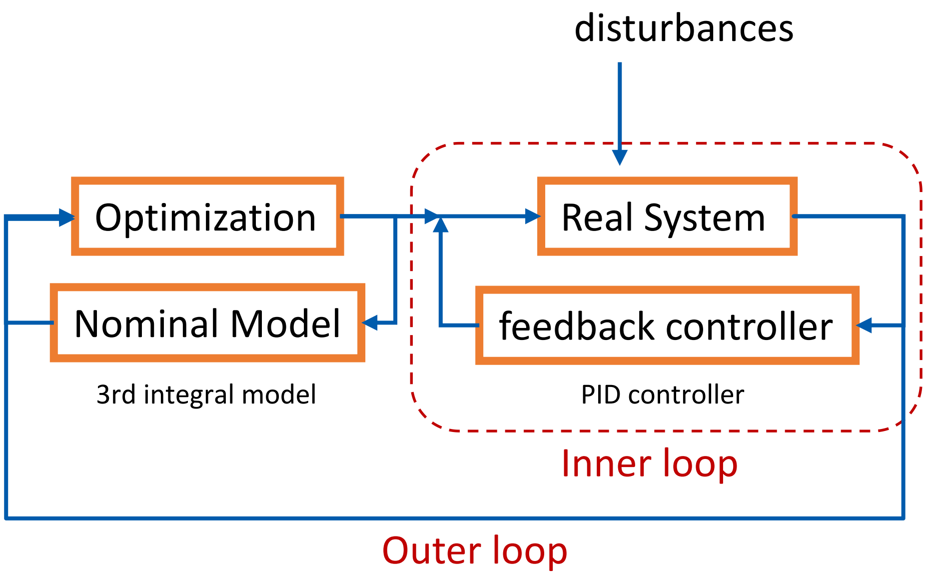
Tube MPC

Tube MPC:

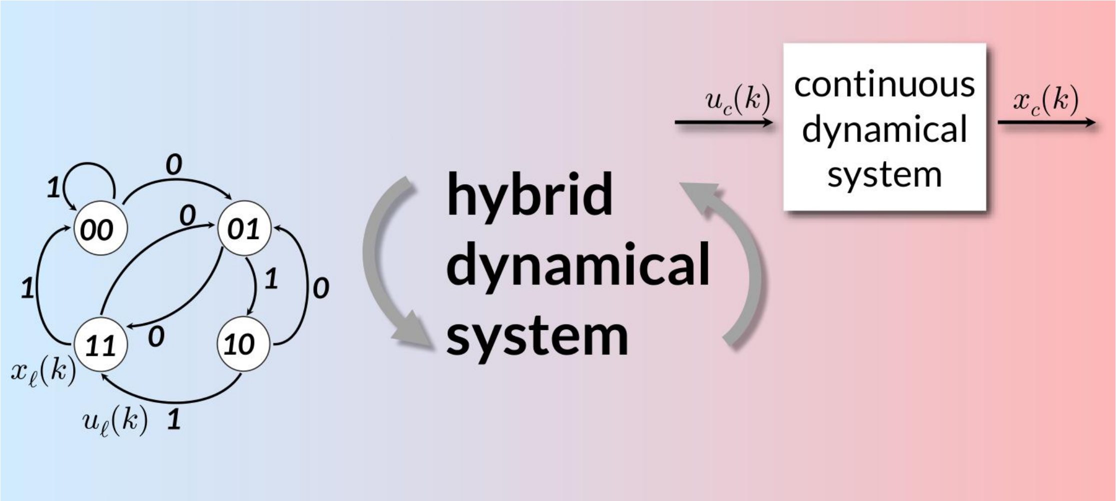
Example: three order integral model

$$\begin{bmatrix} p \\ v \\ a \end{bmatrix}_{k+1} = \begin{bmatrix} 1 & T_s & \frac{1}{2}T_s^2 \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ v \\ a \end{bmatrix}_k + \begin{bmatrix} \frac{1}{6}T_s^3 \\ \frac{1}{2}T_s^2 \\ T_s \end{bmatrix} j_k$$

$p,\ v,\ a,\ j$ denote the position, velocity, acceleration and jerk of the quadrotor.



Hybrid MPC



- Variables are binary-valued

  $x_l \in \{0,1\}^{n_l}$, $u_l \in \{0,1\}^{m_l}$

- Dynamics = finite state machine
- Logic constraints

- Variables are real-valued

  $x_c \in \mathbb{R}^{n_c}$, $u_c \in \mathbb{R}^{m_c}$

- Difference/differential equations
- Linear inequality constraints

Hybrid MPC

- Vehicle

continuous variables
(speed, torque, …)

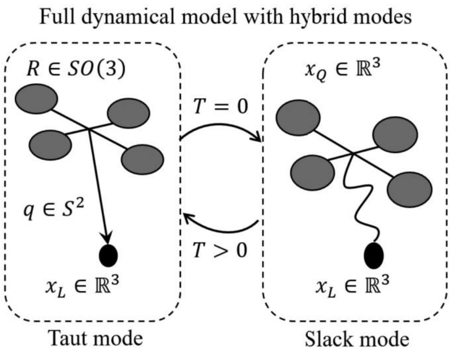
discrete command
(R, N, 1, 2, 3, …)

continuous commands
(brakes & gas pedal)

Hybrid MPC

Mixed Logical Dynamical (MLD) system

- converting logic relations into mixed-integer linear inequalities
- MLD models allow solving MPC, verification, state estimation, and fault detection problems via mixed-integer programming.
- Further equivalences exist with other classes of hybrid dynamical systems, such as Linear Complementarity (LC) systems.

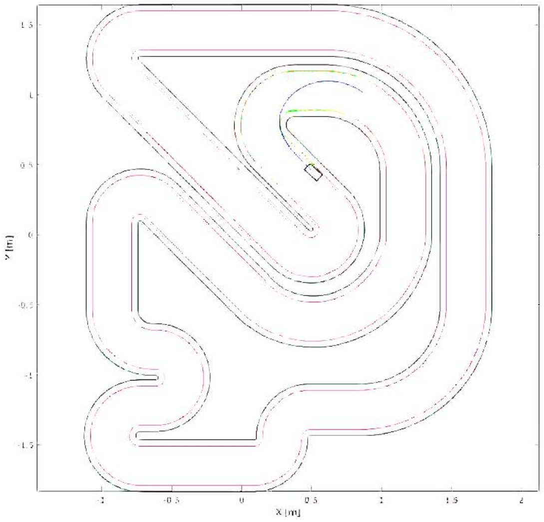


$$-Tq = m_L \ddot{p}_L + m_L g \boldsymbol{e_3}$$

$$T(l - l_0) = 0$$

Model Predictive Contouring Control (MPCC)

Optimization-based autonomous racing of 1: 43 scale RC cars

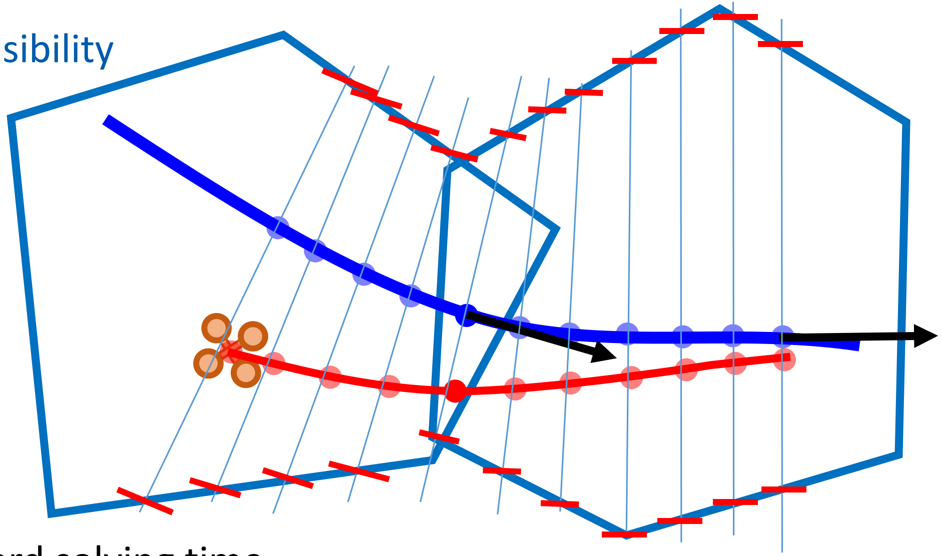


Liniger, Alexander, Alexander Domahidi, and Manfred Morari. "Optimization-based autonomous racing of 1: 43 scale RC cars." Optimal Control Applications and Methods 36.5 (2015): 628-647.

Model Predictive Contouring Control (MPCC)

CMPCC: Corridor-based Model Predictive Contouring Control for Aggressive Drone Flight

- Trade off tracking accuracy and travelling progress
- Terminal speed constraints guarantee feasibility
- Naturally resist external disturbances
- Corridor constraints guarantee safety



- QP formulation with less than 5ms onboard solving time

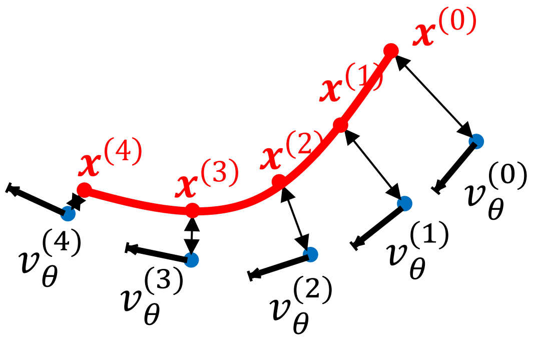
Ji, Jialin, et al. "CMPCC: Corridor-based model predictive contouring control for aggressive drone flight." International Symposium on Experimental Robotics. Springer, Cham, 2020.

Model Predictive Contouring Control (MPCC)

CMPCC: Corridor-based Model Predictive Contouring Control for Aggressive Drone Flight

State: $\boldsymbol{x}^{(k)} = \left[x, v_x, a_x, y, v_y, a_y, z, v_z, a_z, \theta, v_\theta, a_\theta\right]^T$

Input: $\boldsymbol{u}^{(k)} = \left[j_x, j_y, j_z, j_\theta\right]^T$

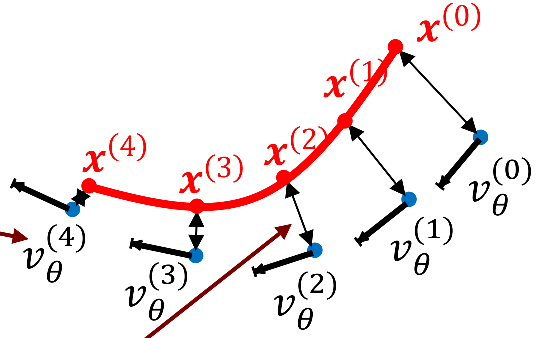


Ji, Jialin, et al. "CMPCC: Corridor-based model predictive contouring control for aggressive drone flight." International Symposium on Experimental Robotics. Springer, Cham, 2020.

Model Predictive Contouring Control (MPCC)

CMPCC: Corridor-based Model Predictive Contouring Control for Aggressive Drone Flight

$$J = \min_{\mathbf{x},\mathbf{u}} \sum_{k=1}^{N} \Big\{ \sum_{\mu=x,y,z} \underbrace{\left(\mu^{(k)} - p_\mu\left(\theta^{(k)}\right)\right)^2}_{\text{Tracking error}} \underbrace{- q \cdot v_\theta^{(k)}}_{\text{Progress}} \Big\}$$



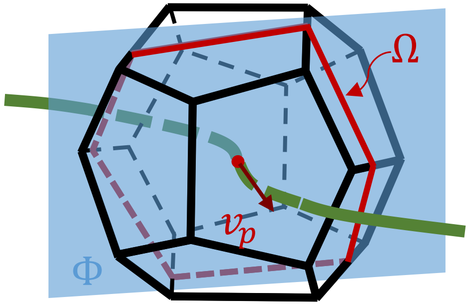
$$\textbf{s.t.}\quad \boldsymbol{x}^{(k+1)} = \boldsymbol{A_d}\boldsymbol{x}^{(k)} + \boldsymbol{B_d}\boldsymbol{u}^{(k)},\ k = 1,2,3,\dots,N-1$$

$$\boldsymbol{x}_l \le \boldsymbol{x}^k \le \boldsymbol{x}_u, k = 1,2,3,\dots,N-1$$

$$\boldsymbol{u}_l \le \boldsymbol{u}^k \le \boldsymbol{u}_u, k = 1,2,3,\dots,N-1$$

$$\boxed{\mathbf{C}^k \cdot \left[\boldsymbol{x}^k, \boldsymbol{y}^k, \boldsymbol{z}^k\right]^T \le \mathbf{b}^k, k = 1,2,3,\dots,N-1}$$

$$\left|v_\mu^{(N)}\right| \le v_{t\mu},\ \mu = x, y, z$$



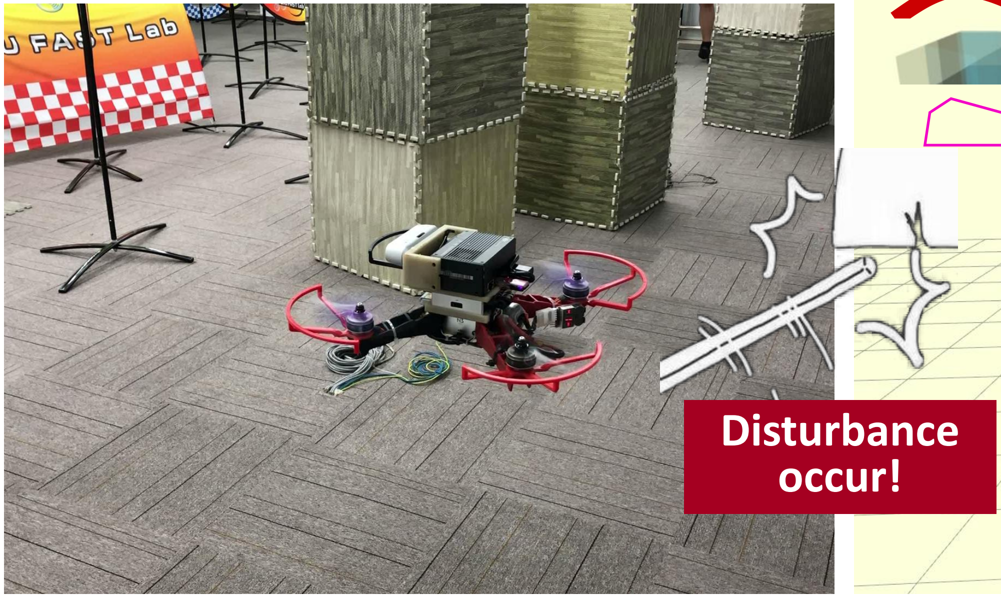
Global trajectory

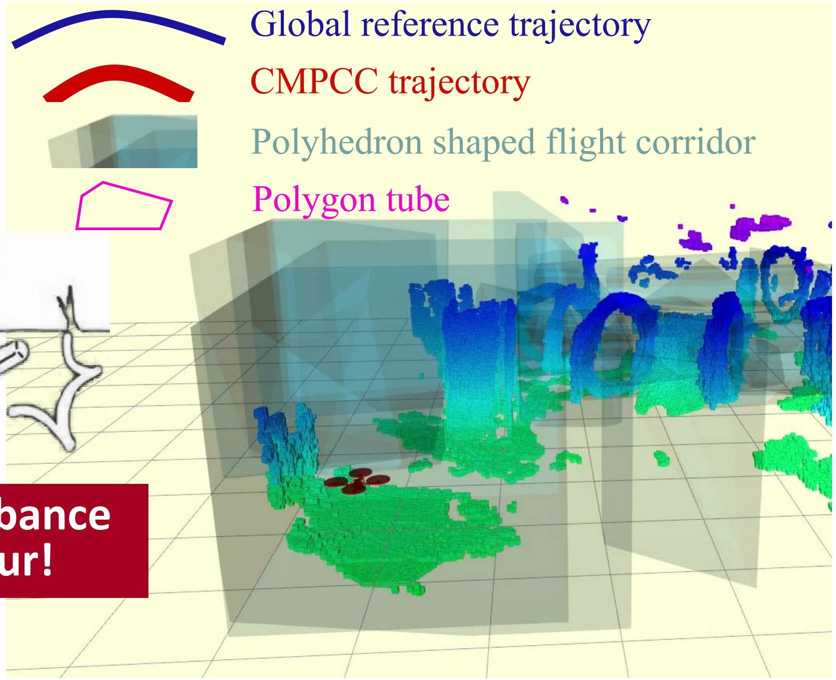
Ji, Jialin, et al. "CMPCC: Corridor-based model predictive contouring control for aggressive drone flight." International Symposium on Experimental Robotics. Springer, Cham, 2020.

Model Predictive Contouring Control (MPCC)

CMPCC: Corridor-based Model Predictive Contouring Control for Aggressive Drone Flight

- Contact disturbance



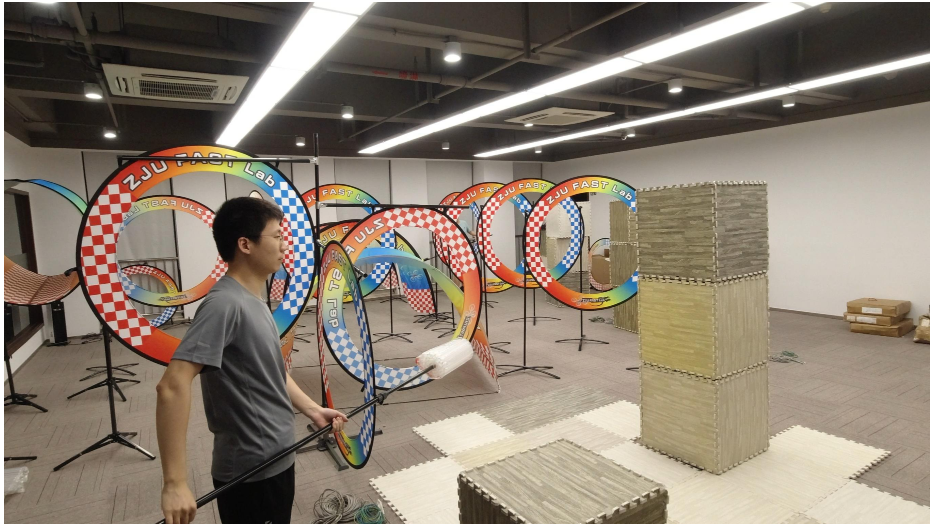


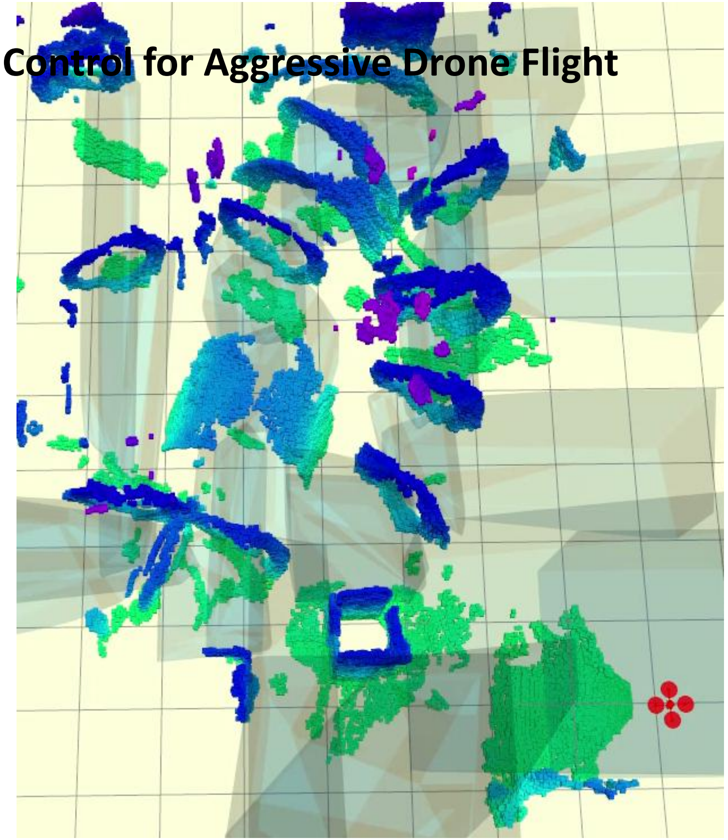
Ji, Jialin, et al. "CMPCC: Corridor-based model predictive contouring control for aggressive drone flight." International Symposium on Experimental Robotics. Springer, Cham, 2020.

Model Predictive Contouring Control (MPCC)

CMPCC: Corridor-based Model Predictive Contouring Control for Aggressive Drone Flight

- Contact disturbance





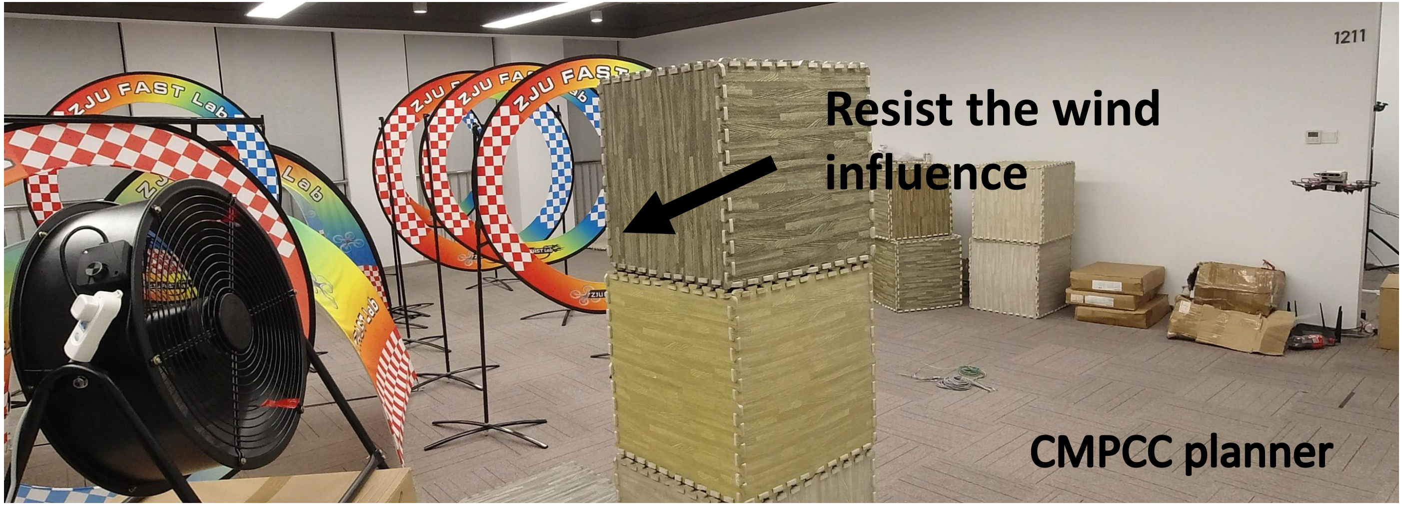
Ji, Jialin, et al. "CMPCC: Corridor-based model predictive contouring control for aggressive drone flight." International Symposium on Experimental Robotics. Springer, Cham, 2020.

Model Pred

CMPCC: Corridor-based Model Predictive Contouring Control for Aggressive Drone Flight

- Wind disturbance

Fan



Summary

- Reactive Control & Optimal Control
- Model Predictive Control → receding horizon
- MPC design
- Linear MPC (unconstrained and constrained case)
- MPC and LQR
- MPC with delays
- Explicit MPC
- Linear Time-Varying MPC and Nonlinear MPC
- Tube MPC
- Hybrid MPC
- Model Predictive Contouring Control

Assignment

1. **Implement MPC of tracking reference trajectory in C++;**
2. **Implement MPC with delays in C++;**
3. **Implement MPCC in C++;**

The framework is ready, only coding for MPC problem formulation is required.

Reference

- http://cse.lab.imtlucca.it/~bemporad/mpc_course.html
- Borrelli, Francesco, Alberto Bemporad, and Manfred Morari. *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.
- Mellinger, Daniel, and Vijay Kumar. "*Minimum snap trajectory generation and control for quadrotors*." 2011 IEEE international conference on robotics and automation. IEEE, 2011.
- Kong, Jason, et al. "*Kinematic and dynamic vehicle models for autonomous driving control design*." 2015 IEEE intelligent vehicles symposium (IV). IEEE, 2015.
- Liniger, Alexander, Alexander Domahidi, and Manfred Morari. "*Optimization-based autonomous racing of 1: 43 scale RC cars*." Optimal Control Applications and Methods 36.5 (2015): 628-647.
- Ji, Jialin, et al. "*CMPCC: Corridor-based model predictive contouring control for aggressive drone flight*." International Symposium on Experimental Robotics. Springer, Cham, 2020.

Thanks for Listening!