

universität freiburg

Model Predictive Control for Nonlinear and Switched Systems

Moritz Diehl¹

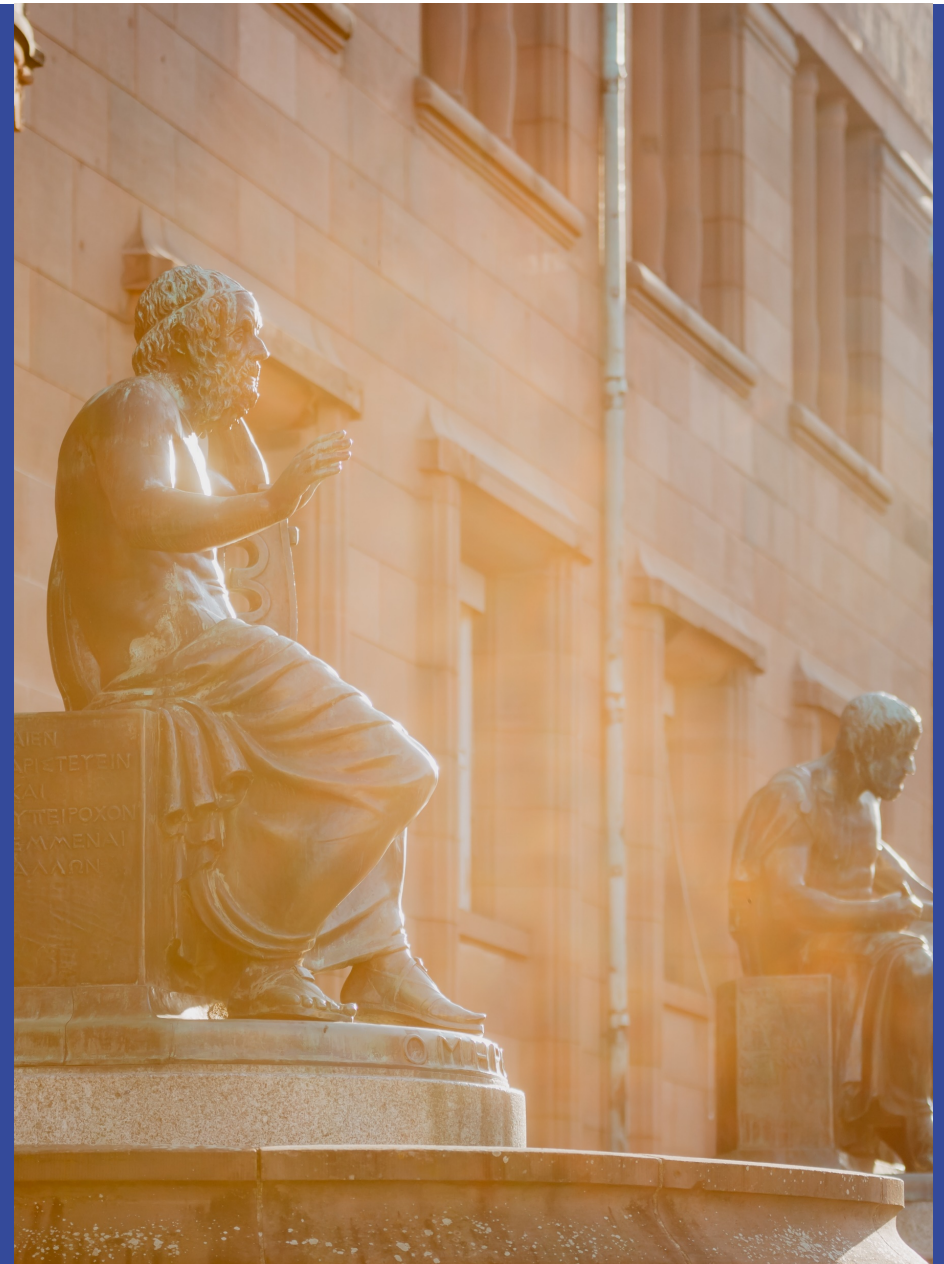
joint work with Armin Nurkanovic¹, Christian Dietz^{1,2}, Anton
Pozharskiy¹, Gianluca Frison^{1,3} Sebastian Albrecht²

¹ Department of Microsystems Engineering and Department
of Mathematics, University of Freiburg, Germany

² Siemens Foundational Technology, Munich, Germany

³ MOSEK ApS, Denmark

NOMICC Summer School Freiburg, Oct. 2, 2026



NMPC needs to solve Nonlinear Programs (NLP)



Continuous Time NMPC Problem

$$\begin{aligned} \min_{x(\cdot), u(\cdot)} \quad & \int_0^T L(x, u) dt + E(x(T)) \\ \text{s.t.} \quad & x(0) = \bar{x}_0 \\ & \dot{x}(t) = f(x(t), u(t)) \\ & 0 \geq h(x(t), u(t)), \quad t \in [0, T] \\ & 0 \geq r(x(T)) \end{aligned}$$

Assume smooth convex L, E, h, r .
Nonlinear f makes problem nonconvex.
Direct methods discretize, then optimize.
E.g. collocation or multiple shooting.

Discretized NMPC Problem (an NLP)

$$\begin{aligned} \min_{x, z, u} \quad & \sum_{k=0}^{N-1} \Phi_L(x_k, z_k, u_k) + E(x_N) \\ \text{s.t.} \quad & x_0 = \bar{x}_0 \\ & x_{k+1} = \Phi_f^{\text{dif}}(x_k, z_k, u_k) \\ & 0 = \Phi_f^{\text{alg}}(x_k, z_k, u_k) \\ & 0 \geq \Phi_h(x_k, z_k, u_k), \quad k = 0, \dots, N-1 \\ & 0 \geq r(x_N) \end{aligned}$$

Again, smooth convex Φ_L, E, Φ_h, r .
Variables $x = (x_0, \dots)$ and $z = (z_0, \dots)$ and $u = (u_0, \dots, u_{N-1})$ can be summarized in vector $w \in \mathbb{R}^{n_w}$.



Nonlinear Programs (NLP) with Convex Structure

Newton-type methods generate a sequence $w_0, w_1, w_2, \dots$ by linearizing and solving convex subproblems. E.g., sequential convex programming (SCP) linearizes nonconvex constraints.

Summarized NLP

$$\begin{aligned} \min_{w \in \mathbb{R}^{n_w}} \quad & J(w) \\ \text{s.t.} \quad & 0 = F(w) \\ & 0 \geq H(w) \end{aligned}$$

Still assume smooth convex J, H .
Nonlinear F makes problem nonconvex.

SCP subproblem at linearization point w_i

$$\begin{aligned} w_{i+1} \in \arg \min_{w \in \mathbb{R}^{n_w}} \quad & J(w) \\ \text{s.t.} \quad & 0 = F_L(w; w_i) \\ & 0 \geq H(w) \end{aligned}$$

First order Taylor series:
 $F_L(w; w_i) := F(w_i) + \frac{\partial F}{\partial w}(w_i)(w - w_i)$

Works extremely well for mildly nonlinear F , also in microsecond NMPC [cf. Zanelli 2021, Lekic 2020, Hausberger 2020]

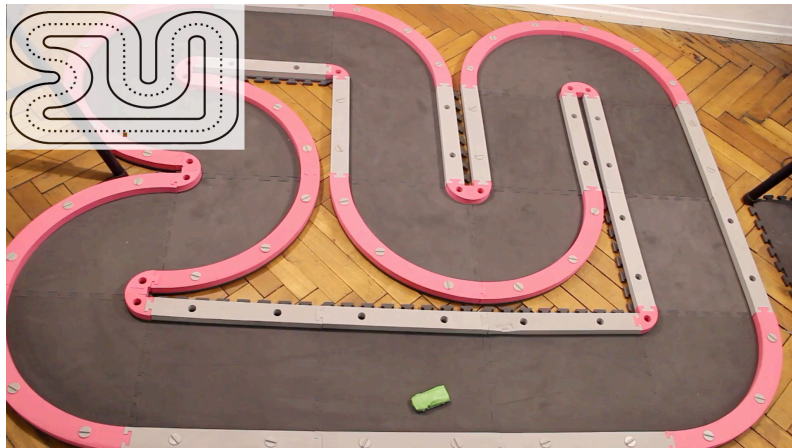
But what if there is significant **nonconvex** structure in the NLP ?

Nonlinear Optimal Control often used for Model Predictive Control (MPC)

One widely used nonlinear MPC package is `acados` [Verscheuren et al. 2021]



Example 1: Autonomous Driving (in Freiburg)



Paper: <https://ieeexplore.ieee.org/abstract/document/9805699>

Example 2: Quadrotor Racing (U Zurich, Scaramuzza)

Video: <https://www.youtube.com/watch?v=zBVpx3bgl6E>

7730

IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 7, NO. 3, JULY 2022

Time-Optimal Online Replanning for Agile Quadrotor Flight

Angel Romero , Robert Penicka , and Davide Scaramuzza

Abstract—In this letter, we tackle the problem of flying a quadrotor using time-optimal control policies that can be replanned online when the environment changes or when encountering unknown disturbances. This problem is challenging as the time-optimal trajectories that consider the full quadrotor dynamics are computationally expensive to generate, on the order of minutes or even hours. We introduce a sampling-based method for efficient generation of time-optimal paths of a point-mass model. These paths are then tracked using a Model Predictive Contouring Control approach that considers the full quadrotor dynamics and the single rotor thrust limits. Our combined approach is able to run in real-time, being the first time-optimal method that is able to adapt to changes *on-the-fly*. We showcase our approach's adaption capabilities by flying a quadrotor at more than 60 km/h in a racing track where gates are moving. Additionally, we show that our online replanning approach can cope with strong disturbances caused by winds of up to 68 km/h.

Index Terms—Aerial systems; Applications, integrated planning and control, motion and path planning.

SUPPLEMENTARY MATERIAL

Video of the experiments: <https://youtu.be/zBVpx3bgl6E>



Fig. 1. The proposed algorithm is able to adapt *on-the-fly* when encountering unknown disturbances. In the figure we show a quadrotor platform flying at speeds of more than 60 km/h. Thanks to our online replanning method, the drone can adapt to wind disturbances of up to 68 km/h while flying as fast as possible.

A. Implementation Details

In order to deploy our MPCC controller, (4) needs to be solved in real-time. To this end, we have implemented our optimization problem using `acados` [24] as a code generation tool, in contrast to [6], where its previous version, `ACADO` [25] was used. It is important to note that for consistency, the optimization problem that is solved online is written in (4) and is exactly the same as in [6]. The main benefit of using `acados` is that it provides an interface to HPIPM (High Performance Interior Point Method) solver [26]. HPIPM solves optimization problems using BLAS-FEO [27], a linear algebra library specifically designed for

Latest `acados` development:
differentiable nonlinear MPC via adjoint approach [Frey et al. 2026, Neurips]

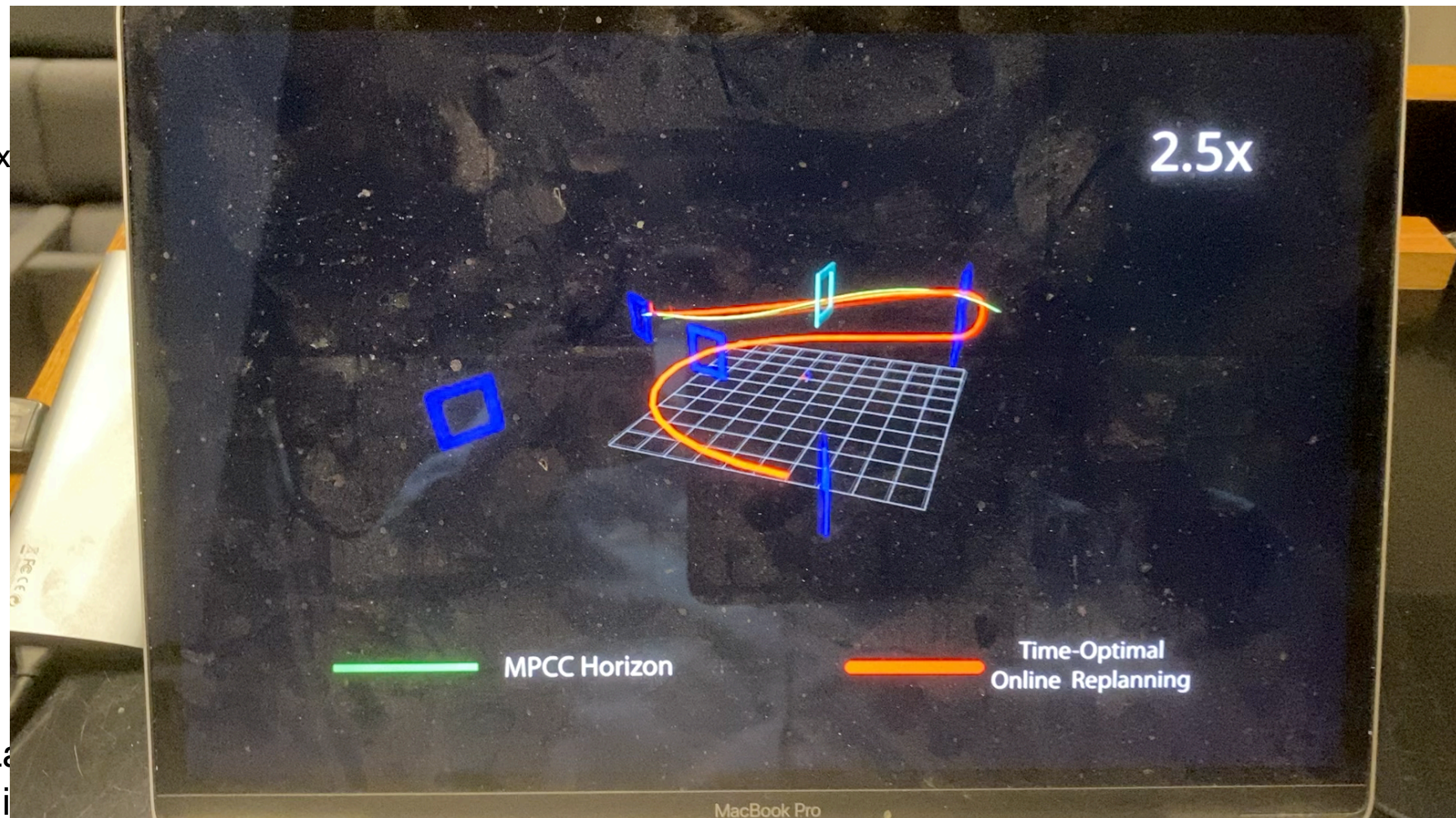
Nonlinear Optimal Control often used for Model Predictive Control (MPC)

One widely used nonlinear MPC package is `acados` [Verscheuren et al. 2021]



Ex

a)



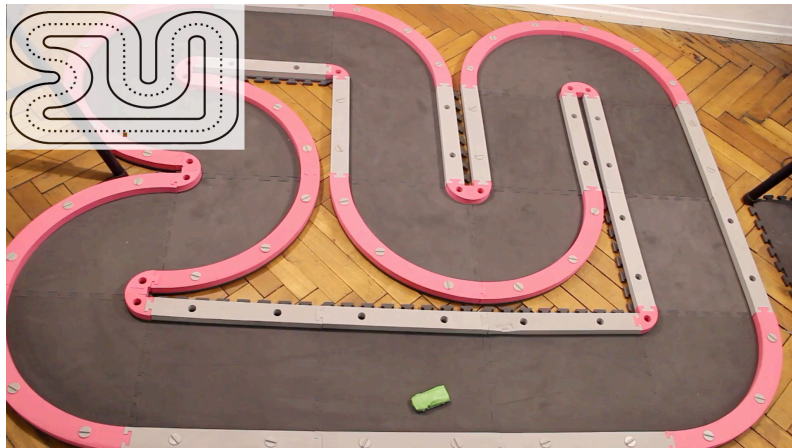
La
di

Nonlinear Optimal Control often used for Model Predictive Control (MPC)

One widely used nonlinear MPC package is `acados` [Verscheuren et al. 2021]



Example 1: Autonomous Driving (in Freiburg)



Latest `acados` development:
differentiable nonlinear MPC via adjoint approach [Frey et al. 2026, Neurips]

Paper: <https://ieeexplore.ieee.org/abstract/document/9805699>

Example 2: Quadrotor Racing (U Zurich, Scaramuzza)

Video: <https://www.youtube.com/watch?v=zBVpx3bgl6E>

7730

IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 7, NO. 3, JULY 2022

Time-Optimal Online Replanning for Agile Quadrotor Flight

Angel Romero , Robert Penicka , and Davide Scaramuzza

Abstract—In this letter, we tackle the problem of flying a quadrotor using time-optimal control policies that can be replanned online when the environment changes or when encountering unknown disturbances. This problem is challenging as the time-optimal trajectories that consider the full quadrotor dynamics are computationally expensive to generate, on the order of minutes or even hours. We introduce a sampling-based method for efficient generation of time-optimal paths of a point-mass model. These paths are then tracked using a Model Predictive Contouring Control approach that considers the full quadrotor dynamics and the single rotor thrust limits. Our combined approach is able to run in real-time, being the first time-optimal method that is able to adapt to changes *on-the-fly*. We showcase our approach's adaption capabilities by flying a quadrotor at more than 60 km/h in a racing track where gates are moving. Additionally, we show that our online replanning approach can cope with strong disturbances caused by winds of up to 68 km/h.

Index Terms—Aerial systems; Applications, integrated planning and control, motion and path planning.

SUPPLEMENTARY MATERIAL

Video of the experiments: <https://youtu.be/zBVpx3bgl6E>

A. Implementation Details

In order to deploy our MPCC controller, (4) needs to be solved in real-time. To this end, we have implemented our optimization problem using `acados` [24] as a code generation tool, in contrast to [6], where its previous version, `ACADO` [25] was used. It is important to note that for consistency, the optimization problem that is solved online is written in (4) and is exactly the same as in [6]. The main benefit of using `acados` is that it provides an interface to HPIPM (High Performance Interior Point Method) solver [26]. HPIPM solves optimization problems using BLAS-FEO [27], a linear algebra library specifically designed for



Fig. 1. The proposed algorithm is able to adapt *on-the-fly* when encountering unknown disturbances. In the figure we show a quadrotor platform flying at speeds of more than 60 km/h. Thanks to our online replanning method, the drone can adapt to wind disturbances of up to 68 km/h while flying as fast as possible.

Dynamic Optimization Problem in NMPC

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

Structured parametric Nonlinear Program

Initial Value $\bar{x}_0$ usually not known beforehand (“online data” in MPC)

Discrete time dynamics often come from ODE/DAE simulation, in direct multiple shooting method [Bock and Plitt, 1984]

Dynamic Optimization Problem in NMPC

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

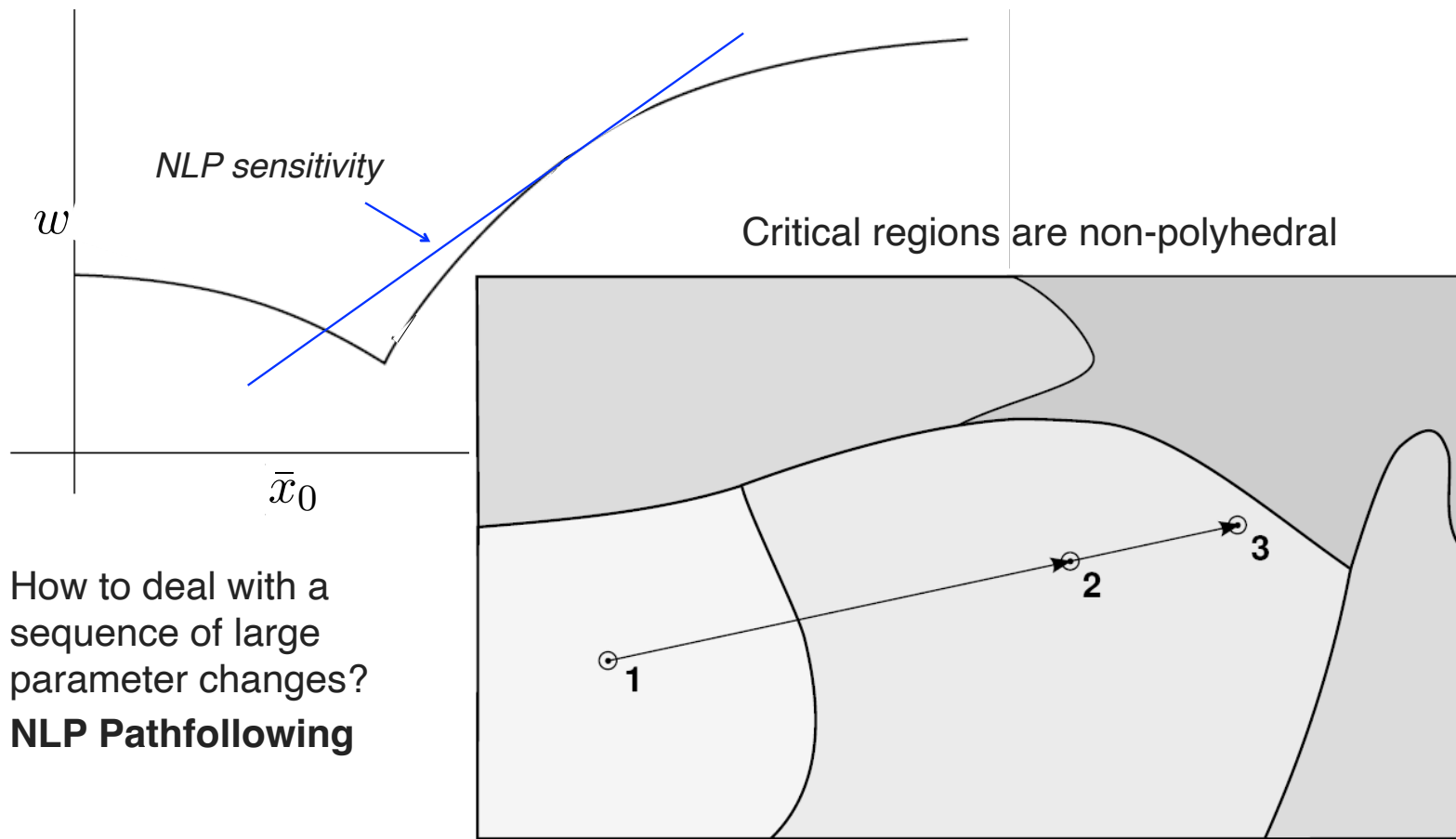
Summarize as

$$\begin{aligned} & \underset{w \in \mathbb{R}^n}{\text{minimize}} && \phi(w) \\ & \text{subject to} && g(w) + M\bar{x}_0 = 0 \\ & && w \in \Omega \end{aligned}$$

with convex ϕ and Ω

Nonlinear MPC = parametric Nonlinear Programming

Solution manifold is piecewise differentiable (kinks at active set changes)



Sequential Convex Programming (SCP) and Real-Time Iteration (RTI)

$\begin{array}{ll} \underset{w \in \mathbb{R}^n}{\text{minimize}} & \phi(w) \\ \text{subject to} & g(w) + M\bar{x}_0 = 0 \\ & w \in \Omega \end{array}$	$\rightarrow$	$\begin{array}{ll} \underset{w \in \mathbb{R}^n}{\text{minimize}} & \phi(w) \\ \text{subject to} & g(w_k) + g'(w_k)(w - w_k) + M \cdot (\bar{x}_0)_{k+1} = 0 \\ & w \in \Omega \end{array}$
---	---------------	---

Repeat each sampling instant k two steps:

Step 1: Linearize nonlinear constraints at w_k to obtain convex problem (right).
(numerical integrations, nonlinear function and derivative evaluations)

Step 2: Obtain new value of parameter $(\bar{x}_0)_{k+1}$ and solve convex problem to
obtain new iterate w_{k+1} (quadratic program solution)

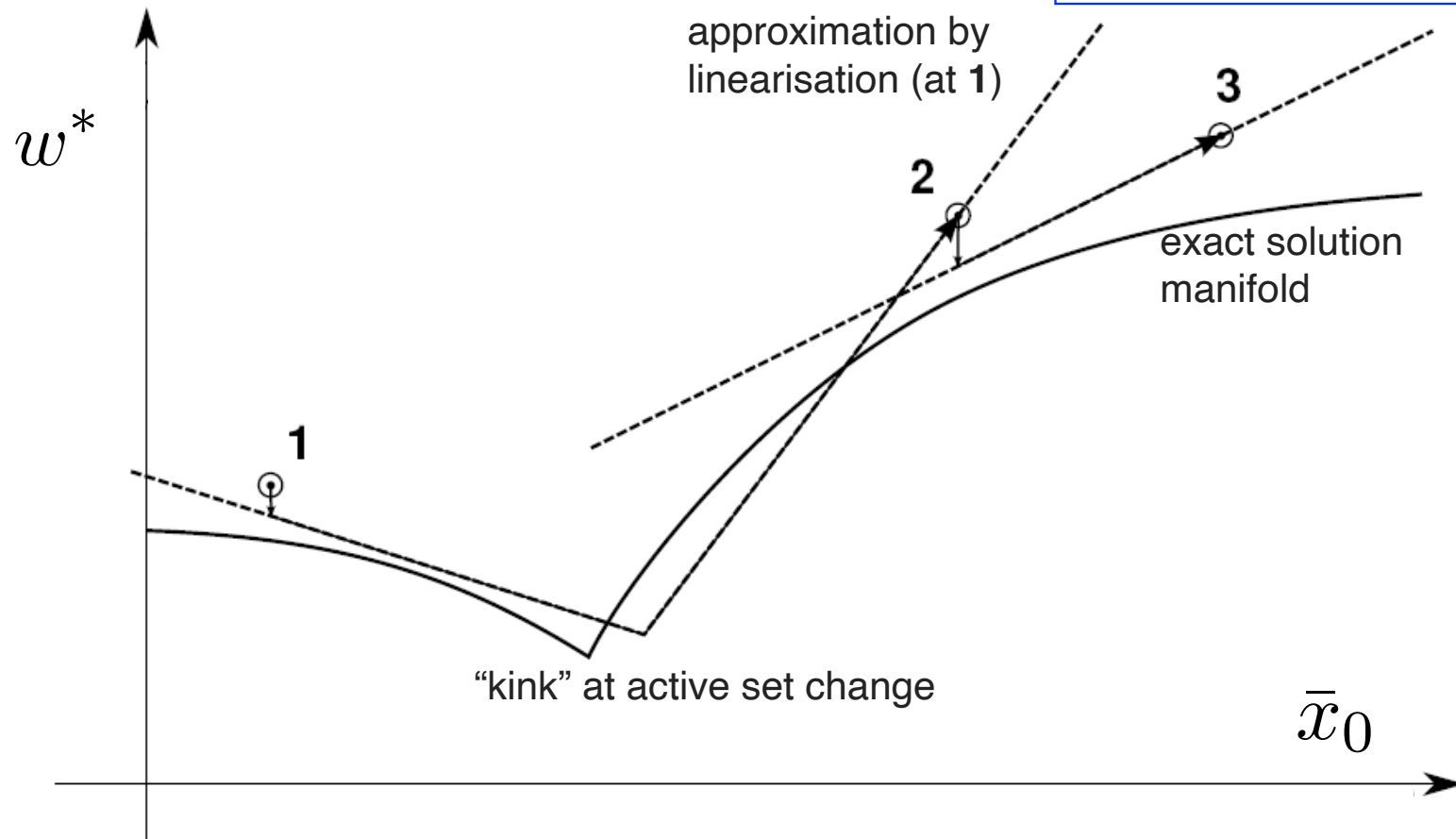
[D., Bock, Schloeder, Findeisen, Nagy, Allgower, JPC, 2002]

[Zavala, Anitescu, SICON, 2010]

[Tran Dinh, Savorgnan, D., SIOPT, 2013]

Real-Time Iteration

$$\begin{array}{ll} \underset{w \in \mathbb{R}^n}{\text{minimize}} & \phi(w) \\ \text{subject to} & g(w) + M\bar{x}_0 = 0 \\ & w \in \Omega \end{array}$$



Tangential prediction even across active set changes

Can divide computations in “preparation” and “feedback phase” [D. 2001]

Real-Time Iterations [PhD Diehl 2001, Heidelberg]

- 1) Keep states in problem - use direct multiple shooting [1]
- 2) Exploit convexity via Generalized Gauss-Newton [2]
- 3) Use tangential predictors for short feedback delay [3]
- 4) Iterate while problem changes (Real-Time Iterations) [4]
- 5) Auto-generate custom solvers in plain-C [5,6,7] (no malloc)

[1] Bock & Plitt, IFAC WC, 1984

[2] Bock 1983

[3] Bock, D. et al, 1999

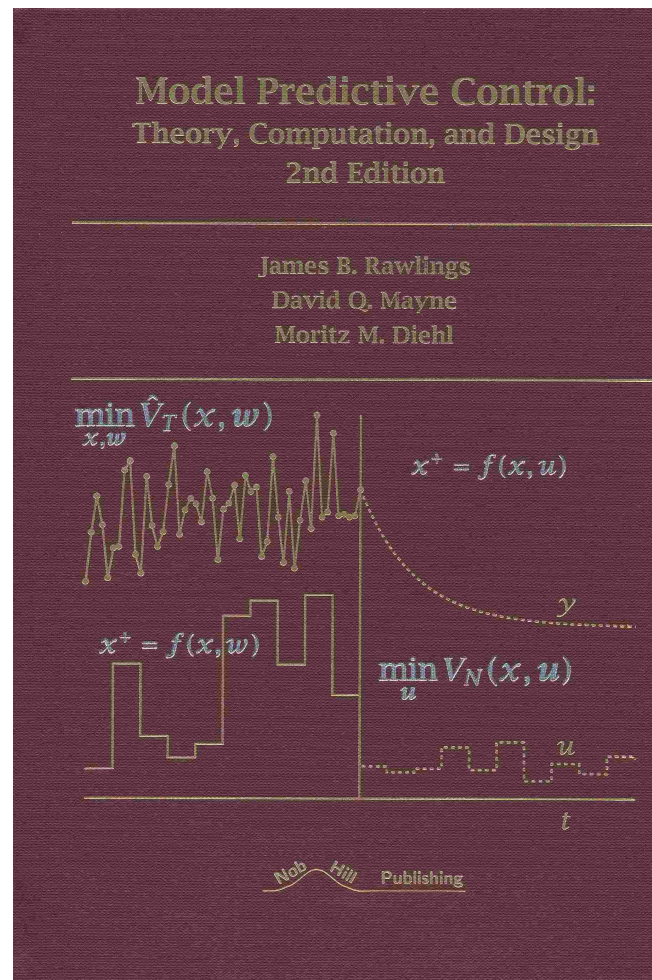
[4] D. et al., 2002 / 2005

[5] Mattingley & Boyd, 2009

[6] Houska et al.: Automatica, 2011

[7] Verschueren et al.: Math.Prog. C, 2022

Also see MPC textbook, Ch. 8 on “Numerical Optimal Control”
(2nd edition, fourth printing, 618 pages, Nob Hill 2022)



Computations in one Real-Time Iteration (RTI)

NLP

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

1) Linearize constraints: Integration & sensitivities

Sparse Quadratic Program (QP)

$$\begin{aligned} & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) &+& E_{\text{QP}}(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\ & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\ & && r' + R x_N \leq 0. \end{aligned}$$

2) Solve sparse QP either

- via condensing, or
- via a Riccati type algorithm, or
- via a combination of both

Computations in one Real-Time Iteration (RTI)

NLP

$$\begin{aligned} & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\ & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\ & && r(x_N) \leq 0. \end{aligned}$$

1) Linearize constraints: Integration & sensitivities

Sparse Quadratic Program (QP)

$$\begin{aligned} & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) &+& E_{\text{QP}}(x_N) \\ & \text{subject to} && x_0 - \bar{x}_0 = 0, \\ & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\ & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\ & && r' + R x_N \leq 0. \end{aligned}$$

2) Solve sparse QP either

- **via condensing**, or
- via a Riccati type algorithm, or
- via a combination of both

OCP-QP Progress 2012-2016

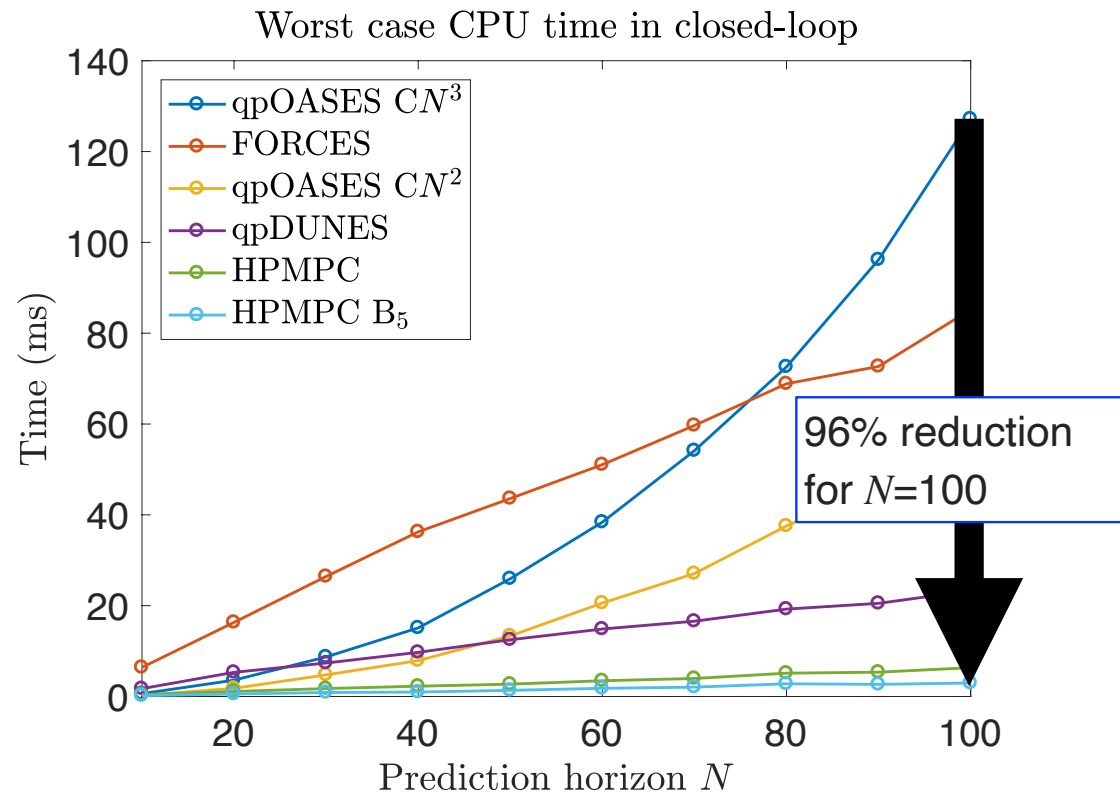
- **partial condensing** (proposed by Daniel Axehill, Linköping) combines advantages of condensing and Riccati recursion and further boost performance of HPMPC (by G. Frison, DTU/Freiburg)



Daniel Axehill



Gianluca Frison



Real-Time Iteration Contraction

$$\begin{array}{ll} \underset{w \in \mathbb{R}^n}{\text{minimize}} & \phi(w) \\ \text{subject to} & g(w) + M\bar{x}_0 = 0 \\ & w \in \Omega \end{array}$$

Regard primal dual iterates $z_k = (w_k, \lambda_k)$
and exact solutions z_k^* solving the full nonconvex problem for $(\bar{x}_0)_k$

Iterations are driven by parameter changes $\Delta\bar{x}_{0,k} := (\bar{x}_0)_{k+1} - (\bar{x}_0)_k$

We can establish a contraction estimate for the primal dual errors:

$$\|z_{k+1} - z_{k+1}^*\| \leq (c_1 + c_2 \|z_k - z_k^*\|) \|z_k - z_k^*\| + (c_3 + c_4 \|\Delta\bar{x}_{0,k}\|) \|\Delta\bar{x}_{0,k}\|$$

Contraction rate depends on bounds on nonlinearity, Jacobian error, and on strong regularity constant.

Contraction rate is independent of active set changes.

[Tran Dinh, Savorgnan, Diehl, SIOPT, 2013]

[Zanelli, Tran Dinh, Diehl, Automatica, 2021]

Computations in one Real-Time Iteration

NLP

$$\begin{aligned}
 & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\
 & \text{subject to} && x_0 - \bar{x}_0 = 0, \\
 & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\
 & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\
 & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\
 & && r(x_N) \leq 0.
 \end{aligned}$$

1) Linearize constraints: Integration & sensitivities

Sparse QP

$$\begin{aligned}
 & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) &+& E_{\text{QP}}(x_N) \\
 & \text{subject to} && x_0 - \bar{x}_0 = 0, \\
 & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\
 & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\
 & && r' + R x_N \leq 0.
 \end{aligned}$$

2) Condense sparse QP

Condensed small QP

$$\begin{aligned}
 & \underset{u}{\text{minimize}} && f_{\text{condQP},i}(\bar{x}_0, u) \\
 & \text{subject to} && \bar{r} + \bar{R}^{x_0} \bar{x}_0 + \bar{R}^u u \leq 0.
 \end{aligned}$$

3) Solve condensed QP

Computations in one Real-Time Iteration

NLP

$$\begin{aligned}
 & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\
 & \text{subject to} && x_0 - \bar{x}_0 = 0, \\
 & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\
 & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\
 & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\
 & && r(x_N) \leq 0.
 \end{aligned}$$

Sparse QP

$$\begin{aligned}
 & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) &+& E_{\text{QP}}(x_N) \\
 & \text{subject to} && x_0 - \bar{x}_0 = 0, \\
 & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\
 & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\
 & && r' + R x_N \leq 0.
 \end{aligned}$$

Condensed small QP

$$\begin{aligned}
 & \underset{u}{\text{minimize}} && f_{\text{condQP},i}(\bar{x}_0, u) \\
 & \text{subject to} && \bar{r} + \bar{R}^{x_0} \bar{x}_0 + \bar{R}^u u \leq 0.
 \end{aligned}$$

Can prepare without knowing $\bar{x}_0$
“Preparation phase”

1) Linearize constraints: Integration & sensitivities

2) Condense sparse QP

3) Solve condensed QP

Computations in one Real-Time Iteration

NLP

$$\begin{aligned}
 & \underset{x, z, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_i(x_i, z_i, u_i) &+& E(x_N) \\
 & \text{subject to} && x_0 - \bar{x}_0 = 0, \\
 & && x_{i+1} - f_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\
 & && g_i(x_i, z_i, u_i) = 0, \quad i = 0, \dots, N-1, \\
 & && h_i(x_i, z_i, u_i) \leq 0, \quad i = 0, \dots, N-1, \\
 & && r(x_N) \leq 0.
 \end{aligned}$$

1) Linearize constraints: Integration & sensitivities

Sparse QP

$$\begin{aligned}
 & \underset{x, u}{\text{minimize}} && \sum_{i=0}^{N-1} L_{\text{redQP},i}(x_i, u_i) &+& E_{\text{QP}}(x_N) \\
 & \text{subject to} && x_0 - \bar{x}_0 = 0, \\
 & && x_{i+1} - c_i - A_i x_i - B_i u_i = 0, \quad i = 0, \dots, N-1, \\
 & && \bar{h}_i + \bar{H}_i^x x_i + \bar{H}_i^u u_i \leq 0, \quad i = 0, \dots, N-1, \\
 & && r' + R x_N \leq 0.
 \end{aligned}$$

2) Condense sparse QP

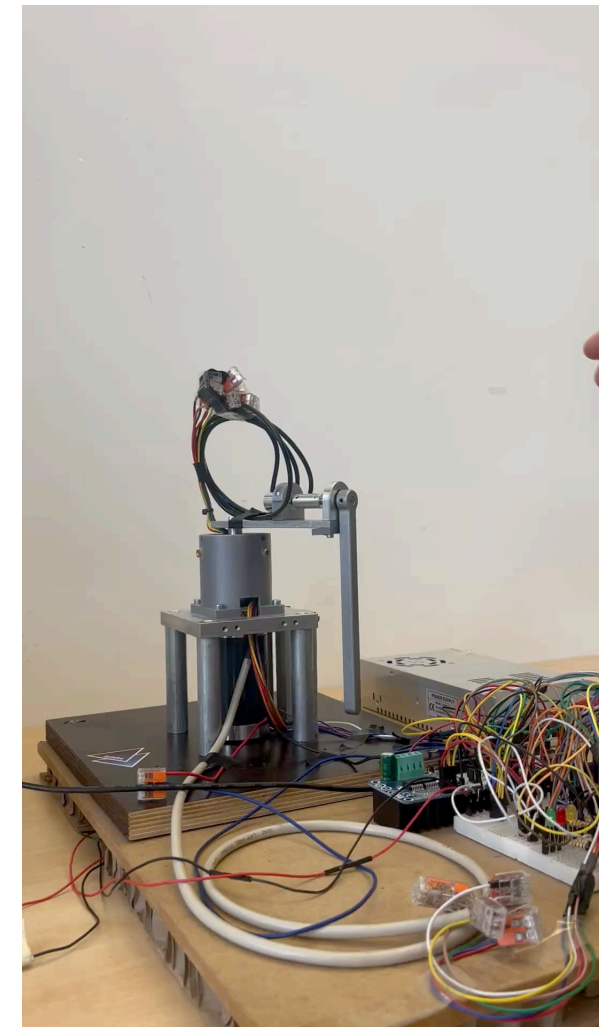
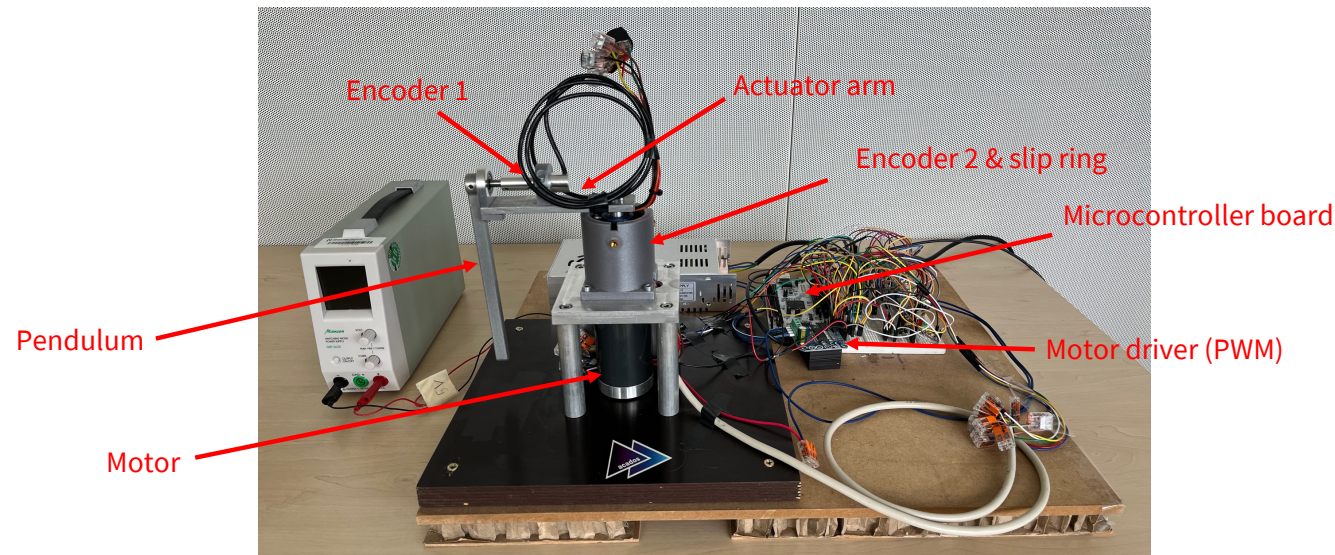
Condensed small QP

$$\begin{aligned}
 & \underset{u}{\text{minimize}} && f_{\text{condQP},i}(\bar{x}_0, u) \\
 & \text{subject to} && \bar{r} + \bar{R}^{x_0} \bar{x}_0 + \bar{R}^u u \leq 0.
 \end{aligned}$$

“Feedback phase”
3) Solve condensed QP

Nonlinear Model Predictive Control of Furuta Pendulum with **acados**

(by Prof. Dr. Jörg Fischer, HS Offenburg, on an embedded Cortex-M7 platform, in Feb 2026)



▶ **μController board: (STM32 NUCLEO-H745ZI)**

- ▶ Prototyping-Board by STMicroelectronics
- ▶ Dual-core MCU:
 - ▶ Cortex-M7 (480 MHz)
 - ▶ Cortex-M4 (240 MHz)
- ▶ 1 MB RAM
- ▶ 2 MB Flash
- ▶ On-board ST-LINK for programming and debugging
- ▶ Widely used STM32Cube ecosystem
- ▶ Costs $\approx$ 30 EUR

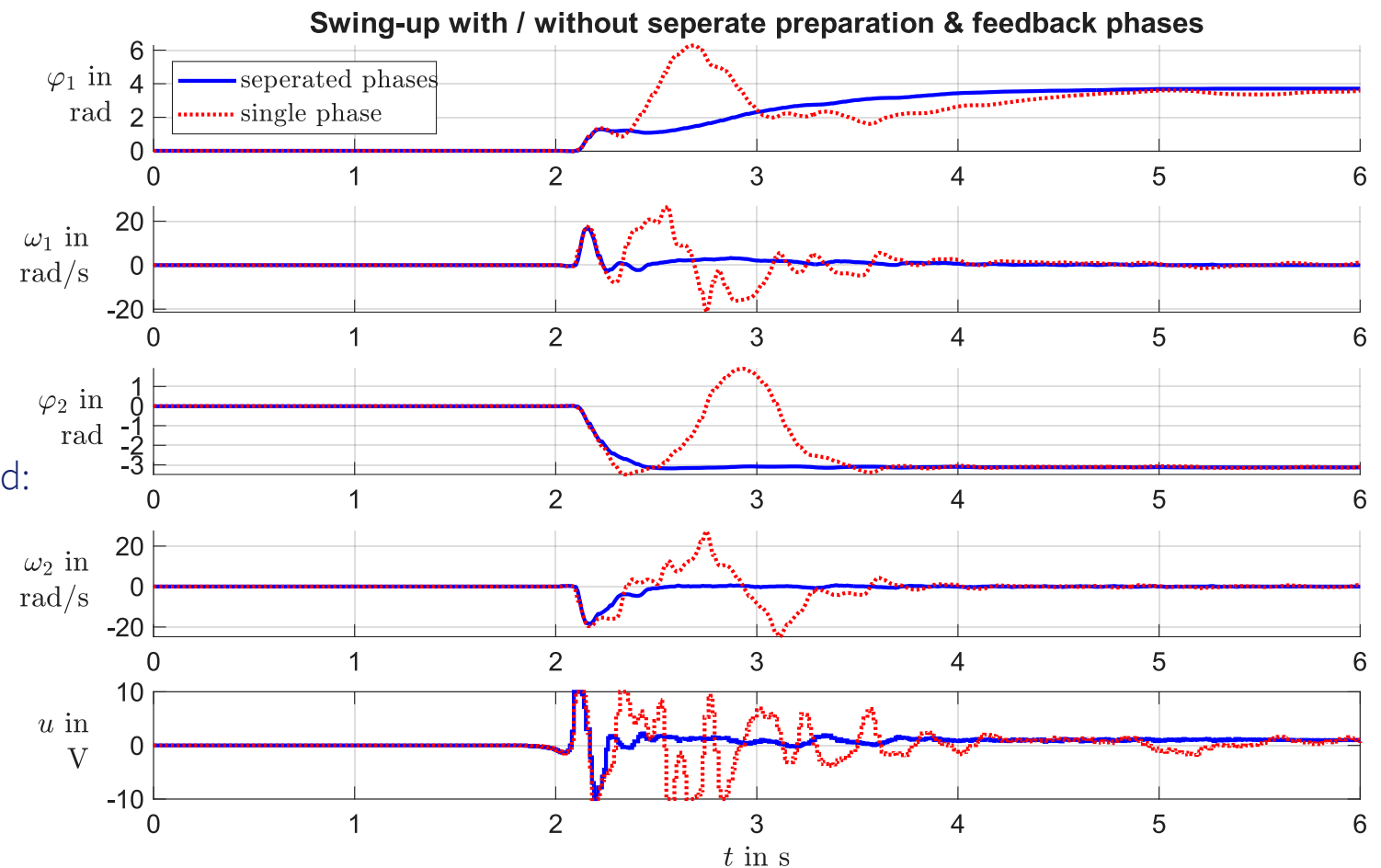
NMPC parameters

- ▶ Control cycle time: 12 ms
- ▶ Integrator: Explicit RK4 with 4 steps
- ▶ Prediction horizon: 200 ms
- ▶ Control intervals: 10
- ▶ Control constraints: $\bar{u} = +10\text{ V}$
 $\underline{u} = -10\text{ V}$
- ▶ State constraints: None
- ▶ NLP-solver: SQP - RTI
(with separate preparation & feedback phases)
- ▶ QP-solver: Full condensing HPIPM
- ▶ Hessian: Exact with regularization (Mirror)

Experimental Validation of the Benefit of Separating Preparation and Feedback Phase (on Furuta pendulum, by J. Fischer)

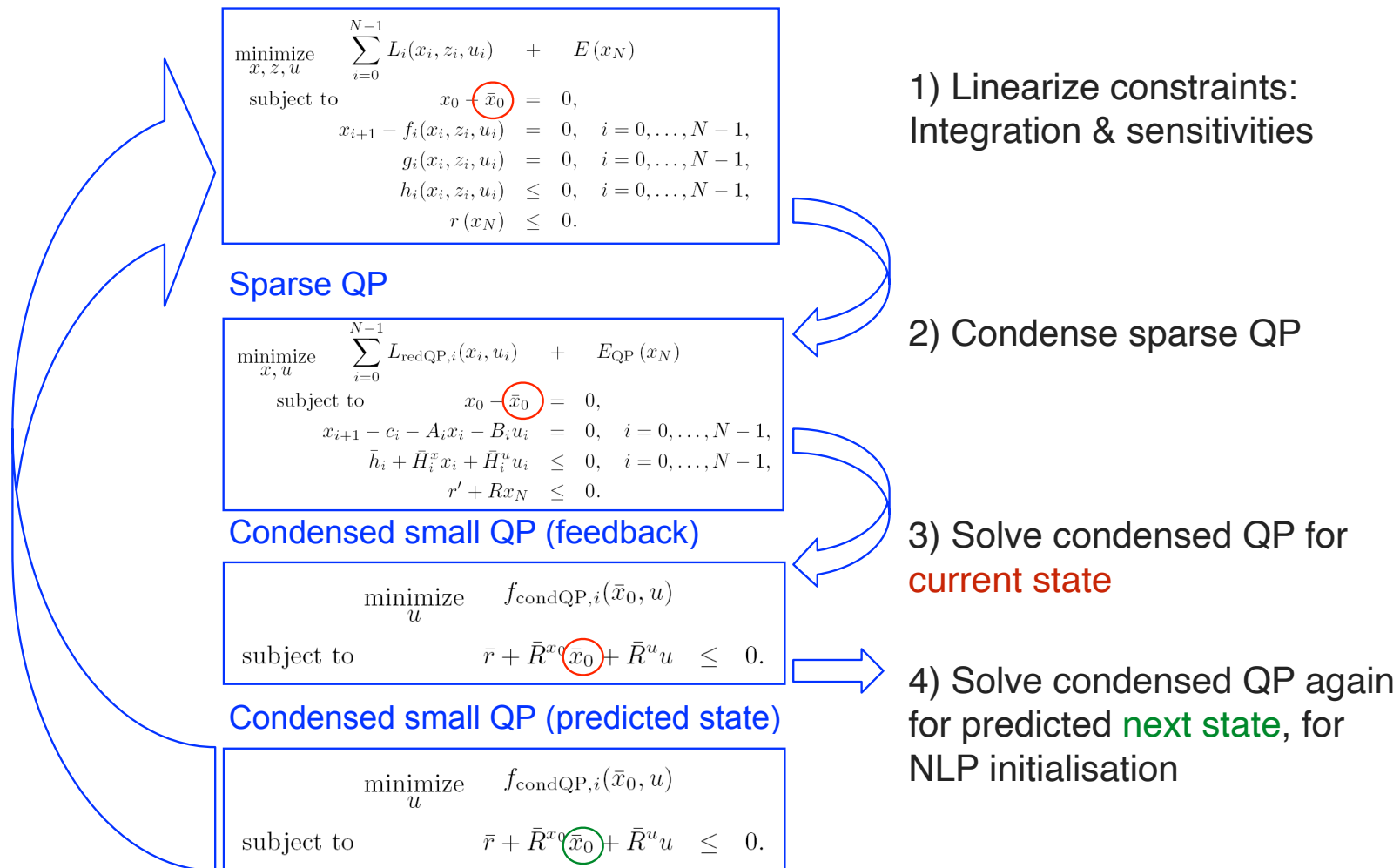


- ▶ Control cycle time: 12 ms
- ▶ RTI preparation phase: 9 ms ... 10 ms
- ▶ RTI feedback phase: 500 μ s ... 1,5 ms
- ▶ Time from sensor read to motor voltage applied: 1,5 ms ... 2 ms
- ▶ Short measurement propagation time gives robustness



*** Advanced-Step Real-Time Iteration (AS-RTI) ***

[Nurkanovic et al. 2020]



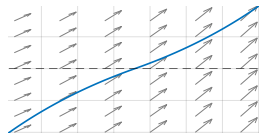


- ▶ Optimization with Complementarity Constraints: Embracing the Nonconvex
- ▶ **Finite Elements with Switch Detection (FESD)**
- ▶ Time Freezing for State Dependent Jumps
- ▶ Three Step Decomposition for Discrete Actuators

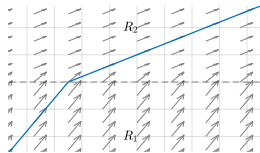
Non-Smooth Dynamics (NSD) - Informal Classification



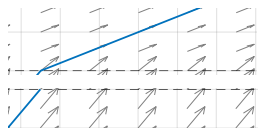
Regard ordinary differential equation (ODE) with **non-smooth** right hand side (RHS).
Distinguish three cases:



NSD1: non-differentiable RHS, e.g. $\dot{x} = 1 + |x|$



NSD2: state dependent ("internal") switch of RHS, e.g. $\dot{x} = 2 - \text{sign}(x)$
(similar but different: external switch by discrete actuator)



NSD3: state dependent jump, e.g. $x(t_+) = 3 + x(t_-)$

NSD2 Systems - State Dependent Switches



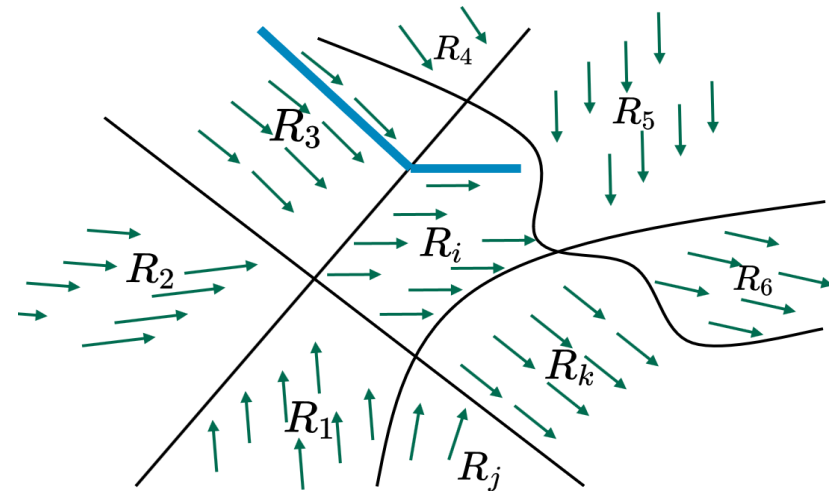
Regard **discontinuous** right hand side, piecewise smooth on disjoint open regions $R_i \subset \mathbb{R}^{n_x}$

Discontinuous ODE (NSD2)

$$\dot{x} = f_i(x, u), \text{ if } x \in R_i, \\ i \in \{1, \dots, m\}$$

Numerical aims:

1. exactly detect switching times
2. obtain exact sensitivities across regions



NSD2 Systems - State Dependent Switches



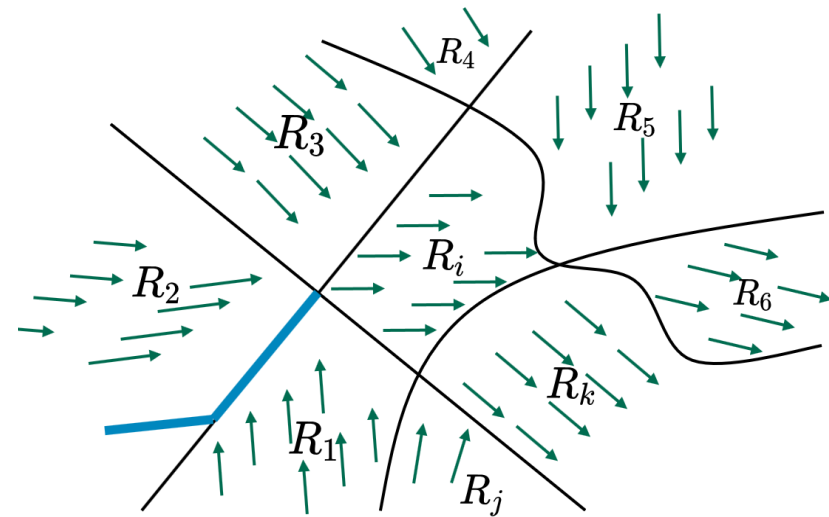
Regard **discontinuous** right hand side, piecewise smooth on disjoint open regions $R_i \subset \mathbb{R}^{n_x}$

Discontinuous ODE (NSD2)

$$\dot{x} = f_i(x, u), \text{ if } x \in R_i, \\ i \in \{1, \dots, m\}$$

Numerical aims:

1. exactly detect switching times
2. obtain exact sensitivities across regions
3. appropriately treat evolution on boundaries (sliding mode $\rightarrow$ Filippov convexification)



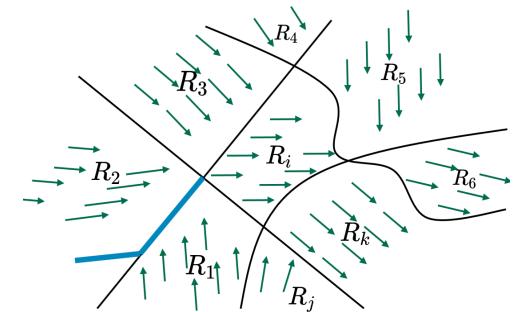
Filippov Convexification



Dynamics not yet well-defined on region boundaries ∂R_i . Idea by A.F. Filippov (1923-2006): replace ODE by differential inclusion, using convex combination of neighboring vector fields.

Filippov Differential Inclusion

$$\dot{x} \in F_F(x, u) := \left\{ \sum_{i=1}^m f_i(x, u) \theta_i \mid \sum_{i=1}^m \theta_i = 1, \right. \\ \left. \begin{array}{l} \theta_i \geq 0, \quad i = 1, \dots, m, \\ \theta_i = 0, \quad \text{if } x \notin \overline{R_i} \end{array} \right\}$$



- ▶ for interior points $x \in R_i$ nothing changes: $F_F(x, u) = \{f_i(x, u)\}$
- ▶ Provides meaningful generalization on region boundaries.
E.g. on $\overline{R_1} \cap \overline{R_2}$ both θ_1 and θ_2 can be nonzero

How to compute convex multipliers θ ?



Assume sets R_i given by [cf. Stewart, 1990]

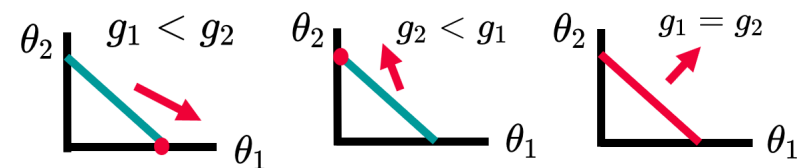
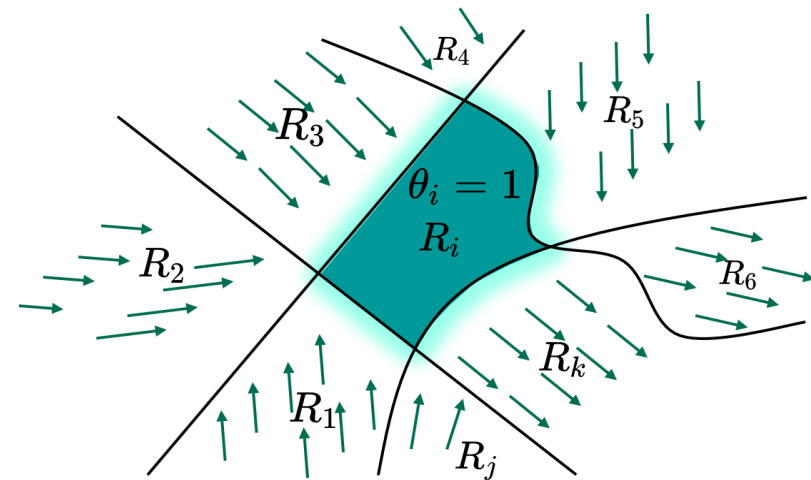
$$R_i = \{x \in \mathbb{R}^n \mid g_i(x) < \min_{j \neq i} g_j(x)\}$$

Linear program (LP) Representation

$$\dot{x} = \sum_{i=1}^m f_i(x, u) \theta_i^* \quad \text{with}$$

$$\theta^* \in \arg \min_{\theta \in \mathbb{R}^m} \sum_{i=1}^m g_i(x) \theta_i$$

$$\begin{aligned} \text{s.t.} \quad & \sum_{i=1}^m \theta_i = 1 \\ & \theta \geq 0. \end{aligned}$$



From Filippov to dynamic complementarity systems

Using the KKT conditions of the parametric LP



LP representation

$$\begin{aligned} \dot{x} &= F(x, u) \theta^* \\ \text{with } \theta^* &\in \operatorname{argmin}_{\theta \in \mathbb{R}^m} g(x)^\top \theta \\ \text{s.t. } &0 \leq \theta \\ &1 = e^\top \theta \end{aligned}$$

where

$$\begin{aligned} F(x, u) &:= [f_1(x, u), \dots, f_m(x, u)] \in \mathbb{R}^{n_x \times m} \\ g(x) &:= [g_1(x), \dots, g_m(x)]^\top \in \mathbb{R}^m \\ e &:= [1, 1, \dots, 1]^\top \in \mathbb{R}^m \end{aligned}$$

Express equivalently by optimality conditions:

Dynamic Complementarity System (DCS)

$$\dot{x} = F(x, u) \theta \quad (1a)$$

$$0 = g(x) - \lambda - e\mu \quad (1b)$$

$$0 \leq \theta \perp \lambda \geq 0 \quad (1c)$$

$$1 = e^\top \theta \quad (1d)$$

- ▶ $\mu \in \mathbb{R}$ and $\lambda \in \mathbb{R}^m$ are Lagrange multipliers
- ▶ $(1c) \Leftrightarrow \min\{\theta, \lambda\} = 0 \in \mathbb{R}^m$
- ▶ Together, (1b), (1c), (1d) determine the $(2m + 1)$ variables θ, λ, μ uniquely

Conventional Discretization by Implicit Runge Kutta (IRK) Method



Continuous time DCS

$$\begin{aligned}x(0) &= \bar{x}_0, \\ \dot{x}(t) &= v(t) \\ v(t) &= F(x(t), u(t)) \theta(t) \\ 0 &= g(x(t)) - \lambda(t) - e\mu(t) \\ 0 &\leq \theta(t) \perp \lambda(t) \geq 0 \\ 1 &= e^\top \theta(t), \quad t \in [0, T]\end{aligned}$$

Discrete time IRK-DCS equation

$$\begin{aligned}x_{0,0} &= \bar{x}_0, \quad x_{k+1,0} = x_{k,0} + h \sum_{n=1}^s b_n v_{k,n} \\ x_{k,j} &= x_{k,0} + h \sum_{n=1}^s a_{jn} v_{k,n} \\ v_{k,j} &= F(x_{k,j}, u_{k,j}) \theta_{k,j} \\ 0 &= g(x_{k,j}) - \lambda_{k,j} - e\mu_{k,j} \\ 0 &\leq \theta_{k,j} \perp \lambda_{k,j} \geq 0 \\ 1 &= e^\top \theta_{k,j}, \quad j = 1, \dots, s, \quad k = 0, \dots, N-1\end{aligned}$$

Notation: $x_{k,r} \in \mathbb{R}^{n_x}$, $\theta_{k,r} \in \mathbb{R}^m$ etc. with:

- ▶ $k \in \{0, 1, \dots, N\}$ - index of integration step; step length $h := T/N$
- ▶ $j, n \in \{0, 1, \dots, s\}$ - index of intermediate IRK stage / collocation point
- ▶ a_{jn} and b_n - Butcher tableau entries of Implicit Runge Kutta method

Conventional Collocation - Illustrative Example



Regard example with $x \in \mathbb{R}^2$ and constants $a, k, c > 0$:

$$\dot{x} = \begin{cases} f_1(x), & x_1 > 0, \\ f_2(x), & x_1 < 0. \end{cases}$$

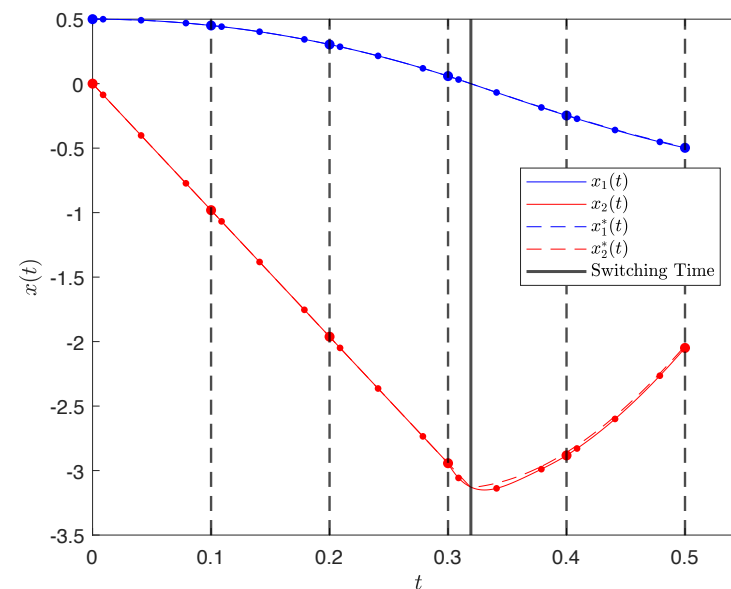
$$f_1(x) = \begin{pmatrix} x_2 \\ -a \end{pmatrix}, \quad f_2(x) = \begin{pmatrix} x_2 \\ -kx_1 - cx_2 \end{pmatrix}$$

$$g_1(x) = -x_1,$$

$$g_2(x) = x_1$$

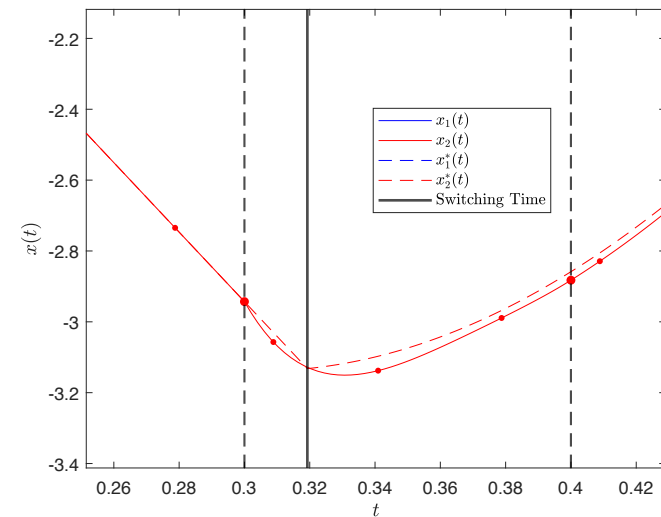
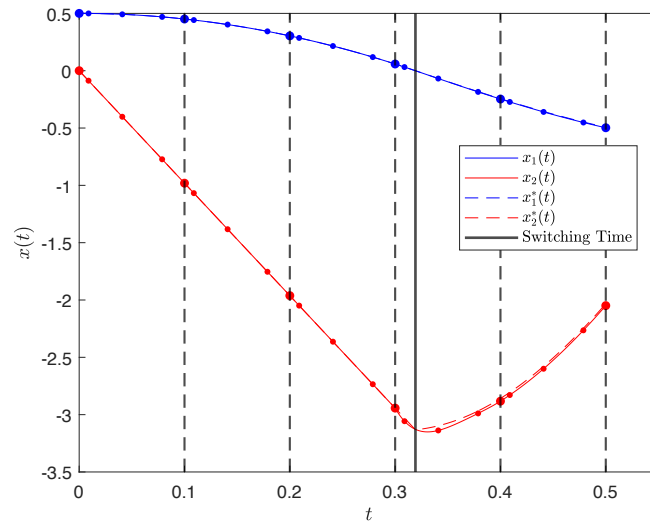
$$\bar{x}_0 = [0.5, 0]^\top$$

Solve with IRK Radau IIA method of order 7, $s = 4$, $N = 5$, $T = 0.5$, $h = 0.1$.



Conventional Collocation - Illustrative Example

Zoom in



High integration accuracy of 7th order IRK method is lost in fourth time step.

Reason: we try to approximate a non-smooth function by a (smooth) polynomial.

Question: could we ensure that switches happen only at element boundaries?

→ **Finite Elements with Switch Detection (FESD)**

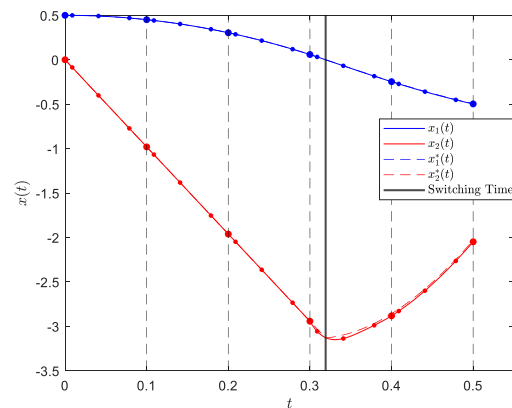
Finite Elements with Switch Detection (FESD)

PhD work of A. Nurkanovic, to be submitted

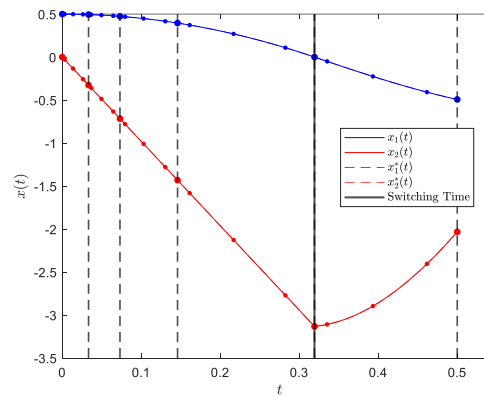


FESD is a novel DCS discretization method based on three ideas:

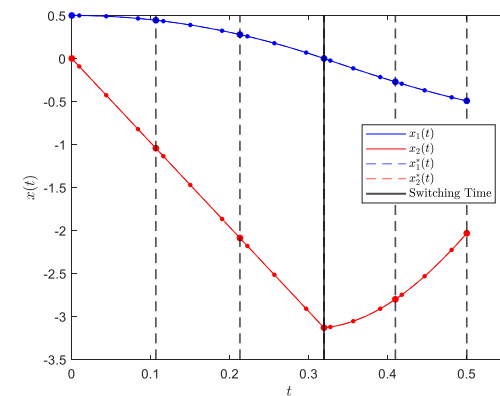
- ▶ make stepsizes h_k free, ensure $\sum_{k=0}^{N-1} h_k = T$ [cf. Baumrucker & Biegler 2009]
- ▶ allow switches only at element boundaries, enforce via *cross-complementarities*
- ▶ remove spurious degrees of freedom via *step equilibration*



conventional
discretization



variable stepsizes and
cross-complementarities



FESD discretization
with step equilibration

Conventional DCS and FESD discretization without step equilibration



Conventional discretization

$$\begin{aligned}
 x_{0,0} &= \bar{x}_0, \quad h = T/N \\
 x_{k+1,0} &= x_{k,0} + h \sum_{n=1}^s b_n v_{k,n} \\
 x_{k,j} &= x_{k,0} + h \sum_{n=1}^s a_{jn} v_{k,n} \\
 v_{k,j} &= F(x_{k,j}, u_{k,j}) \theta_{k,j} \\
 0 &= g(x_{k,j}) - \lambda_{k,j} - e\mu_{k,j} \\
 0 &\leq \theta_{k,j} \perp \lambda_{k,j} \geq 0 \\
 1 &= e^\top \theta_{k,j}
 \end{aligned}$$

for $j = 1, \dots, s$
and $k = 0, \dots, N-1$

FESD discretization without step equilibration

$$\begin{aligned}
 x_{0,0} &= \bar{x}_0, \quad \sum_{k=0}^{N-1} h_k = T \\
 x_{k+1,0} &= x_{k,0} + h_k \sum_{n=1}^s b_n v_{k,n} \\
 x_{k,j} &= x_{k,0} + h_k \sum_{n=1}^s a_{jn} v_{k,n} \\
 v_{k,j} &= F(x_{k,j}, u_{k,j}) \theta_{k,j} \\
 0 &= g(x_{k,j'}) - \lambda_{k,j'} - e\mu_{k,j'} \\
 0 &\leq \theta_{k,j} \perp \lambda_{k,j'} \geq 0 \quad (\text{cross-complementarities}) \\
 1 &= e^\top \theta_{k,j}
 \end{aligned}$$

for $j = 1, \dots, s$ and $k = 0, \dots, N-1$
and $j' = 0, 1, \dots, s$

- ▶ N extra variables ($h_0, \dots, h_{N-1}$) restricted by one extra equality
- ▶ additional multipliers $\lambda_{k,0}, \mu_{k,0}$ are uniquely determined

Conventional DCS and FESD discretization with step equilibration



Conventional discretization

$$\begin{aligned}
 x_{0,0} &= \bar{x}_0, \quad h = T/N \\
 x_{k+1,0} &= x_{k,0} + h \sum_{n=1}^s b_n v_{k,n} \\
 x_{k,j} &= x_{k,0} + h \sum_{n=1}^s a_{jn} v_{k,n} \\
 v_{k,j} &= F(x_{k,j}, u_{k,j}) \theta_{k,j} \\
 0 &= g(x_{k,j}) - \lambda_{k,j} - e \mu_{k,j} \\
 0 &\leq \theta_{k,j} \perp \lambda_{k,j} \geq 0 \\
 1 &= e^\top \theta_{k,j}
 \end{aligned}$$

for $j = 1, \dots, s$
and $k = 0, \dots, N-1$

FESD discretization with step equilibration

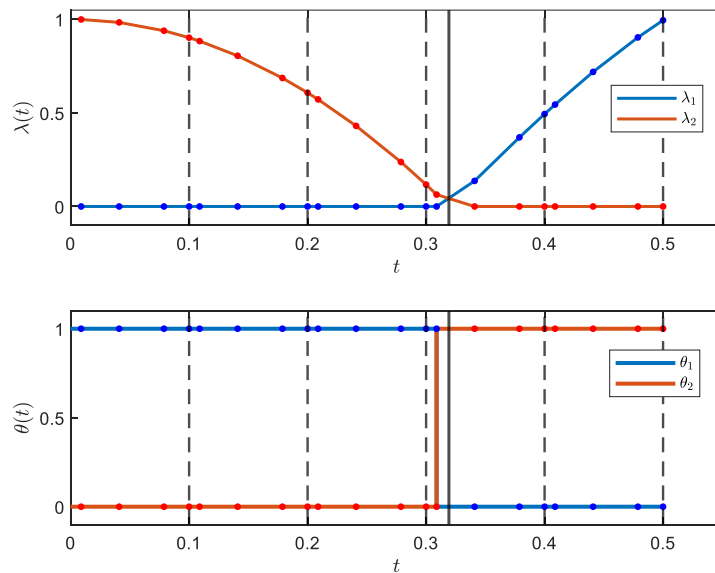
$$\begin{aligned}
 x_{0,0} &= \bar{x}_0, \quad \sum_{k=0}^{N-1} h_k = T \\
 x_{k+1,0} &= x_{k,0} + h_k \sum_{n=1}^s b_n v_{k,n} \\
 x_{k,j} &= x_{k,0} + h_k \sum_{n=1}^s a_{jn} v_{k,n} \\
 v_{k,j} &= F(x_{k,j}, u_{k,j}) \theta_{k,j} \\
 0 &= g(x_{k,j'}) - \lambda_{k,j'} - e \mu_{k,j'} \\
 0 &\leq \theta_{k,j} \perp \lambda_{k,j'} \geq 0 \quad (\text{cross-complementarities}) \\
 1 &= e^\top \theta_{k,j} \\
 0 &= \nu(\theta_{k'}, \theta_{k'+1}, \lambda_{k'}, \lambda_{k'+1}) \cdot (h_{k'} - h_{k'+1})^2 \\
 \text{for } j &= 1, \dots, s \quad \text{and } k = 0, \dots, N-1 \\
 \text{and } j' &= 0, 1, \dots, s \quad \text{and } k' = 0, \dots, N-2
 \end{aligned}$$

- ▶ N extra FESD variables $(h_0, \dots, h_{N-1})$ now locally uniquely determined by N constraints
- ▶ "Nurkanovic's indicator function" $\nu(\theta_{k'}, \theta_{k'+1}, \lambda_{k'}, \lambda_{k'+1})$ only zero if a switch occurs

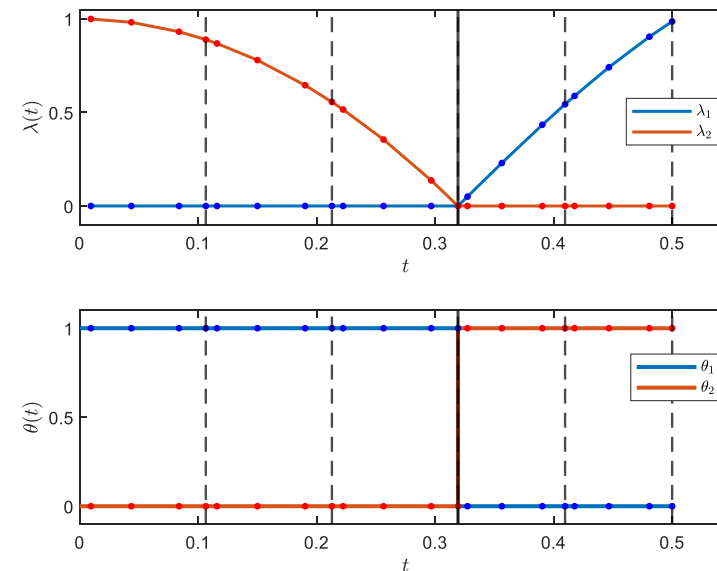
Multipliers in Conventional and FESD Discretization



Conventional Collocation:



FESD Discretization:



FESD's cross-complementarities exploit the fact that the multiplier $\lambda_i(t)$ is continuous in time. On boundary, $\lambda_i(t_k)$ must be zero if $\theta_i(t) > 0$ for any $t \in [t_{k-1}, t_{k+1}]$ on the adjacent intervals.

Numerical simulation example: unstable switched oscillator



Regard an unstable non-smooth oscillator

$$\dot{x}(t) = \begin{cases} A_1 x, & c(x) < 0, \\ A_2 x, & c(x) > 0, \end{cases}$$

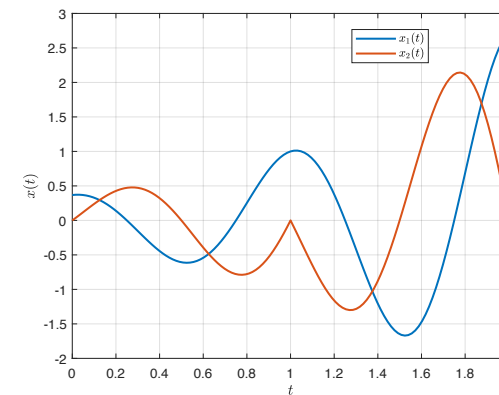
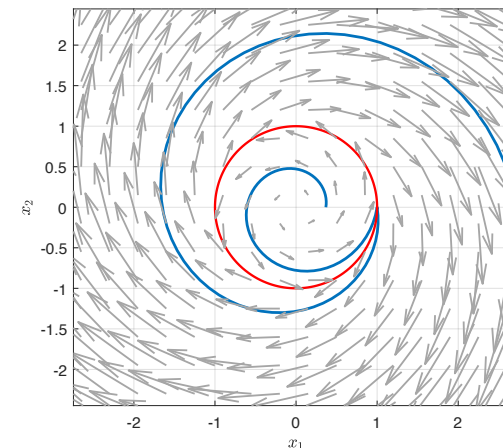
with

$$A_1 = \begin{bmatrix} 1 & \omega \\ -\omega & 1 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 & -\omega \\ \omega & 1 \end{bmatrix},$$

$$c(x) = x_1^2 + x_2^2 - 1, \quad \omega = 2\pi, \quad x(0) = [e^{-1} \ 0]^\top$$

For $t \in [0, 2]$, we have

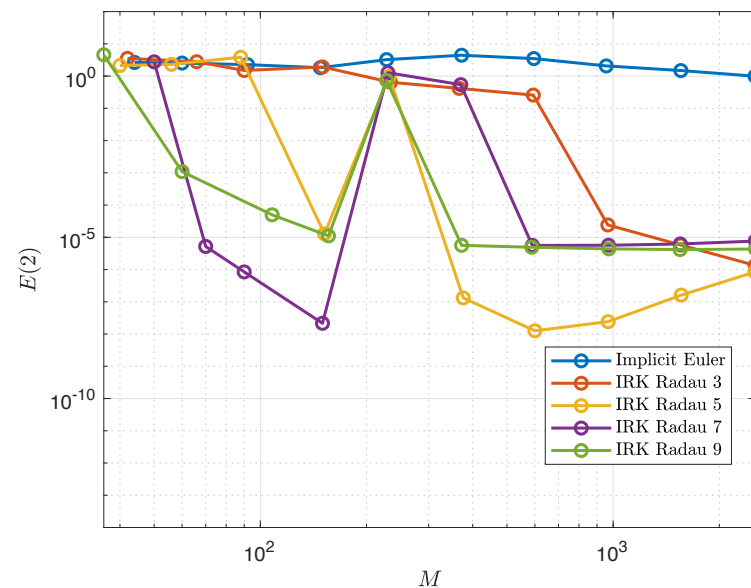
$$x(2) = [e \ 0]^\top.$$



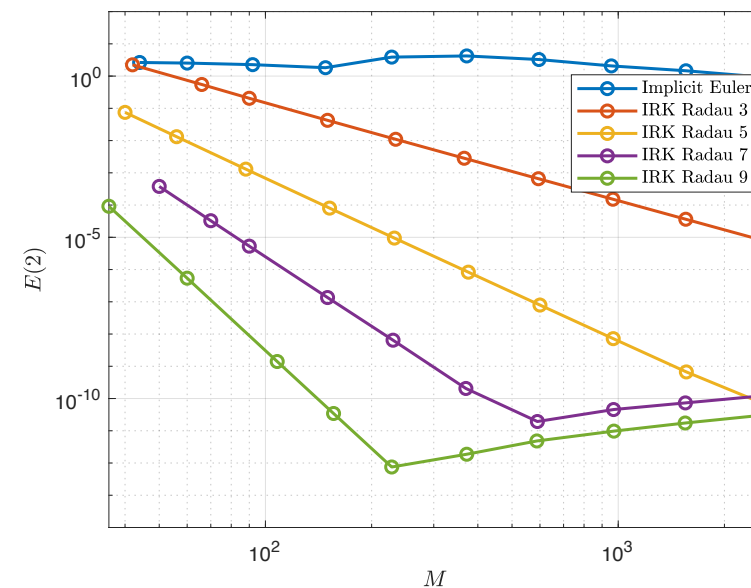
FESD recovers high integration order for switched systems



Conventional Collocation:



FESD Discretization:



Integration error $E(T)$ at time $T = 2$ vs. total number $M = s N$ of collocation points, for different Radau IIA methods.

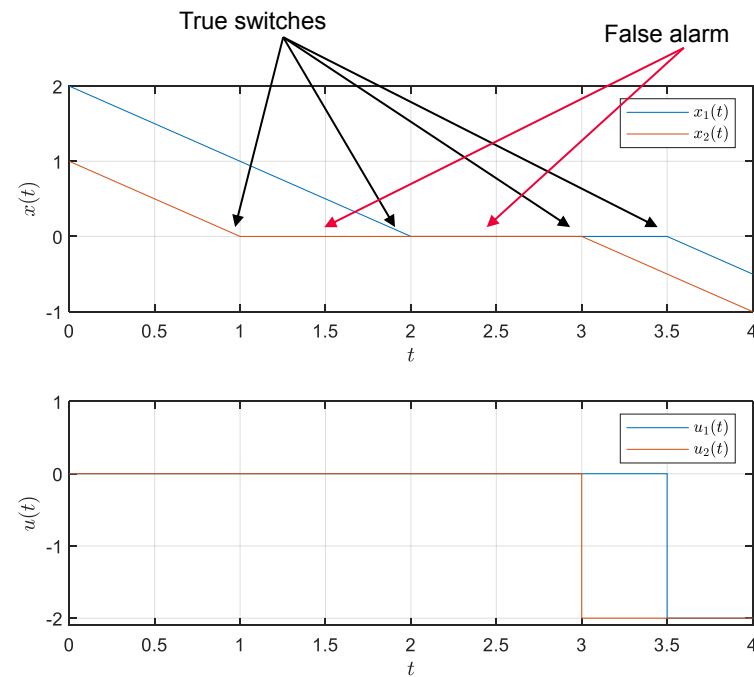
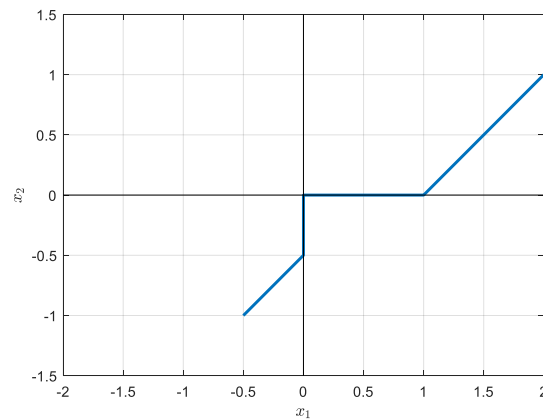
FESD discretization delivers versatile MPCC formulation with high integration order

Optimal Control Example: Solution Trajectory with 3 Sliding Modes



Regard the following OCP

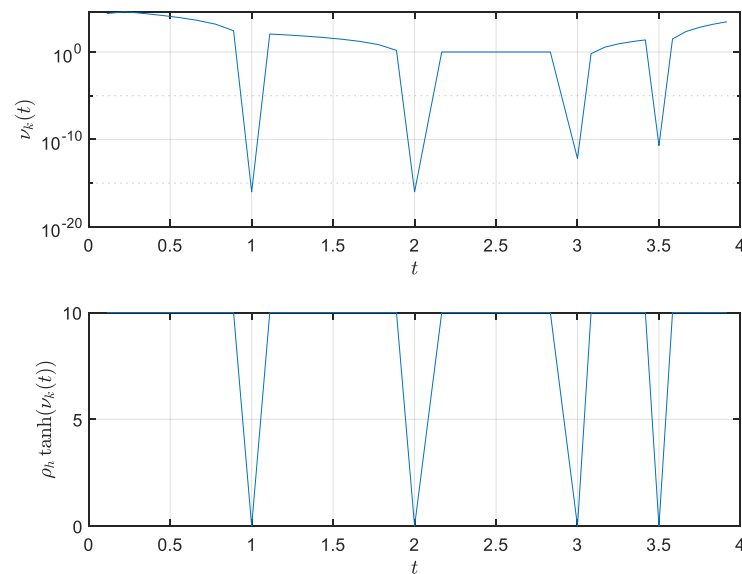
$$\begin{aligned} \min_{x(\cdot), u(\cdot)} \quad & \int_0^4 u(t)^\top u(t) dt \\ \text{s.t.} \quad & x(0) = (2, 1), \\ & \dot{x}(t) \in -\text{sign}(x(t)) + u(t), \quad t \in [0, 4], \\ & -2e_2 \leq u(t) \leq 2e_2, \quad t \in [0, 4], \\ & x(4) = (-1, -0.5). \end{aligned}$$



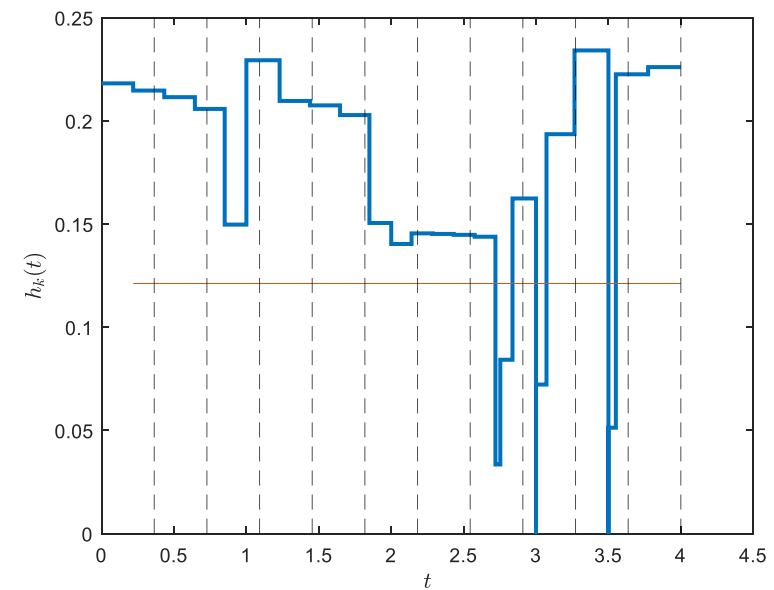
Numerical Solution without Equilibration



Nurkanovic's indicator function over time:



Step size over time:

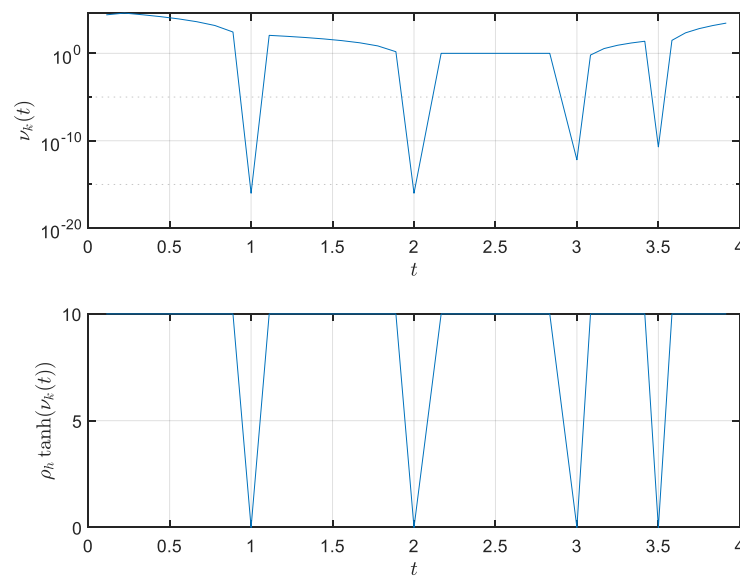


Optimizer varies step size randomly, potentially playing with integration errors.

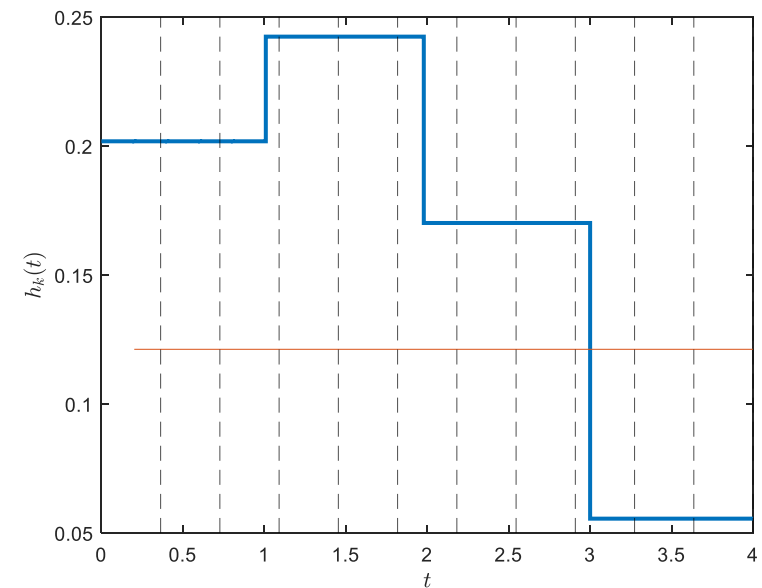
Numerical Solution with Equilibration



Nurkanovic's indicator function over time:



Step size over time:



Equidistant grid on each "switching stage". Jumps exactly at switching times.

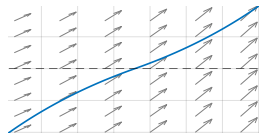


- ▶ Optimization with Complementarity Constraints: Embracing the Nonconvex
- ▶ Finite Elements with Switch Detection (FESD)
- ▶ **Time Freezing for State Dependent Jumps**
- ▶ Three Step Decomposition for Discrete Actuators

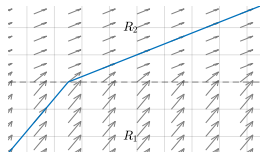
Non-Smooth Dynamics (NSD) - Informal Classification



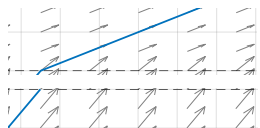
Regard ordinary differential equation (ODE) with **non-smooth** right hand side (RHS).
Distinguish three cases:



NSD1: non-differentiable RHS, e.g. $\dot{x} = 1 + |x|$



NSD2: state dependent ("internal") switch of RHS, e.g. $\dot{x} = 2 - \text{sign}(x)$
(similar but different: external switch by discrete actuator)



NSD3: state dependent jump, e.g. $x(t_+) = 3 + x(t_-)$

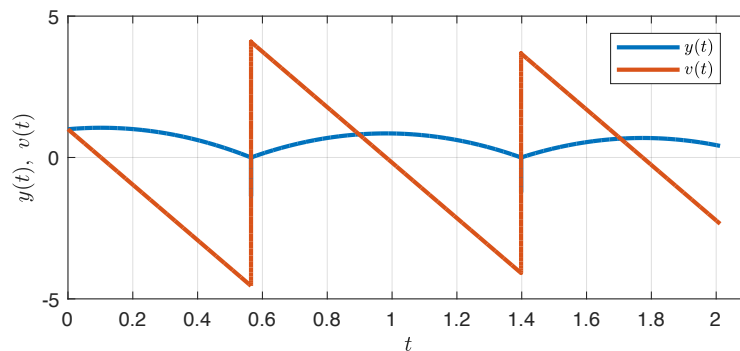
NSD3 State Jump Example: Bouncing Ball



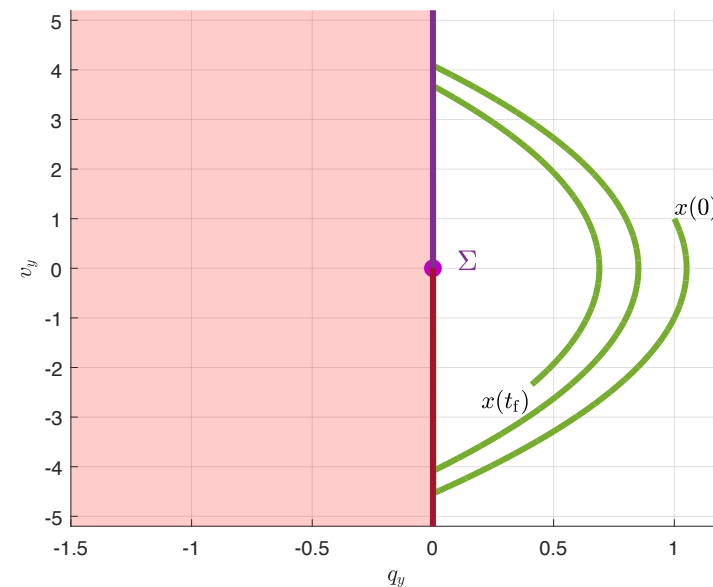
Bouncing ball with state $x = (y, v)$:

$$\begin{aligned} m\dot{v} &= -mg, & \text{if } y > 0 \\ v(t^+) &= -0.9v(t^-), & \text{if } y(t^-) = 0 \text{ and } v(t^-) < 0 \end{aligned}$$

Time plot of bouncing ball trajectory:



Phase plot of bouncing ball trajectory:



Question: could we transform NSD3 systems into (easier) NSD2 systems?

Three ideas:



1. mimic state jump by **auxiliary dynamic system** $\dot{x} = \varphi(x)$ on prohibited region
2. introduce a **clock state** $t(\tau)$ that stops counting when the auxiliary system is active
3. adapt speed of time, $\frac{dt}{d\tau} = s$ with $s \geq 1$, and **impose terminal constraint** $t(T) = T$

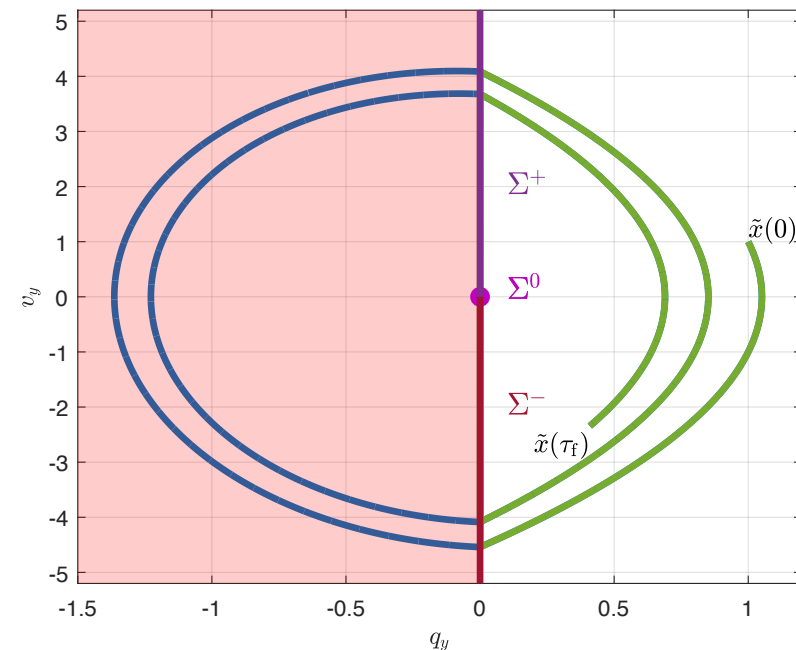
The time-freezing reformulation



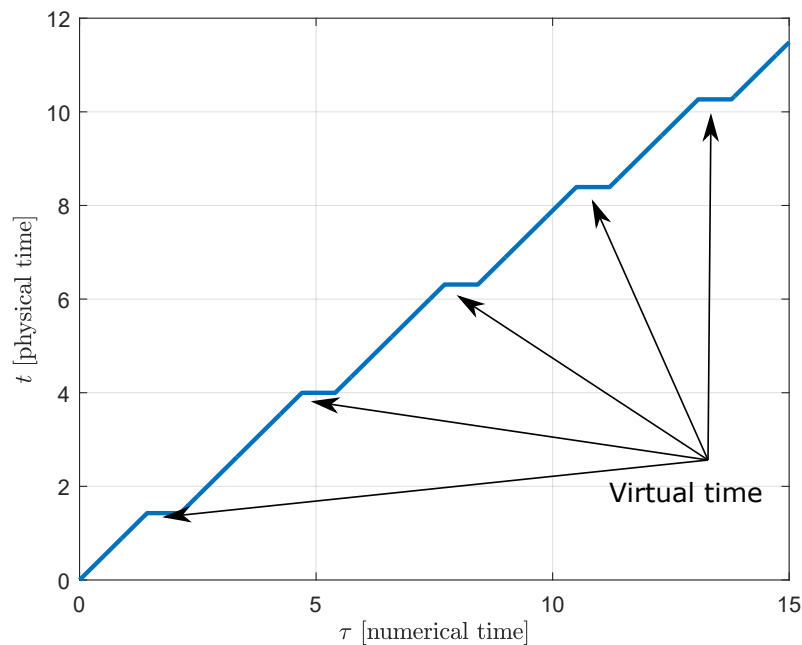
Augmented state $(x, t) \in \mathbb{R}^{n+1}$ evolves in **numerical time** τ . Augmented system is nonsmooth, of NSD2 type:

$$\frac{d}{d\tau} \begin{bmatrix} x \\ t \end{bmatrix} = \begin{cases} \textcolor{red}{s} \begin{bmatrix} f(x) \\ 1 \end{bmatrix}, & \text{if } c(x) \geq 0 \\ \begin{bmatrix} \varphi(x) \\ \textcolor{red}{0} \end{bmatrix}, & \text{if } c(x) < 0 \end{cases}$$

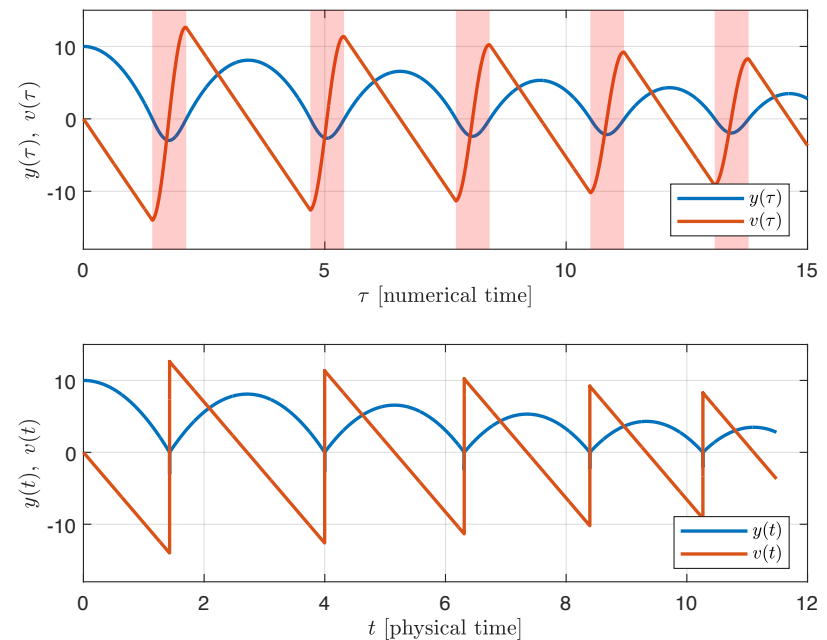
- ▶ During normal times, system and clock state evolve with adapted speed $\textcolor{red}{s} \geq 1$.
- ▶ Auxiliary system $\frac{dx}{d\tau} = \varphi(x)$ mimics state jump while time is frozen, $\frac{dt}{d\tau} = \textcolor{red}{0}$.



Time-freezing for bouncing ball example



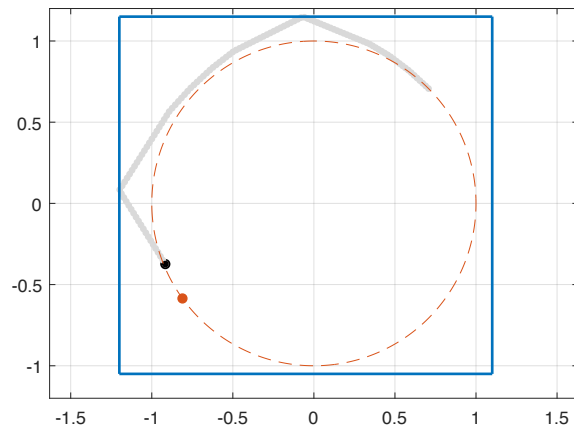
Evolution of physical time (clock state) during augmented system simulation ($s = 1$).



We can recover the true solution by plotting $x(\tau)$ vs. $t(\tau)$ and disregarding "frozen pieces".

A Tracking NMPC Example with Time-Freezing and FESD

Regard bouncing ball in two dimensions driven by bounded force: $\ddot{q} = u$



- ▶ augmented state
 $x = (q, \dot{q}, t) \in \mathbb{R}^5$
- ▶ $m = 9$ regions (8 with auxiliary dynamics for state jumps)

$$\min_{\substack{x(\cdot), u(\cdot), s(\cdot), \\ \theta(\cdot), \lambda(\cdot), \mu(\cdot)}} \int_0^T (q - q_{\text{ref}}(\tau))^\top (q - q_{\text{ref}}(\tau)) s(\tau) d\tau$$

$$\text{s.t. } x(0) = x_0, \quad t(T) = T,$$

$$x'(\tau) = \sum_{i=1}^m \theta_i(\tau) f_i(x(\tau), u(\tau), s(\tau)),$$

$$0 = g(x(\tau)) - \lambda(\tau) - \mu(\tau)e,$$

$$0 \leq \lambda(\tau) \perp \theta(\tau) \geq 0,$$

$$1 = e^\top \theta(\tau),$$

$$\|u(\tau)\|_2^2 \leq u_{\text{max}}^2,$$

$$1 \leq s(\tau) \leq s_{\text{max}}, \quad \tau \in [0, T].$$

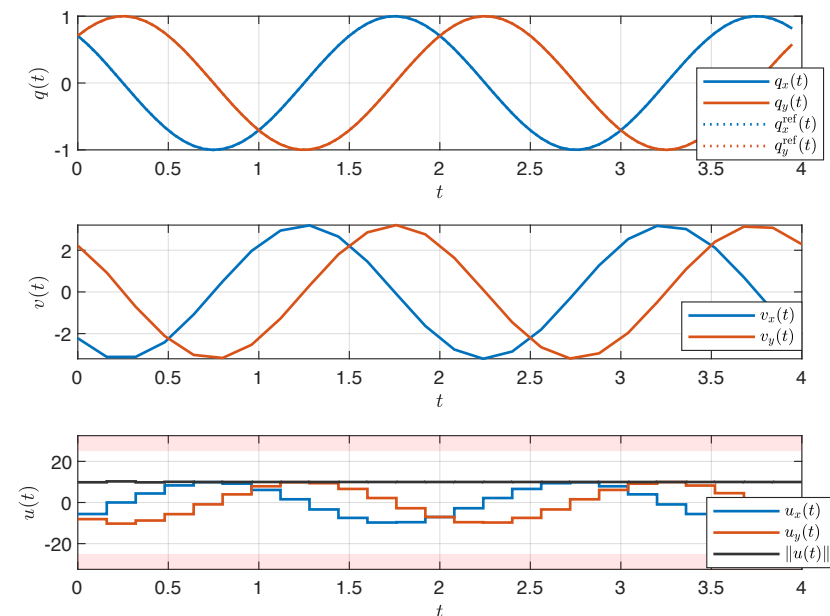
$$q_{\text{ref}}(\tau) = (R \cos(\omega t(\tau)), R \sin(\omega t(\tau))).$$

Results with slowly moving reference

For $\omega = \pi$, tracking is easy: no jumps occur in optimal solution.



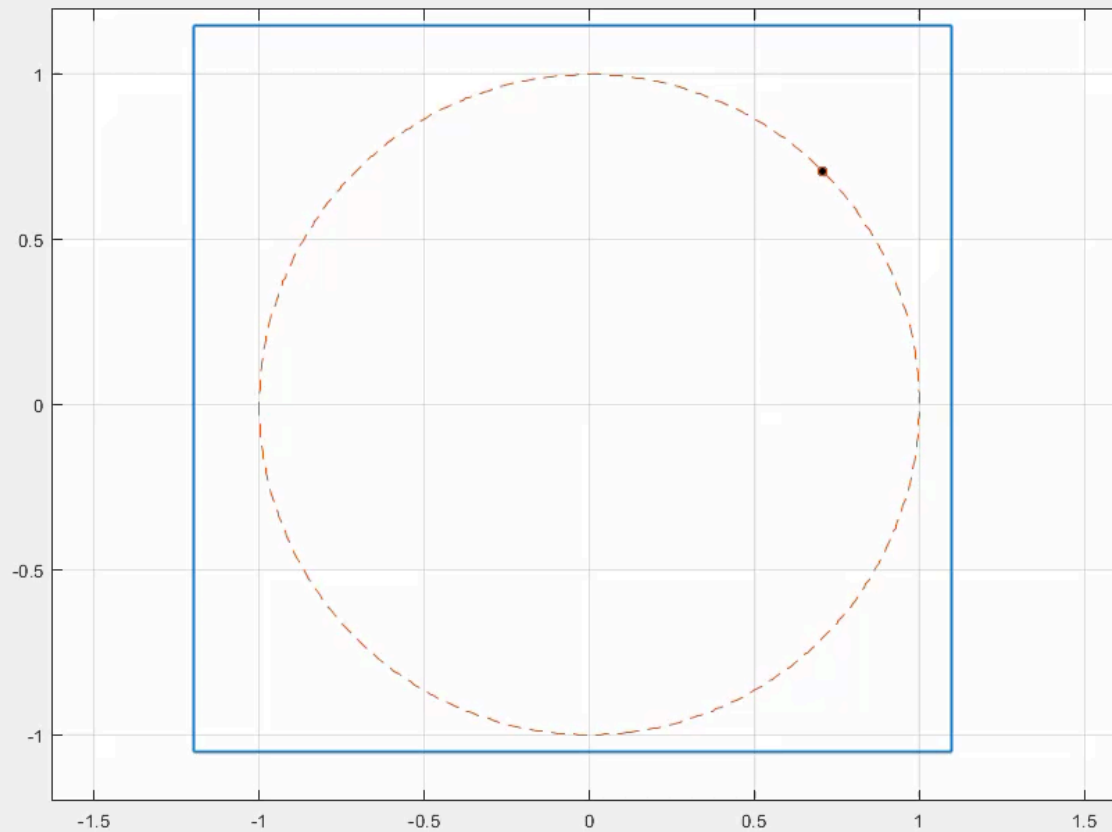
- ▶ Regard time horizon of two periods
- ▶ $N = 25$ equidistant control intervals
- ▶ use FESD with $N_{\text{FESD}} = 3$ finite elements with Radau 3 on each control interval
- ▶ each FESD interval has one constant control u and one speed of time s
- ▶ MPCC solved via ℓ_∞ penalty reformulation and homotopy
- ▶ For homotopy convergence: in total 4 NLPs solved with IPOPT via CasADi



States and controls in physical time.

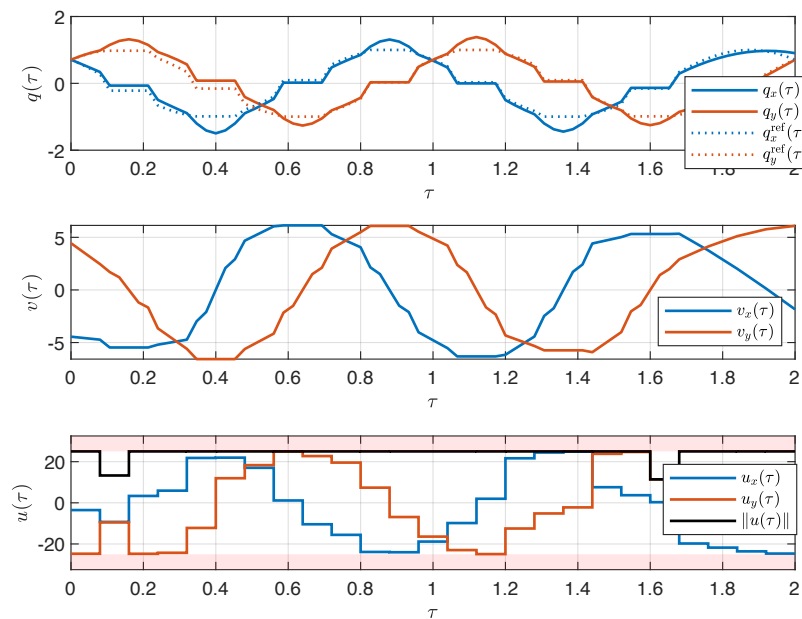
Results with slowly moving reference - Movie

For $\omega = \pi$, tracking is easy: no jumps occur in optimal solution.

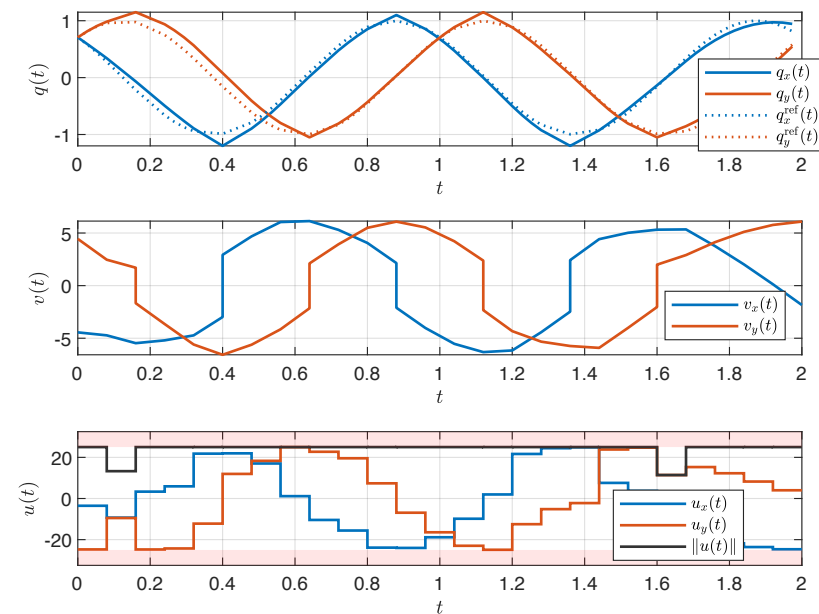


Results with fast reference

For $\omega = 2\pi$, tracking is only possible if ball bounces against walls.



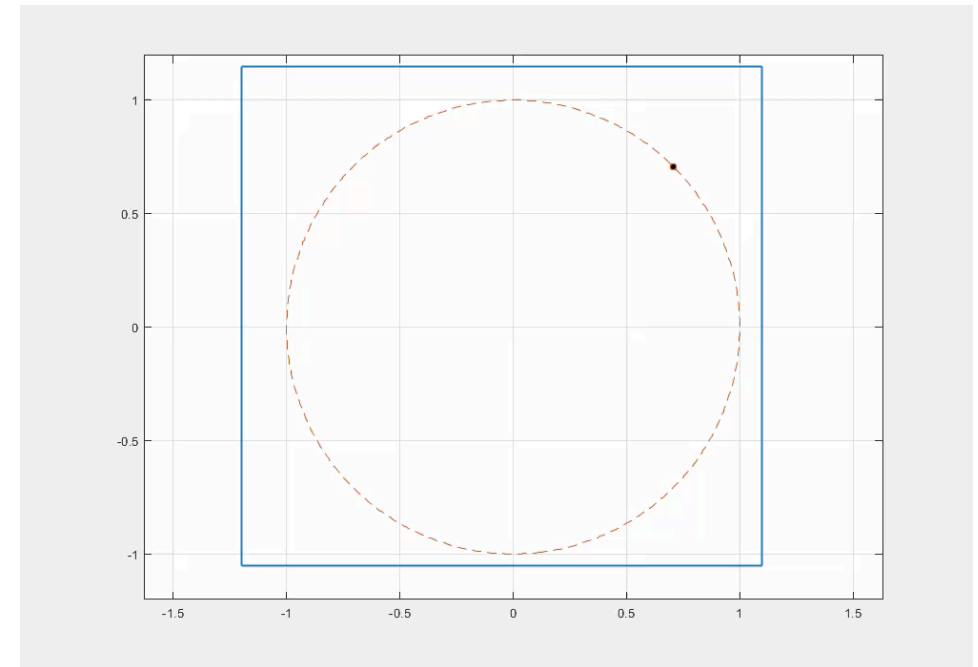
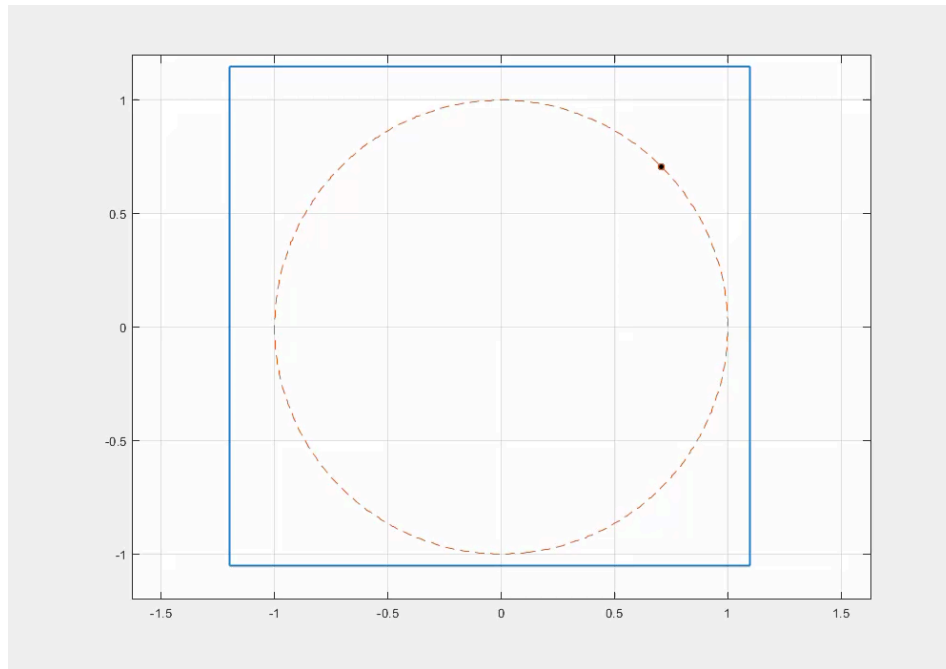
States and controls in numerical time.



States and controls in physical time time.

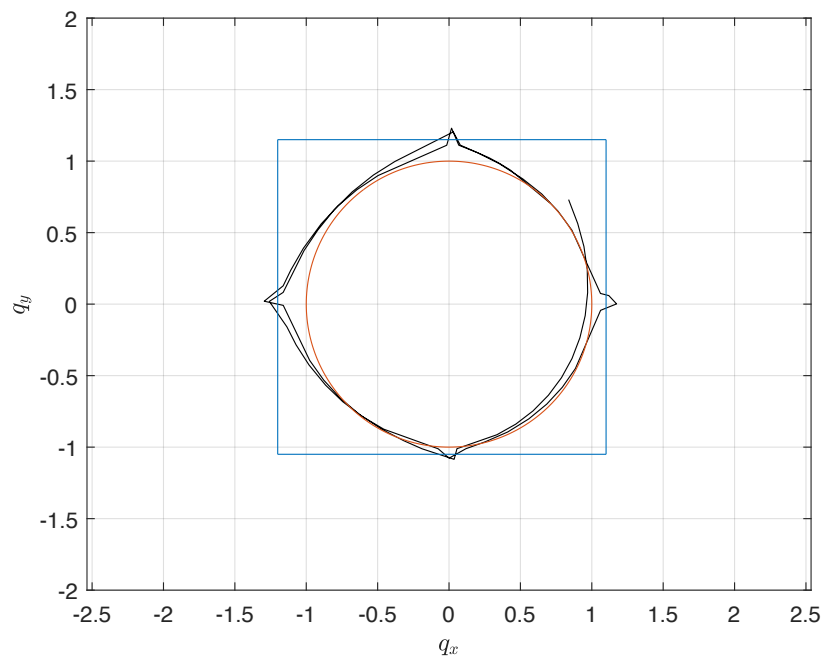
Results with fast reference - Movie

For $\omega = 2\pi$, tracking is only possible if ball bounces against walls.

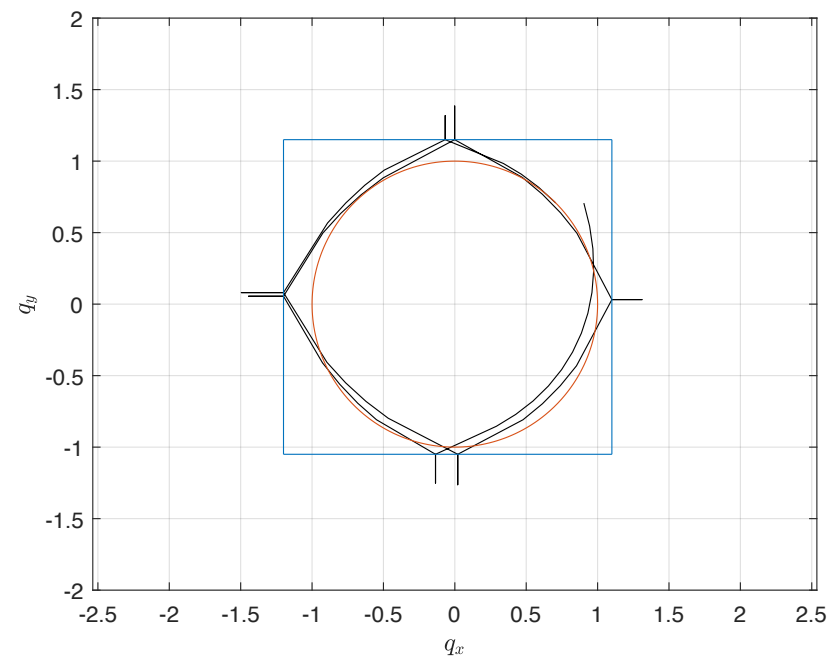


Homotopy: first iteration vs converged solution

Geometric trajectory



After the first homotopy iteration

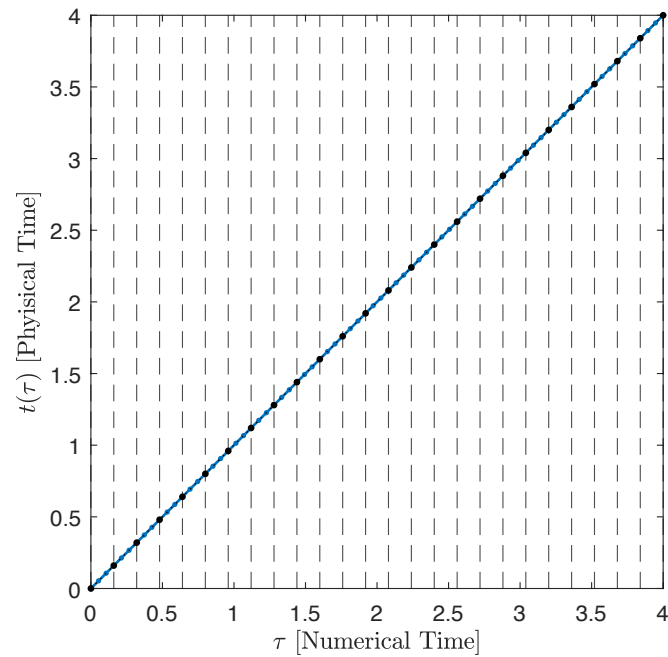


The solution trajectory after convergence

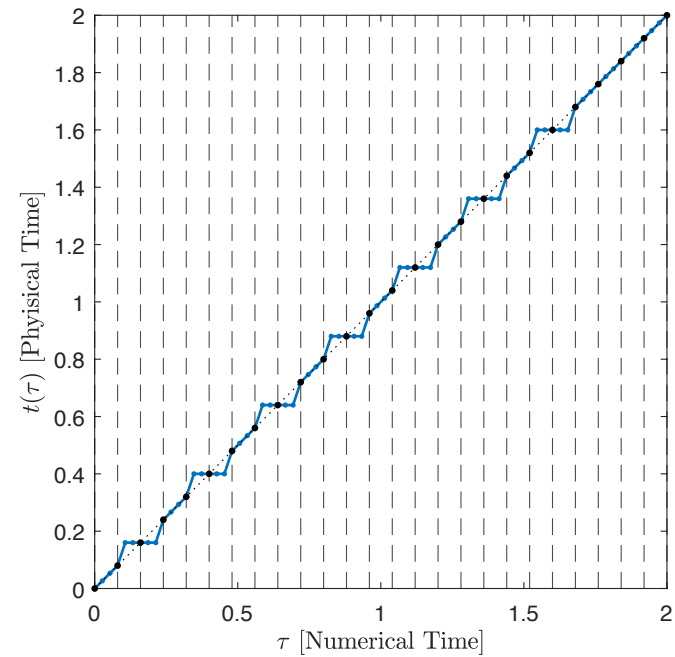
Physical vs. Numerical Time



for $\omega = \pi$

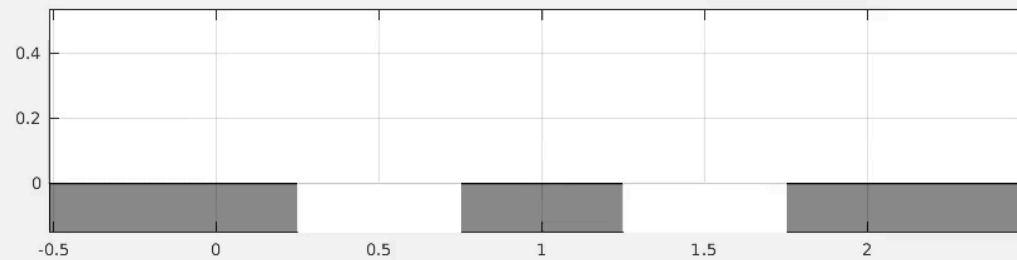


for $\omega = 2\pi$

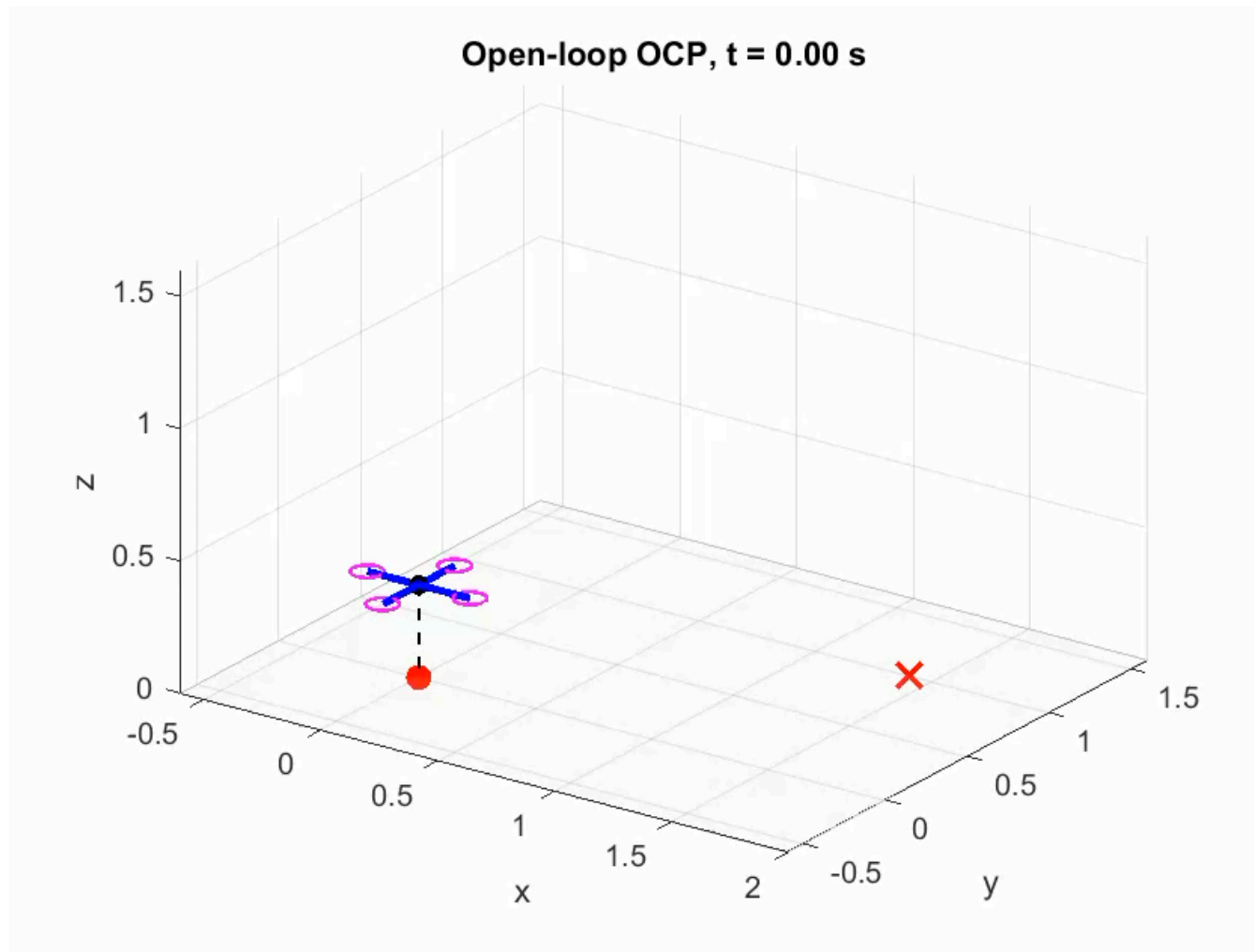


Hopping Robot - move with minimal effort from start to end position

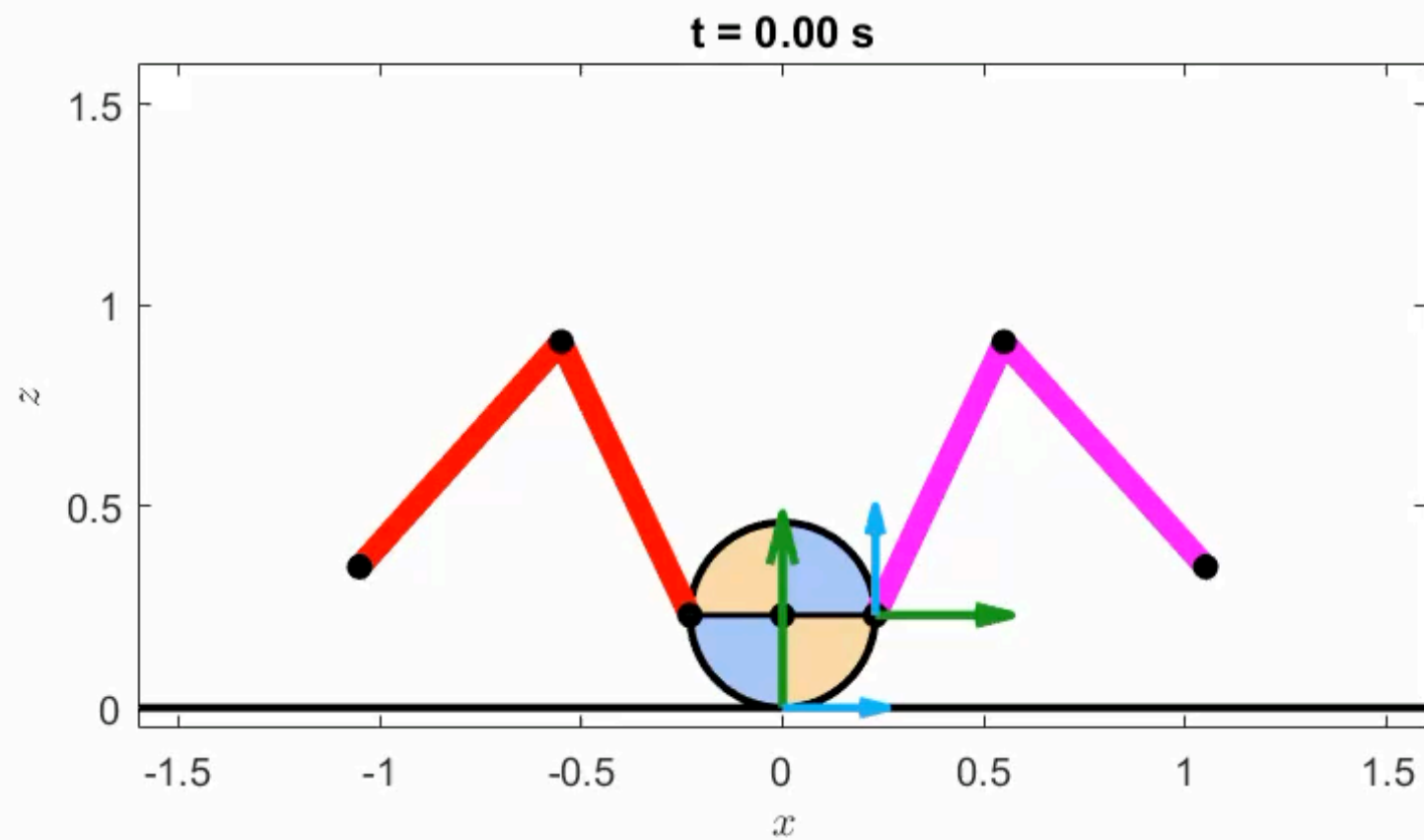
Homotopy initialized with start position everywhere. Optimizer finds creative solution. Not with FESD yet.



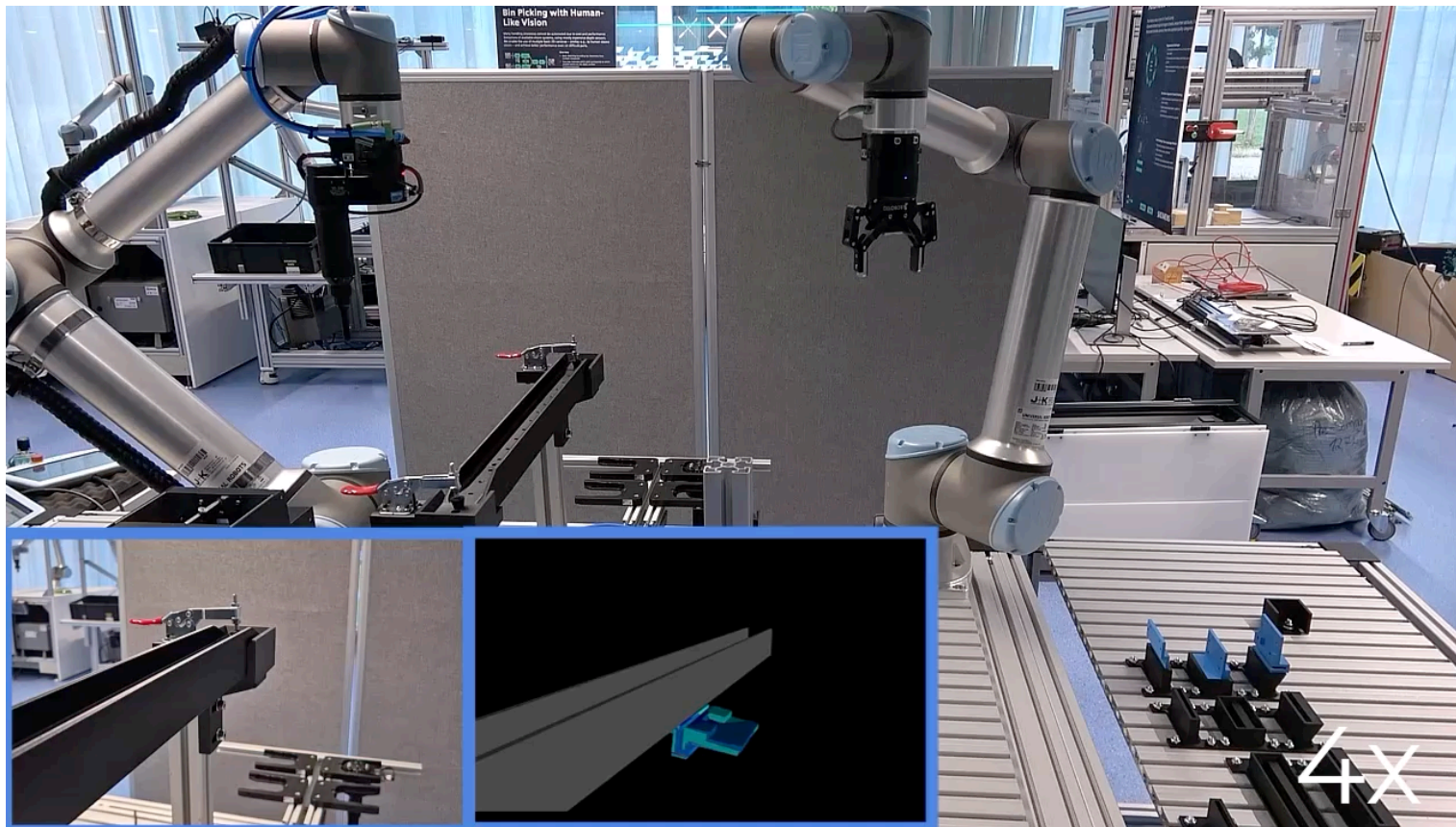
Optimal drone control:
lifting a mass on a tether with ground contacts



Optimal Control of Collaborative Manipulation



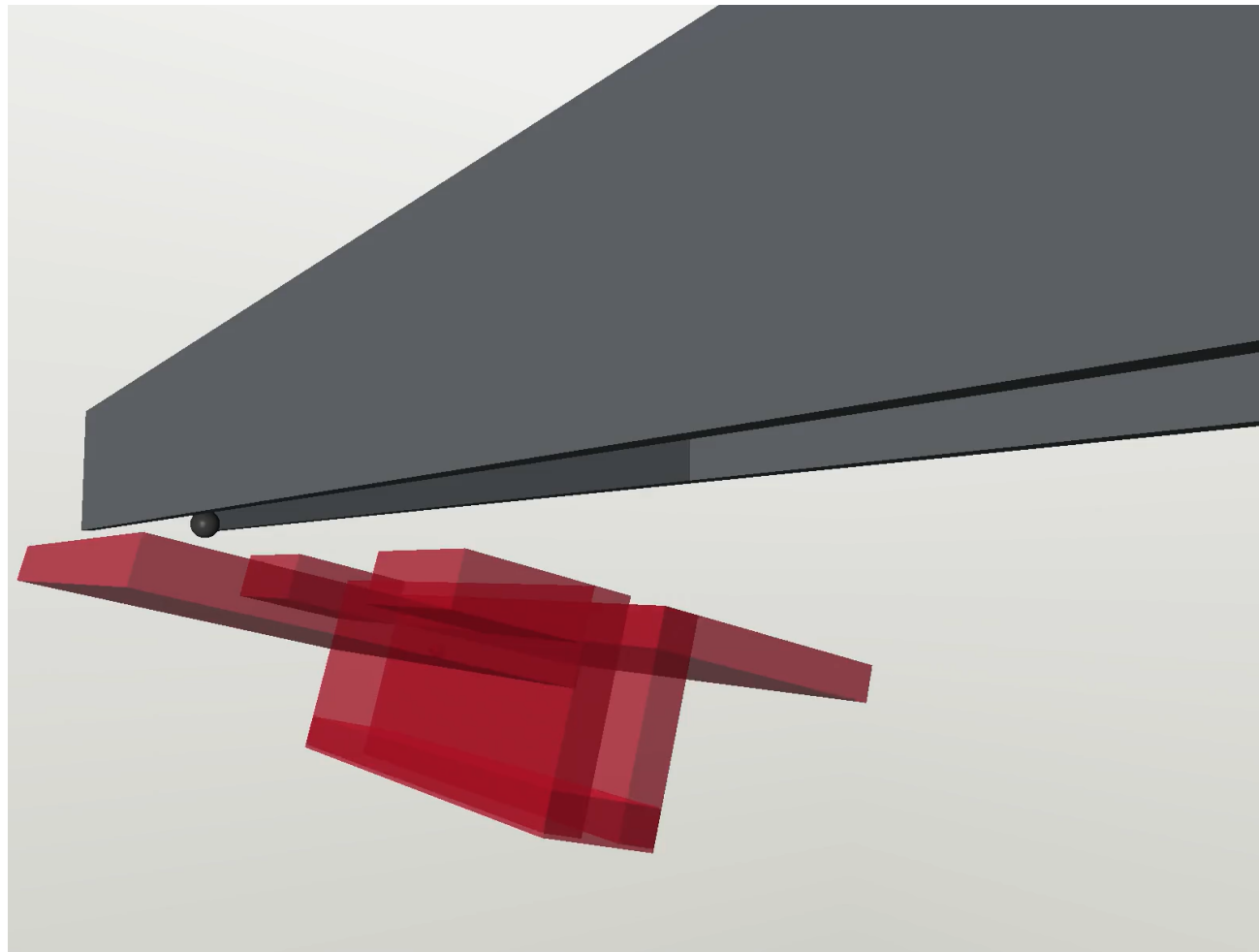
Real-World Application: Assembly Robots at Siemens in Munich (NSD3)



Christian Dietz
(MSc Mathematics)
industrial PhD
student at University
of Freiburg,
supervised by Armin
Nurkanovic,
Sebastian Albrecht,
and MD

Dream: Use Optimal Control to Move Robot to Desired End Position

(simulated solution of optimal control problem, L2-control penalty)



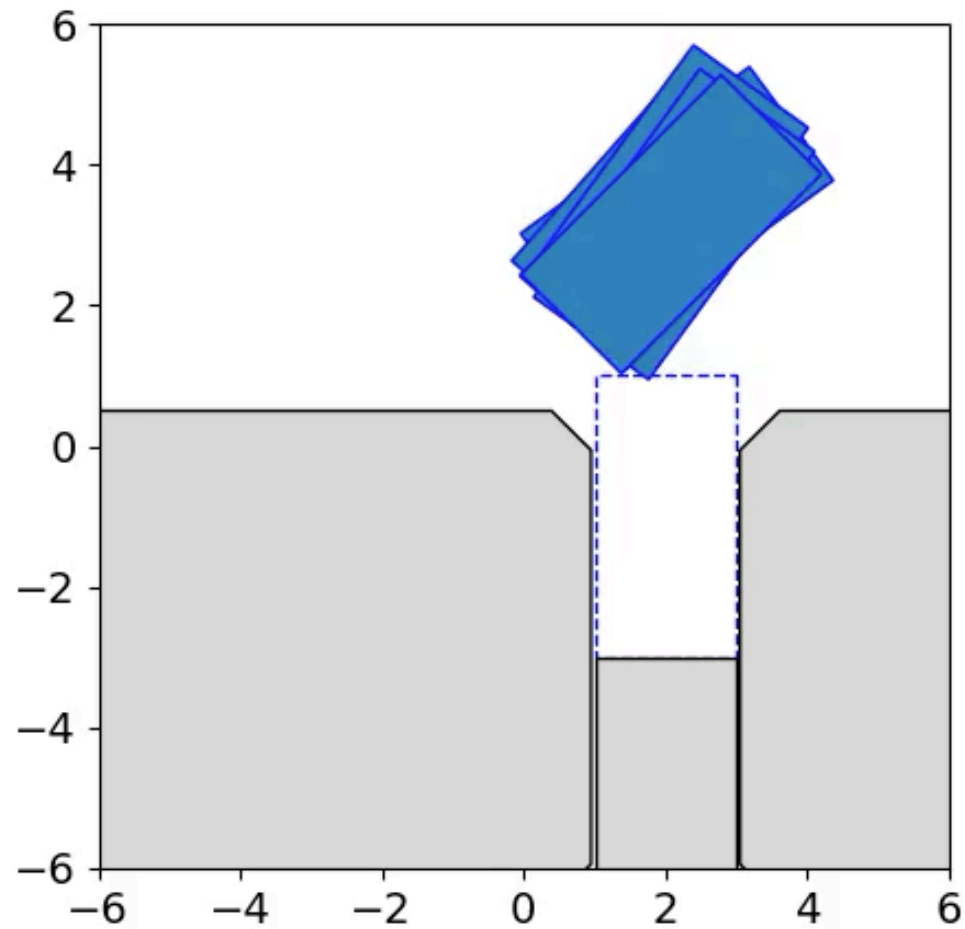
Dream: Use Optimal Control to Move Robot to Desired End Position (experiment)



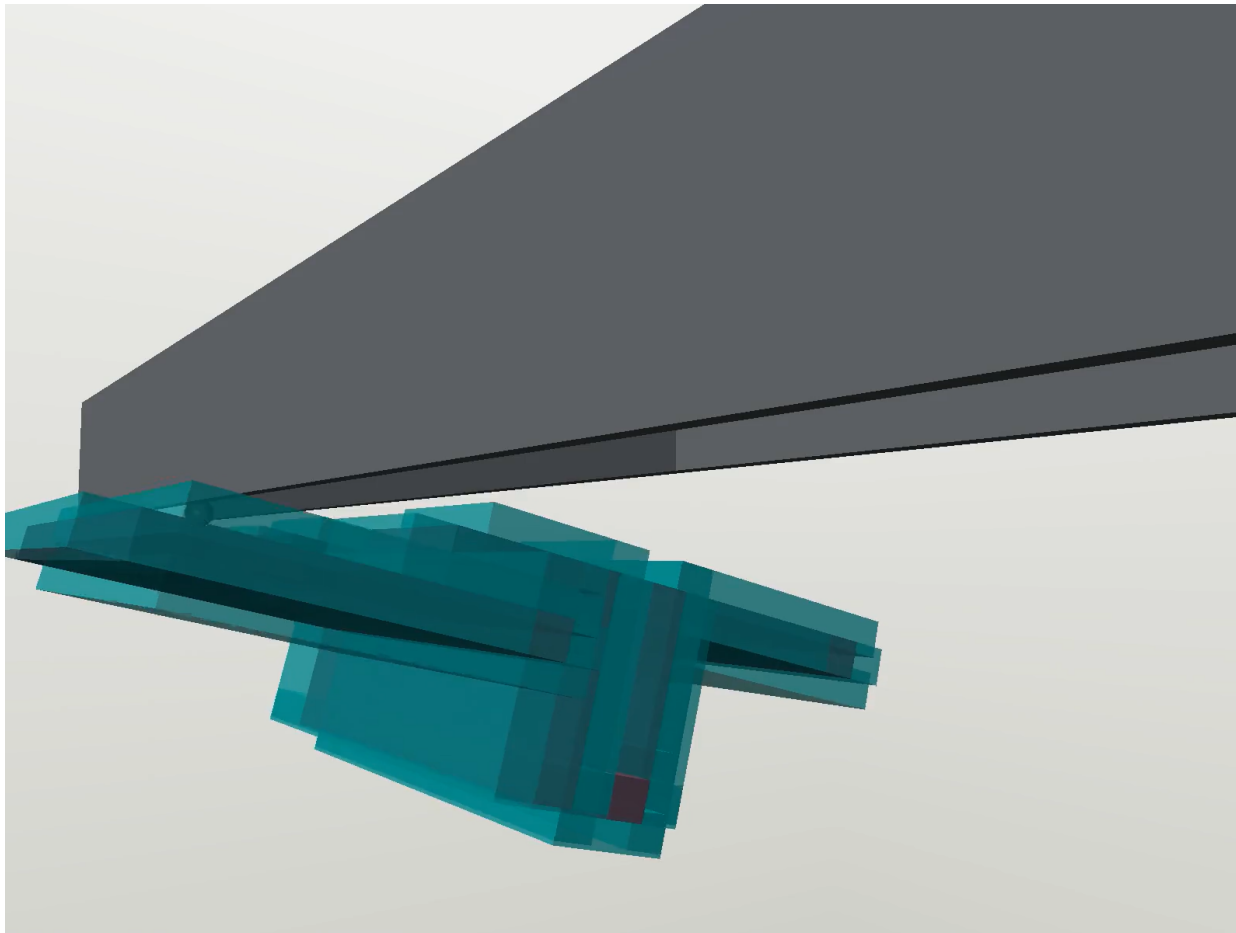
Dream: Use Optimal Control to Move Robot to Desired End Position
(experiment)



Idea: robustify motions by optimizing several reference trajectories simultaneously



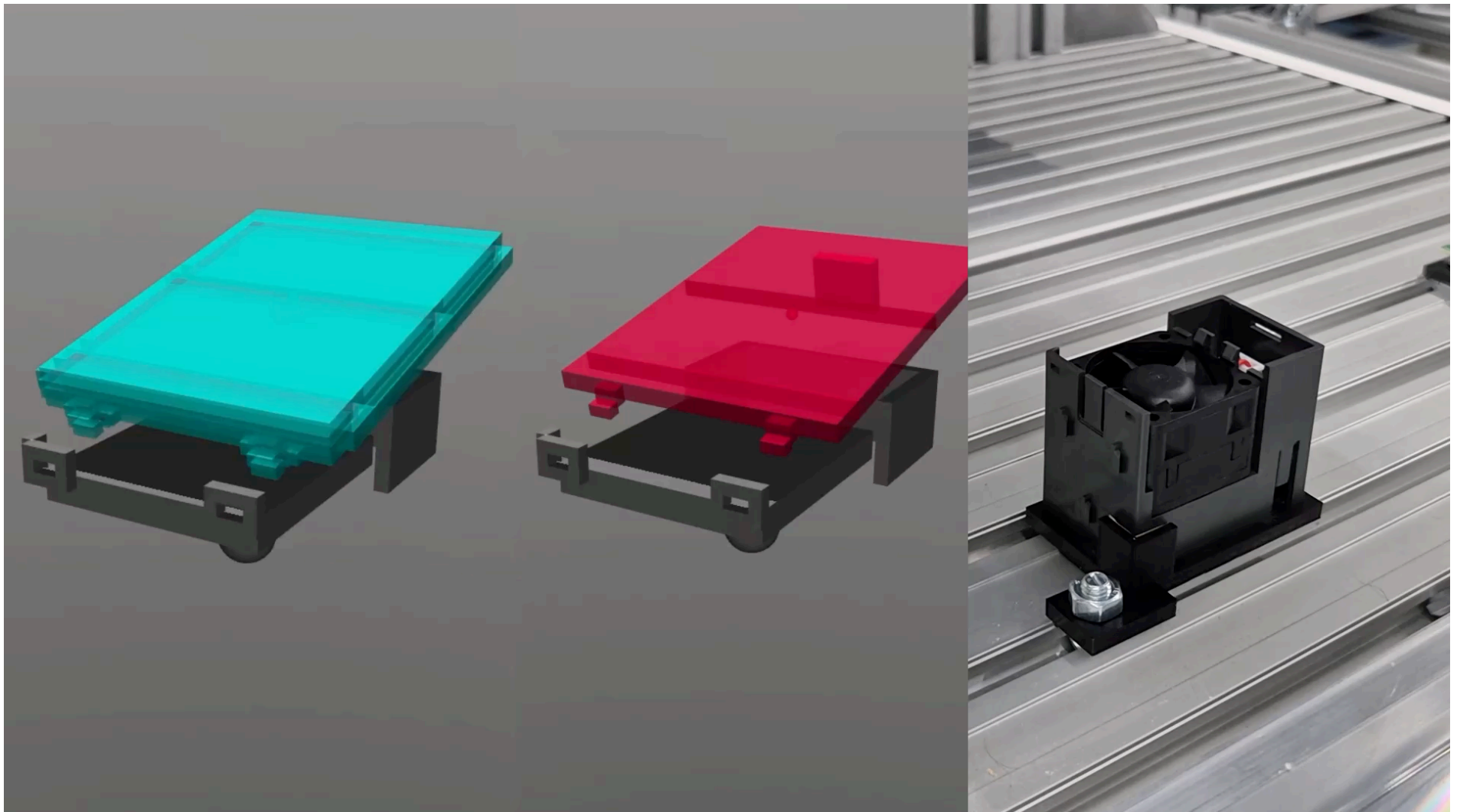
Robust Optimal Control Solution (5 scenarios) (simulation)



Robust Optimal Control Solution (5 scenarios) (experiment)



Open Loop Robustified Fan Assembly (by C. Dietz at Siemens)





- ▶ Optimization with Complementarity Constraints: Embracing the Nonconvex
- ▶ Finite Elements with Switch Detection (FESD)
- ▶ Time Freezing for State Dependent Jumps
- ▶ **Three Step Decomposition for Discrete Actuators**

Mixed Integer Optimal Control Problem with Binary Inputs $b(t)$

Formulated in outer convexified form. Can equivalently be formulated with complementarity constraints.



$$\begin{aligned} \underset{x(\cdot), u(\cdot), b(\cdot), s(\cdot)}{\text{minimize}} \quad & \int_0^T L(x, u, b, s) \, dt + M(x(T)) \end{aligned} \quad (2a)$$

$$\text{subject to} \quad x(0) = \bar{x}_0 \quad (2b)$$

$$\frac{dx}{dt} = \sum_{i=1}^{n_b} b_i \cdot f_i(x, u, c), \quad \sum_{i=1}^{n_b} b_i(t) = 1, \quad (2c)$$

$$b_i(t) \in \{0, 1\} \quad \left[\Leftrightarrow 0 \leq b_i(t) \perp (1 - b_i(t)) \geq 0 \right] \quad \text{for } i = 1, \dots, n_b, \quad (2d)$$

$$-s + r_l \leq r(x, u, b, c) \leq r_u + s, \quad \text{for } t \in [0, T] \quad (2e)$$

$$(\text{ + additional combinatorial constraints}) \quad (2f)$$

$x(t)$: states, $u(t)$: continuous controls, $b(t)$: binary controls, $s(t)$: slack variables

$c(t)$: time-varying parameters, f : system dynamics, $r_l \leq r \leq r_u$: path constraints

Discretize to obtain MINLP. Global solution usually prohibitive (cf. Ruth Misener's plenary).

Three step decomposition for fast approximate MIOCP solution



Combinatorial Integral Approximation (CIA)¹

1. Solve relaxed NLP with $b(t) \in [0, 1]^{n_b}$ to obtain relaxed solution $b^*(t)$ for $t \in [0, T]$.
2. Solve minimum distance problem to find binary trajectory $b^{**}(\cdot)$ closest to $b^*(\cdot)$.
3. Solve an NLP where the binary controls are fixed to $b^{**}(t)$, to adjust $x(\cdot)$ and $u(\cdot)$.

Distance function in Step 2 is the "CIA distance" which measures the maximum of the integral of the difference of the trajectories. Fast tailored solvers for this special problem – an MILP – exist, e.g. in the python package pycombina [Bürger 2019].

¹S. Sager, M. Jung, and C. Kirches: Combinatorial Integral Approximation, Mathematical Methods of Operations Research, vol. 73, no. 3, pp. 363-380, 2011.

NMPC for a solar thermal test plant

at Karlsruhe University of Applied Sciences, with two discrete actuators



Control cabinet, cold storage, ACM, hot storage, pumps (cellar)



Plate collectors (roof)



Recooling unit (roof)



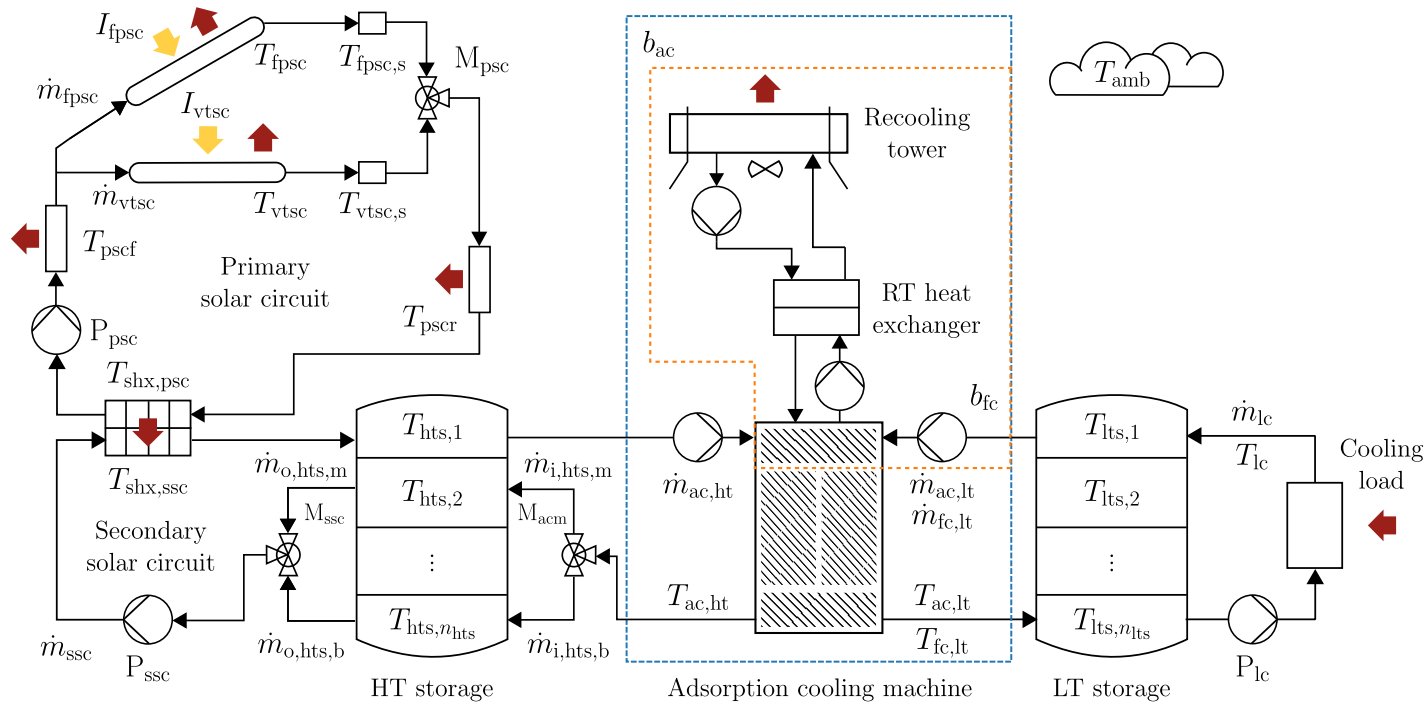
Ambient sensors (roof)



Vacuum tube collectors (roof)

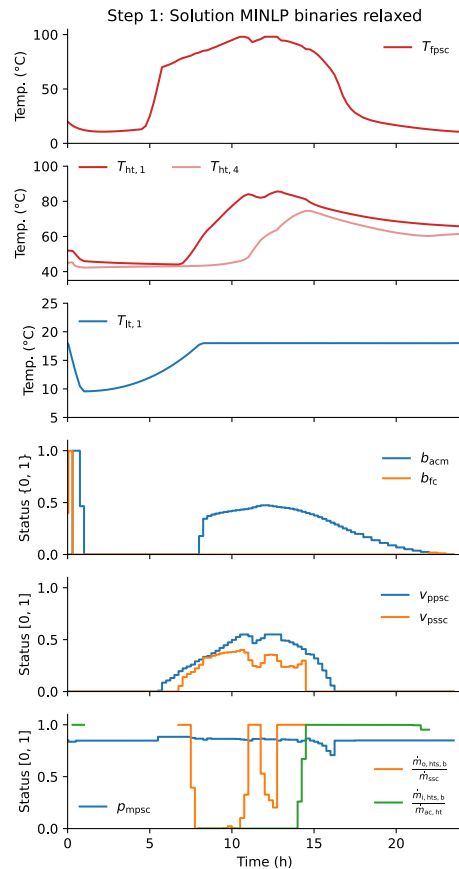
Control-oriented modeling

Schematic depiction of the system model

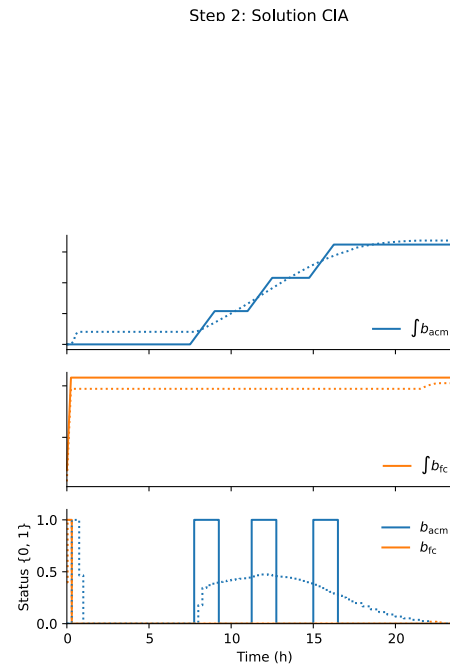


Nonlinear switched system ODE model with $n_x = 20$, $n_b = 2$, $n_u = 5$, and $n_c = 4$,
differentiable in all arguments within the domain of interest

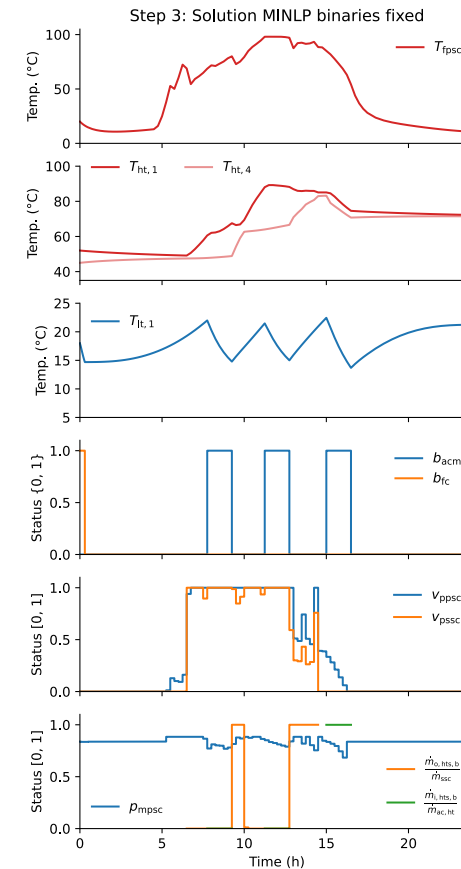
Numerical results: Three Step CIA Decomposition



(25 CPU sec)

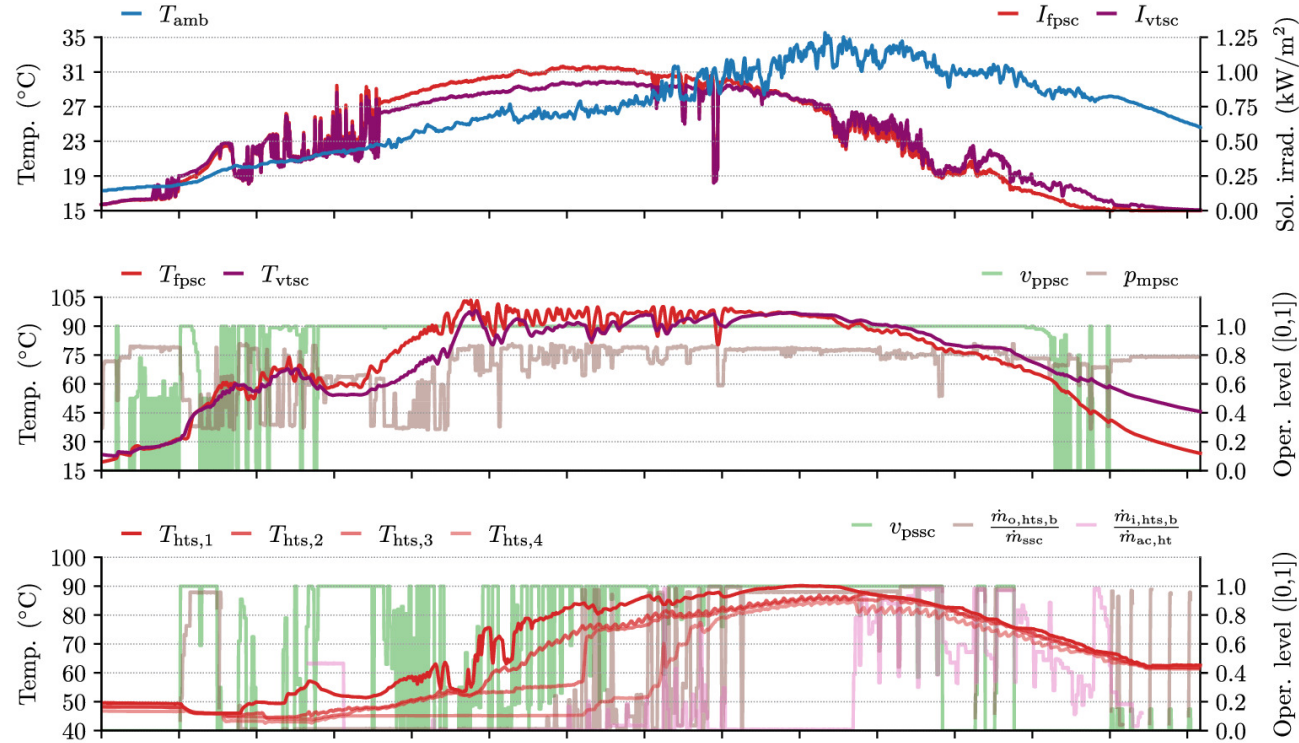


(0.02 CPU sec)



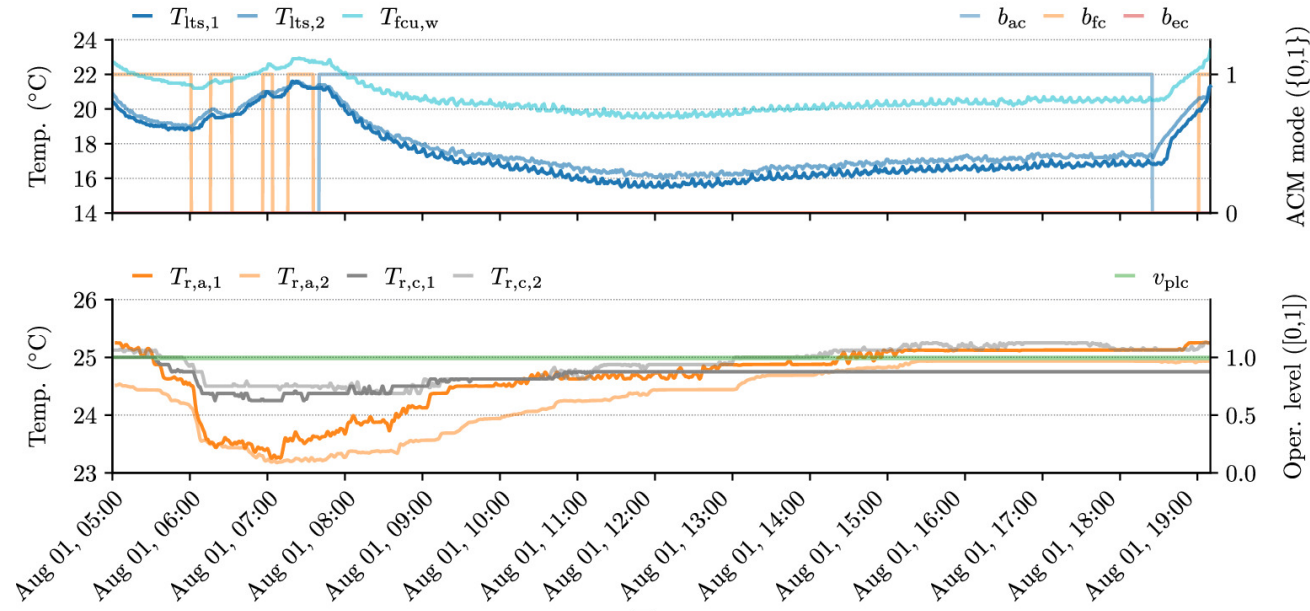
(18 CPU sec)

Experimental Results from Aug 1st, 2019



Every 2 minutes, a new nonlinear mixed integer optimal control problem is solved, using a parallel real-time algorithm based on CasADi, IPOPT [Wächter and Biegler 2006], and Pycombina [Bürger et al, 2019], an implementation of the combinatorial integral approximation (CIA) method [Sager 2009].

Experimental Results from Aug 1st, 2019



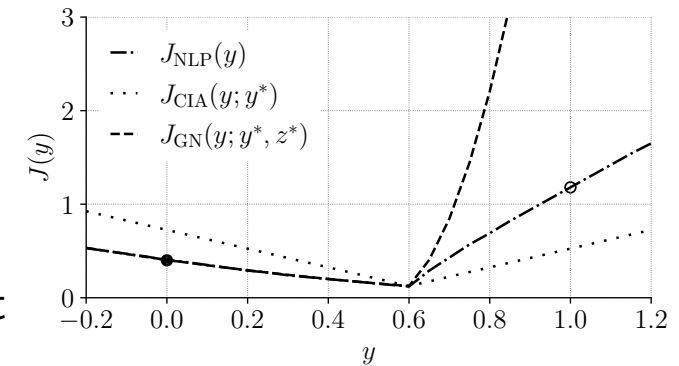
Every 2 minutes, a new nonlinear mixed integer optimal control problem is solved, using a parallel real-time algorithm based on CasADi, IPOPT [Wächter and Biegler 2006], and Pycombina [Bürger et al, 2019], an implementation of the combinatorial integral approximation (CIA) method [Sager 2009].

Alternative to CIA Decomposition: Gauss-Newton based MIQP

[Bürger et al., Automatica, 2023]



- ▶ Derive convex Gauss-Newton-type approximation of original MINLP from linearization at relaxed MINLP solution.
- ▶ Solution of resulting MIQP can yield improved integer solution in terms of objective and feasibility of the original MINLP.
- ▶ MIQP is equivalent to minimization of a distance function that is a first order accurate approximation of the true objective.



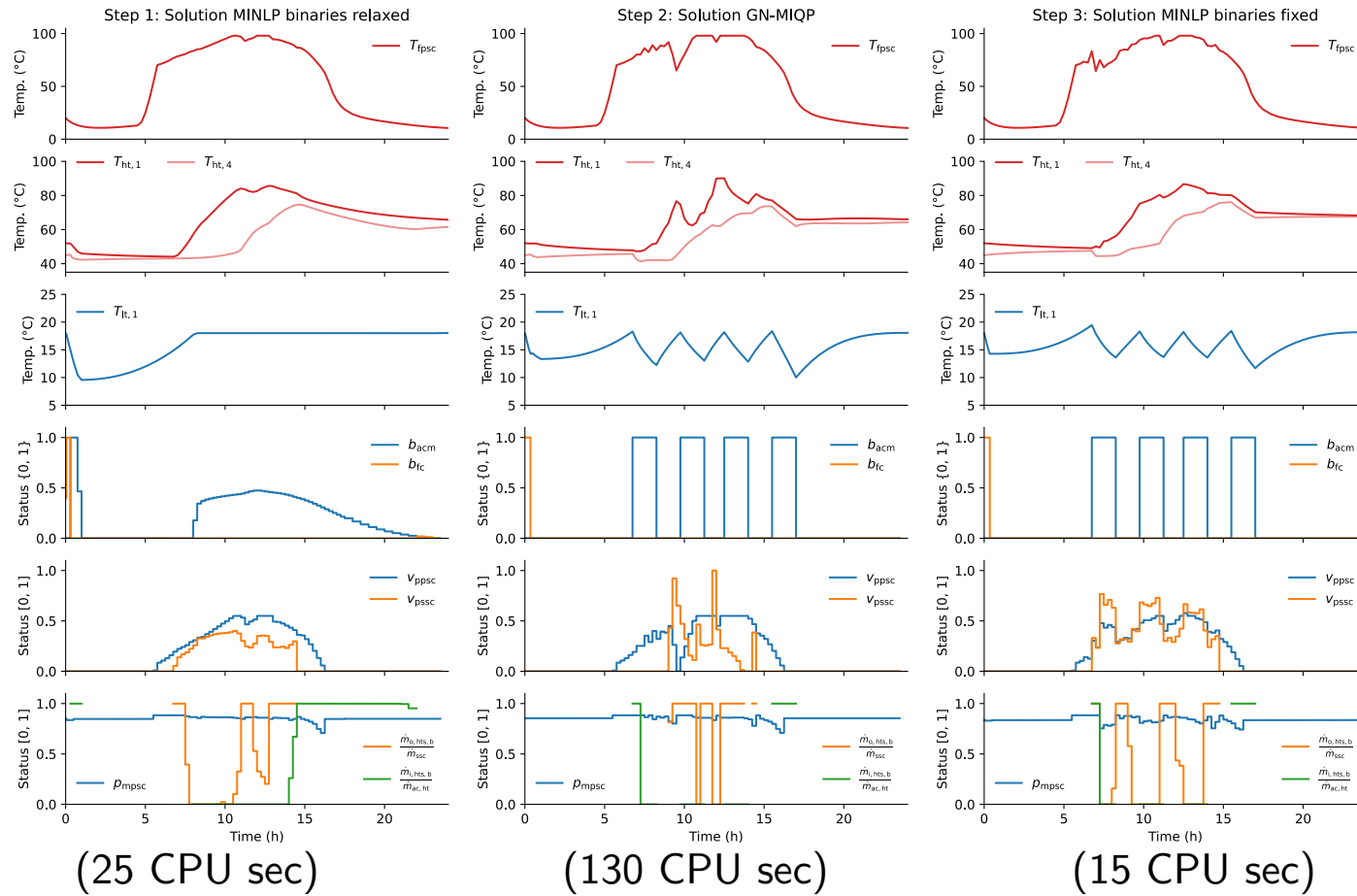
Original MINLP

$$\begin{aligned} \min_{x,y} \quad & \frac{1}{2} \|F_1(x, y)\|_2^2 + f_2(x, y) \\ \text{s. t.} \quad & g(x, y) \leq 0 \\ & h(x, y) = 0 \\ & y \in \mathbb{Z}^{n_y} \end{aligned}$$

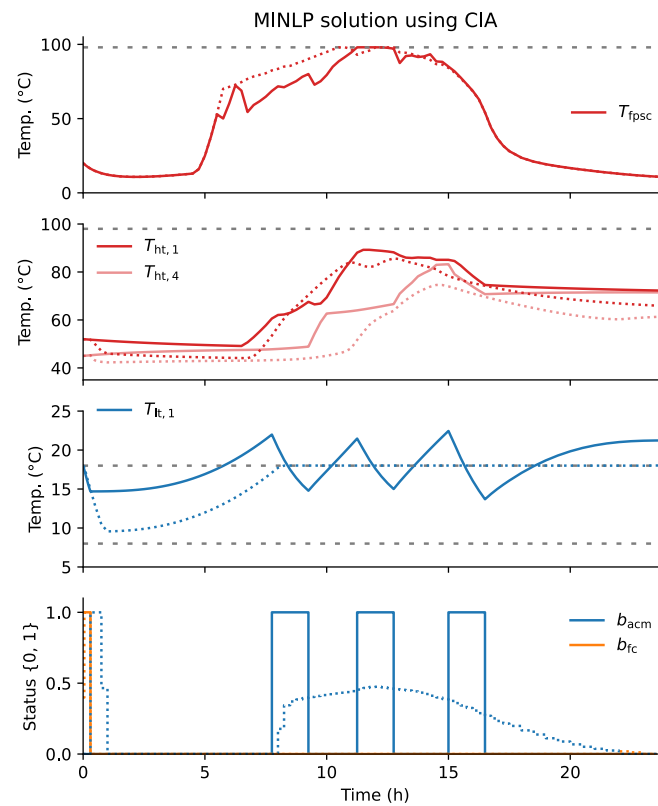
GN-MIQP from linearization at (x^*, y^*)

$$\begin{aligned} \min_{x,y} \quad & \frac{1}{2} \|F_{1,L}(x, y; x^*, y^*)\|_2^2 + f_{2,L}(x, y; x^*, y^*) \\ \text{s. t.} \quad & g_L(x, y; x^*, y^*) \leq 0 \\ & h_L(x, y; x^*, y^*) = 0 \\ & y \in \mathbb{Z}^{n_y} \end{aligned}$$

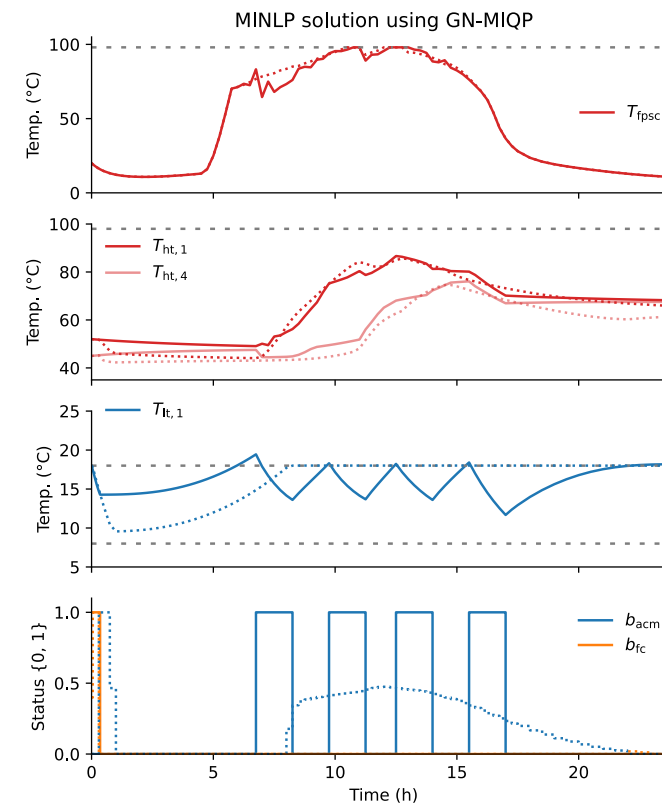
Numerical results: Three Step GN-MIQP Decomposition



Comparison of CIA and GN-MIQP Solution



(43 CPU sec)



(170 CPU sec)

GN-MIQP delivers significant feasibility improvements, at the expense of increased computational cost.



- ▶ Mathematical Programs with Complementarity Constraints (MPCC) are a powerful tool to formulate and solve nonsmooth and nonconvex optimization problems.
- ▶ Finite Elements with Switch Detection (FESD) allow highly accurate simulation and optimal control for switched systems of level NSD2.
- ▶ Time-Freezing allows us to transform systems with state jumps of level NSD3 to the easier level NSD2 (which can be treated with FESD).
- ▶ NMPC with discrete actuators can efficiently be addressed by a three-step decomposition method. In Step 2, either a cheap MILP or a more accurate MIQP can be solved.

References 1 (own, most related to this talk)



- ▶ Survey of Sequential Convex Programming and Generalized Gauss-Newton Methods. F. Messerer, K. Baumgärtner and M. Diehl, ESAIM: Proc. Surv., accepted, 2022.
- ▶ A Sequential Convex Programming Approach to Solving Quadratic Programs and Optimal Control Problems with Linear Complementarity Constraints. J. Hall, A. Nurkanovic, F. Messerer and M. Diehl, in IEEE Cont. Sys. Lett., 2022.
- ▶ A time-freezing approach for numerical optimal control of nonsmooth differential equations with state jumps. A. Nurkanovic, T. Sartor, S. Albrecht, and M. Diehl, IEEE Cont. Sys. Lett., 2021.
- ▶ Design, implementation and simulation of an MPC algorithm for switched nonlinear systems under combinatorial constraints. A. Bürger, C. Zeile, A. Altmann-Dieses, S. Sager, M. Diehl. Journal of Process Control, 2019.
- ▶ A Gauss-Newton-based Decomposition Algorithm for Nonlinear Mixed-Integer Optimal Control Problems. A. Bürger, C. Zeile, A. Altmann-Dieses, S. Sager, M. Diehl., submitted to CDC, 2021.

References 2 (external, most related to this talk)



- ▶ A. F. Filippov, Differential equations with discontinuous right-hand side, Matematicheskii sbornik, 1960.
- ▶ B. Baumrucker and L. Biegler, MPEC strategies for optimization of a class of hybrid dynamic systems, Journal of Process Control, 2009.
- ▶ D. E. Stewart, A high accuracy method for solving ODEs with discontinuous right-hand side, Numerische Mathematik, 1990.
- ▶ A. Wächter and L. T. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, Mathematical Programming, 2006.
- ▶ S. Sager, M. Jung, C. Kirches. Combinatorial Integral Approximation, Mathematical Methods of Operations Research, 2011.



References 3 (own, related to this talk, software)

- ▶ pycombina: An Open-Source Tool for Solving Combinatorial Approximation Problems Arising in Mixed-Integer Optimal Control. A. Bürger, C. Zeile, M. Hahn, A. Altmann-Dieses, S. Sager, M. Diehl Proceedings of the IFAC World Congress, 2020).
- ▶ CasADi: a software framework for nonlinear optimization and optimal control. J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, Mathematical Programming Computation, 2019.
- ▶ qpOASES: A parametric active-set algorithm for quadratic programming, Mathematical Programming Computation. H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, vol. 6, no. 4, pp. 327363, 2014.
- ▶ acados: a modular open-source framework for fast embedded optimal control. R. Verschueren, G. Frison, D. Kouzoupis, J. Frey, N. van Duijkeren, A. Zanelli, B. Novoselnik, T. Albin, R. Quirynen, and M. Diehl, Math. Prog. Comp. con. acc., 2022.

References 4 (own, related to this talk)



- ▶ Limits of MPCC Formulations in Direct Optimal Control with Nonsmooth Differential Equations.
A. Nurkanovic, S. Albrecht, and M. Diehl, European Control Conference (ECC), 2020.
- ▶ Continuous Control Set Nonlinear Model Predictive Control of Reluctance Synchronous Machines. A. Zanelli, J. Kullick, H. Eldeeb, G. Frison, C. Hackl, and M. Diehl, IEEE Trans. Cont. Sys. Tech., 2021.
- ▶ Experimental operation of a solar-driven climate system with thermal energy storages using mixed-integer nonlinear model predictive control.
A. Bürger, D. Bull, P. Sawant, M. Bohlayer, A. Klotz, D. Beschtz, A. Altmann-Dieses, M. Braun, M. Diehl, Optimal Control Applications and Methods, 2021.
- ▶ A whole-year simulation study on nonlinear mixed-integer model predictive control for a thermal energy supply system with multi-use components. A. Bürger, M. Bohlayer, S. Hoffmann, A. Altmann-Dieses, M. Braun, M. Diehl, Applied Energy, 2020.

References 5 (external, related to this talk)



- ▶ D. E. Stewart and M. Anitescu, Optimal control of systems with discontinuous differential equations, Numerische Mathematik, 2010.
- ▶ M. Ferris and F. Tin-Loi, On the solution of a minimum weight elastoplastic problem involving displacement and complementarity constraints, Computer Methods in Applied Mechanics and Engineering, 1999.
- ▶ D. Ralph and S. J. Wright, Some properties of regularization and penalization schemes for MPECs, Optimization Methods and Software, 2004.
- ▶ S. Sager. Reformulations and algorithms for the optimization of switching decisions in nonlinear optimal control, Journal of Process Control, 2009.
- ▶ A. Lekic, B. Hermans, N. Jovicic, P. Patrinos, Microsecond nonlinear model predictive control for DC-DC converters, Int J Circ Theor Appl., 2020.
- ▶ T. Hausberger, A. Kugi, A. Eder, W. Kemmetmüller, High-speed nonlinear model predictive control of an interleaved switching DC/DC-converter, Cont. Eng. Pract., 2020.

Thank you very much for your attention!