# SAB Meeting

# NMPC of the carousel-setup with the flying ball

Jonas Koenemann

11.06.2015

University of Freiburg

UNI FREIBURG

IMTEK

Ampyx Power
airborne wind energy

# Freiburg Carousel

# Motivation

- Well identified model available (Ernesto)
- Rather simple system (one control input)
- Define a non-trivial objective
- First step towards half-wing, tethered airplane NMPC

# Current State

- Open loop control (no state feedback)
- State evolution by integration
- Optimization running in the real-time loop
  - Integration of auto-generated code
  - Check for timing

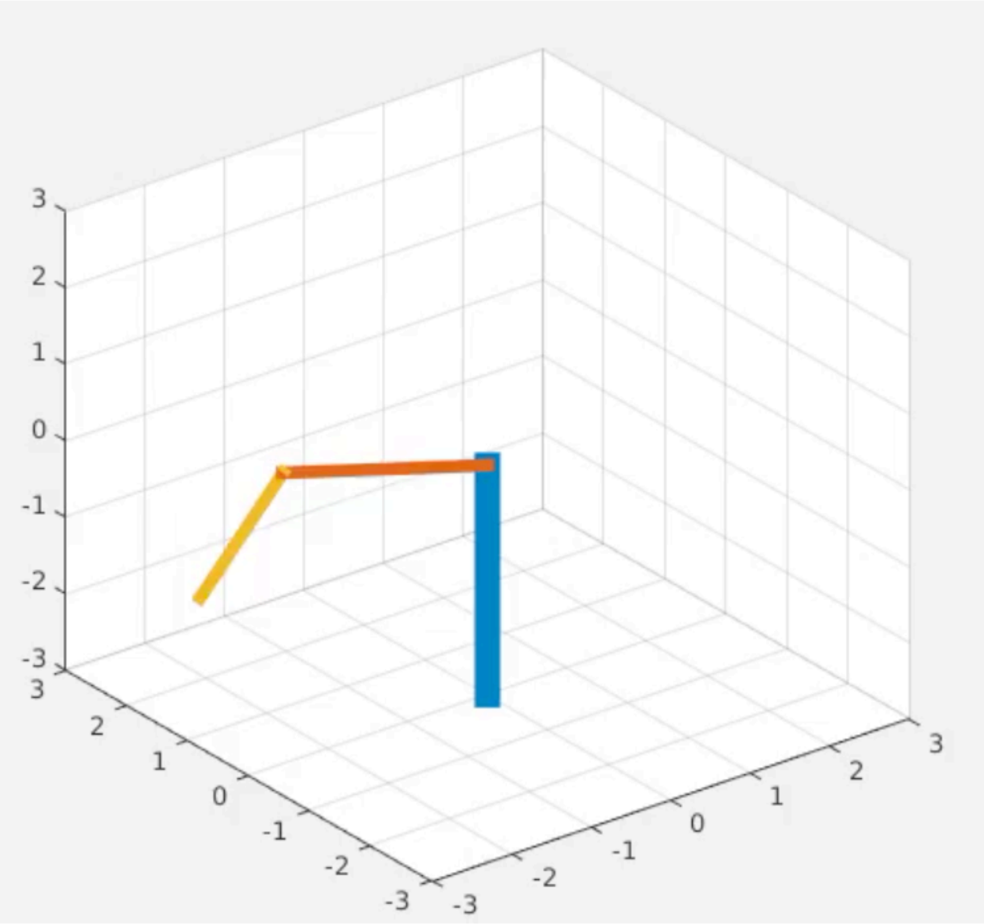
# Model

$$x = [\delta, \alpha, \beta, \dot{\delta}, \dot{\alpha}, \dot{\beta}, \ddot{\delta}, \dot{\delta}_{\mathrm{sp}}]^\top$$

$$u = \ddot{\delta}_{\mathrm{sp}}$$

$$p = (m_{\mathrm{ball}}, \mu_\alpha, \mu_\beta, \mu_{\mathrm{air}}, l_{\mathrm{tether}}, I_{\mathrm{arm}}, I_{\mathrm{LAS}})$$

$$\dot{x} = \begin{bmatrix} \dot{\delta} \\ \dot{\alpha} \\ \dot{\beta} \\ \ddot{\delta} \\ a(x,p) \\ b(x,p) \\ w_n^2 \dot{\delta}_{sp} - 2 d w_n \ddot{\delta} - w_n \dot{\delta} \\ u \end{bmatrix}$$



# MPC workflow

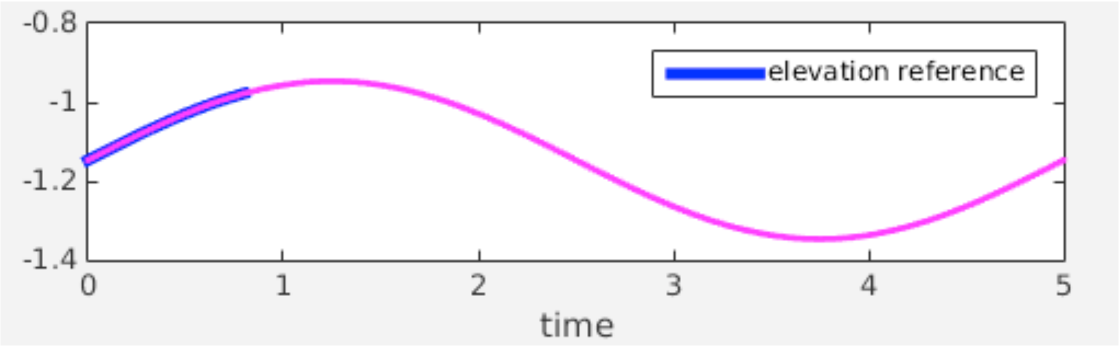
- Formulation of the Optimal Control Problem
- Implementation of OCP in ACADO Toolkit
  - Generation of optimized code
- Implementation of real-time loop

# Optimal Control Problem

$$
\begin{aligned}
\underset{x(\cdot),u(\cdot)}{\text{minimize}} \quad & \int_0^T \left\| \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} - y(t) \right\|_W^2 \mathrm{d}t \; + \; \|x(T) - y_N\|_{W_N}^2 \\
\text{subject to} \quad & x(0) \;=\; \bar{x}_0, \\
& \dot{x}(t) \;=\; f(x(t), u(t)), \qquad \forall t \in [0, T], \\
& u_{\min} \le u \le u_{\max}
\end{aligned}
$$

$$T = 40 \cdot 20\text{ms}$$

$$y_\alpha(t) =$$




$$y_{\dot{\delta}} = 0.2 \cdot 2\pi \frac{\text{rad}}{s}$$

$$y_u = 0$$

$$W = \text{diag}([10^{-8}\; 1.0\; 10^{-8}\; 0.4\; 10^{-8}\; 10^{-8}\; 10^{-8}\; 10^{-8}\; 10^{-3}])$$

$$x = [\delta, \alpha, \beta, \dot{\delta}, \dot{\alpha}, \dot{\beta}, \ddot{\delta}, \dot{\delta}_{\text{sp}}]^\top$$

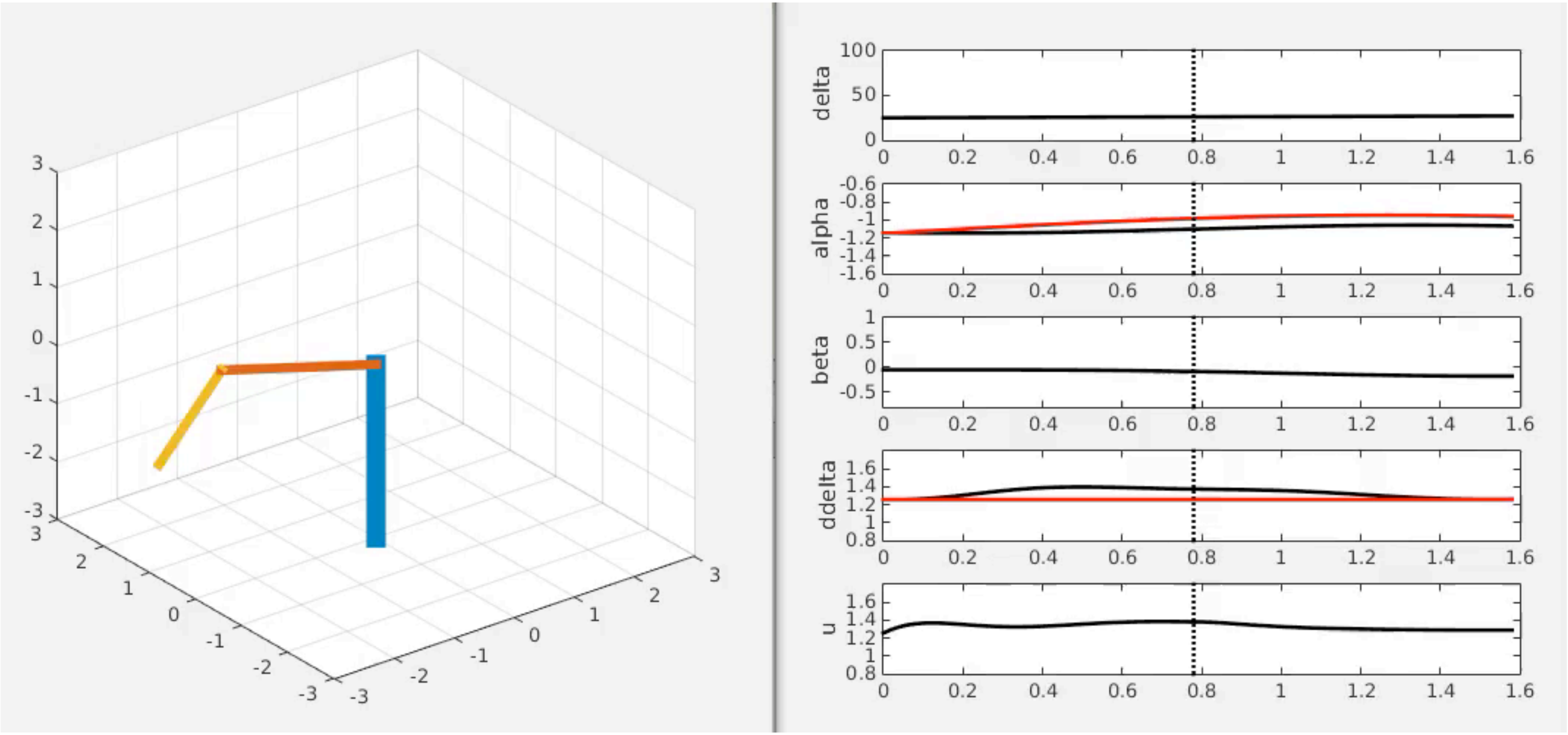
# OCP in ACADO

```
ocp = acado.OCP(0.0, T, N);
ocp.minimizeLSQ(W,[x;u]);
ocp.minimizeLSQEndTerm(WN,[xN;uN]);
ocp.subjectTo(-umin <= u <= umax);
ocp.setModel(ode);
mpc = acado.OCPexport( ocp );
mpc.set( 'HESSIAN_APPROXIMATION',    'GAUSS_NEWTON'         );
mpc.set( 'DISCRETIZATION_TYPE',      'MULTIPLE_SHOOTING'    );
mpc.set( 'SPARSE_QP_SOLUTION',       'FULL_CONDENSING_N2'   );
mpc.set( 'INTEGRATOR_TYPE',          'INT_RK4'              );
mpc.set( 'NUM_INTEGRATOR_STEPS',      N                     );
mpc.set( 'QP_SOLVER',                'QP_QPOASES'           );
---------------------EXPORT CODE---------------------
```

# Control loop

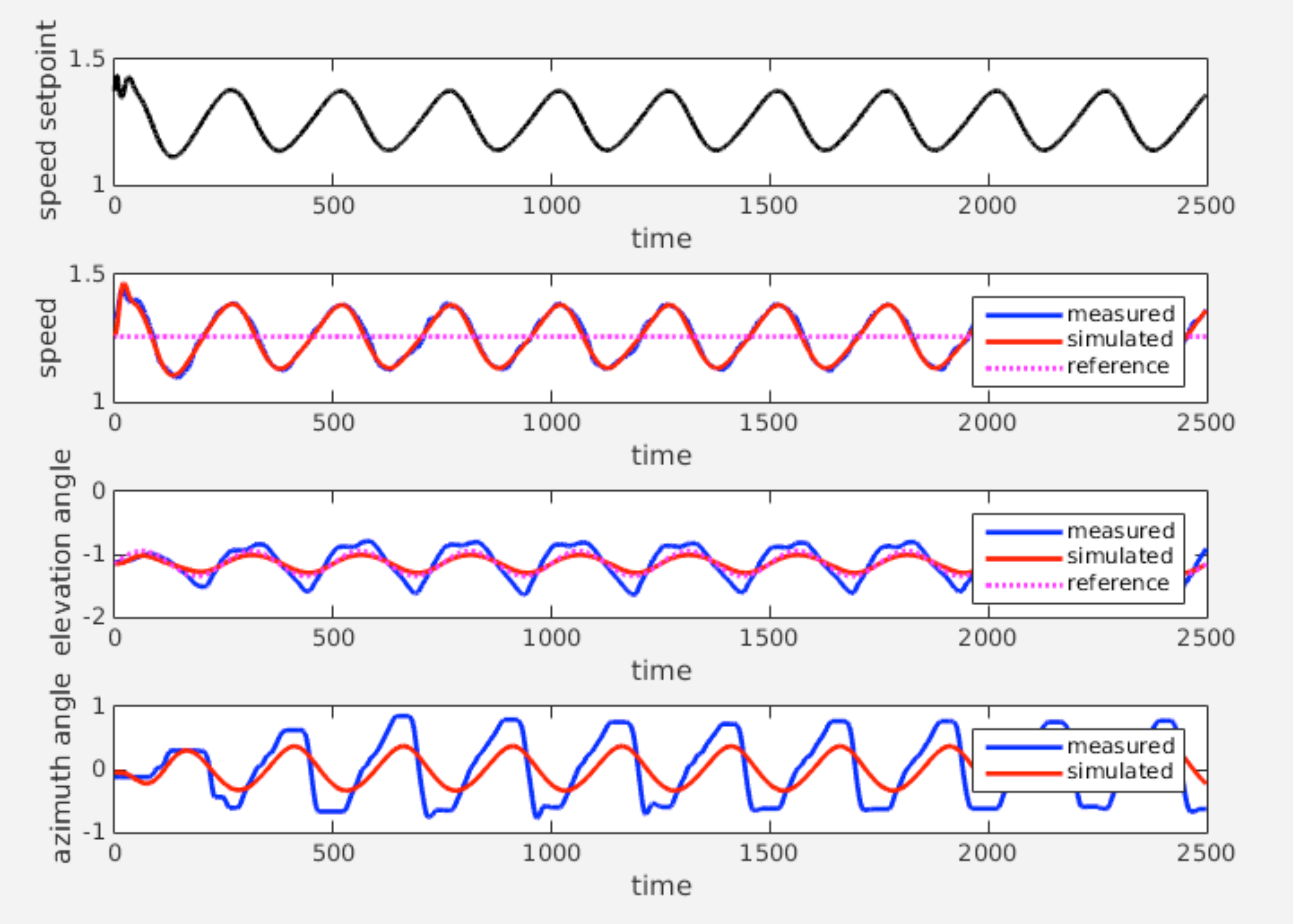
```
#include <acado_functions.h>

InitializeSolver();

acadoVariables.x = x_initialization;
acadoVariables.u = u_initialization;
preperationStep();

real-time loop:
        acadoVariables.y = y;
        acadoVariables.yN = yN;
        acadoVariables.x0 = state_feedback;
        feedBackStep();
        drives.write(control);
        shiftStates(2,0,0);
        shiftControls(0);
        preparationsStep();
```

# NMPC in Simulation

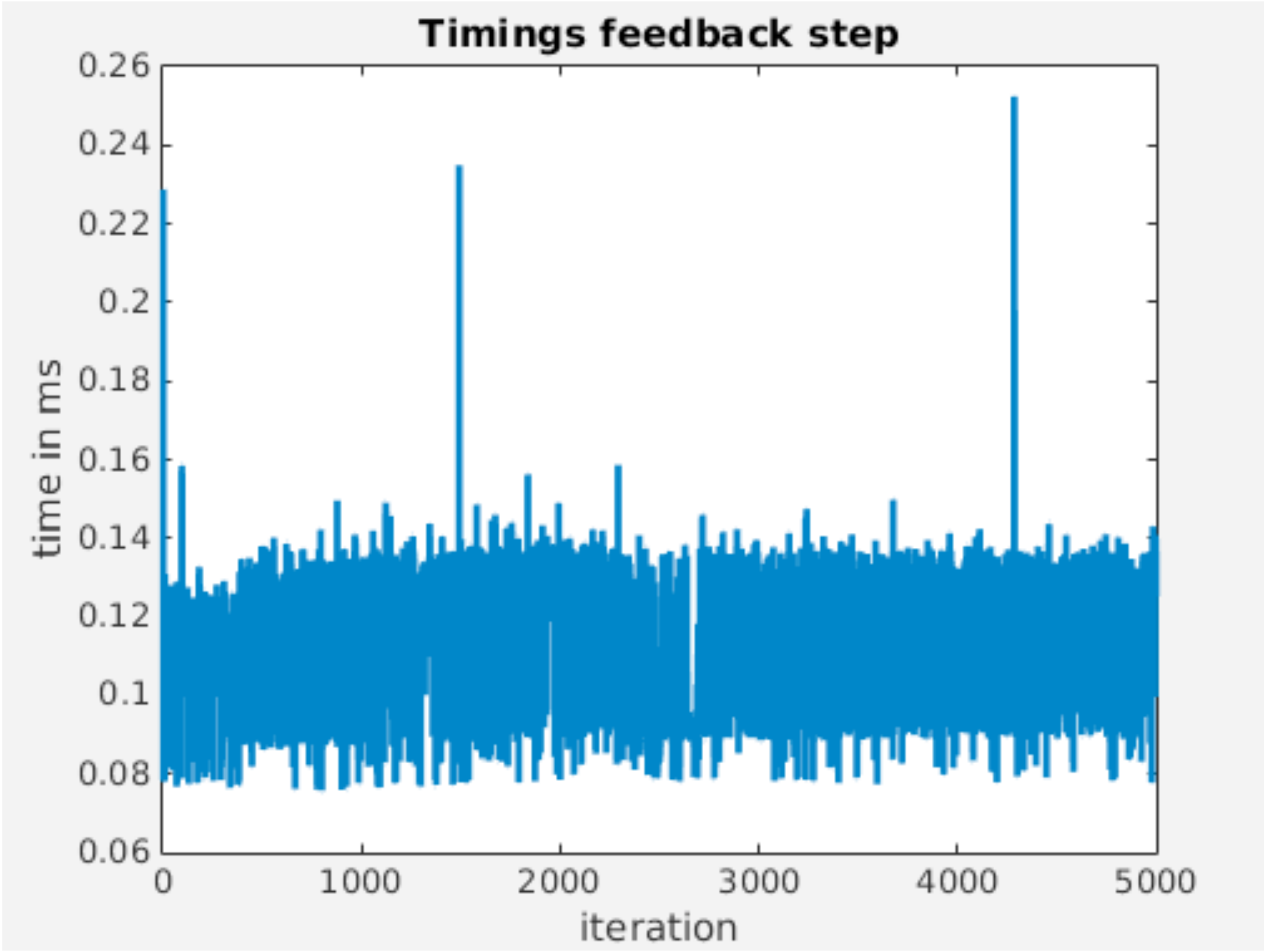


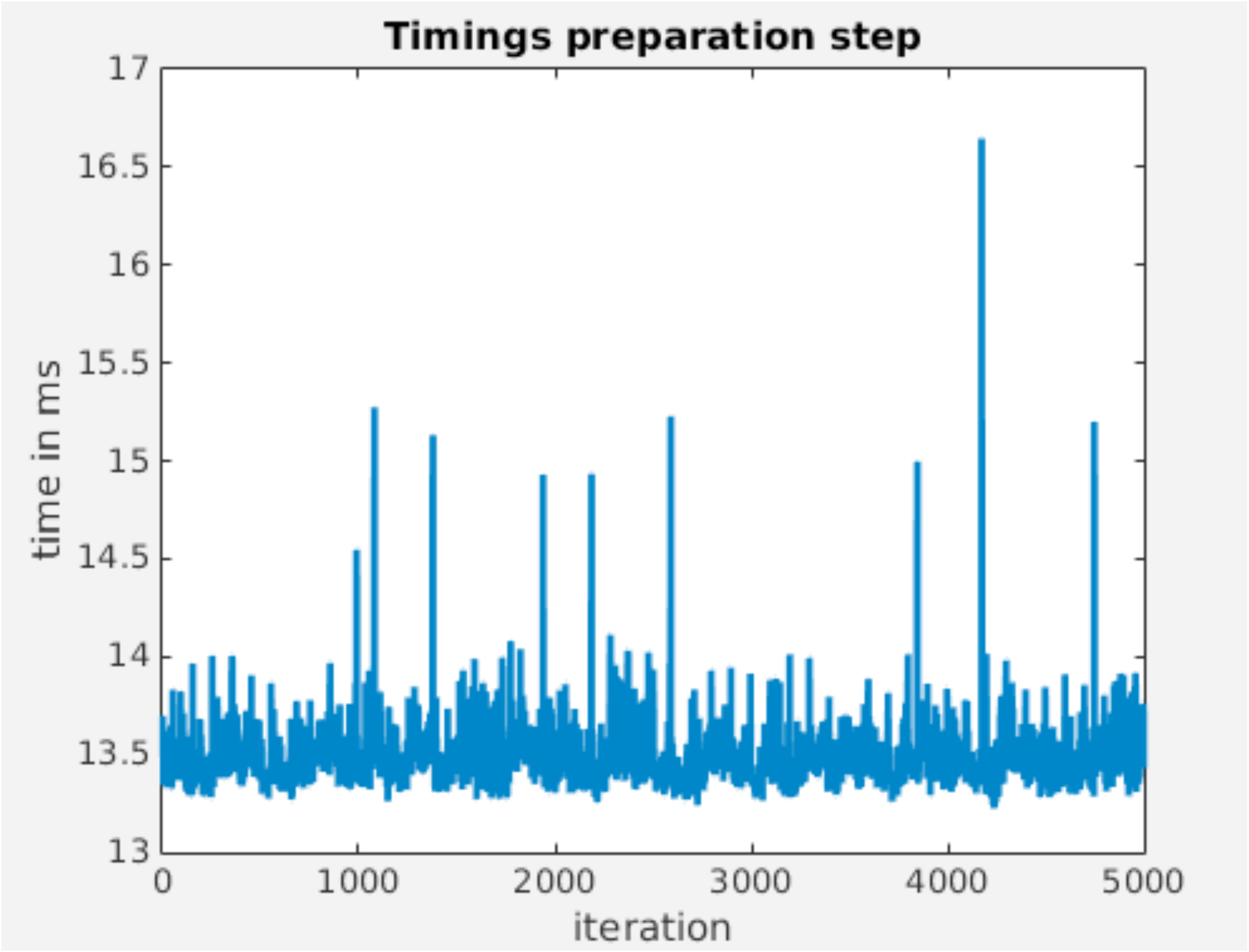
# Open loop control

# Measurements



# Timings




- Average: 13.604ms, Maximum: 16.727ms

# Conclusion

- Open loop seems to work pretty good
- Fast calculation times
- Next:
  - Integrate estimator and feedback
  - Half-wing setup

# Thank you for your attention!

Questions?