Master Project

Performance and Comparison Analysis of Linear Model Predictive Control on Reference Tracking Quadrotors

Final Presentation

Student: Hugo Grall Lucas
Supervisor: Izzet Kagan Erünsal

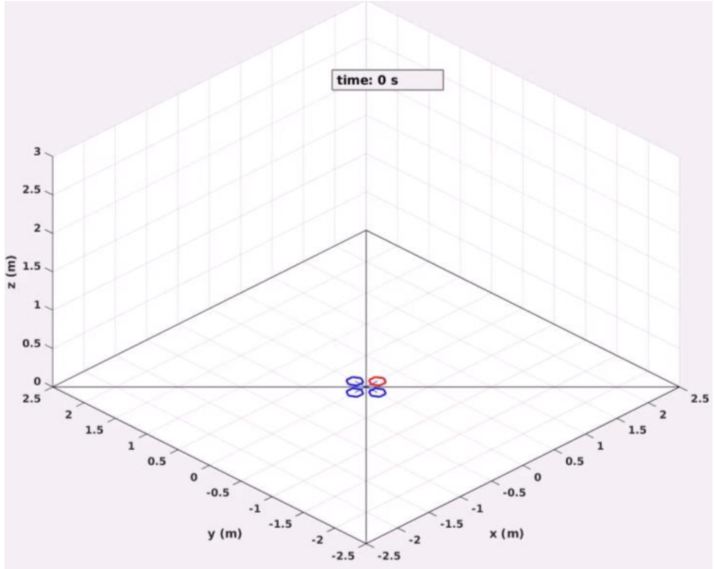
EPFL

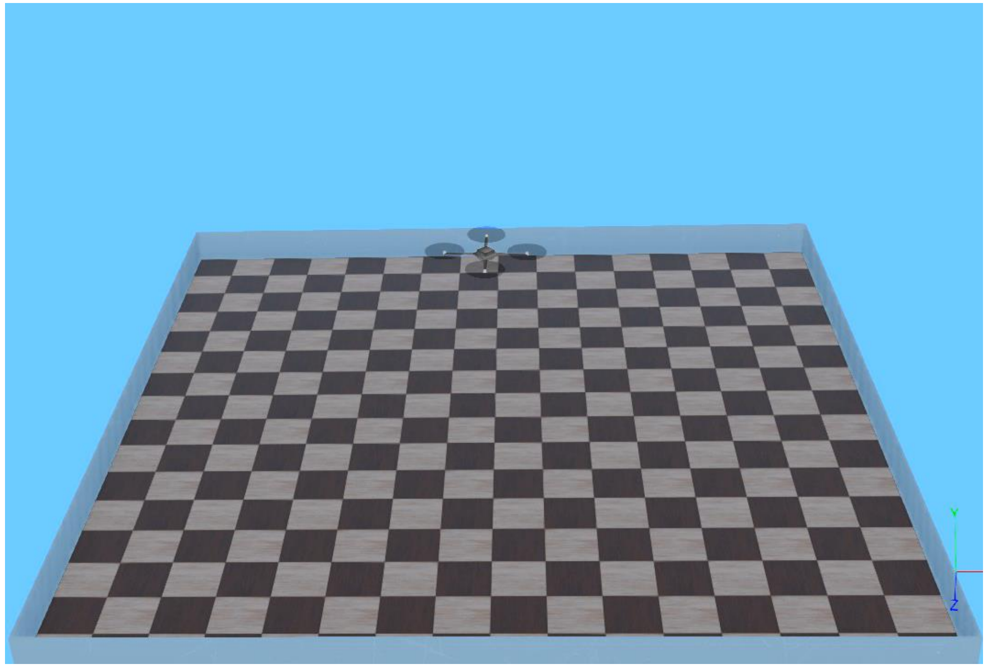
Professor: Martinoli Alcherio
Expert: Mansolino David

Introduction

Motivations

- Inspection, formation, search and rescue
- Performing high precision trajectory tracking
- Multi-variables system under constraints
- Evolution of embedded electronics capability
- Linear MPC





Introduction

Real Model : Quadrotor

Name : Helipal Storm Drone-4 v3
Mass : 1.29 Kg
Autopilot : PixHawk Cube 2
Onboard Computer : Rasberry Pi 3B+
Sensors : IMU, OpticFlow, External Tracking and External Compass

Simulation model : MATLAB and Webots

Plant : Non-linear equations for the dynamics
Attitude controller: PD controller
Parameters: Identified during experiments on the real drone by Kagan

Introduction

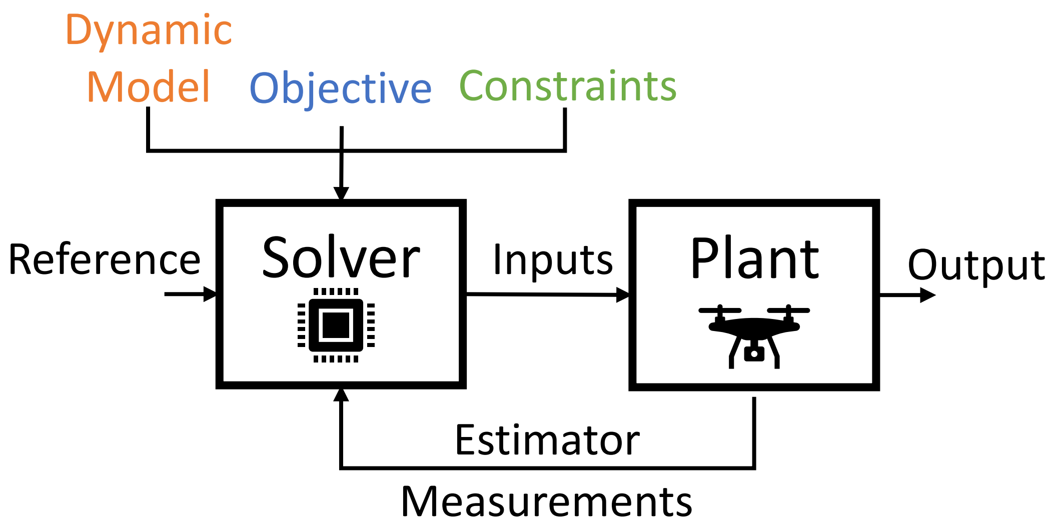
Objectives

1. Chose a dynamical model
2. Linearization, decoupling and discretization
3. Design the Optimization Problem (objective, constraints and model)
4. Select a solver for embedded optimization
5. Design a state estimator
6. MATLAB (microscopic) simulator
7. Webots (sub-microscopic) simulator
8. Update the parameters
9. Use a Raspberry PI 4 to evaluate the computational time
10. Comparative study between Linear and Nonlinear MPC [Erunsal et al. (2019)]

Model Predictive Control (MPC) [Colin Jones, 2018]

Concept

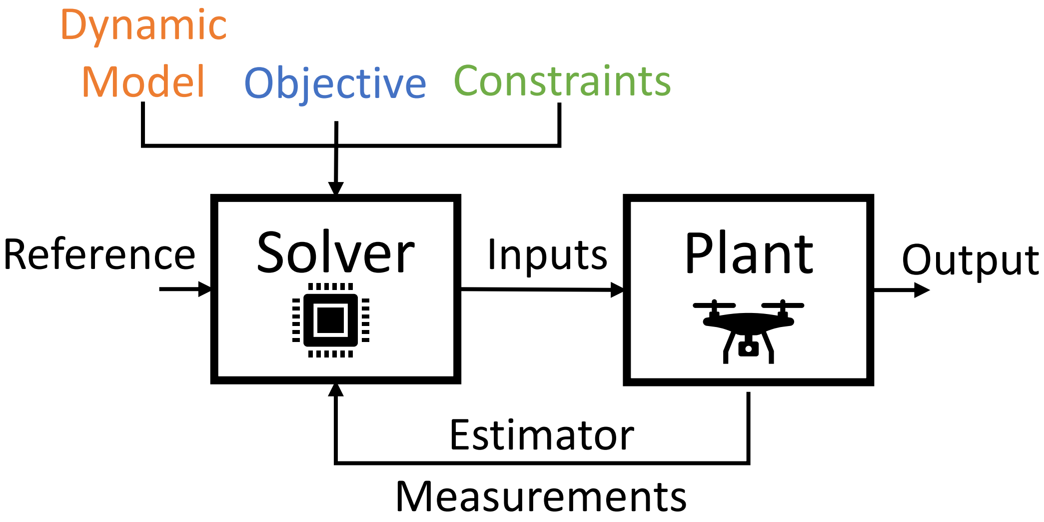
- Time variant control law
- Aim to minimize an objective
- Satisfy constraints
- Solve an optimization problem in-the-loop
- Make prediction of the state (Horizon)
- Use the first optimal input as command to the Plant



Model Predictive Control (MPC) [Colin Jones, 2018]

Concept

- Time variant control law
- Aim to minimize an objective
- Satisfy constraints
- Solve an optimization problem in-the-loop
- Make prediction of the state (Horizon)
- Use the first optimal input as command to the Plant



Optimization Problem

$$u^\star(x_0) = \operatorname{argmin} \boxed{\sum_{i=0}^{N-1} l(x_i, u_i) + V_f(x_N)}$$

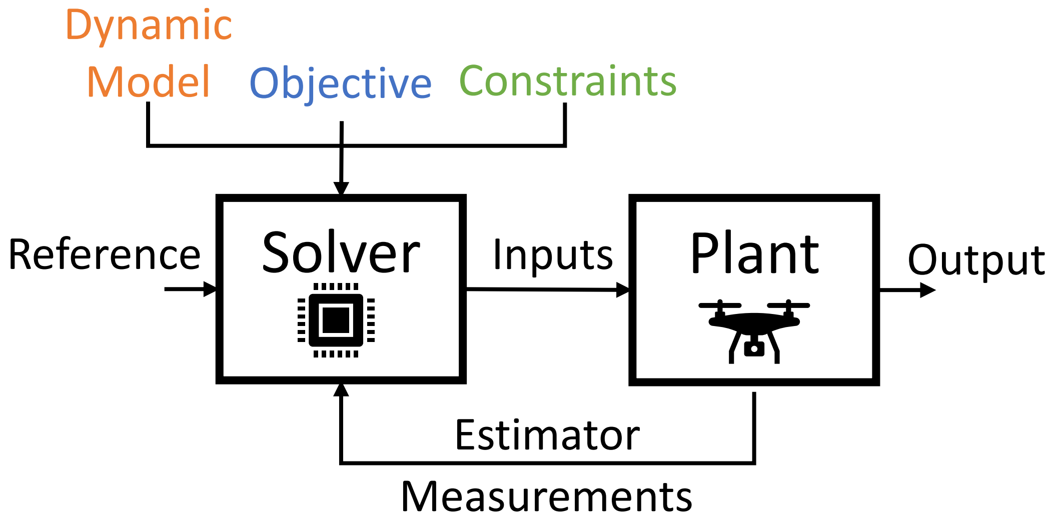
s.t.

$$\boxed{\begin{aligned} &\boxed{x_{i+1} = f(x_i, u_i)} && \forall i = 0, \ldots, N-1 \\ &g(x_i, u_i) \le 0 && \forall i = 0, \ldots, N-1 \\ &h(x_N) \le 0 \end{aligned}}$$

Model Predictive Control (MPC) [Colin Jones, 2018]

Concept

- Time variant control law
- Aim to minimize an objective
- Satisfy constraints
- Solve an optimization problem in-the-loop
- Make prediction of the state (Horizon)
- Use the first optimal input as command to the Plant



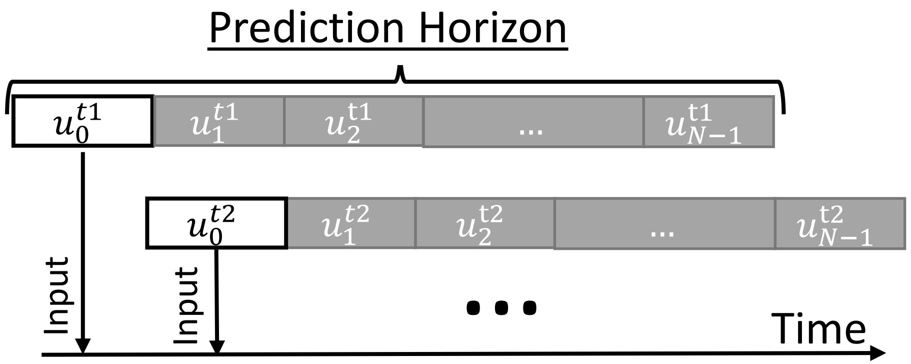
Optimization Problem

$$u^\star(x_0) = \operatorname{argmin} \boxed{\sum_{i=0}^{N-1} l(x_i, u_i) + V_f(x_N)}$$

s.t.

$$\begin{aligned} \boxed{x_{i+1} = f(x_i, u_i)} & \quad \forall i = 0, \ldots, N-1 \\ g(x_i, u_i) \leq 0 & \quad \forall i = 0, \ldots, N-1 \\ h(x_N) \leq 0 & \end{aligned}$$

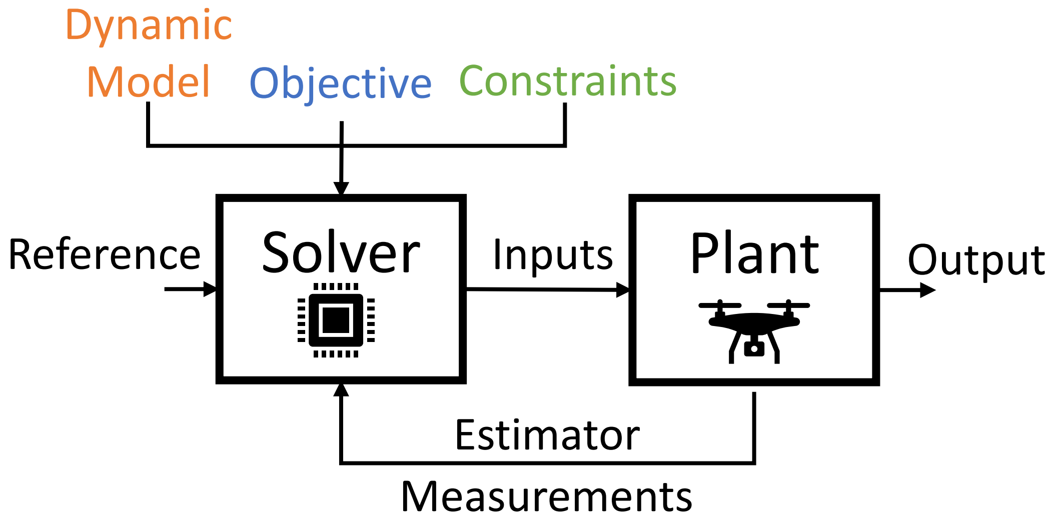
Receding horizon control



Model Predictive Control (MPC) [Colin Jones, 2018]

Concept

- Time variant control law
- Aim to minimize an objective
- Satisfy constraints
- Solve an optimization problem in-the-loop
- Make prediction of the state (Horizon)
- Use the first optimal input as command to the Plant



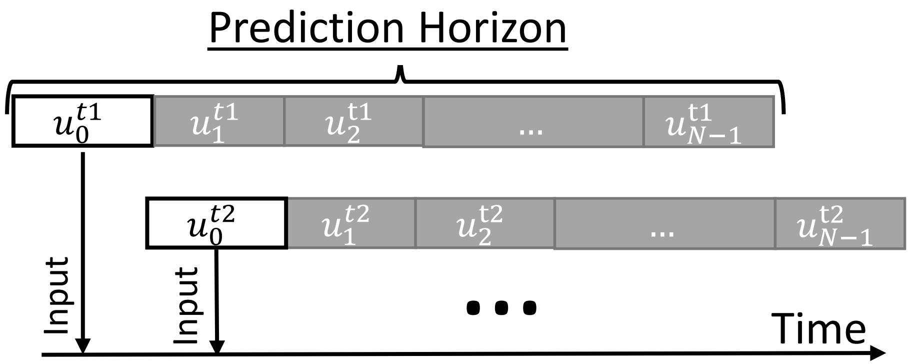
Optimization Problem

$$u^\star(x_0) = \operatorname{argmin} \boxed{\sum_{i=0}^{N-1} l(x_i, u_i) + V_f(x_N)}$$

s.t.

$$\begin{aligned} &\boxed{x_{i+1} = f(x_i, u_i)} && \forall i = 0, \ldots, N-1 \\ &g(x_i, u_i) \leq 0 && \forall i = 0, \ldots, N-1 \\ &h(x_N) \leq 0 && \end{aligned}$$

Receding horizon control



Modeling

Plant [Mahony Kumar and Corke, 2012]

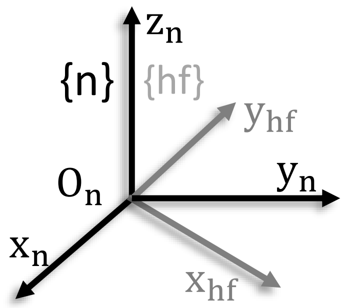
- Allocation matrix binds the squared propellers speed to force and torques

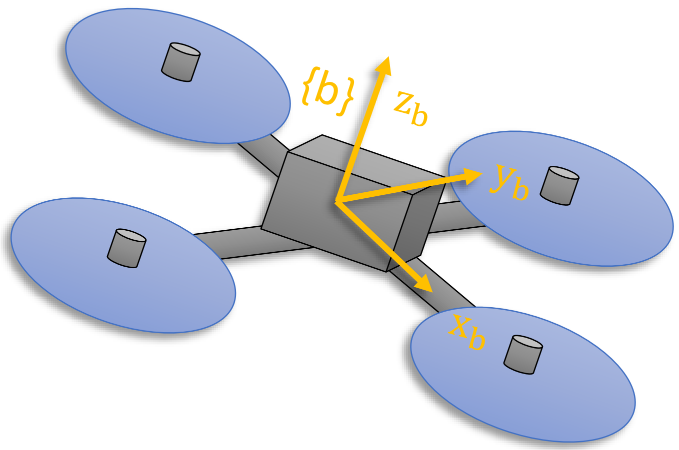
$$\begin{bmatrix} \boldsymbol{F}^b_{b/n,3} \\ \boldsymbol{\tau}^b_{b/n,1} \\ \boldsymbol{\tau}^b_{b/n,2} \\ \boldsymbol{\tau}^b_{b/n,3} \end{bmatrix} = \begin{bmatrix} c_T & c_T & c_T & c_T \\ 0 & dc_T & 0 & -dc_T \\ -dc_T & 0 & dc_T & 0 \\ c_Q & -c_Q & c_Q & -c_Q \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}$$

- Well-known model for Quadrotor dynamics use the force and torques to compute the states

$$\begin{aligned} \dot{\boldsymbol{x}}^n_{b/n} &= \boldsymbol{v}^n_{b/n} \\ m\dot{\boldsymbol{v}}^n_{b/n} &= m_b\boldsymbol{g} + \boldsymbol{R}^n_b\boldsymbol{F}^b_{b/n} \\ \dot{\boldsymbol{t}}^n_{b/n} &= \boldsymbol{T}^n_b\boldsymbol{w}^b_{b/n} \\ \boldsymbol{I}_b\dot{\boldsymbol{w}}^b_{b/n} &= \boldsymbol{\tau}^b_{b/n} - \boldsymbol{w}^b_{b/n} \times \boldsymbol{I}_b\boldsymbol{w}^b_{b/n} \end{aligned}$$

Modeling





Model for MPC [Kamel Burri and Siegwart, 2017]

- Assume cascaded architecture
- Attitude is controlled by the Autopilot
- Add drag forces

$$\dot{x}^{hf}_{b/hf} = v^{hf}_{b/hf}$$

$$\dot{v}^{hf}_{b/hf} = R^{hf}_{b} \begin{bmatrix} 0 \\ 0 \\ \frac{T_{cmd}}{m} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} - \begin{bmatrix} \kappa_{drag} & 0 & 0 \\ 0 & \kappa_{drag} & 0 \\ 0 & 0 & 0 \end{bmatrix} v^{hf}_{b/hf}$$

$$\dot{t}^{n}_{b/n} = \begin{bmatrix} \frac{1}{\tau_\phi} & 0 & 0 \\ 0 & \frac{1}{\tau_\theta} & 0 \\ 0 & 0 & \frac{1}{\tau_\psi} \end{bmatrix} \left(\begin{bmatrix} K_\phi & 0 & 0 \\ 0 & K_\theta & 0 \\ 0 & 0 & K_\psi \end{bmatrix} t^{n}_{cmd_{b/n}} - t^{n}_{b/n} \right)$$

Observations

- Dynamics doesn't depend on position
- **hf** frame is the **n** frame rotated by Yaw around Z axis

Linearization, Decoupling and Discretization

- Linearization around Hovering positions
- 1 dynamical model + infinity of Hovering position
- 4 Subsystems: **[Siegwart et al., 2012]**
  - Pitch : $x = [x, v_x, \theta]^T \quad u = \theta_{cmd}$
  - Roll : $x = [y, v_y, \phi]^T \quad u = \phi_{cmd}$
  - Z : $x = [z, v_z]^T \quad u = T_{cmd}$
  - Yaw : $x = [\psi] \quad u = \psi_{cmd}$
- Use ZOH methode (c2d MATLAB command)

Taylor expansion at steady state

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

$$f(x^*,u^*) = 0$$

$$x^* = \begin{bmatrix} x_{ref} & y_{ref} & z_{ref} & 0 & 0 & 0 & 0 & 0 & \psi_{ref} \end{bmatrix}^T$$

$$u^* = \begin{bmatrix} mg & 0 & 0 & \frac{\psi_{ref}}{K_\psi} \end{bmatrix}^T$$

$$\forall x_{ref}, y_{ref}, z_{ref}, \psi_{ref} \in \Re$$

$$A = \left.\frac{\partial f(x,u)}{\partial x}\right|_{x^*,u^*} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{-K_{drag}}{m} & 0 & 0 & 0 & g & 0 \\ 0 & 0 & 0 & 0 & \frac{-K_{drag}}{m} & 0 & -g & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{\tau_\phi} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{\tau_\theta} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{-1}{\tau_\psi} \end{bmatrix}$$

$$B = \left.\frac{\partial f(x,u)}{\partial u}\right|_{x^*,u^*} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{m} & 0 & 0 & 0 \\ 0 & \frac{K_\phi}{\tau_\phi} & 0 & 0 \\ 0 & 0 & \frac{K_\theta}{\tau_\theta} & 0 \\ 0 & 0 & 0 & \frac{K_\psi}{\tau_\psi} \end{bmatrix}$$

Linearization, Decoupling and Discretization

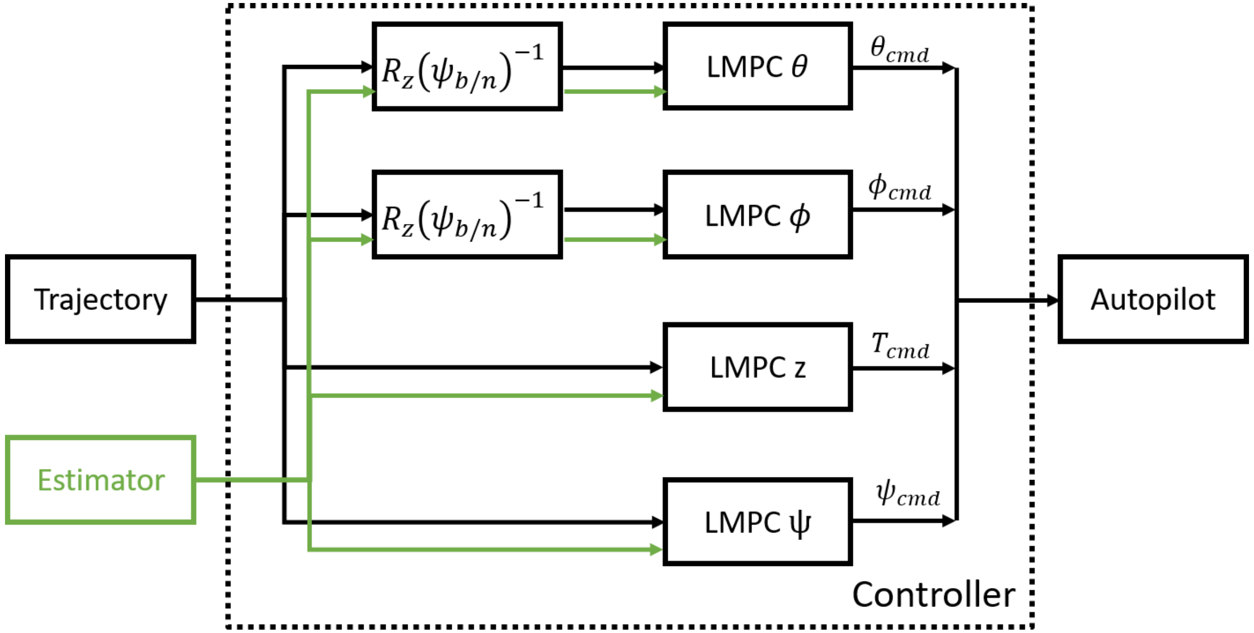
- Linearization around Hovering positions
- 1 dynamical model + infinity of Hovering position
- 4 Subsystems: **[Siegwart et al., 2012]**

  Pitch : $x = [x, v_x, \theta]^T \quad u = \theta_{cmd}$

  Roll : $x = [y, v_y, \phi]^T \quad u = \phi_{cmd}$

  Z : $x = [z, v_z]^T \quad u = T_{cmd}$

  Yaw : $x = [\psi] \quad u = \psi_{cmd}$
- Use ZOH methode (c2d MATLAB command)



Taylor expansion at steady state

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

$$f(x^*,u^*) = 0$$

$$x^* = \begin{bmatrix} x_{ref} & y_{ref} & z_{ref} & 0 & 0 & 0 & 0 & 0 & \psi_{ref} \end{bmatrix}^T$$

$$u^* = \begin{bmatrix} mg & 0 & 0 & \frac{\psi_{ref}}{K_\psi} \end{bmatrix}^T$$

$$\forall x_{ref}, y_{ref}, z_{ref}, \psi_{ref} \in \Re$$

Objective : Cost Functions Component

Standard Quadratic Cost

$$l_i^{LQR}(x,u) = x_i^T Q x_i + u_i^T R u_i$$

Command Rate Cost

[Kamel Burri and Siegwart, 2017]

$$l_i^{rate}(x,u) = (u_i - u_{i-1})^T R_\Delta (u_i - u_{i-1})$$

Terminal Cost

$$V_f(x_N) = x_N^T P x_N$$

$$P = Q + A^T P A - A^T P B \left(R + B^T P B\right)^{-1} B^T P A$$

Objective

$$V(x,u) := \sum_{i=0}^{N-1} l_i^{LQR}(x,u) + l_i^{rate}(x,u) + V_f(x,u)$$

Constraints Components

Dynamic contraints

$$x_{i+1} = A(x_i - x^*) + B(u_i - u^*) \quad \forall i = 0, ..., N-1$$

Inputs contraints

Thrust: limited by the motors

Attitude angles : limited by the linearization

$$u_{min} \leq u \leq u_{max}$$

States contraints

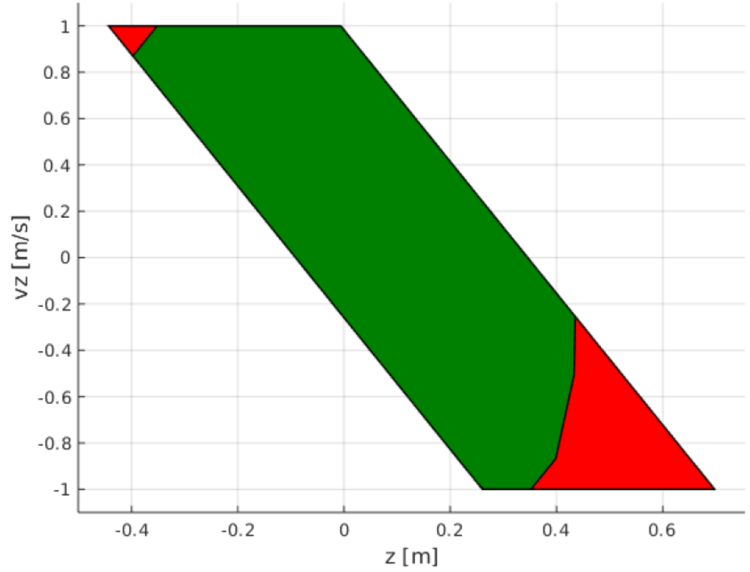
Attitude angles: also limited by the linearization

Velocity: Limited by the motion capture system

$$x_{min} \leq x \leq x_{max}$$

Terminal constraints (set)

- Needed for theorical proof of feasibility and stability of the closed-loop controller
- Defined as the maximum invariant set under the control law that respect the constraints



Solver : Embedded computing

Comparative study [Adriano Silva Martins Brandão et al., 2018]

	Obj. Function	Constraints	Code Size	Performance
CVXGEN	Flexibility in the formulation	Flexible and allows slack variables	several MB	fast and reliable
FalcOpt	Imposed by the solver	Only on control inputs	several KB	really fast but not reliable
muAO	Imposed by the solver	Combinations of states and inputs	several MB	fast and reliable

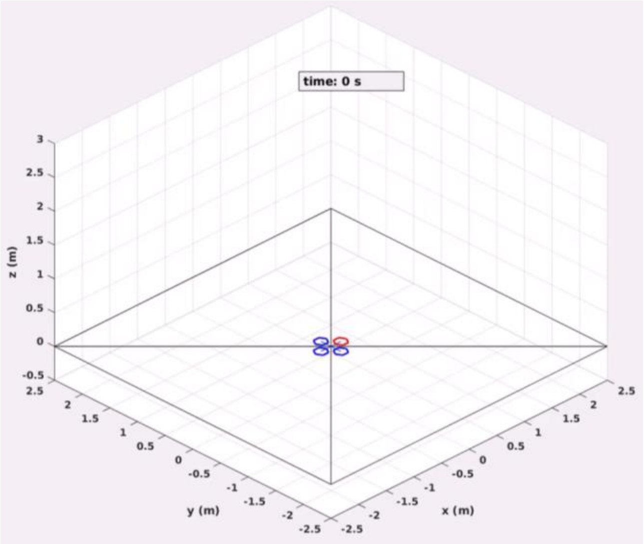
CVXGEN [Mattingley and Boyd, 2012]

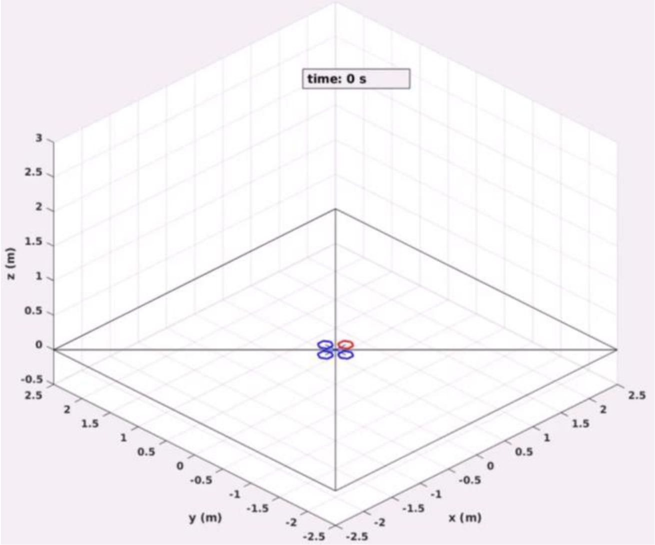
- Flexible formulation of the objective function
- Quick to solve and reliable
- Successfully used in [Kamel Burri and Siegwart, 2017]
- C Code generation + MATLAB interface
- (Sometimes) Slow to generate C code from the web interface

Sensors and estimator

Sensors

- ❖ The states are fully observable
  - Positions : Motion capture system
  - Velocity : Optical Flow
  - Attitude : Autopilot estimator
- ❖ Simulation : Noise modelled as gaussian

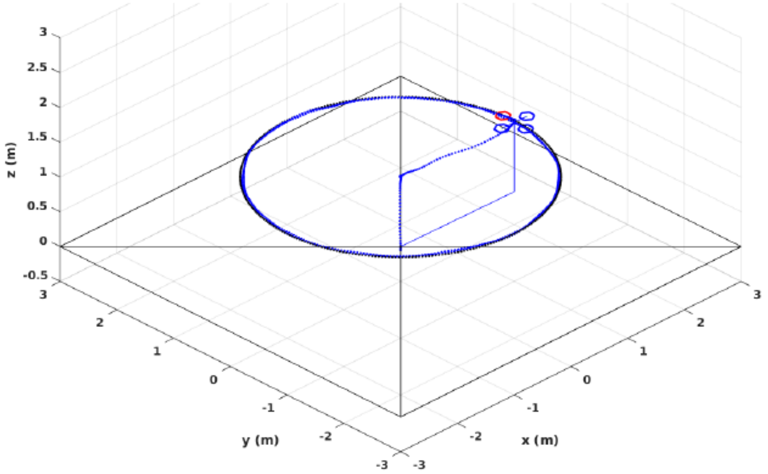


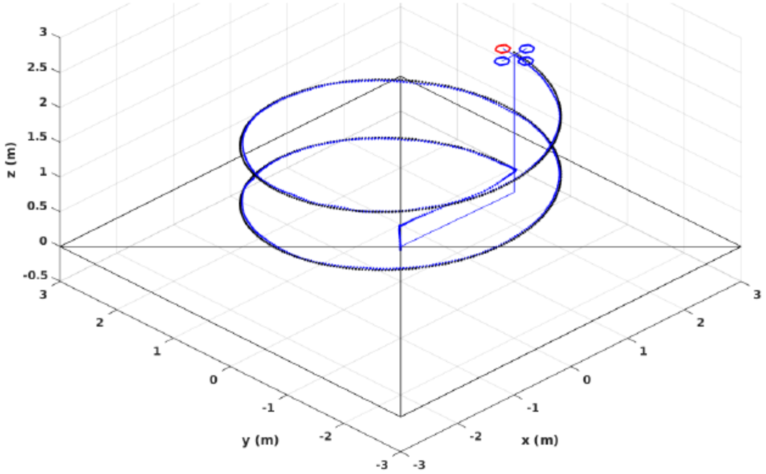


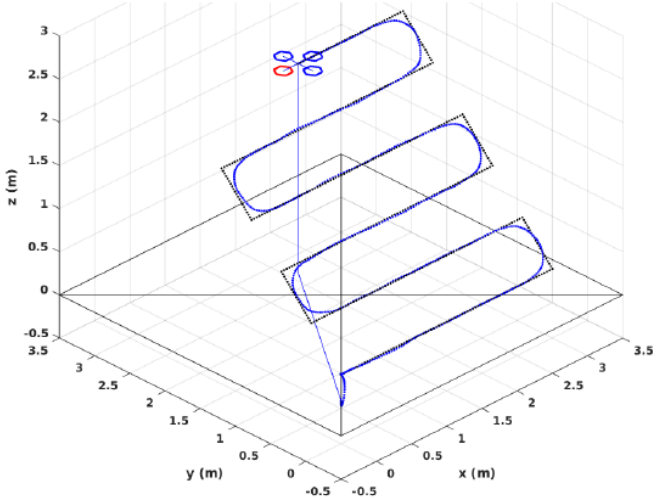
Extended Kalman Filter (EKF)

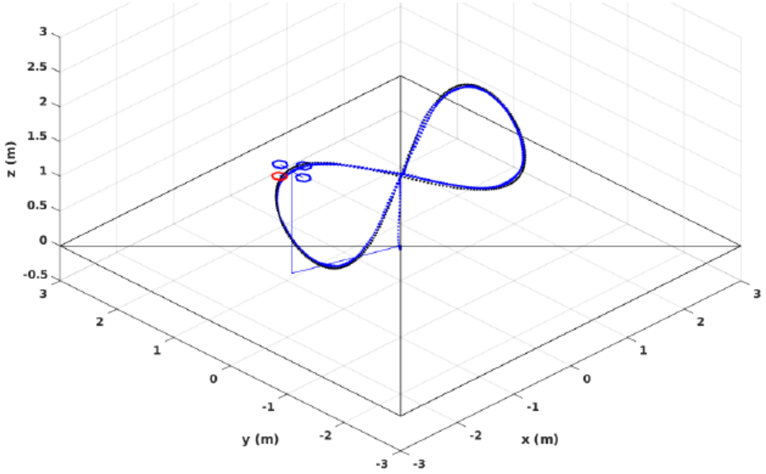
- ❖ Measurement model: Identity matrix
- ❖ Process model: non-linear model used for control but expressed in world frame
- ❖ Discretization : Use Euler forward method

Trajectory tracking : MATLAB

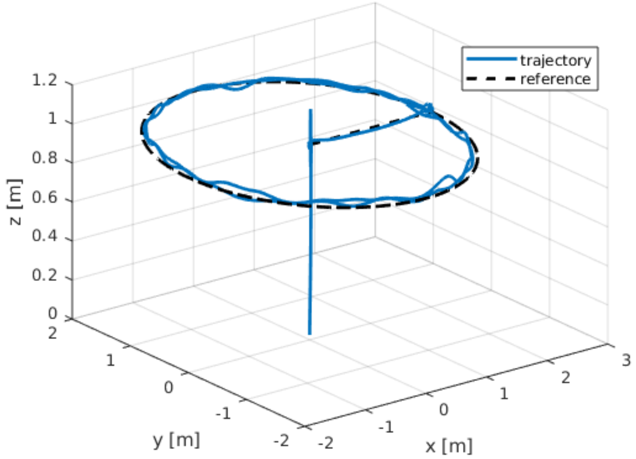


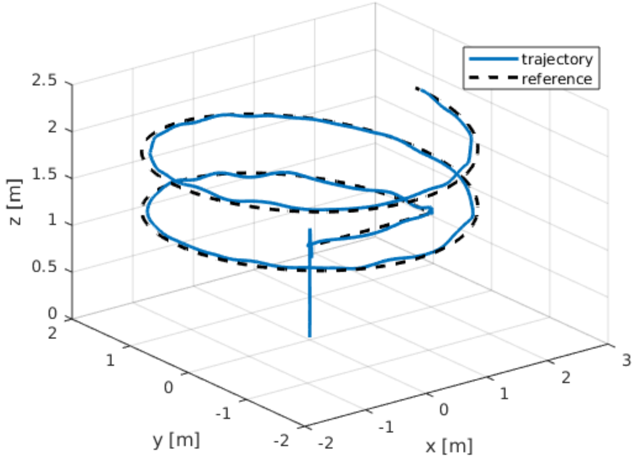






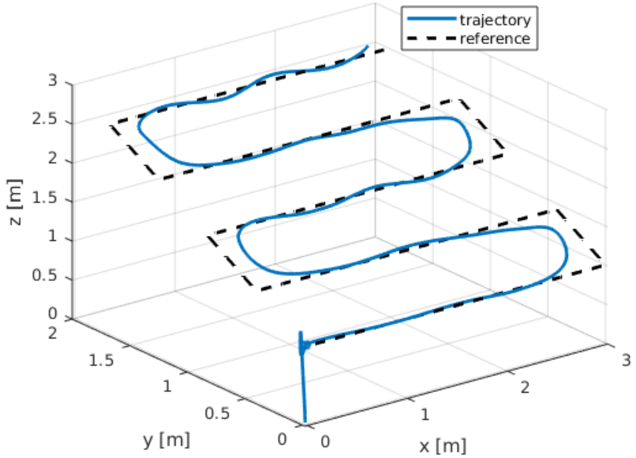
Trajectory tracking : Webots and ROS

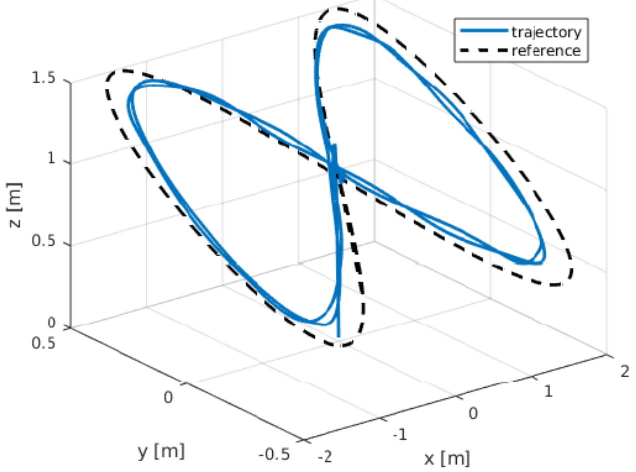




Observations

- Drag forces
- Noise on the forces
- Closed-loop effect
- High acceleration





Nonlinear vs linear MPC

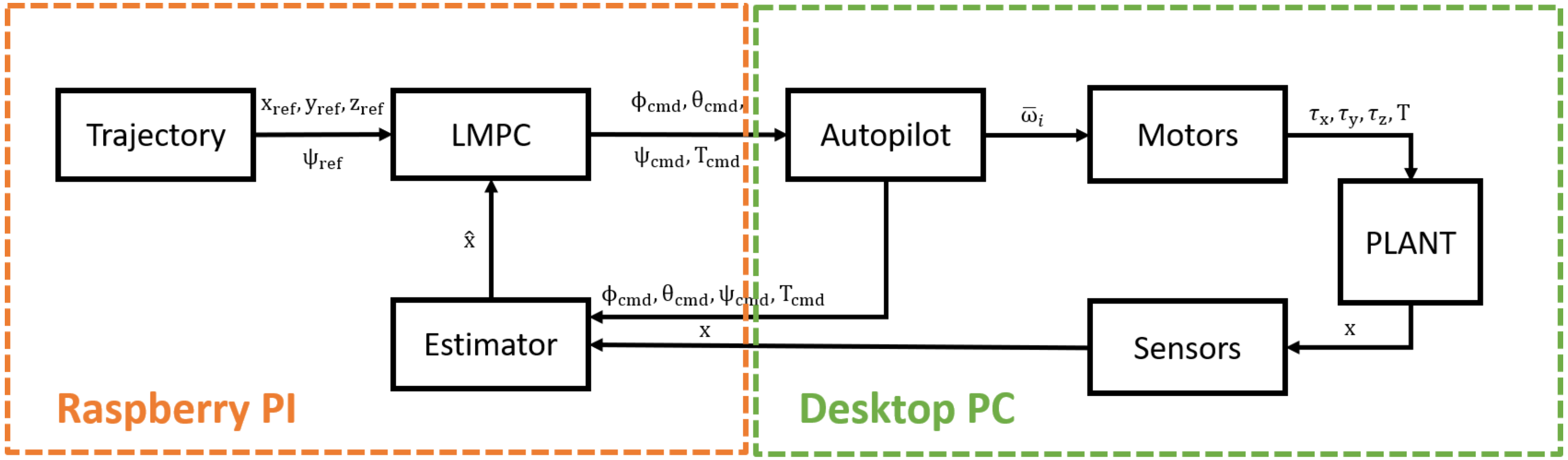
Comparison

	Linear	Nonlinear
Linear Dynamics	Yes	No
Decoupling	Yes	No
Small angles	Yes	No
Solver	CVXGEN	ACADO
Computational Load	Medium	High

How to be fair ?

- Tuning procedure
- Allocation of the computational time
- Different N
- Different the controller frequency
- Increase the validity of the linearization
- Use same solver

Nonlinear vs linear MPC



Budget allocation

- Step responses on Raspberry PI 4
- Use maximum N for LMPC
- Test two N values for NMPC
- Controller loop time 0.05 s

Step Response

	LMPC	NMPC	
N	30	15	25
mean time [s]	0.0142	0.0171	0.0338
max time [s]	0.0298	0.1042	0.5421

Nonlinear vs linear MPC

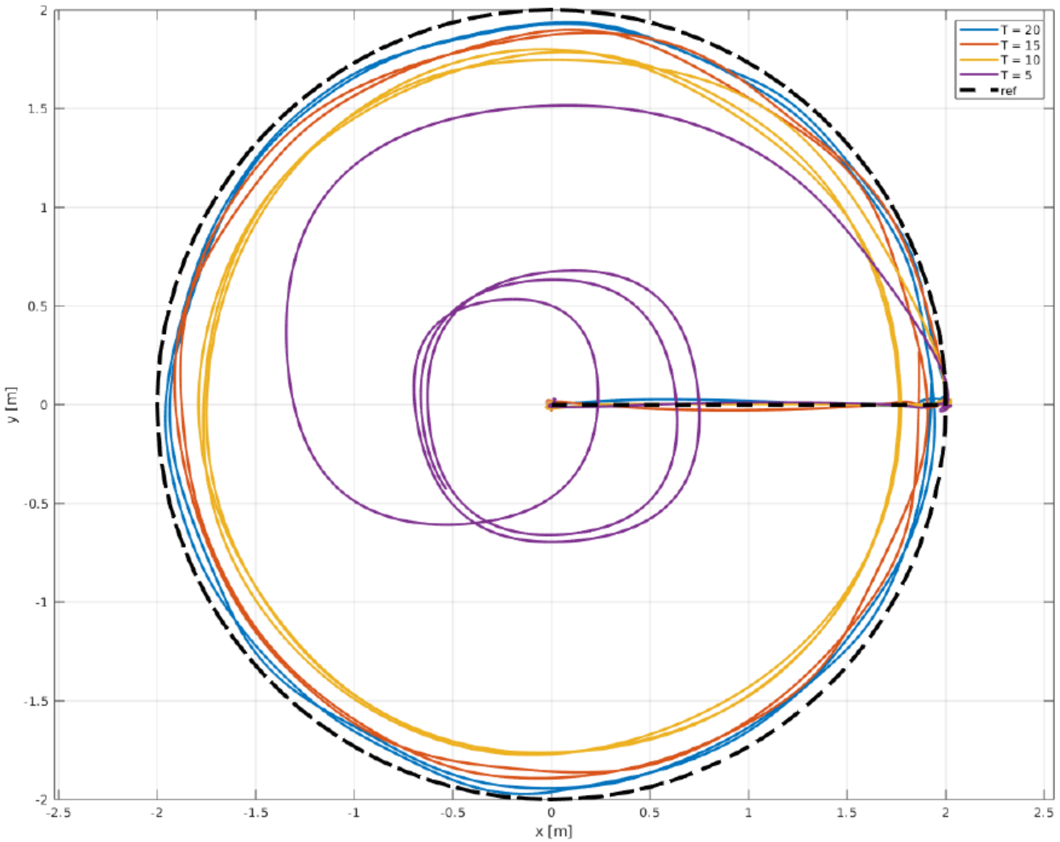
Experiment

- Track circle in x-y plan
- Measure the accuracy with RMSE in meter
- Measure the agility of the controller
- Decrease the period of the circle T

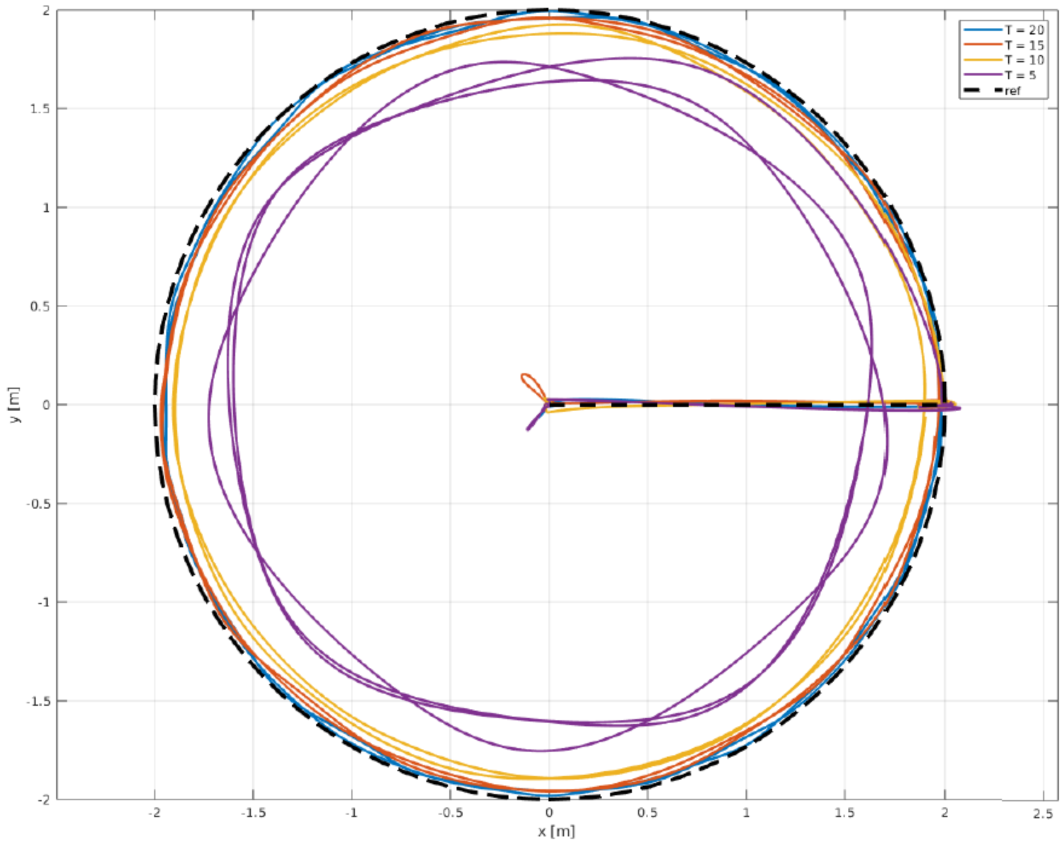
	LMPC				NMPC			
T	20	15	10	5	20	15	10	5
$\mathrm{RMSE_x}$	0.0878	0.1226	0.1901	1.3793	0.0381	0.0504	0.0903	0.2466
$\mathrm{RMSE_y}$	0.0831	0.1163	0.1837	1.4064	0.0357	0.0473	0.0856	0.2336
$\mathrm{RMSE_z}$	0.0172	0.0195	0.016	0.0151	0.023	0.0337	0.0369	0.0603

Nonlinear vs linear MPC

LMPC



NMPC



Conclusion

Work summary

- Implement LMPC for tracking trajectories
- Set up and update two simulation models
- Run the proposed controller in a Raspberry PI 4 while the physics is computed on Webots
- Propose a comparative study between Linear and Nonlinear MPC

Future possible work

- Explore more the fairly comparison
- Valid the work on real hardware
- Disturbance observer (offset free tracking)
- Extended to Robust Linear MPC
- Explore other linearization (feedback linearization)

References

[Adriano Silva Martins Brandão, 2018] Adriano Silva Martins Brandão, Daniel Martins Lima, Marcus Vinicius Americano da Costa Filho, and Julio Elias Normey-Rico (2018). A Comparative Study On Embedded MPC For Industrial Processes. João Pessoa, Paraba, Brasil.

[Colin Jones, 2018] Colin Jones (2018), lecture notes from Model Predictive Control course ME-425, EPFL (Switzerland).

[Mahony Kumar and Corke, 2012] Robert Mahony, Vijay Kumar and Peter Corke (2012), Multirotor Aerial Vehicles – Modeling, Estimation, and Control of Quadrotor, IEEE ROBOTICS & AUTOMATION MAGAZINE.

[Kamel Burri and Siegwart, 2017] Mina Kamel, Michael Burry and Roland Siegwart (2017), Linear Vs Nonlinear MPC for Trajectory Tracking Applied to Rotary Wind Micro Aerial Vehicles, IFAC-PapersOnLine.

[Siegwart et al., 2012] Michael Burri, Janosh Nikilic, Christoph Hürzeler, Gilles Caprari and Roland Siegwart (2012), Aerial Service Robots for Visual Inspection of Thermal Power Plant Boiler system, CARPI.

[Mattingley and Boyd, 2012] Jacob Mattingley and Stephan Boyd (2012), CVXGEN: a code generator for embedded convex optimization, Springer.

Questions

Illustration: https://www.lecoindesentrepreneurs.fr/wp-content/uploads/2015/06/liste-de-questions-%C3%A0-poser-au-franchiseur.png [visited: 29.04.2019]

Sensors and estimator