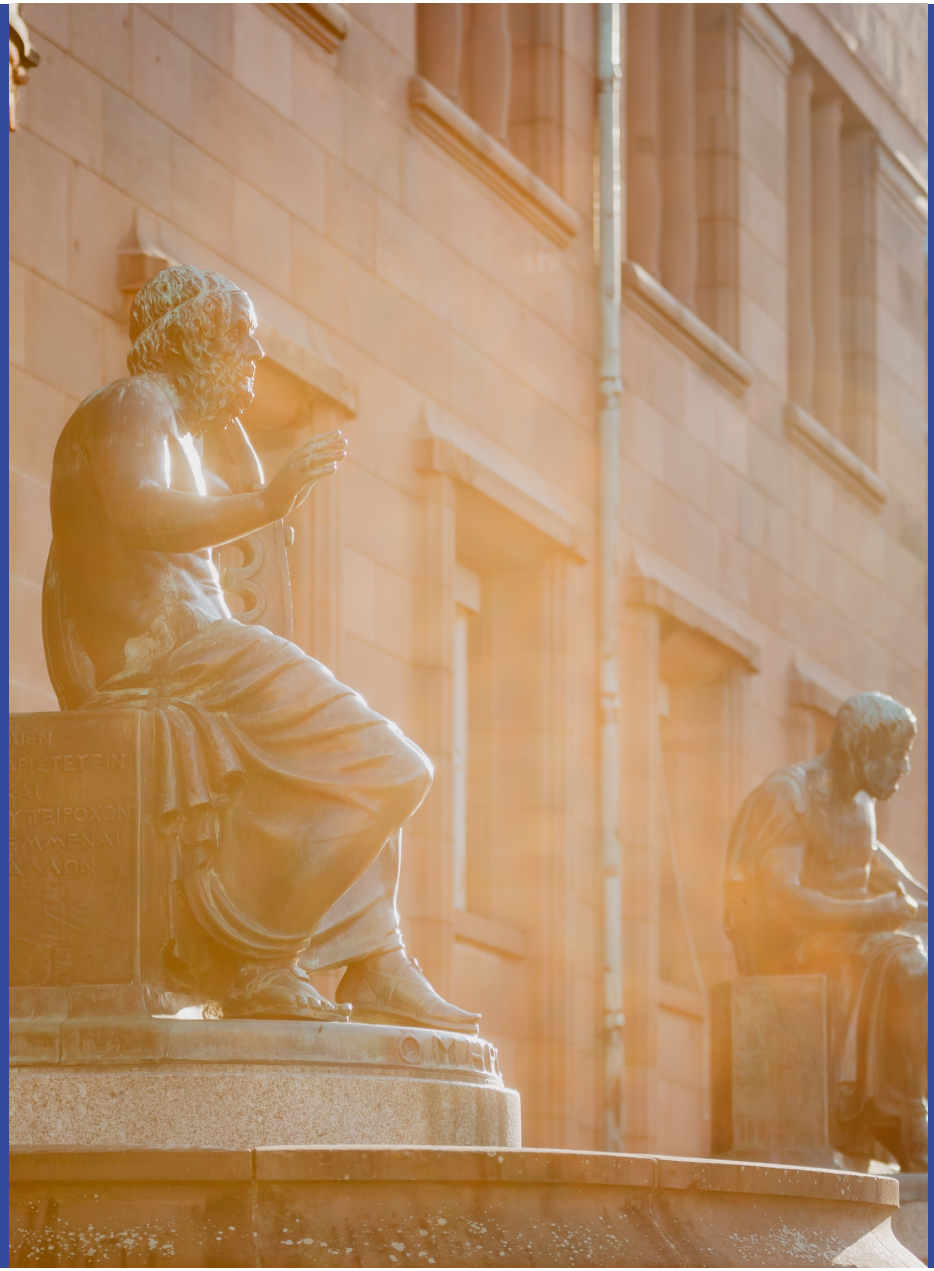
universität freiburg

Real-Time Algorithms for Nonlinear Model Predictive Control - Part II

Moritz Diehl

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

MPCRL26 Fall School Freiburg, Oct. 12, 2026



Continuous-Time Optimal Control Problems (OCP)



Continuous-Time OCP with Ordinary Differential Equation (ODE) Constraints

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

Continuous-Time Optimal Control Problems (OCP)



Continuous-Time OCP with Ordinary Differential Equation (ODE) Constraints

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

Can in most applications assume convexity of all "outer" problem functions: L_{c}, E, h, r.

Three Levels of Difficulty in Continuous-Time OCP



Continuous-Time OCP

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

Three levels of difficulty:

Three Levels of Difficulty in Continuous-Time OCP



Continuous-Time OCP

$$\min_{x(\cdot),u(\cdot)} \int_0^T L_c(x(t),u(t))\,\mathrm{d}t + E(x(T))$$

$$\text{s.t.} \quad x(0) = \bar{x}_0$$

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

$$0 \geq h(x(t),u(t)),\ t \in [0,T]$$

$$0 \geq r(x(T))$$

Three levels of difficulty:

(a) Linear ODE: $f(x,u) = Ax + Bu$

Three Levels of Difficulty in Continuous-Time OCP



Continuous-Time OCP

$$\min_{x(\cdot),u(\cdot)} \int_0^T L_c(x(t),u(t))\,\mathrm{d}t + E(x(T))$$

$$\text{s.t.} \quad x(0) = \bar{x}_0$$

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

$$0 \geq h(x(t),u(t)),\ t \in [0,T]$$

$$0 \geq r(x(T))$$

Three levels of difficulty:

(a) Linear ODE: $f(x,u) = Ax + Bu$

(b) Nonlinear smooth ODE: $f \in \mathcal{C}^1$

Three Levels of Difficulty in Continuous-Time OCP



Continuous-Time OCP

$$\min_{x(\cdot),u(\cdot)} \int_0^T L_c(x(t),u(t))\,\mathrm{d}t + E(x(T))$$

$$\text{s.t.} \quad x(0) = \bar{x}_0$$
$$\dot{x}(t) = f(x(t),u(t))$$
$$0 \geq h(x(t),u(t)),\ t \in [0,T]$$
$$0 \geq r(x(T))$$

Three levels of difficulty:

(a) Linear ODE: $f(x,u) = Ax + Bu$

(b) Nonlinear smooth ODE: $f \in \mathcal{C}^1$

(c) **Nonsmooth Dynamics (NSD):**

- f not differentiable (NSD1),
- f not continuous (NSD2), or even
- f not finite valued, discontinuous state $x(t)$ (NSD3)

Three Levels of Difficulty in Continuous-Time OCP



Continuous-Time OCP

$$\min_{x(\cdot),u(\cdot)} \int_0^T L_c(x(t),u(t))\,\mathrm{d}t + E(x(T))$$

$$\text{s.t.} \quad x(0) = \bar{x}_0$$
$$\dot{x}(t) = f(x(t),u(t))$$
$$0 \geq h(x(t),u(t)),\ t \in [0,T]$$
$$0 \geq r(x(T))$$

Three levels of difficulty:

(a) Linear ODE: $f(x,u) = Ax + Bu$

(b) Nonlinear smooth ODE: $f \in \mathcal{C}^1$

(c) **Nonsmooth Dynamics (NSD):**

- f not differentiable (NSD1),
- f not continuous (NSD2), or even
- f not finite valued, discontinuous state $x(t)$ (NSD3)

focus on smooth cases (a) and (b).

Three Levels of Difficulty in Continuous-Time OCP



Continuous-Time OCP

$$\min_{x(\cdot),u(\cdot)} \int_0^T L_c(x(t),u(t))\,\mathrm{d}t + E(x(T))$$

$$\text{s.t.} \quad x(0) = \bar{x}_0$$

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

$$0 \geq h(x(t),u(t)),\ t \in [0,T]$$

$$0 \geq r(x(T))$$

Three levels of difficulty:

(a) Linear ODE: $f(x,u) = Ax + Bu$

(b) Nonlinear smooth ODE: $f \in \mathcal{C}^1$

focus on smooth cases (a) and (b).

Recall: Runge-Kutta Discretization for Smooth Systems



Ordinary Differential Equation (ODE)

$$\dot{x}(t) = \underbrace{f(x(t), u(t))}_{=:v(t)}$$

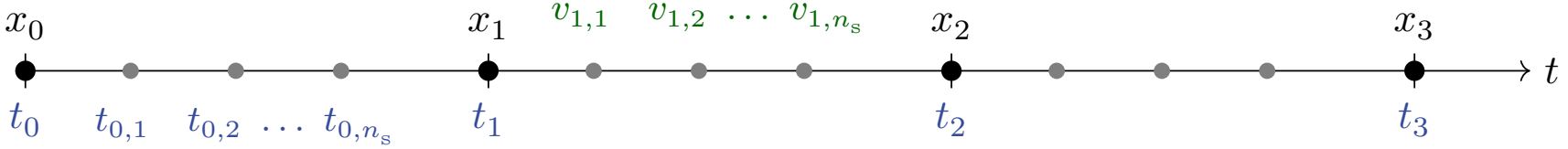
Initial Value Problem (IVP)

$$\begin{aligned} x(0) &= \bar{x}_0 \\ v(t) &= f(x(t), u(t)) \\ \dot{x}(t) &= v(t) \\ t &\in [0, T] \end{aligned}$$

Discretization: N Runge-Kutta steps of each n_s stages

$$\begin{aligned} x_{0,0} &= \bar{x}_0, & \Delta t &= \tfrac{T}{N} \\ v_{k,j} &= f(x_{k,j}, u_k) \\ x_{k,j} &= x_{k,0} + \Delta t \textstyle\sum_{n=1}^{n_s} a_{jn} v_{k,n} \\ x_{k+1,0} &= x_{k,0} + \Delta t \textstyle\sum_{n=1}^{n_s} b_n v_{k,n} \\ j &= 1, \ldots, n_s, \quad k = 0, \ldots, N-1 \end{aligned}$$

For fixed controls and initial value: square system with $n_x + N(2n_s + 1)n_x$ unknowns, implicitly defined via $n_x + N(2n_s + 1)n_x$ equations.
(trivial eliminations in case of explicit RK methods)



Direct Methods Transform OCP into Nonlinear Program (NLP)



Continuous time OCP

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

- Direct methods "first discretize, then optimize"

Direct Methods Transform OCP into Nonlinear Program (NLP)



Continuous time OCP

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

- Direct methods "first discretize, then optimize"

1. Parameterize controls, e.g. $u(t) = u_n, t \in [t_n, t_{n+1}]$.

Direct Methods Transform OCP into Nonlinear Program (NLP)



Continuous time OCP

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

- Direct methods "first discretize, then optimize"

1. Parameterize controls, e.g. $u(t) = u_n, t \in [t_n, t_{n+1}]$.
2. Discretize cost and dynamics

$$L_{\mathrm{d}}(x_n, z_k, u_n) \approx \int_{t_n}^{t_{n+1}} L_{\mathrm{c}}(x(t),u(t))\,\mathrm{d}t$$

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

$$\begin{aligned} x_{n+1} &= \phi_f(x_n, z_n, u_n) \\ 0 &= \phi_{\mathrm{int}}(x_n, z_n, u_n)\end{aligned}$$

Direct Methods Transform OCP into Nonlinear Program (NLP)



Continuous time OCP

$$\min_{x(\cdot),u(\cdot)} \int_0^T L_{\mathrm{c}}(x(t),u(t))\,\mathrm{d}t + E(x(T))$$
$$\text{s.t.} \quad x(0) = \bar{x}_0$$
$$\dot{x}(t) = f(x(t),u(t))$$
$$0 \geq h(x(t),u(t)),\ t \in [0,T]$$
$$0 \geq r(x(T))$$

- Direct methods "first discretize, then optimize"

1. Parameterize controls, e.g. $u(t) = u_n, t \in [t_n, t_{n+1}]$.
2. Discretize cost and dynamics

$$L_{\mathrm{d}}(x_n, z_k, u_n) \approx \int_{t_n}^{t_{n+1}} L_{\mathrm{c}}(x(t),u(t))\,\mathrm{d}t$$

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

$$x_{n+1} = \phi_f(x_n, z_n, u_n)$$
$$0 = \phi_{\mathrm{int}}(x_n, z_n, u_n)$$

3. Also discretize path constraints

$$0 \geq \phi_h(x_n, z_n, u_n),\ n = 0, \ldots N-1.$$

Direct Methods Transform OCP into Nonlinear Program (NLP)



Continuous time OCP

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

- Direct methods "first discretize, then optimize"

Discrete time OCP (an NLP)

$$\begin{aligned}\min_{\mathbf{x},\mathbf{z},\mathbf{u}} \quad & \textstyle\sum_{k=0}^{N-1} L_{\mathrm{d}}(x_k,z_k,u_k) + E(x_N) \\ \text{s.t.} \quad & x_0 = \bar{x}_0 \\ & x_{n+1} = \phi_f(x_n,z_n,u_n) \\ & 0 = \phi_{\mathrm{int}}(x_n,z_n,u_n) \\ & 0 \geq \phi_h(x_n,z_n,u_n),\ n = 0,\ldots,N-1 \\ & 0 \geq r(x_N)\end{aligned}$$

Variables $\mathbf{x} = (x_0,\ldots,x_N)$, $\mathbf{z} = (z_0,\ldots,z_N)$ and $\mathbf{u} = (u_0,\ldots,u_{N-1})$.
Here, $\mathbf{z}$ are the intermediate variables of the integrator (e.g. Runge-Kutta)

Simplest Direct Transcription: Single Step Explicit Euler

(not recommended in practice, other Runge-Kutta methods are much more efficient)



Continuous time OCP

$$\min_{x(\cdot),u(\cdot)} \int_0^T L_c(x(t),u(t))\,\mathrm{d}t + E(x(T))$$

$$\begin{aligned} \text{s.t.} \quad x(0) &= \bar{x}_0 \\ \dot{x}(t) &= f(x(t),u(t)) \\ 0 &\geq h(x(t),u(t)),\ t\in[0,T] \\ 0 &\geq r(x(T)) \end{aligned}$$

- Direct methods: first discretize, then optimize

Single Step Explicit Euler NLP, with $\Delta t = \frac{T}{N}$

$$\min_{\mathbf{x},\mathbf{u}} \sum_{k=0}^{N-1} L_c(x_k,u_k)\Delta t + E(x_N)$$

$$\begin{aligned} \text{s.t.} \quad x_0 &= \bar{x}_0 \\ x_{n+1} &= x_n + f(x_n,u_n)\Delta t \\ 0 &\geq h(x_n,u_n),\ n=0,\ldots,N-1 \\ 0 &\geq r(x_N) \end{aligned}$$

Variables $\mathbf{x} = (x_0,\ldots,x_N)$ and $\mathbf{u} = (u_0,\ldots,u_{N-1})$.
(single step explicit Euler has no internal integrator variables $\mathbf{z}$)

2nd Simplest Direct Transcription: Midpoint Rule



Continuous time OCP

$$\min_{x(\cdot),u(\cdot)} \int_0^T L_c(x(t),u(t))\,\mathrm{d}t + E(x(T))$$

$$\begin{aligned} \text{s.t.} \quad x(0) &= \bar{x}_0 \\ \dot{x}(t) &= f(x(t),u(t)) \\ 0 &\geq h(x(t),u(t)),\ t\in[0,T] \\ 0 &\geq r(x(T)) \end{aligned}$$

Midpoint Rule NLP, with $\Delta t = \frac{T}{N}$

$$\min_{\mathbf{x},\mathbf{z},\mathbf{u}} \sum_{k=0}^{N-1} L_c(z_k,u_k)\Delta t + E(x_N)$$

$$\begin{aligned} \text{s.t.} \quad x_0 &= \bar{x}_0 \\ x_{n+1} &= x_n + f(z_n,u_n)\Delta t \\ 0 &= z_n - \frac{x_n + x_{n+1}}{2} \\ 0 &\geq h(z_n,u_n),\ n=0,\ldots,N-1 \\ 0 &\geq r(x_N) \end{aligned}$$

Variables $\mathbf{x} = (x_0,\ldots,x_N)$, $\mathbf{z} = (z_0,\ldots,z_{N-1})$, and $\mathbf{u} = (u_0,\ldots,u_{N-1})$.

Sparse NLP resulting from direct transcription



Discrete time OCP (an NLP)

$$
\begin{aligned}
\min_{\mathbf{x},\mathbf{z},\mathbf{u}} \ & \textstyle\sum_{k=0}^{N-1} L_{\mathrm{d}}(x_k, z_n, u_k) + E(x_N) \\
\text{s.t.} \quad & x_0 = \bar{x}_0 \\
& x_{n+1} = \phi_f(x_n, z_n, u_n) \\
& 0 = \phi_{\mathrm{int}}(x_n, z_n, u_n) \\
& 0 \geq \phi_h(x_n, z_n, u_n), \ n = 0, \ldots, N-1 \\
& 0 \geq r(x_N)
\end{aligned}
$$

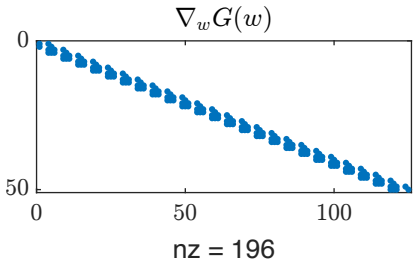
Variables $w = (\mathbf{x}, \mathbf{z}, \mathbf{u})$

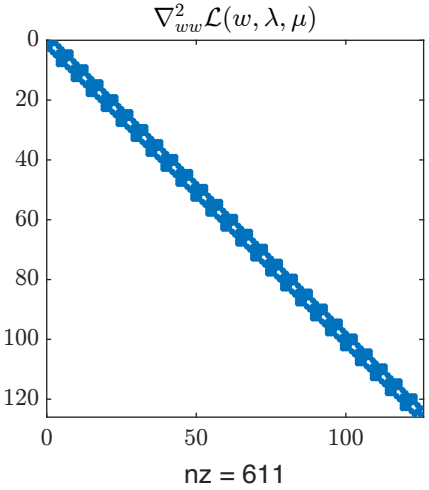
Nonlinear Program (NLP)

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

Large and sparse NLP

Sparse NLP resulting from direct transcription





Variables $w = (\mathbf{x}, \mathbf{z}, \mathbf{u})$

Nonlinear Program (NLP)

$$\min_{w \in \mathbb{R}^{n_x}} \; F(w)$$
$$\text{s.t. } G(w) = 0$$
$$H(w) \geq 0$$

Large and sparse NLP

Illustrative example of direct collocation with Newton-type optimization

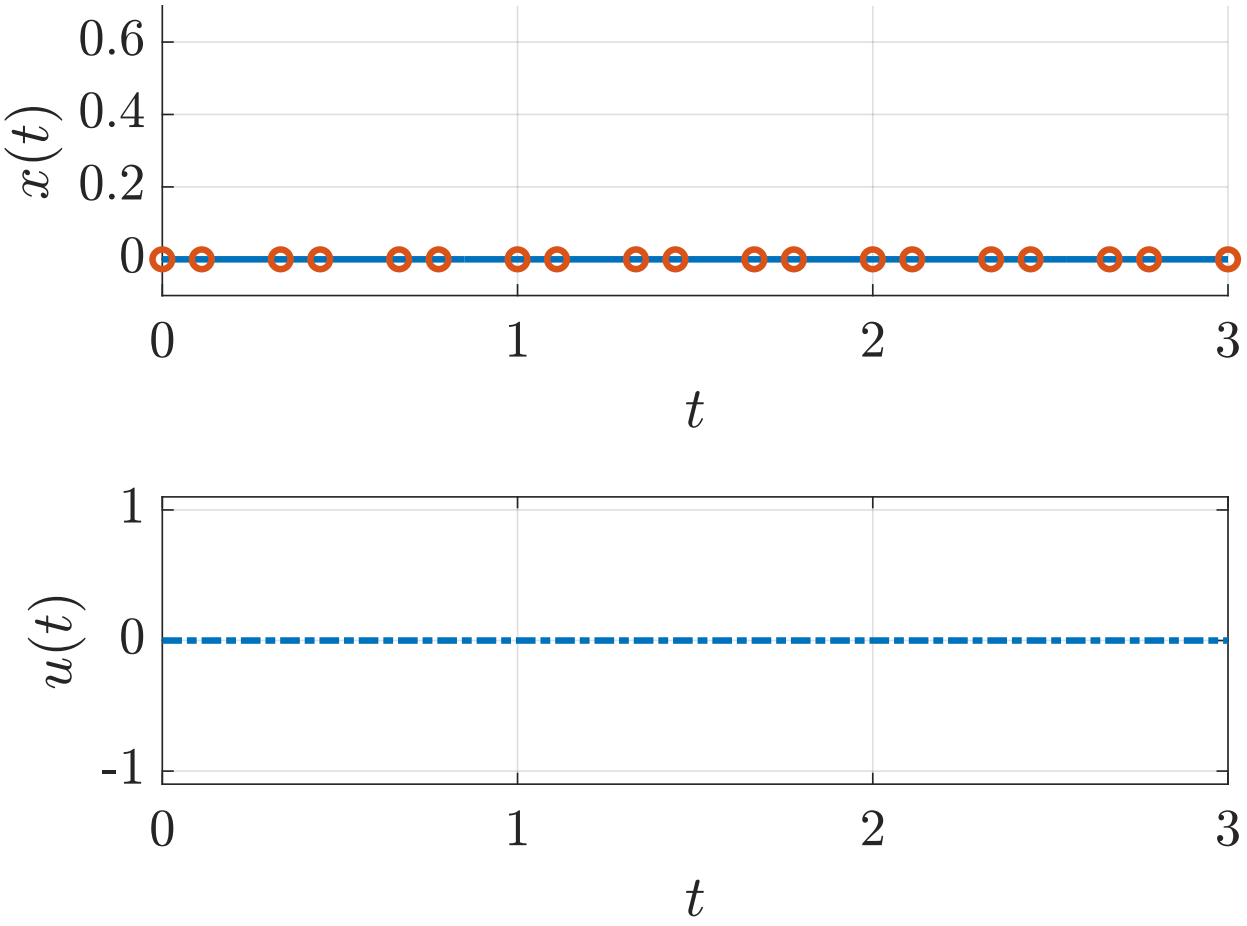


Illustrative nonlinear optimal control problem (with one state and one control)

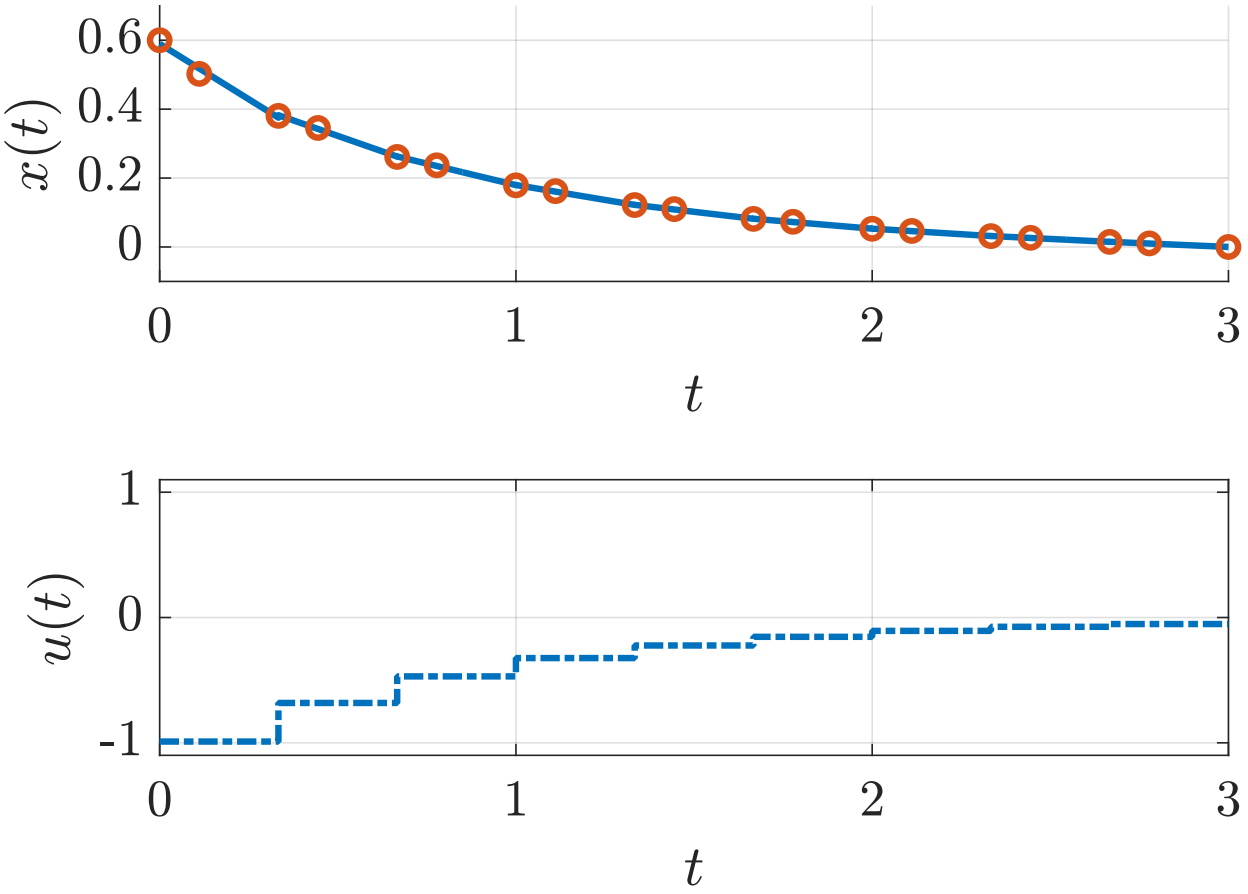
$$
\begin{aligned}
&\underset{x(\cdot),u(\cdot)}{\text{minimize}} && \int_0^3 x(t)^2 + u(t)^2 \, dt && \\
&\text{subject to} && && \\
&\qquad x(0) = \bar{x}_0 && && \text{(initial value, } \bar{x}_0 = 0.6\text{)} \\
&\qquad \dot{x} = (1+x)x + u, && && \text{(ODE model)} \\
&\qquad -1 \le u(t) \le 1, && t \in [0,3] && \text{(bounds)} \\
&\qquad x(3) = 0 && && \text{(terminal constraint)}
\end{aligned}
$$

- choose $N = 9$ equal intervals and Radau-IIA collocation with $n_s = 2$ stages
- obtain nonlinear program with $n_x + (2n_s + 1)Nn_x + Nn_u$ variables
- initialize with zeros everywhere, solve with CasADi and Ipopt (interior point)

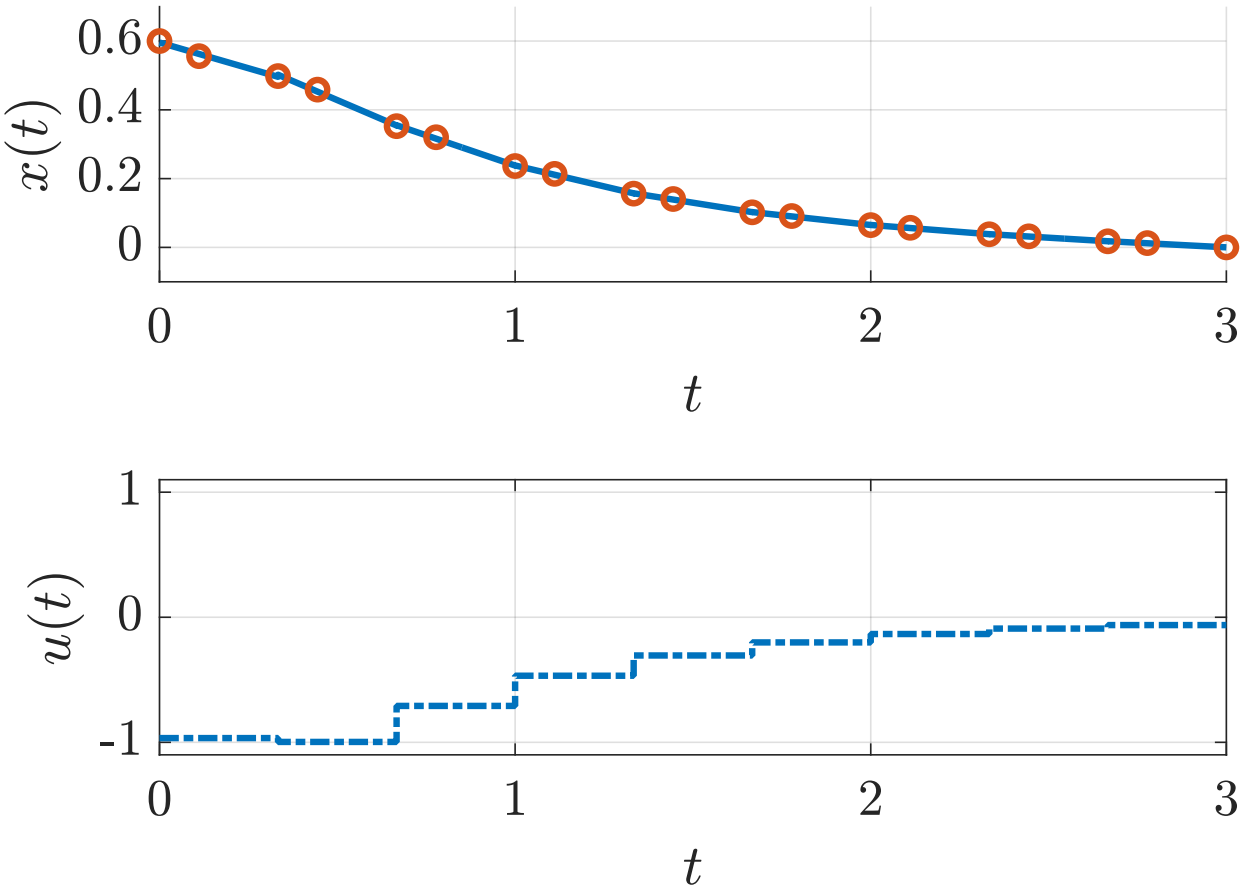
Illustrative example: Initialization



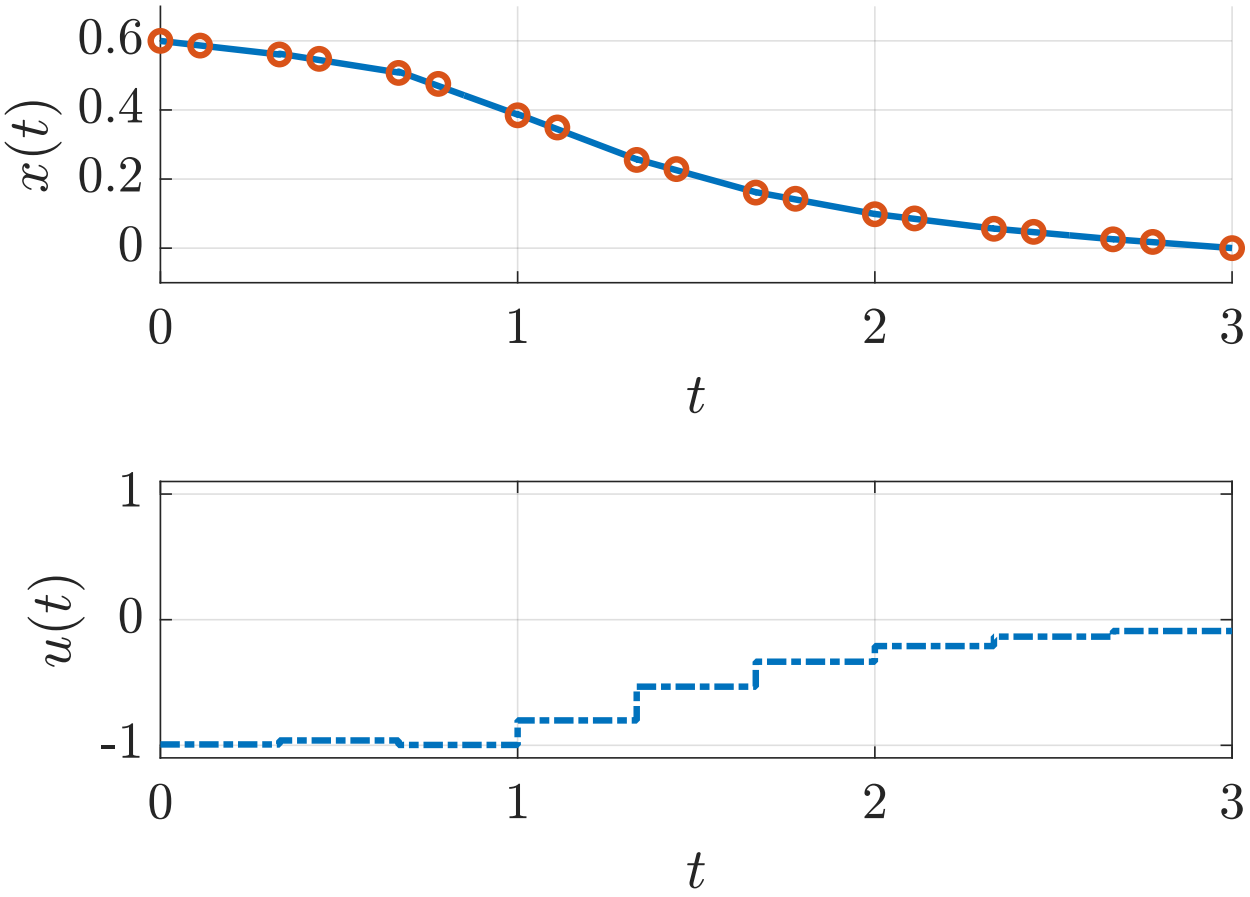
Illustrative example: First Iterate



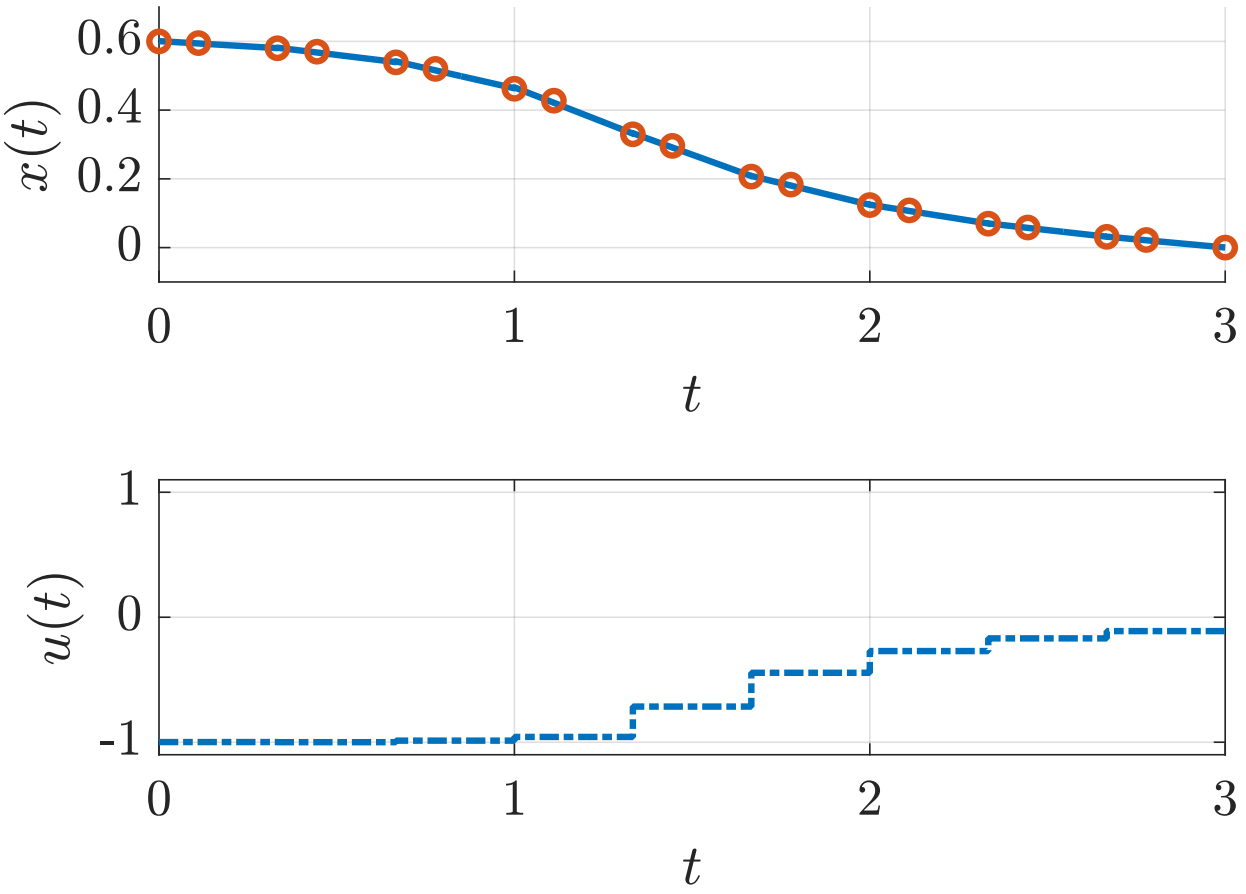
Illustrative example: Second Iterate



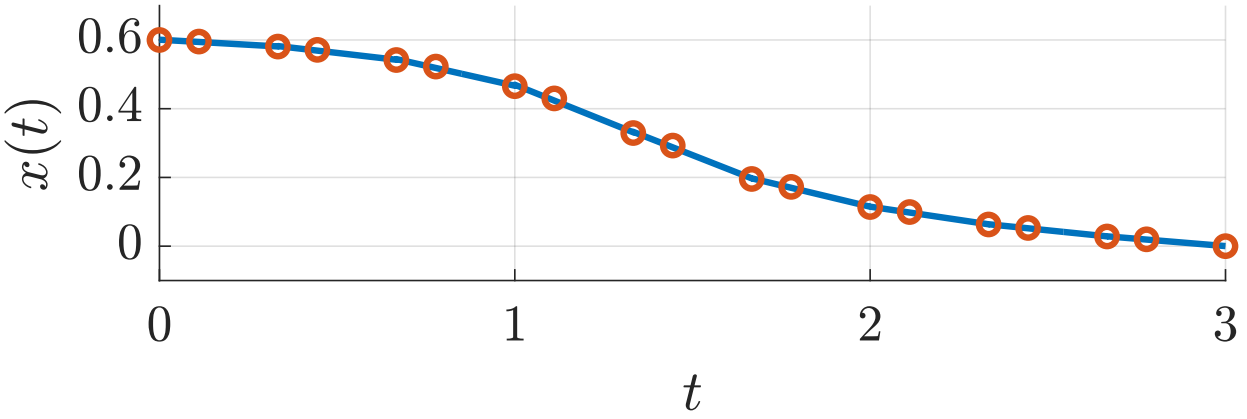
Illustrative example: Third Iterate

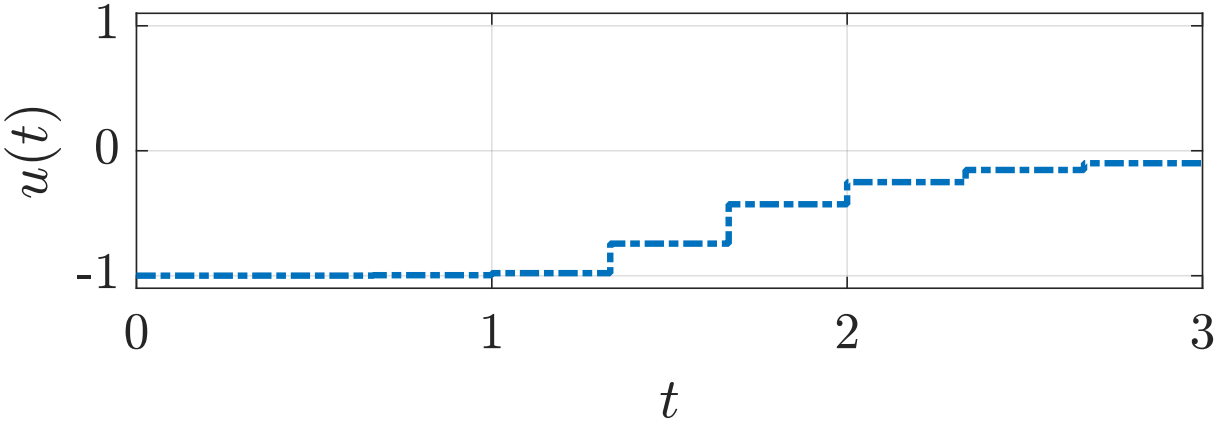


Illustrative example: Fourth Iterate

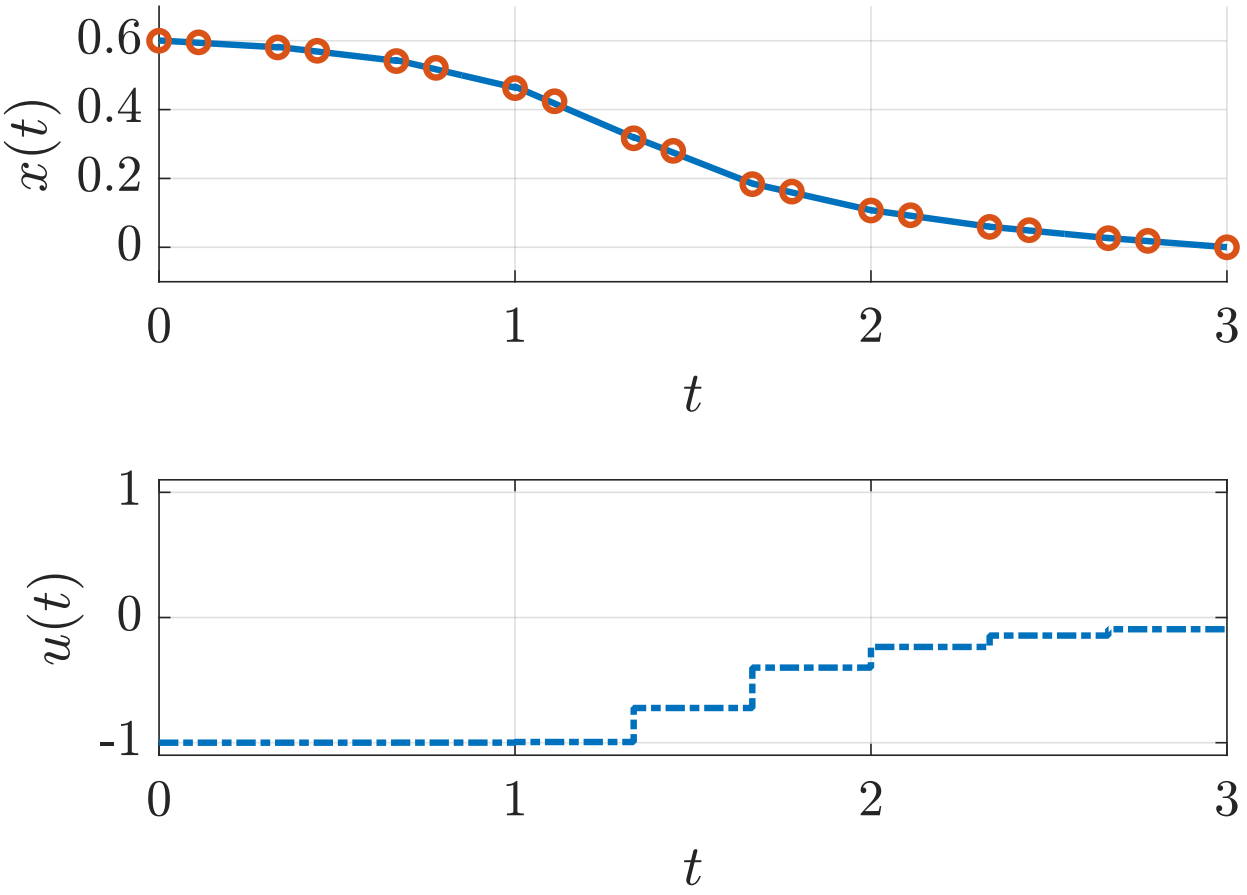


Illustrative example: Fifth Iterate

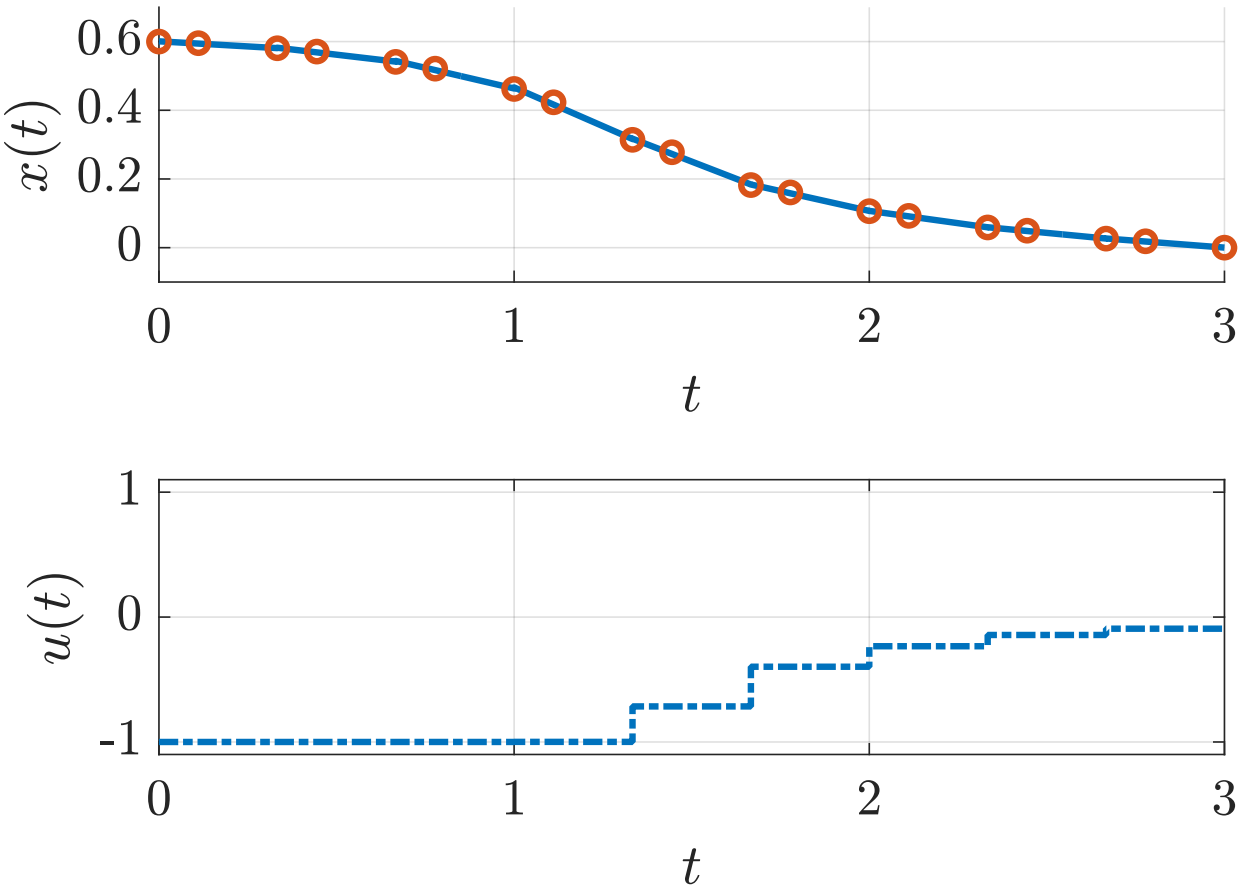




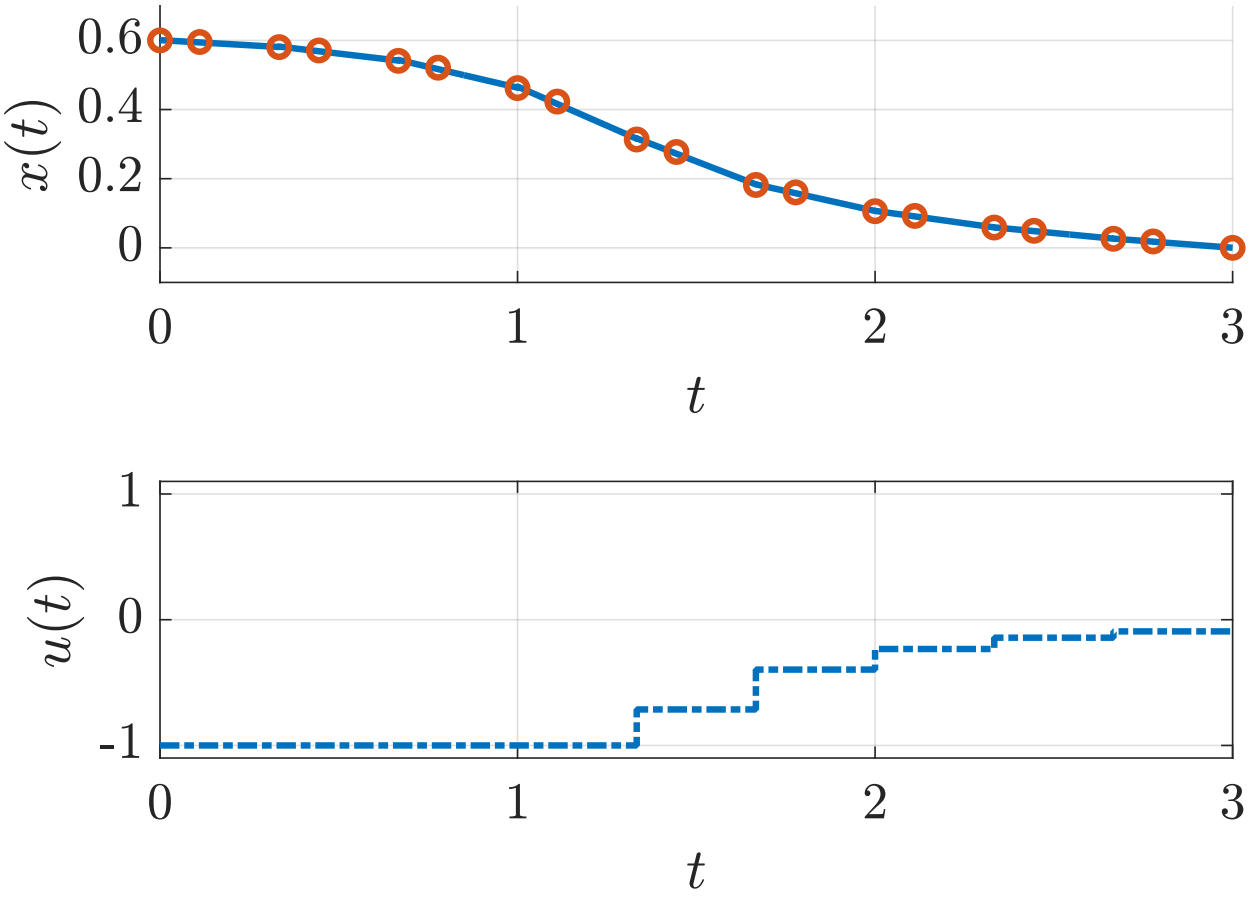
Illustrative example: Sixth Iterate



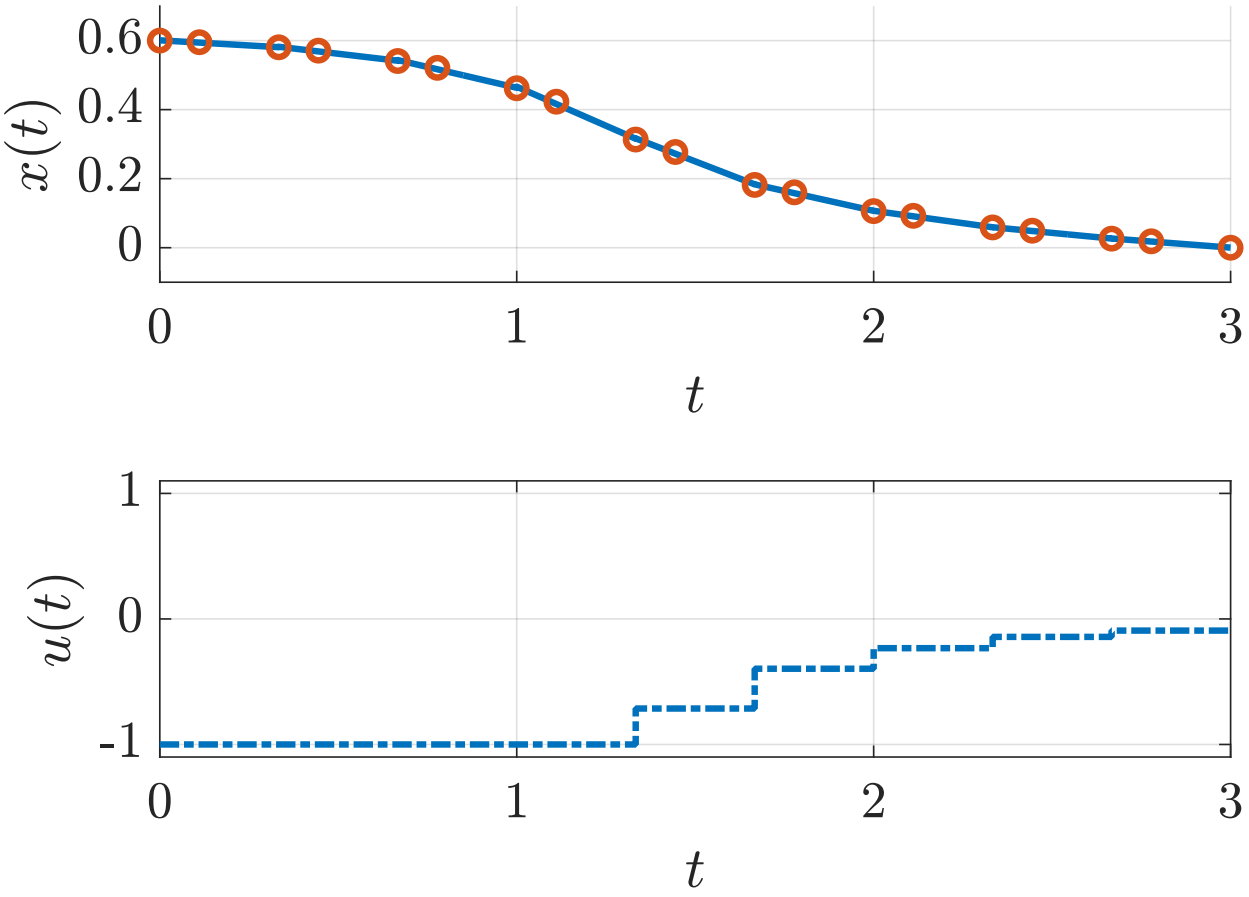
Illustrative example: Seventh Iterate



Illustrative example: Eighth Iterate



Illustrative example: Solution after Nine Newton-type Iterations



More Complex Example: Power Optimal Trajectories in Airborne Wind Energy (AWE)

formulated and solved daily by practitioners using open-source python package “AWEBox” [De Schutter et al. 2023]



For simple plane attached to a tether:

- 20 differential states (3+3 trans, 9+3 rotation, 1+1 tether)
- 1 algebraic state (tether force)
- 8 invariants (6 rotation, 2 due to tether constraint)
- 3 control inputs (aileron, elevator, tether length)

Translational:

$$\begin{bmatrix} m & 0 & 0 & x \\ 0 & m & 0 & y \\ 0 & 0 & m & z \\ x & y & z & 0 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \lambda \end{bmatrix} = \begin{bmatrix} F_x + m\left(\dot{\delta}^2 r_A + \dot{\delta}^2 x + 2\dot{\delta}\dot{y} + \ddot{\delta} y\right) \\ F_y + m\left(y\dot{\delta}^2 - 2\dot{x}\dot{\delta} - \ddot{\delta}(rA + x)\right) \\ F_z - gm \\ -\dot{x}^2 - \dot{y}^2 - \dot{z}^2 \end{bmatrix}$$

Rotational:

$$\dot{R} = R\omega_\times - R^T \begin{bmatrix} 0 \\ 0 \\ \dot{\delta} \end{bmatrix}, \qquad J\dot{\omega} = T - \omega \times J\omega, \qquad R = \begin{bmatrix} \vec{E}_x & \vec{E}_y & \vec{E}_z \end{bmatrix}$$

Aero. coefficients:

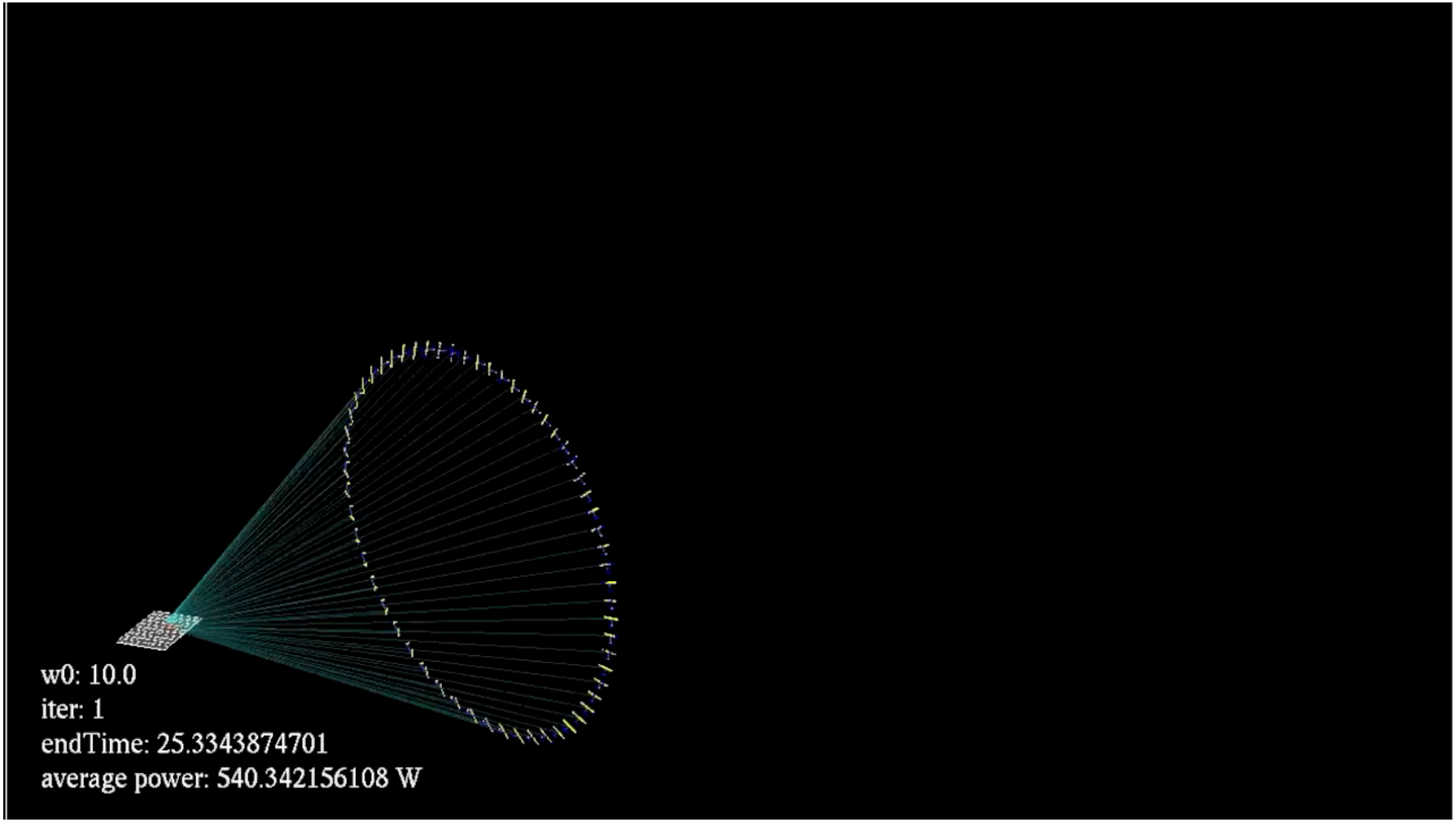
$$\vec{v} = \begin{bmatrix} \dot{x} - \dot{\delta} y \\ \dot{y} + \dot{\delta}(r_{\mathrm{A}} + x) \\ \dot{z} \end{bmatrix} - \vec{w}(x, y, z, \delta, t), \qquad \alpha = -\frac{\vec{E}_z^T \vec{v}}{\vec{E}_x^T \vec{v}}, \qquad \beta = \frac{\vec{E}_y^T \vec{v}}{\vec{E}_x^T \vec{v}}$$

Aero. forces/torques:

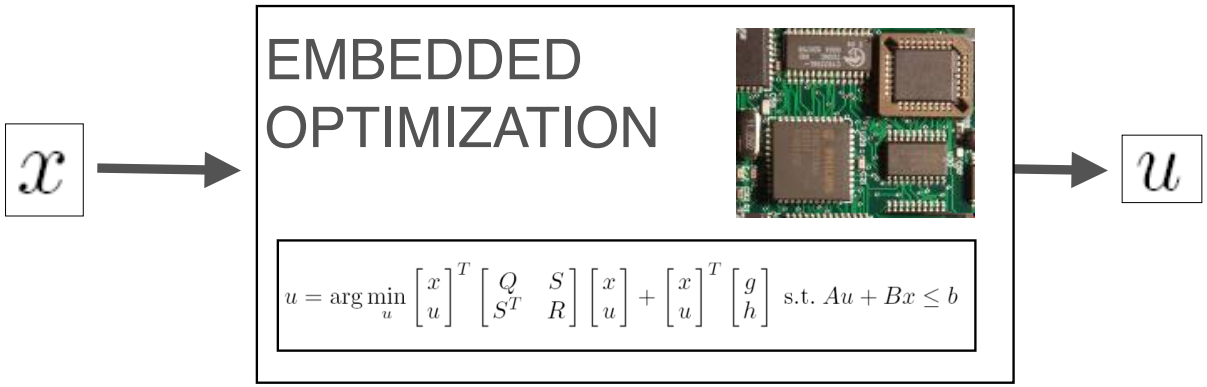
$$\vec{F}_{\mathrm{A}} = \frac{1}{2}\rho A \|\vec{v}\|(C_{\mathrm{L}}\vec{v} \times \vec{E}_y - C_{\mathrm{D}}\vec{v}), \quad \vec{T}_{\mathrm{A}} = \frac{1}{2}\rho A \|\vec{v}\|^2 \begin{bmatrix} C_{\mathrm{R}} \\ C_{\mathrm{P}} \\ C_{\mathrm{Y}} \end{bmatrix}$$

Newton-Type Optimization Iterations for Power Optimal Flight

(video by Greg Horn, using CasADi and Ipopt as optimization engine)



Embedded Optimization: a CPU-Intensive Map





Solve, in real-time and repeatedly, an optimization problem that depends on the incoming stream of input data, to generate a stream of output data.

The ubiquity of parametric convex optimization

THEOREM [Baes, D., Necoara, 2008]
Every continuous map

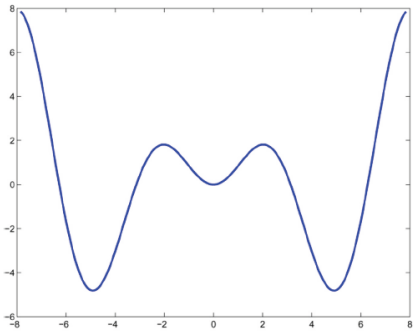
$$\begin{aligned} \mu : \mathbb{R}^{n_x} &\rightarrow \mathbb{R}^{n_u} \\ x &\mapsto u = \mu(x) \end{aligned}$$

can be represented as parametric convex program (PCP):

$$\mu(x) = \arg\min_u g(u, x) \quad \text{s.t.} \quad (u, x) \in \Gamma$$

PCP: objective and feasible set jointly convex in parameters and variables (x, u).

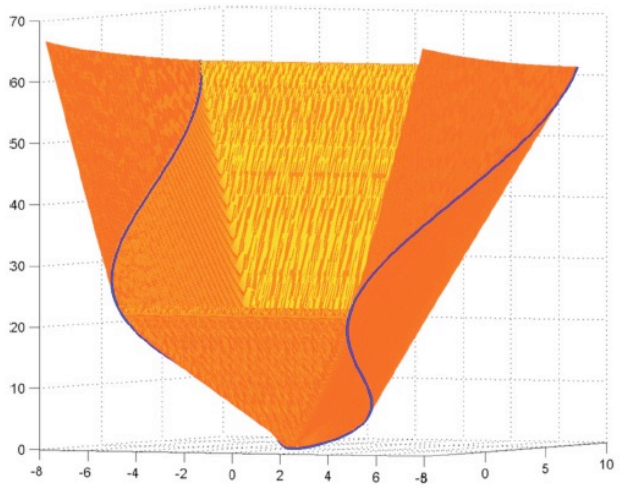
(Sketch of Proof)



Given: graph of $\mu(x)$

Construct epigraph E of $g(u, x)$

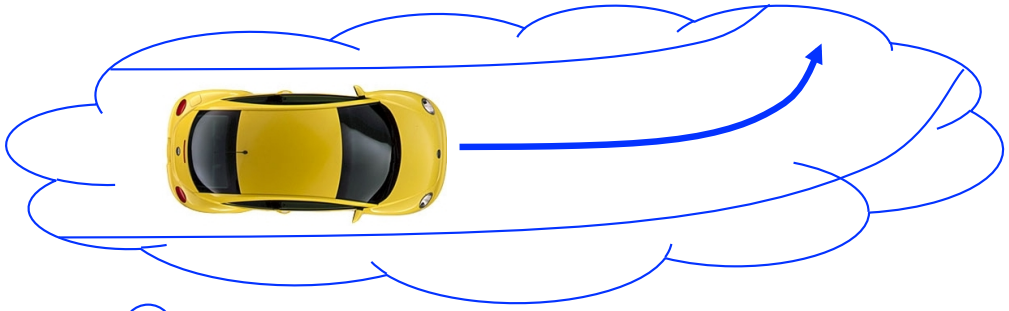
1. "Bend" graph of $\mu(x)$ using strictly convex $g^0(x)$
2. Add upward rays.
3. Take convex hull.
4. Show that minima are preserved.



$$S := \{(x, \mu(x), t) | x \in \Omega, g^0(x) \leq t\}$$
$$E := \text{conv}(S)$$

Prime Example: Model Predictive Control (MPC)

Always look a bit into the future

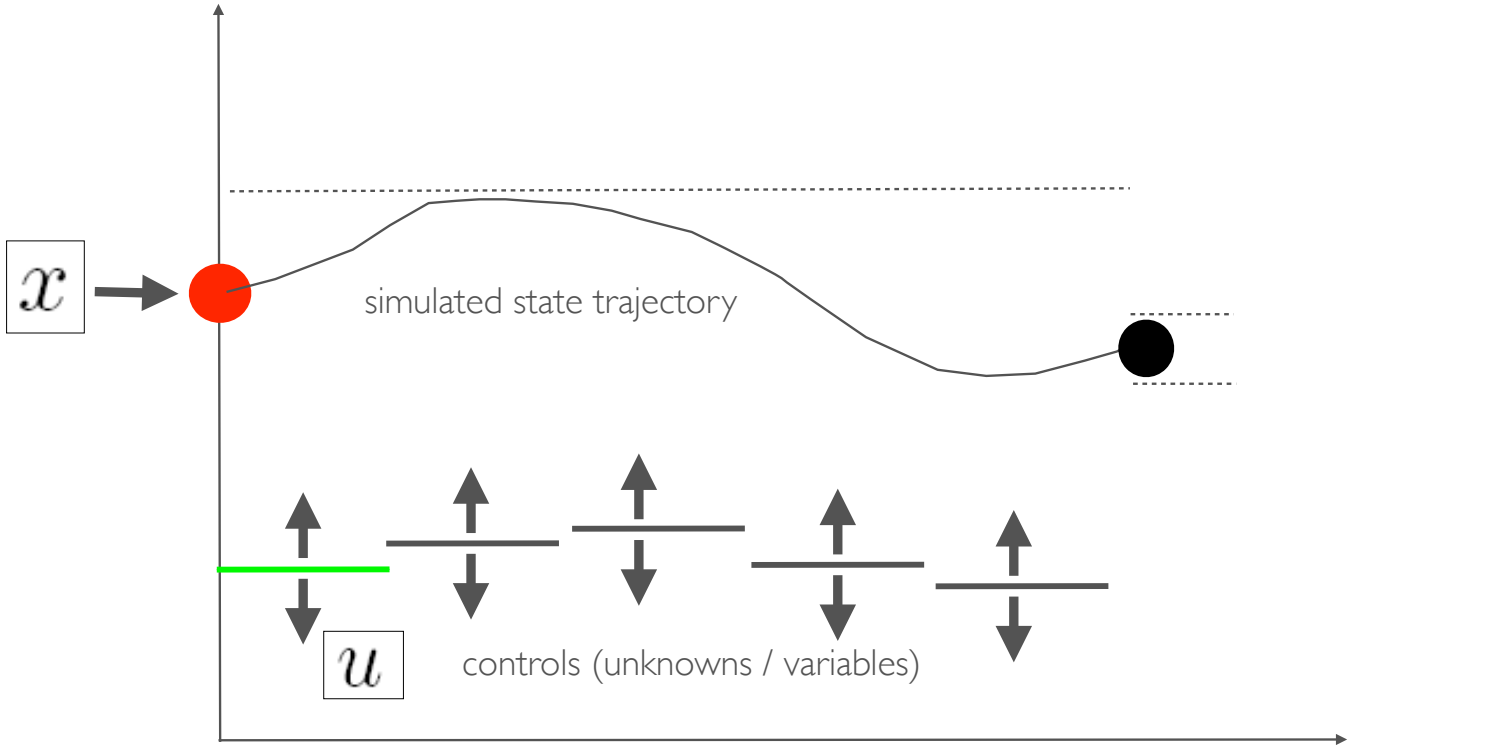




Example: driver predicts and optimizes,
and therefore slows down before a curve

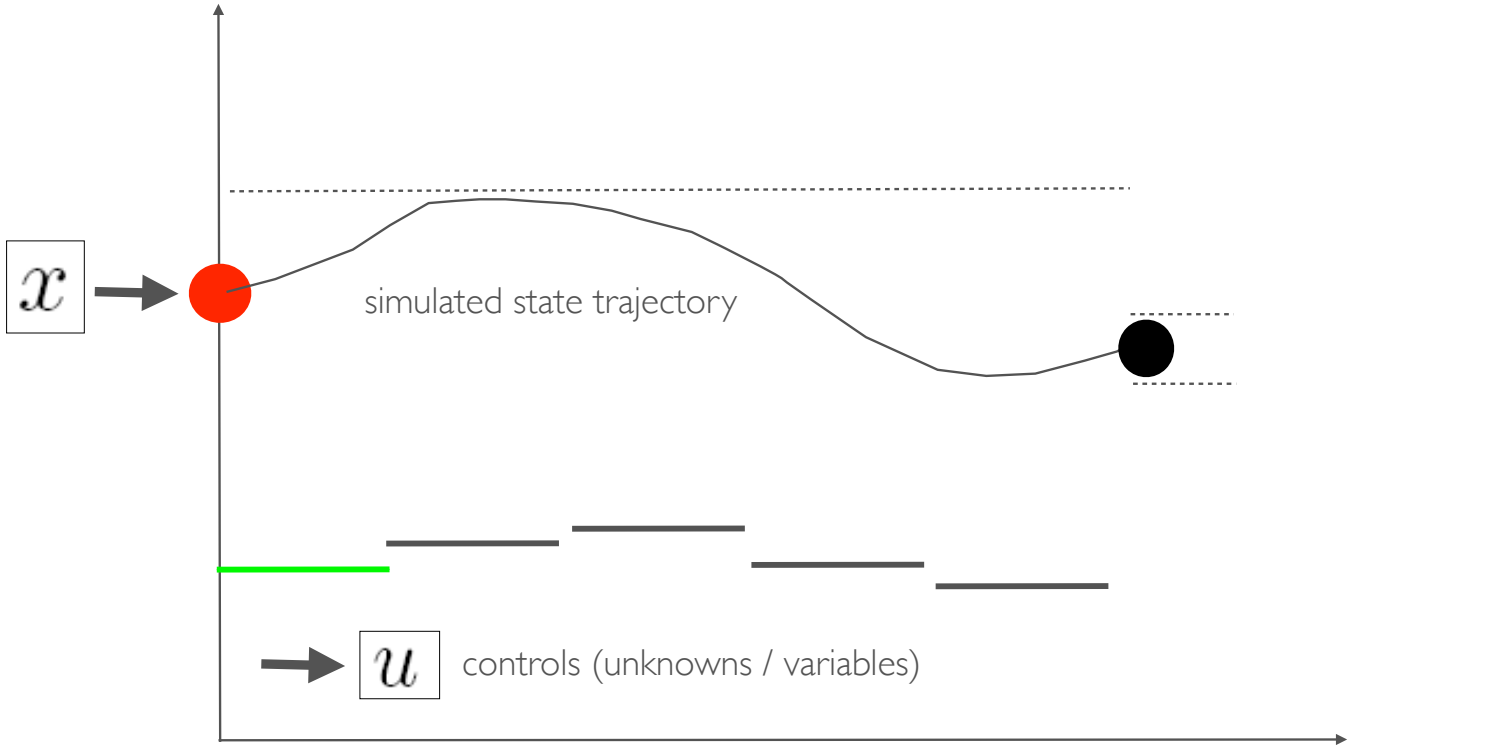
Open Loop Optimal Control Problem in MPC

For given system state $\boldsymbol{x}$, which controls $\boldsymbol{u}$ lead to the best objective value without violation of constraints ?



Open Loop Optimal Control Problem in MPC

For given system state $\boldsymbol{x}$, which controls $\boldsymbol{u}$ lead to the best objective value without violation of constraints ?



MPC creates a map from the initial value $\boldsymbol{x}$ to the first control $\boldsymbol{u}$
(which in fact approximates the optimal feedback control from the HJB equation)

MPC Example: Point-To-Point Motions [PhD Vandenbrouck 2012]





Fast oscillating systems (cranes, plotters, wafer steppers, …)

Control aims:

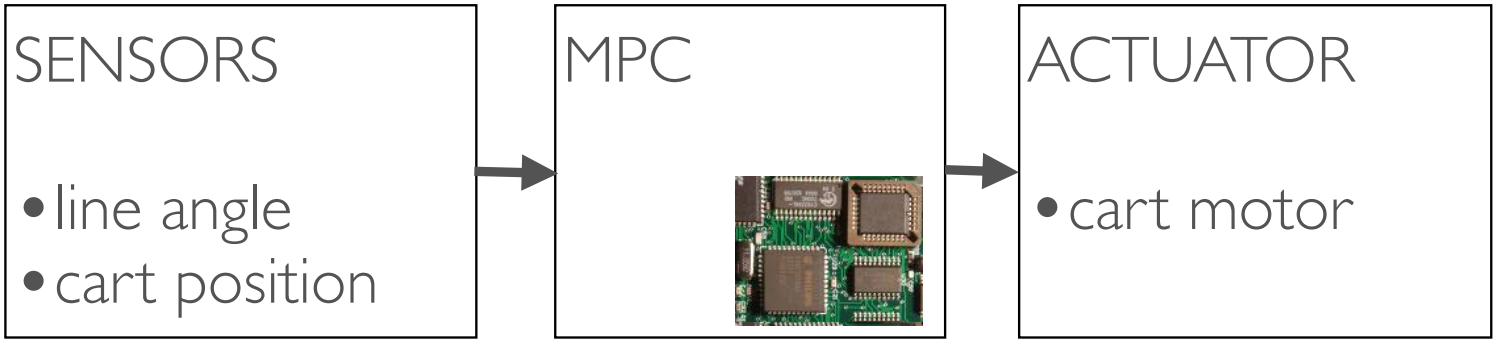
- reach end point as fast as possible
- do not violate constraints
- no residual vibrations



Idea: formulate as embedded optimization problem
in form of Model Predictive Control (MPC)

Time Optimal MPC of a Crane

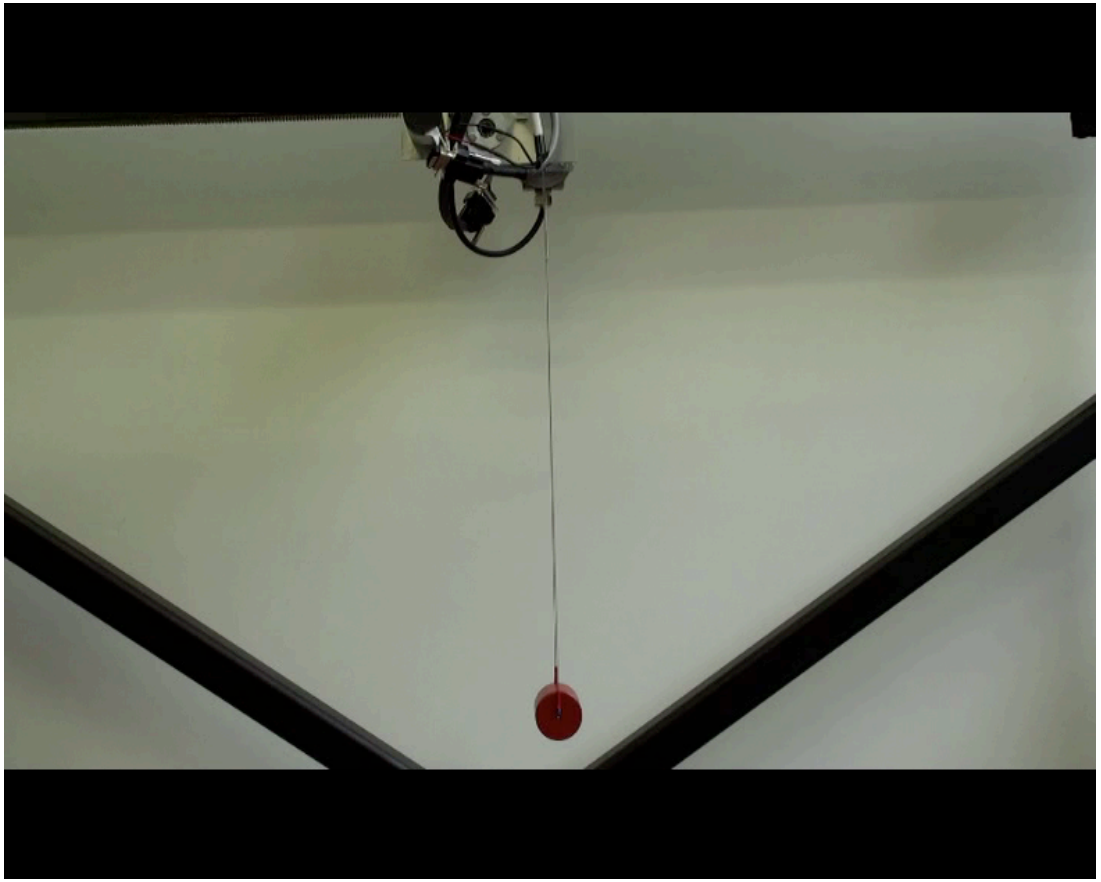




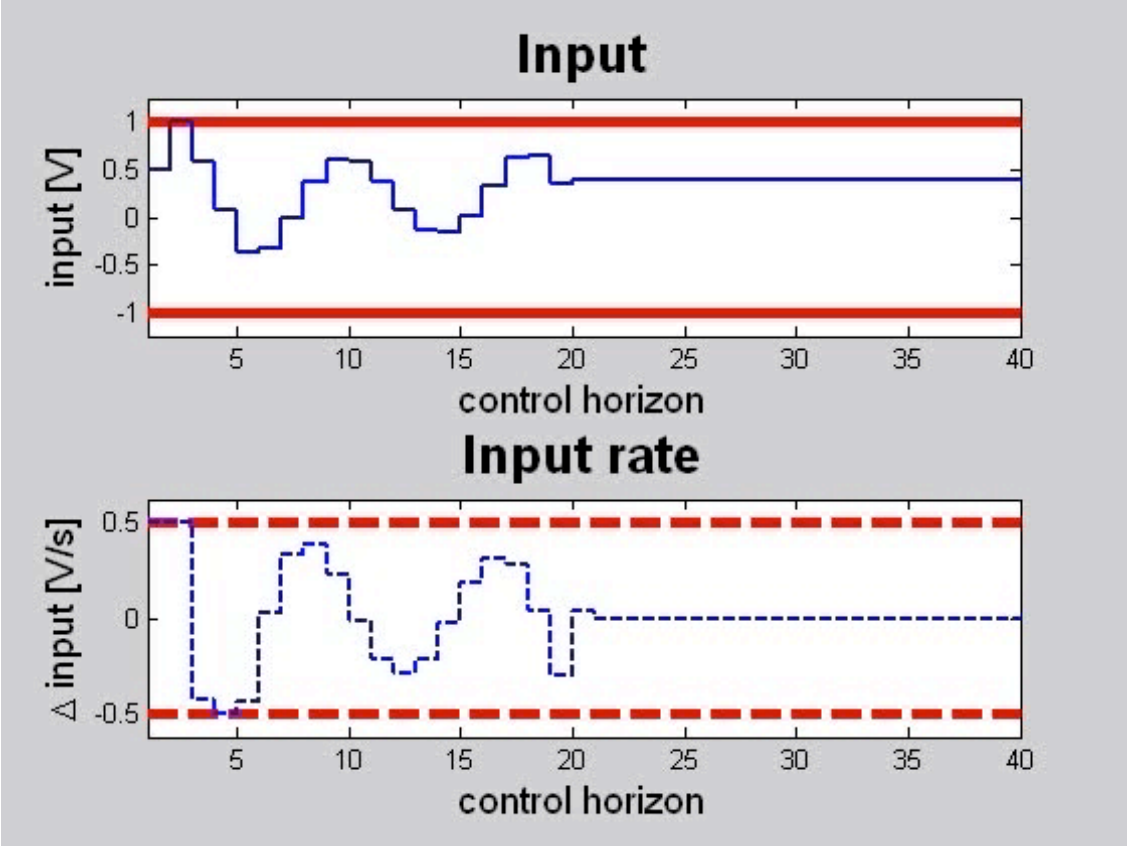
Hardware: xPC Target. Software: qpOASES [Ferreau, D., Bock, 2008]

Time Optimal MPC of a Crane

Univ. Leuven [Vandenbrouck, Swevers, D.]



Optimal solutions varying in time (inequalities matter)



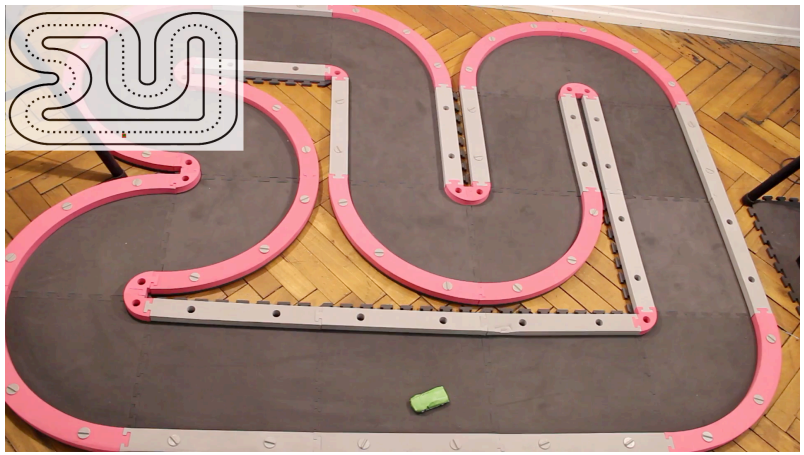
Solver qpOASES [PhD H.J. Ferreau, 2011], [Ferreau, Kirches, Potschka, Bock, D. , A parametric active-set algorithm for quadratic programming, Mathematical Programming Computation, 2014]

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=zBVpx3bgI6E

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.**



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.

SUPPLEMENTARY MATERIAL

Video of the experiments: https://youtu.be/zBVpx3bgI6E

A. Implementation Details

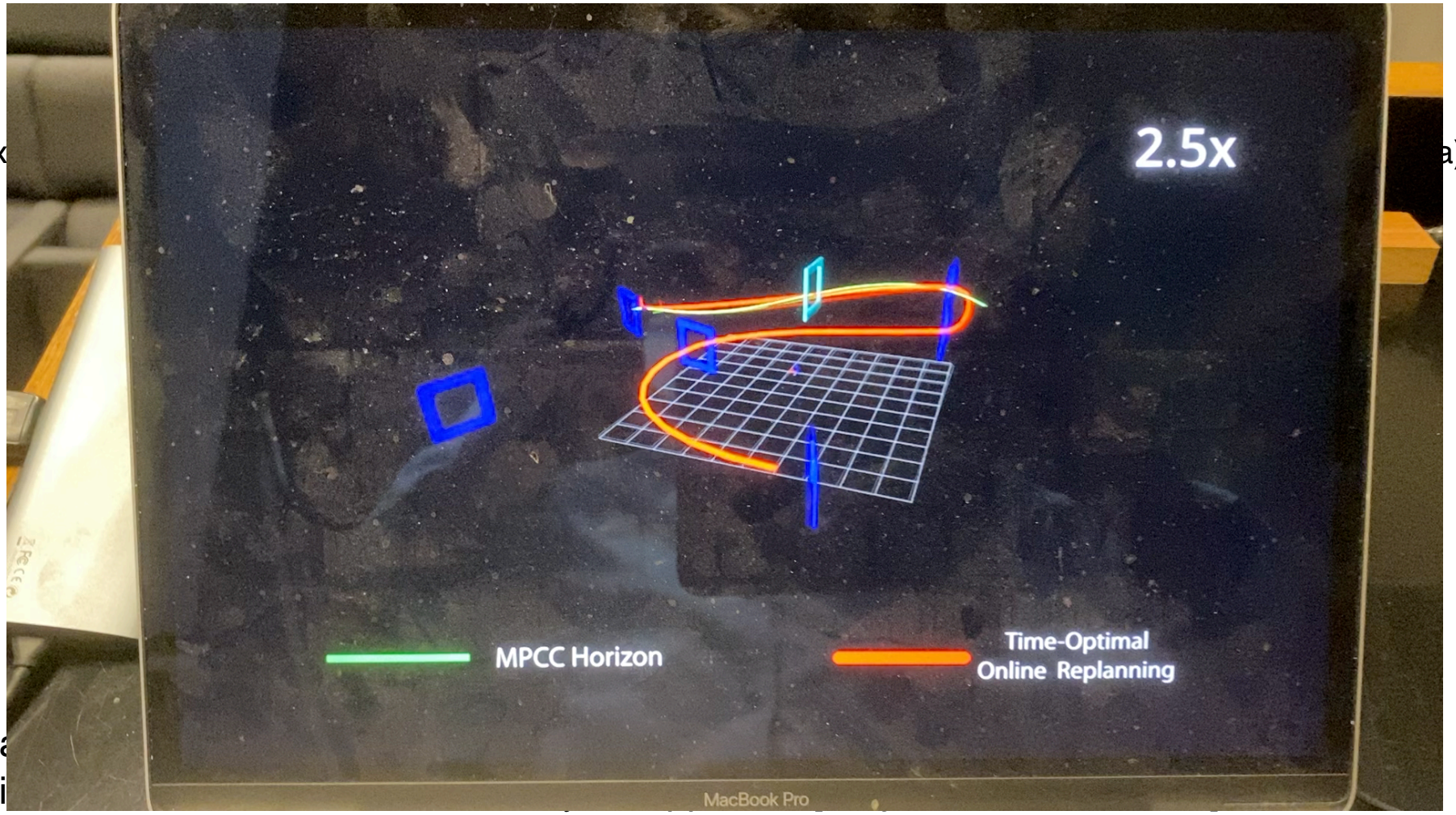
In order to deploy our MPCC controller, (4) needs to be solved in real-time. To this end, we have implemented our optimization problem using acados [24] as a code generation tool, in contrast to [6], where its previous version, ACADO [25] was used. It is important to note that for consistency, the optimization 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 BLASFEO [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

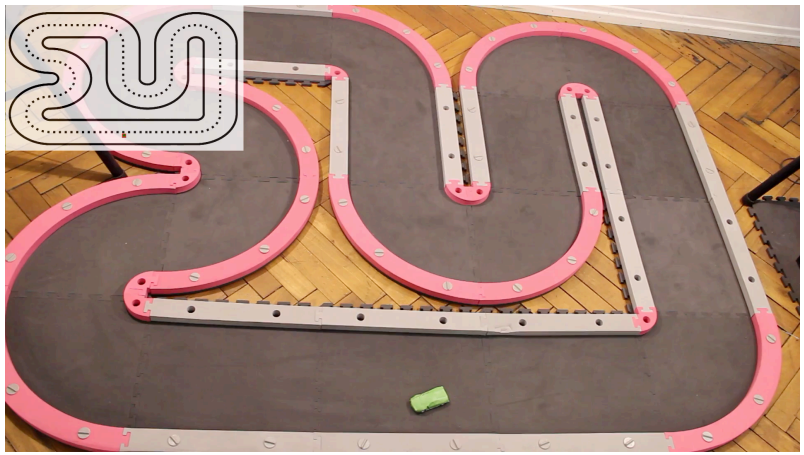
di

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)



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

Example 2: Quadrotor Racing (U Zurich, Scaramuzza)

Video: https://www.youtube.com/watch?v=zBVpx3bgI6E

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.**



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.

SUPPLEMENTARY MATERIAL

Video of the experiments: https://youtu.be/zBVpx3bgI6E

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 BLASFEO [27], a linear algebra library specifically designed for

Latest **acados** development:

differentiable nonlinear MPC via adjoint approach [Frey et al. 2026, Neurips]

NMPC needs to solve Nonlinear Programs (NLP)



Continuous Time NMPC Problem

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

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)

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

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

Nonlinear Programs (NLP) with Convex Structure



Newton-type methods generate a sequence $w_0, w_1, w_2, \ldots$ 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

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

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

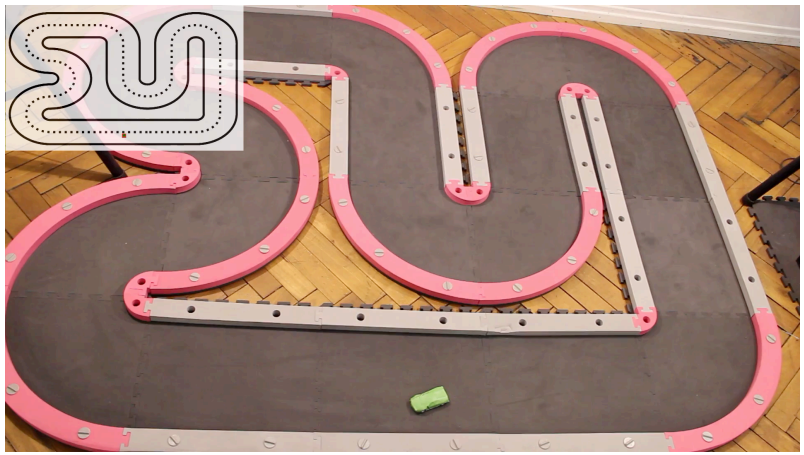
Works extremely well for mildly nonlinear F, also in microsecond NMPC [cf. Zanelli 2021, Lekic 2020, Hausberger 2020]

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=zBVpx3bgI6E

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.**



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.

SUPPLEMENTARY MATERIAL

Video of the experiments: https://youtu.be/zBVpx3bgI6E

A. Implementation Details

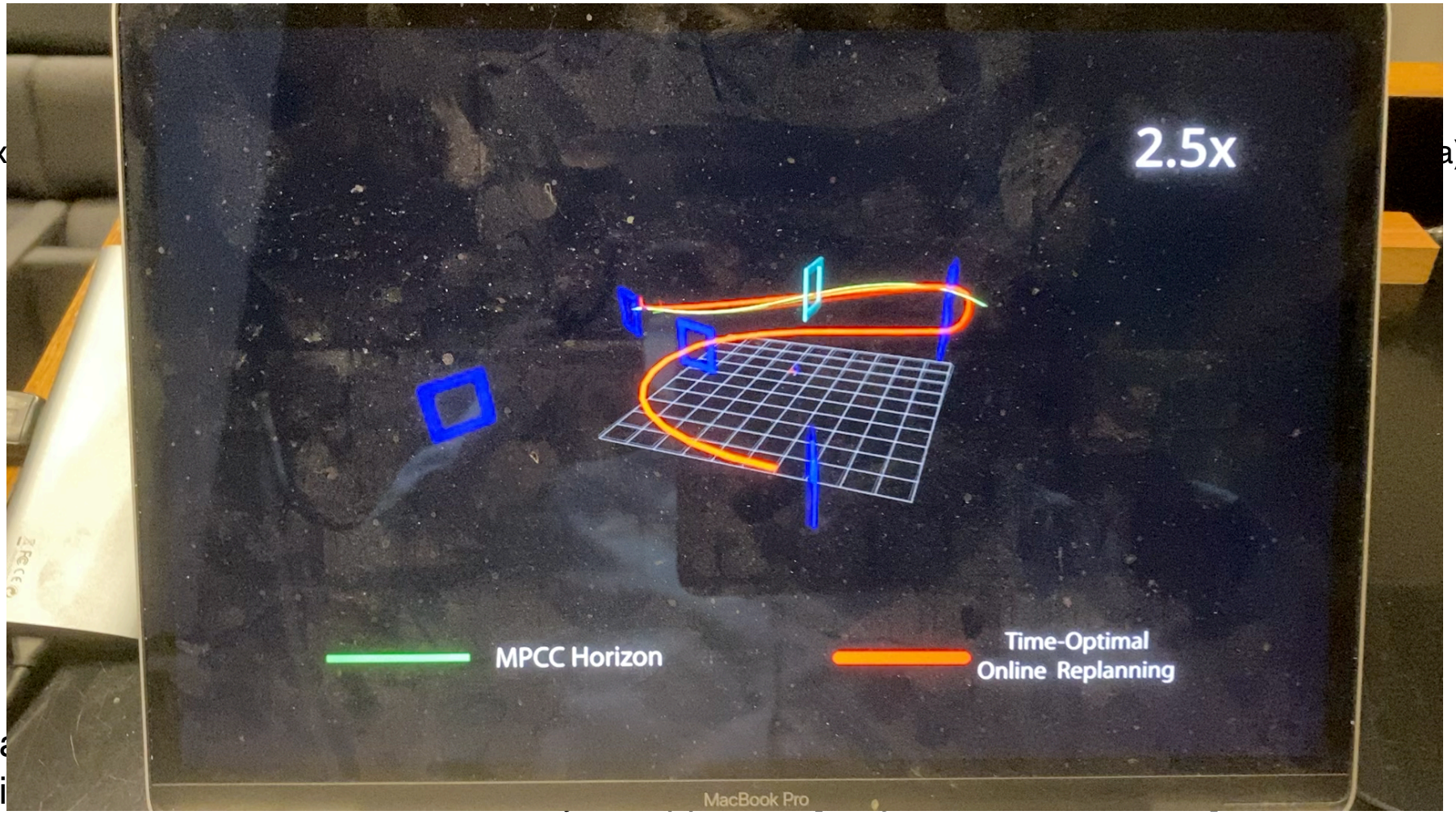
In order to deploy our MPCC controller, (4) needs to be solved in real-time. To this end, we have implemented our optimization problem using acados [24] as a code generation tool, in contrast to [6], where its previous version, ACADO [25] was used. It is important to note that for consistency, the optimization 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 BLASFEO [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

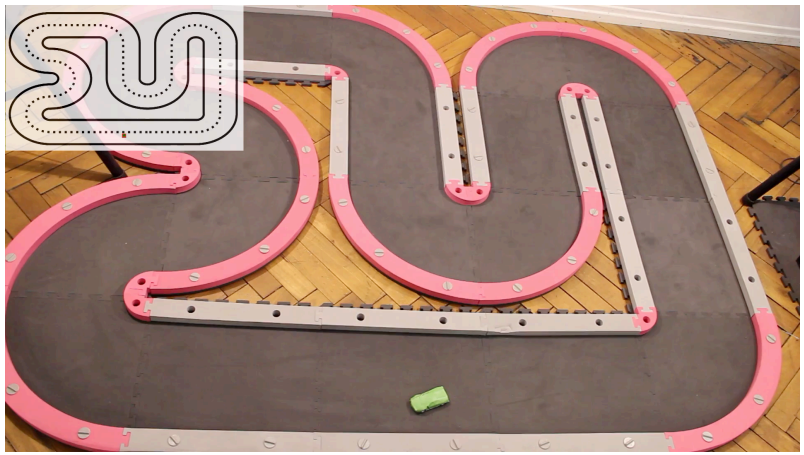
di

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)



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

Example 2: Quadrotor Racing (U Zurich, Scaramuzza)

Video: https://www.youtube.com/watch?v=zBVpx3bgI6E

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.**



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.

SUPPLEMENTARY MATERIAL

Video of the experiments: https://youtu.be/zBVpx3bgI6E

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 BLASFEO [27], a linear algebra library specifically designed for

Latest **acados** development:

differentiable nonlinear MPC via adjoint approach [Frey et al. 2026, Neurips]

Dynamic Optimization Problem in NMPC

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

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

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

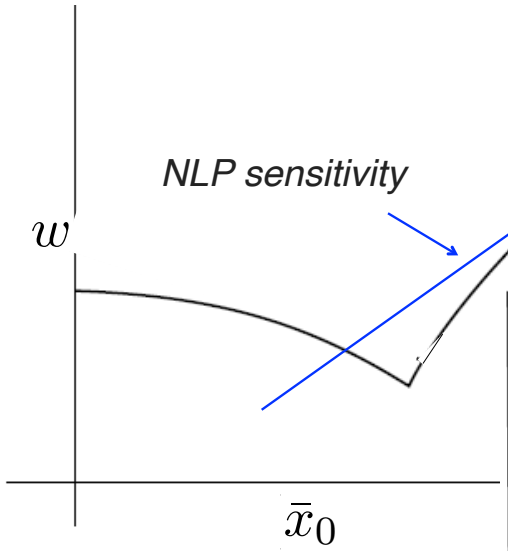
Summarize as

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

with convex ϕ and Ω

Nonlinear MPC = parametric Nonlinear Programming

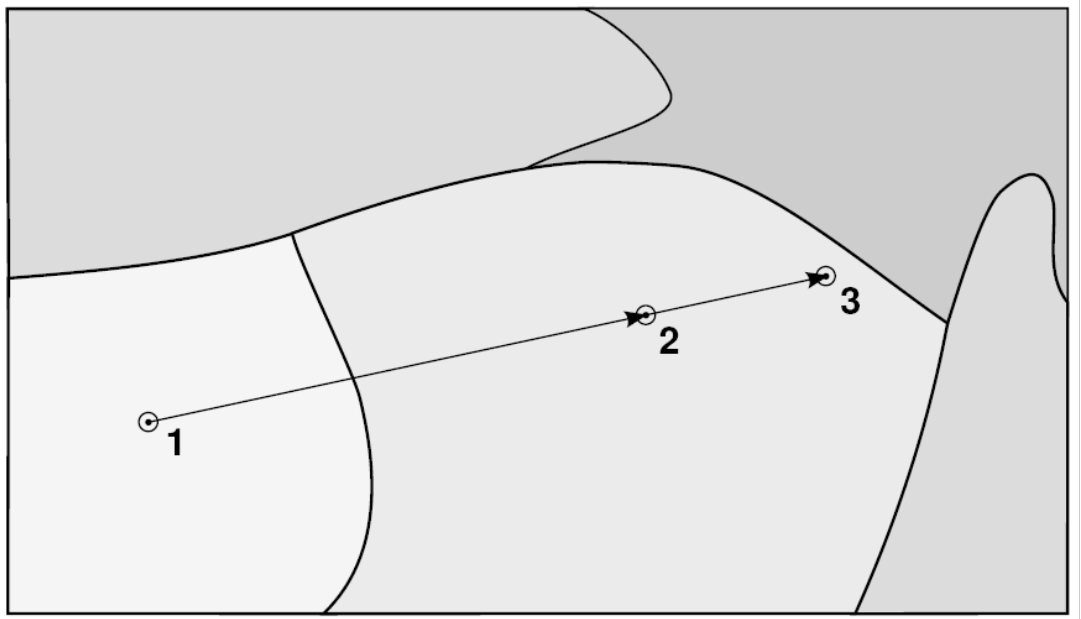
Solution manifold is piecewise differentiable (kinks at active set changes)



How to deal with a sequence of large parameter changes?

NLP Pathfollowing

Critical regions are non-polyhedral



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

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

Repeat each sampling instant k two steps:

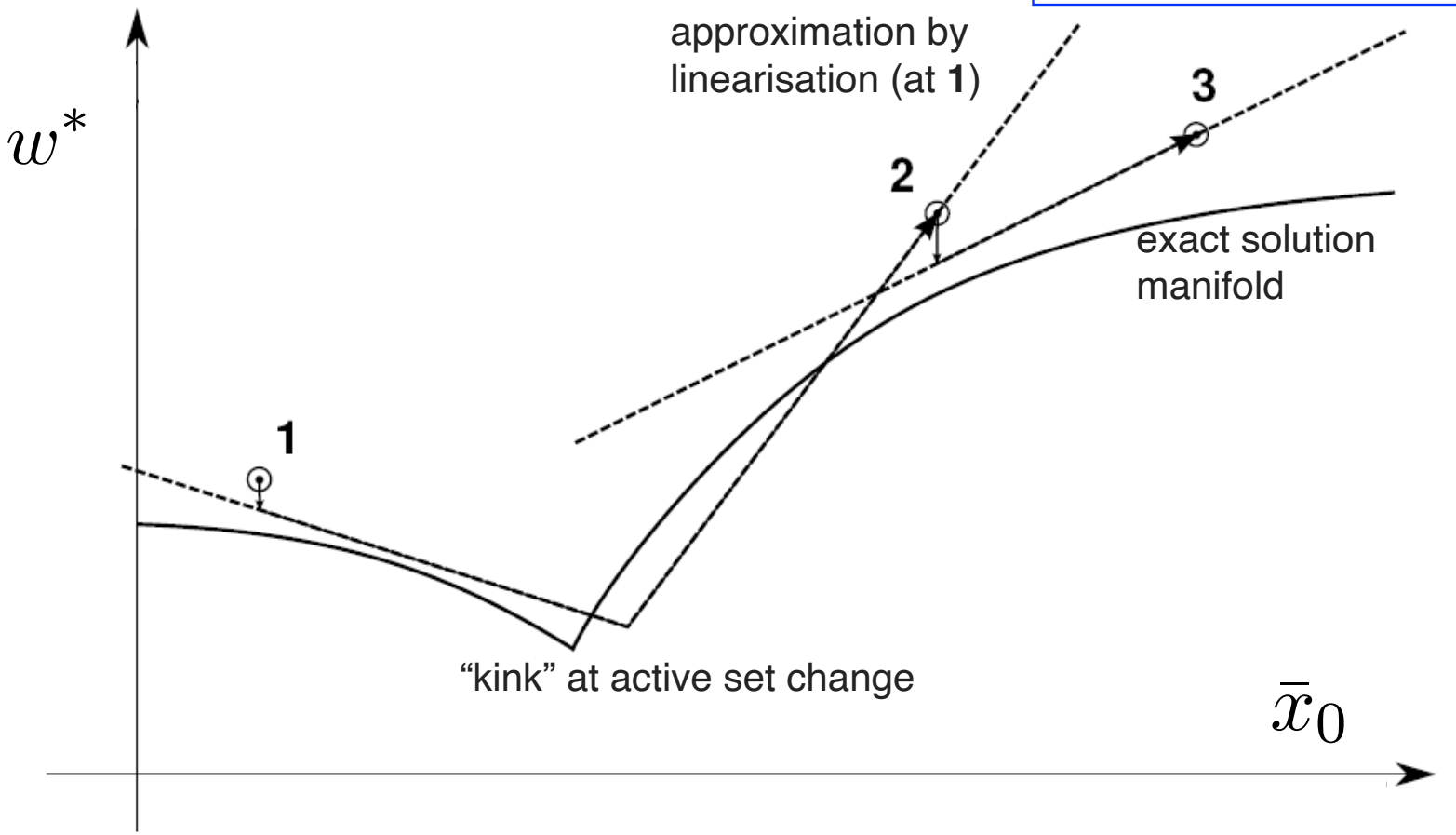
Step 1: Linearize nonlinear constraints at w_k to obtain convex problem (right).
(numerical integrations, nonlinear function and derivative evaluations)

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

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



Tangential prediction even across active set changes

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

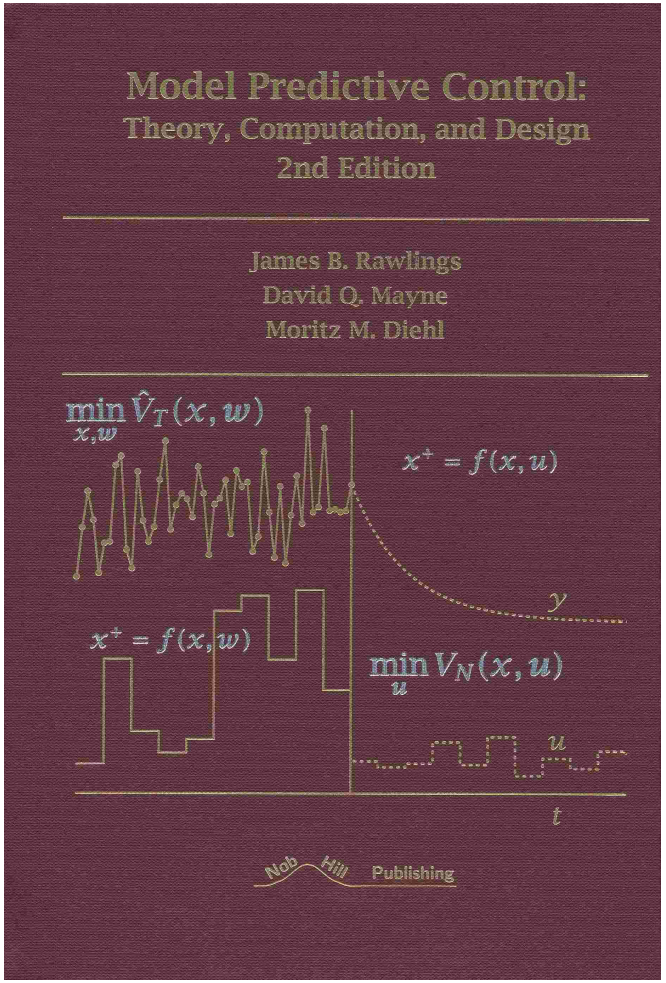
Real-Time Iterations [PhD Diehl 2001, Heidelberg]

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

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

Sparse Quadratic Program (QP)

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

1) Linearize constraints: Integration & sensitivities

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

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

Sparse Quadratic Program (QP)

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

1) Linearize constraints: Integration & sensitivities

2) Solve sparse QP either
- **via condensing,** or
- via a Riccati type algorithm, or
- via a combination of both

OCP-QP Progress 2012-2016

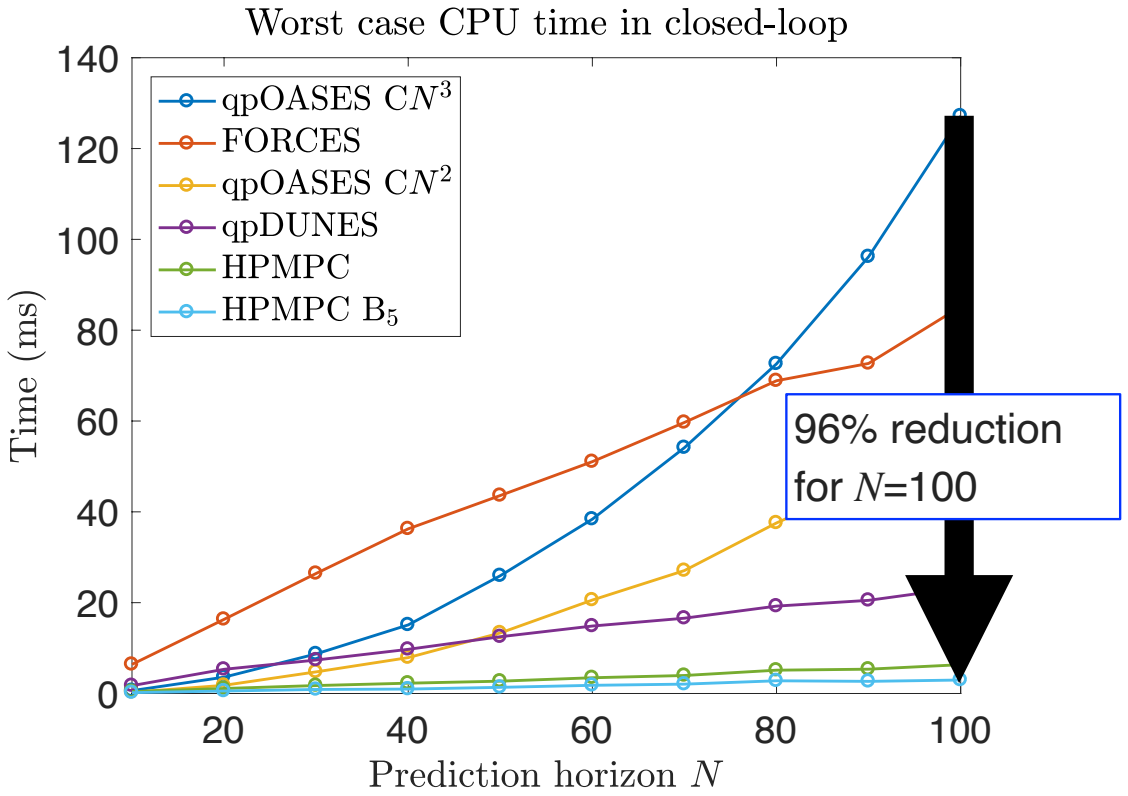
- **partial condensing** (proposed by Daniel Axehill, Linköping) combines advantages of condensing and Riccati recursion and further boost performance of HPMPC (by G. Frison, DTU/Freiburg)



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^* olving 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

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

1) Linearize constraints: Integration & sensitivities

Sparse QP

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

2) Condense sparse QP

Condensed small QP

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

3) Solve condensed QP

Computations in one Real-Time Iteration

NLP

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

Sparse QP

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

Condensed small QP

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

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

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

1) Linearize constraints: Integration & sensitivities

Sparse QP

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

2) Condense sparse QP

Condensed small QP

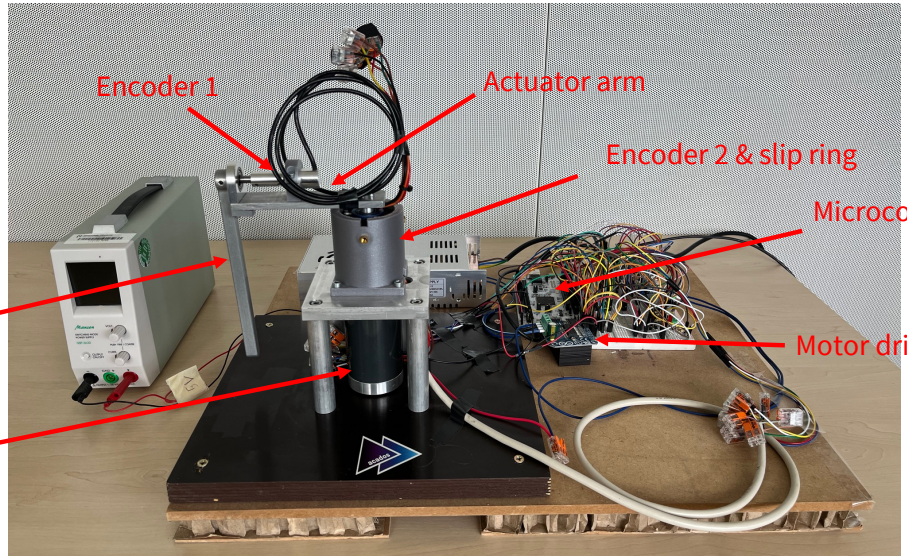
$$
\begin{aligned}
\underset{u}{\text{minimize}} \quad & f_{\text{condQP},i}(\bar{x}_0, u) \\
\text{subject to} \quad & \bar{r} + \bar{R}^{x_0} \bar{x}_0 + \bar{R}^u u \leq 0.
\end{aligned}
$$

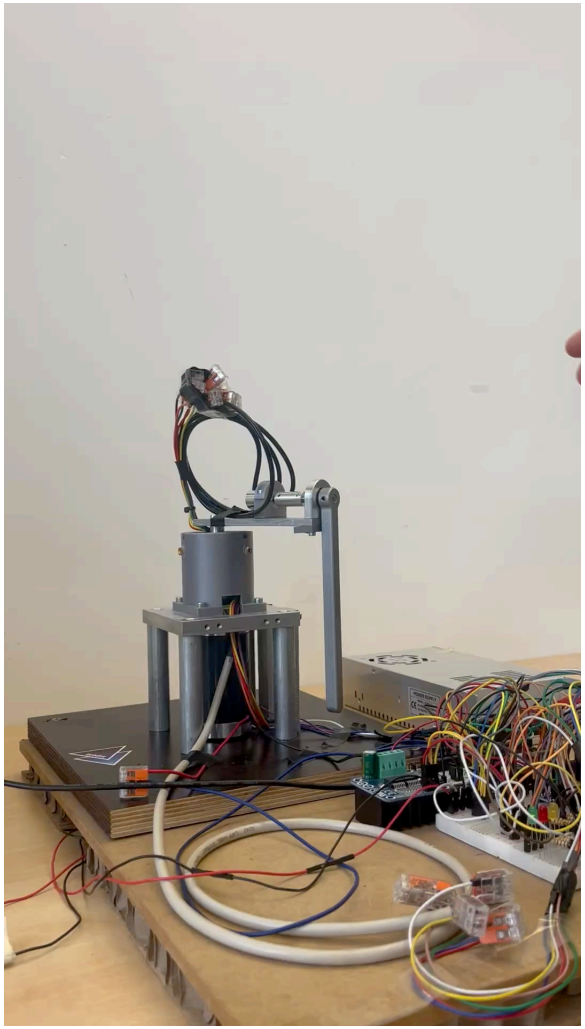
"Feedback phase"

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)





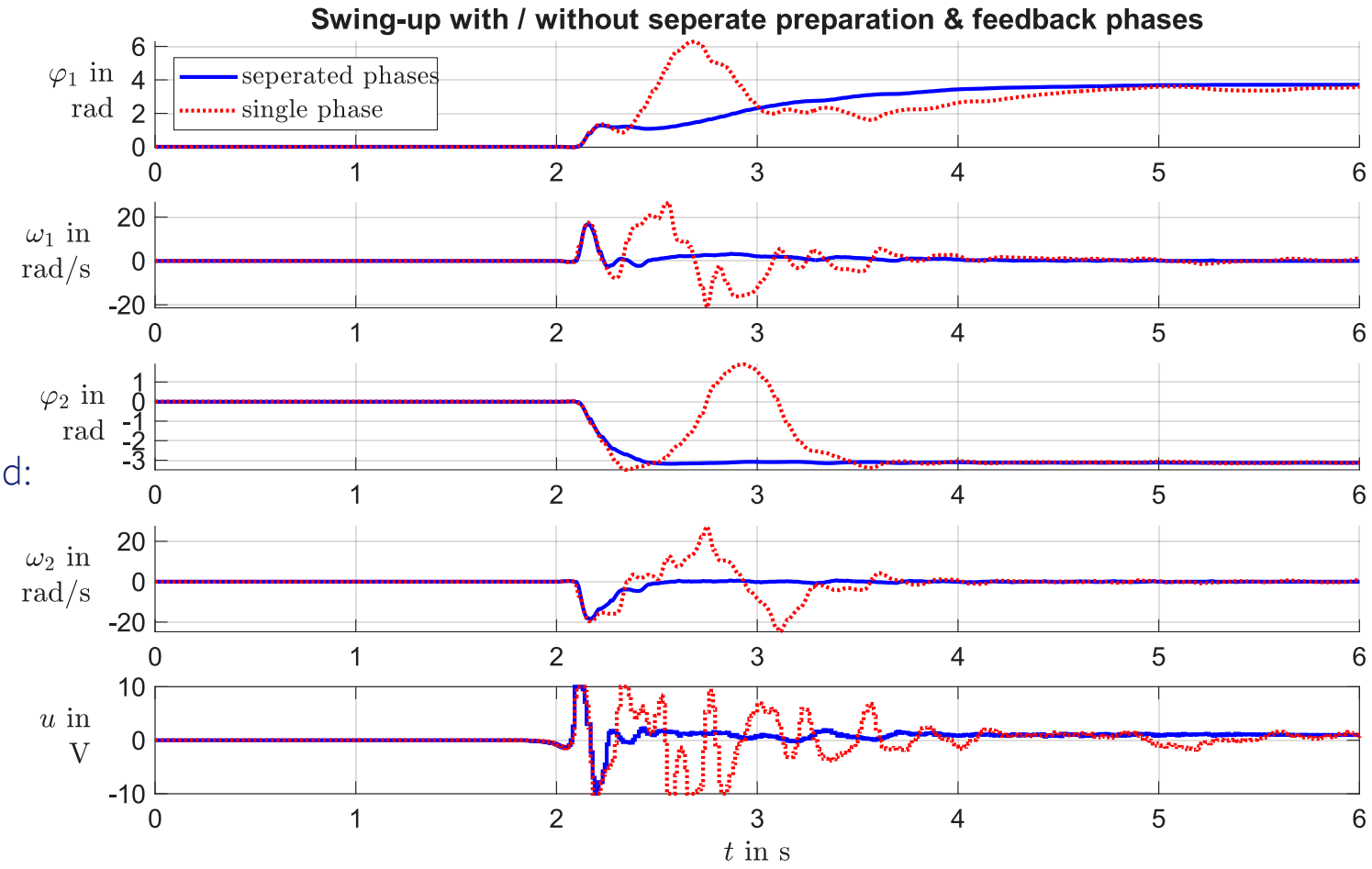
- µController board: (STM32 NUCLEO-H745ZI)
  - Prototyping-Board by STMicroelectronics
  - Dual-core MCU:
    - Cortex-M7 (480 MHz)
    - Cortex-M4 (240 MHz)
  - 1 MB RAM
  - 2 MB Flash
  - On-board ST-LINK for programming and debugging
  - Widely used STM32Cube ecosystem
  - Costs ≈ 30 EUR

NMPC parameters

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

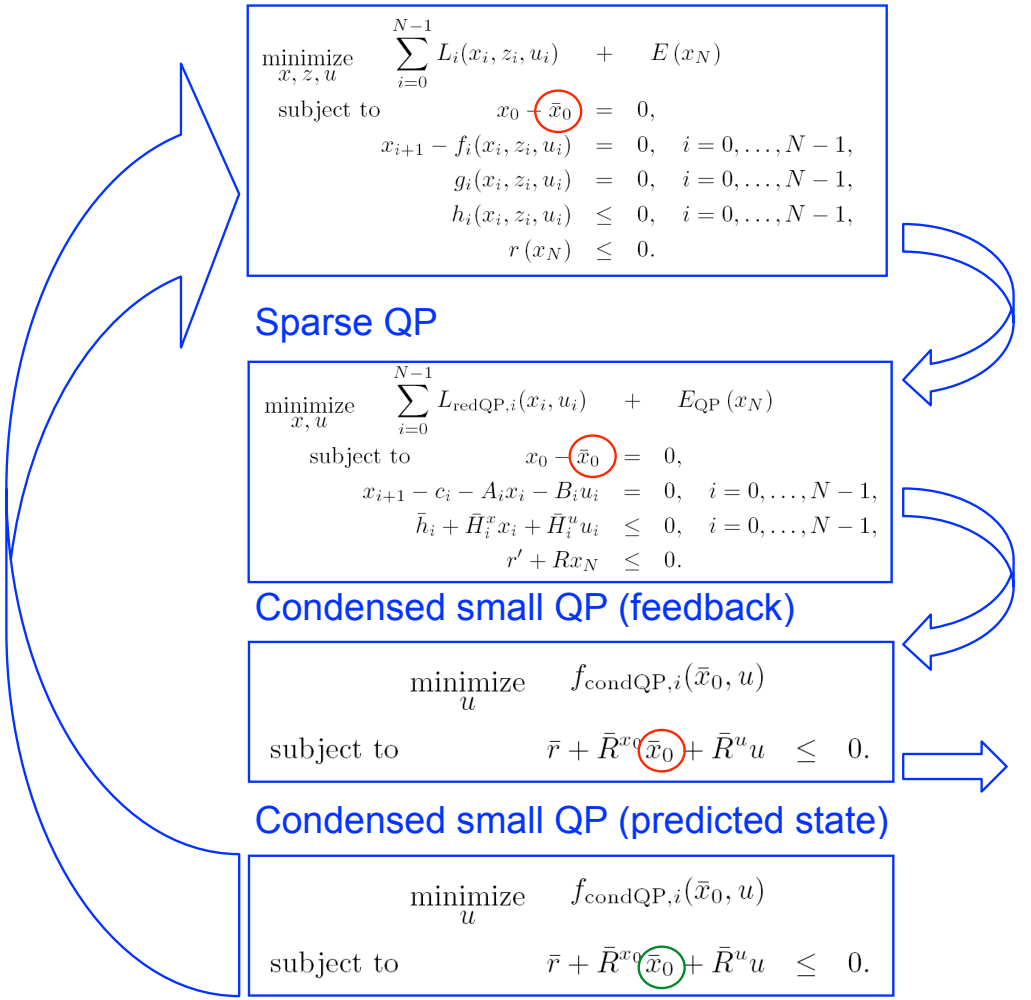
Experimental Validation of the Benefit of Separating Preparation and Feedback Phase (on Furuta pendulum, by J. Fischer)

- Control cycle time: 12 ms
- RTI preparation phase: 9 ms … 10 ms
- RTI feedback phase: 500 µs … 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]



1) Linearize constraints: Integration & sensitivities

2) Condense sparse QP

3) Solve condensed QP for current state

4) Solve condensed QP again for predicted next state, for NLP initialisation

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!