Linear vs Nonlinear MPC for Trajectory Tracking Applied to Rotary Wing Micro Aerial Vehicles

Mina Kamel, Michael Burri and Roland Siegwart [*]

[*] *Authors are with the Autonomous Systems Lab, ETH Zürich, Switzerland,*
fmina@ethz.ch

Abstract: Precise trajectory tracking is a crucial property for Micro Air Vehicles (MAVs) to operate in cluttered environment or under disturbances. In this paper we present a detailed comparison between two state-of-the-art model-based control techniques for MAV trajectory tracking. A classical Linear Model Predictive Controller (LMPC) is presented and compared against a more advanced Nonlinear Model Predictive Controller (NMPC) that considers the full system model. In a careful analysis we show the advantages and disadvantages of the two implementations in terms of speed and tracking performance. This is achieved by evaluating hovering performance, step response, and aggressive trajectory tracking under nominal conditions and under external wind disturbances.

Keywords: UAVs, Predictive Control, Trajectory Tracking and Path Following, Real-time control.

1. INTRODUCTION

MAVs are gaining a growing attention recently thanks to their agility and ability to perform tasks that humans are unable to do. Many researches have employed MAVs successfully to perform infrastructure inspection as shown in Bircher et al. (2016b), exploration tasks in unknown environment as in Bircher et al. (2016a), search and rescue operations as presented in Oettershagen et al. (2016) or forest resources monitoring as shown in Steich et al. (2016). Precise trajectory tracking is an important feature for aerial robots when operating in real environment under external disturbances that may heavily affect the flight performance, especially in vicinity to structures. Furthermore, there is a big boost in personal drones which need to be able to track a moving object and take nice aerial footage requiring fast and agile trajectory tracking.

Currently, it is possible to perform mapping, 3D reconstruction, localization, planning and control completely on-board thanks to the great advances in electronics and semiconductor technology. To fully exploit the robot capabilities and to take advantage of the available computation power, optimization-based control techniques are becoming suitable for real-time MAV control.

In this paper, we present a comparison between two of the state-of-the-art trajectory tracking controllers. A LMPC based on a linearized model of the MAV and a more advanced NMPC that considers the full system dynamics. The goal of this comparison is to emphasis the benefits and drawbacks of considering the full system dynamics in terms of performance improvement, disturbance rejection and computation effort.

The general control structure followed in this paper is a cascade control approach, where a reliable and system-specific low-level attitude controller is present as inner-loop, while a model-based trajectory tracking controller is running as an outer-loop. This cascade approach is justified by the fact that critical flight control algorithm is running on a separate navigation hardware, typically based on miro-controller such as PixHawk by Meier (2016), while high level tasks are typically executed on a more powerful on-board computer. This introduces a separation layer to keep critical code running despite any failure in the high-level computer.

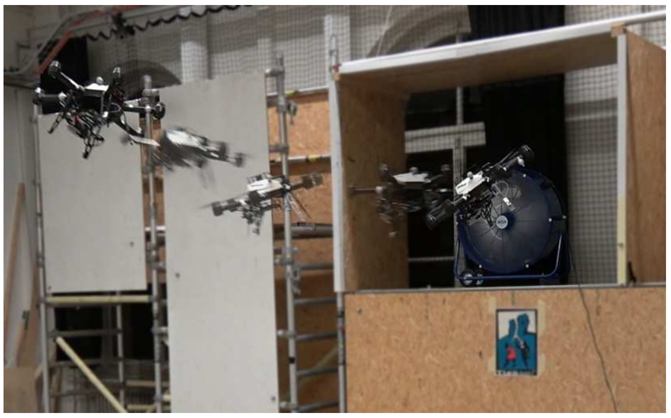
Fig. 1. Sequence of the MAV poses during aggressive trajectory tracking experiment under wind disturbances.

To achieve high performance with the cascade approach, it is necessary to account for the inner-loop dynamics in the trajectory tracking controller. Therefore, a system identification has been performed on the MAV to identify the closed loop attitude dynamics.

Paper Contributions

The main goal of this paper is to perform a thorough comparison between a LMPC and NMPC controller enabling dynamic trajectory tracking for MAVs together with an open source C++ implementation of both controllers on Kamel (2016). The comparison aims to highlight the benefits of full system dynamics consideration and its effect on the flight envelope.

⋆ This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 644128 and from the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract number 15.0044.

The remainder of this paper is organized as follows: In Section 2 the related work of the problem of MAV trajectory tracking control is discussed. In Section 3 the MAV model is presented. The Linear and Nonlinear Model Predictive Controller are presented in Section 4 and Section 5 respectively. The external disturbance observer is briefly discussed in Section 6. Finally, experimental results are presented and discussed in Section 7.

2. RELATED WORK

Many researchers have been focusing on the control problem for MAVs. Among the first controller evaluation on MAV is the work done by Samir Bouabdallah in Bouabdallah et al. (2004) where a comparison between classic PID controller and LQ controller has been conducted for attitude stability. Surprisingly the authors found that a simple PID controller was performing better than an LQ controller for attitude stability. The authors justify this result as a result of imperfect model.

A commonly used nonlinear controller was proposed by Lee et al. (2010), where the attitude error is directly calculated on the manifold to have a globally stable controller. Using a very similar formulation, impressive trajectory tracking results were obtained using high gain feedback control Mellinger et al. (2012). These controllers, however, are not able to guarantee any state or input constraints and the trajectory needs to be selected carefully. Model Predictive Controller (MPC) on the other hand is able to directly include constraints in the optimization, which is important for real systems with physical constraints.

In our previous work presented in Kamel et al. (2015) a NMPC approach is successfully employed to control the MAV attitude on the Special Orthogonal group $SO(3)$. The controller is based on a geometric formulation of the attitude error. Experimental results showed the ability of the controller to recover from any configuration and to handle propeller failure. However, the computation cost limits the use of such controller in practice.

In Bemporad et al. (2009) the authors employed a hierarchical MPC to achieve stability and autonomous navigation for MAVs. To achieve stable flight, a linear MPC is employed while a hybrid MPC approach is used to generate collision-free trajectory which is tracked by the linear MPC. However, the proposed stabilizing controller has not been tested experimentally and controller performance is not evaluated under disturbances.

To add robustness with respect to time delays in on-board estimation Blösch et al. (2010) proposed to separate the controller into an attitude controller running on the micro controller and an Linear-Quadratic Regulator (LQR) for position tracking. For better tracking performance the closed loop dynamics of the attitude controller was identified and included in the LQR This approach was extended in Burri et al. (2012) with a LMPC for position tracking. In this work we extended this controller with a simple aerodynamic drag model and use it as our reference implementation.

A similar formulation was also used in Alexis et al. (2016). The authors present a Robust MPC approach to stabilize the vehicle and demonstrate the effectiveness of the proposed controller under wind disturbances and slung load.

The authors in Berkenkamp and Schoellig (2015) propose a novel approach to deal with conservativeness of robust control techniques. The system uncertainty is learned online using Gaussian Process and the linear robust controller is updated according to the learned uncertainties achieving less conservative linear robust controller.

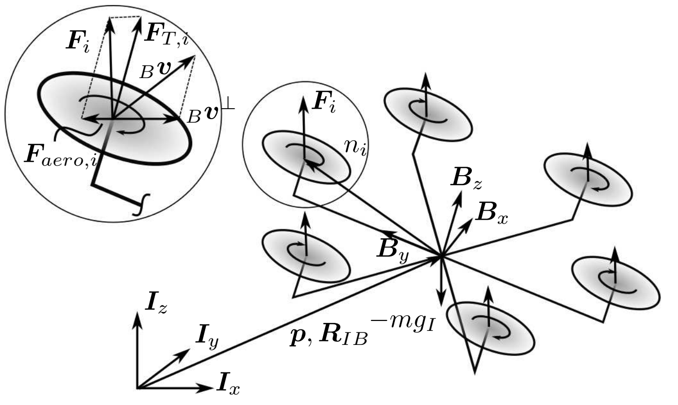

Fig. 2. A schematic of MAV showing Forces and torques acting on the MAV and aerodynamic forces acting on a single rotor. Inertial and CoG frames are also shown.

3. MAV MODEL

In this section we present the MAV model employed in the controllers formulation. We first introduce the full vehicle model and explain the forces and moment acting on the system. Next, we will briefly discuss the closed-loop attitude model employed in the trajectory tracking controller.

System model We define the world fixed inertial frame I and the body fixed frame B attached to the MAV in the Center of Gravity (CoG) as shown in Figure 2. The vehicle configuration is described by the position of the CoG in the inertial frame $\boldsymbol{p} \in \mathbb{R}^3$, the vehicle velocity in inertial frame $\boldsymbol{v}$, the vehicle orientation $\boldsymbol{R}_{IB} \in SO(3)$ and the body angular rate $\boldsymbol{\omega}$.

The main forces acting on the vehicle are generated from the propellers. Each propeller generates thrust proportional to the square of the propeller rotation speed and angular moment due to the drag force. The generated thrust $\boldsymbol{F}_{T,i}$ and moment $\boldsymbol{M}_i$ from the $i-th$ propeller is given by:

$$\boldsymbol{F}_{T,i} = k_n n_i^2 \boldsymbol{e}_z, \tag{1a}$$

$$\boldsymbol{M}_i = (-1)^{i-1} k_m \boldsymbol{F}_{T,i}, \tag{1b}$$

where n_i is the $i-th$ rotor speed, k_n and k_m are positive constants and $\boldsymbol{e}_z$ is a unit vector in z direction.

Moreover, we consider two important effects that appear in case of dynamic maneuvers. These effects are the blade flapping and induced drag. The importance of these effects stems from the fact that they introduce additional force in the $x-y$ rotor plane adding more damping to the MAV as shown in Mahony et al. (2012). It is possible to combine these effects as shown in Omari et al. (2013) into one lumped drag coefficient k_D.

This leads to the aerodynamic force $\boldsymbol{F}_{aero,i}$:

$$\boldsymbol{F}_{aero,i} = f_{T,i} \boldsymbol{K}_{drag} \boldsymbol{R}_{IB}^T \boldsymbol{v} \tag{2}$$

where $\boldsymbol{K}_{drag} = diag(k_D, k_D, 0)$ and $f_{T,i}$ is the z component of the $i-th$ thrust force.

The motion of the vehicle can be described by the following equations:

$$\dot{\boldsymbol{p}} = \boldsymbol{v}, \tag{3a}$$

$$\dot{\boldsymbol{v}} = \frac{1}{m}\left(\boldsymbol{R}_{IB}\sum_{i=0}^{N_r}\boldsymbol{F}_{T,i} - \boldsymbol{R}_{IB}\sum_{i=0}^{N_r}\boldsymbol{F}_{aero,i} + \boldsymbol{F}_{ext}\right) + \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix}, \tag{3b}$$

$$\dot{\boldsymbol{R}}_{IB} = \boldsymbol{R}_{IB}\lfloor\boldsymbol{\omega}\times\rfloor, \tag{3c}$$

$$\boldsymbol{J}\dot{\boldsymbol{\omega}} = -\boldsymbol{\omega}\times\boldsymbol{J}\boldsymbol{\omega} + \boldsymbol{\mathcal{A}}\begin{bmatrix} n_1^2 \\ \vdots \\ n_{N_r}^2 \end{bmatrix}, \tag{3d}$$

where m is the mass of the vehicle and $\boldsymbol{F}_{ext}$ is the external forces acting on the vehicle (i.e wind). $\boldsymbol{J}$ is the inertia matrix, $\mathcal{A}$ is the allocation matrix and N_r is the number of propellers.

Attitude model We follow a cascaded approach as described in Blösch et al. (2010) and assume that the vehicle attitude is controlled by an attitude controller. For completeness we quickly summarize the findings in the following.

To achieve accurate trajectory tracking, it is crucial for the high level controller to consider the inner loop system dynamics. Therefore, it is necessary to consider a simple model of the attitude closed-loop response. These dynamics can either be calculated by simplifying the closed loop dynamic equations (if the controller is known) or by a simple system identification procedure in case of an unknown attitude controller (on commercial platforms for instance). In this work we used the system identification approach to identify a first order closed-loop attitude response.

The inner-loop attitude dynamics are then expressed as follows:

$$\dot{\phi} = \frac{1}{\tau_{\phi}}\left(k_{\phi}\phi_{cmd} - \phi\right), \tag{4a}$$

$$\dot{\theta} = \frac{1}{\tau_{\theta}}\left(k_{\theta}\theta_{cmd} - \theta\right), \tag{4b}$$

$$\dot{\psi} = \dot{\psi}_{cmd} \tag{4c}$$

where k_{ϕ}, k_{θ} and $\tau_{\phi}, \tau_{\theta}$ are the gains and time constant of roll and pitch angles respectively. ϕ_{cmd} and θ_{cmd} are the commanded roll and pitch angles and $\dot{\psi}_{cmd}$ is commanded angular velocity of the vehicle heading.

The aforementioned model will be employed in the subsequent trajectory tracking controllers to account for the inner-loop dynamics. Note that the vehicle heading angular rate $\dot{\psi}$ is assumed to track the command instantaneously. This assumption is reasonable because the MAV heading angle has no effect on the MAV position.

4. LINEAR MPC

In this section, we describe a Model Predictive Controller to achieve trajectory tracking using a simple model of the MAV Borrelli et al. (2015). In Subsection 4.1, the vehicle model is linearized around hovering condition. Finally, in Subsection 4.2, we formulate the LMPC controller and discuss non-linearity compensation.

4.1 Model Linearization

The vehicle model can be linearize around hovering condition assuming small attitude angles and vehicle heading aligned with the first axis of inertial frame ($\psi = 0$). We define the following state vector:

$$\boldsymbol{x} = \left(\boldsymbol{p}^T \ \boldsymbol{v}^T \ {}_I\phi \ {}_I\theta\right)^T, \tag{5}$$

and control input vector:

$$\boldsymbol{u} = \left({}_I\phi_{cmd} \ {}_I\theta_{cmd} \ T_{cmd}\right)^T \tag{6}$$

where T_{cmd} is the commanded thrust, which we assume can be achieve instantaneously as the motors dynamics are typically very fast. ${}_I\phi, {}_I\theta$ are the roll and pitch angles which we denote in inertial frame to get rid of the vehicle heading ψ from the model. The transformation between attitude angles and heading free attitude angles is given by:

$$\begin{pmatrix}\phi \\ \theta\end{pmatrix} = \begin{pmatrix}\cos\psi & \sin\psi \\ -\sin\psi & \cos\psi\end{pmatrix}\begin{pmatrix}{}_I\phi \\ {}_I\theta\end{pmatrix}. \tag{7}$$

After linearization and discretization of the model described in Section 3, the following linear state-space model is obtained:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{B}_d\boldsymbol{F}_{ext,k}. \tag{8}$$

4.2 Controller Formulation

The LMPC scheme repeatedly solves the following Optimal Control Problem (OCP):

$$\begin{aligned}\min_{\boldsymbol{U},\boldsymbol{X}} \ & \sum_{k=0}^{N-1}\left(\left\|(\boldsymbol{x}_k - \boldsymbol{x}_{ref,k}\right\|_{\boldsymbol{Q}_x}^2 + \left\|\boldsymbol{u}_k - \boldsymbol{u}_{ref,k}\right\|_{\boldsymbol{R}_u}^2\right) \\ & + \left\|\boldsymbol{x}_N - \boldsymbol{x}_{ref,N}\right\|_{\boldsymbol{P}}^2 \\ \text{subject to} \ & \boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{B}_d\boldsymbol{F}_{ext,k}; \\ & \boldsymbol{F}_{ext,k+1} = \boldsymbol{F}_{ext,k}, \quad k = 0,\ldots,N-1 \\ & \boldsymbol{u}_k \in \mathbb{U} \\ & \boldsymbol{x}_0 = \boldsymbol{x}(t_0), \quad \boldsymbol{F}_{ext,0} = \boldsymbol{F}_{ext}(t_0).\end{aligned} \tag{9}$$

where $\boldsymbol{Q}_x \succeq 0$ is the penalty on the state error, $\boldsymbol{R}_u \succ 0$ is the penalty on control input error and $\boldsymbol{P} \succeq 0$ is the terminal state error penalty. $\boldsymbol{x}_{ref,k}$ and $\boldsymbol{u}_{ref,k}$ are respectively the target state vector and target control input at time k. $\mathbb{U}$ is the control input constraint given by:

$$\mathbb{U} = \left\{\boldsymbol{u}\in\mathbb{R}^3 | \begin{bmatrix}\phi_{min} \\ \theta_{min} \\ T_{cmd,min}\end{bmatrix} \leq \boldsymbol{u} \leq \begin{bmatrix}\phi_{max} \\ \theta_{max} \\ T_{cmd,max}\end{bmatrix}\right\} \tag{10}$$

Note that we assume constant disturbances along the prediction horizon. Only the first control input $\boldsymbol{u}_0$ is applied to the system, and the process is repeated the next time step in a receding horizon fashion.

The attitude commands ${}_I\phi_{cmd}, {}_I\theta_{cmd}$ is transformed into the MAV body frame by applying (7). Moreover, the non-zero MAV attitude effects on the generated lift can be compensated. The actual thrust control input to the MAV low-level attitude controller is given by:

$$\tilde{T}_{cmd} = \frac{T_{cmd} + g}{\cos\phi\cos\theta} \tag{11}$$

To achieve a better tracking performance of dynamic trajectories, we can include a feed-forward term by setting the reference control input $\boldsymbol{u}_{ref}$ as follows:

$$\boldsymbol{u}_{ref,k} = \left(-{}_B\ddot{y}_{ref,k}/g \ \ {}_B\ddot{x}_{ref,k}/g \ \ {}_B\ddot{z}_{ref} \right)^T \tag{12}$$

where ${}_B\ddot{x}_{ref,k}$, ${}_B\ddot{y}_{ref,k}$, ${}_B\ddot{z}_{ref,k}$ are the desired trajectory acceleration expressed in MAV body frame B.

5. NONLINEAR MPC

In this section, a continuous time NMPC controller that considers the full system dynamics explained in Section 3 is formulated. In Subsection 5.1 we formulate the OCP and in Subsection 5.2 we discuss the technique employed to solve the OCP to achieve real-time implementation.

5.1 Controller Formulation

To formulate the NMPC, we first define the following state vector:

$$\boldsymbol{x} = \left(\boldsymbol{p}^T \ \boldsymbol{v}^T \ \phi \ \theta \ \psi \right)^T, \tag{13}$$

and control input vector:

$$\boldsymbol{u} = \left(\phi_{cmd} \ \theta_{cmd} \ T_{cmd} \right)^T \tag{14}$$

Now, we can define the OCP as follows:

$$\begin{aligned} \min_{\boldsymbol{U},\boldsymbol{X}} \int_{t=0}^{T} & \left\| \boldsymbol{x}(t) - \boldsymbol{x}_{ref}(t) \right\|^2_{\boldsymbol{Q}_x} + \left\| \boldsymbol{u}(t) - \boldsymbol{u}_{ref}(t) \right\|^2_{\boldsymbol{R}_u} dt \\ & + \left\| \boldsymbol{x}(T) - \boldsymbol{x}_{ref}(T) \right) \|^2_{\boldsymbol{P}} \\ \text{subject to} \quad & \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}); \\ & \boldsymbol{u}(t) \in \mathbb{U} \\ & \boldsymbol{x}(0) = \boldsymbol{x}(t_0). \end{aligned} \tag{15}$$

where $\boldsymbol{f}$ is composed by Equations (3a), (3b) and (4). The controller is implemented in a receding horizon fashion, where the aforementioned optimization problem needs to be solved every time step in and only the first control input is actually applied to the system. Solving (15) repeatedly in real-time is not a trivial task. *Direct methods* techniques has gained a particular attention recently to address optimal control problems. In the next subsection we describe the method employed by the solver to solve (15).

5.2 OCP Solution

Multiple shooting technique is employed to solve (15). The system dynamics and constraints are discretized over a coarse discrete time grid $t_0, \dots, t_N$ within the time interval $[t_k, t_{k+1}]$. For each interval, a Boundary Value Problem (BVP) is solved, where additional continuity constrains are imposed. An implicit *RK* integrator of order 4 is employed to forward simulate the system dynamics along the interval. At this point, the OCP can be expressed as a Nonlinear Program (NLP) that can be solved using Sequential Quadratic Programming (SQP) technique where an active set or interior point method can be used to solve the Quadratic Program (QP).

6. EXTERNAL DISTURBANCES ESTIMATION

In this section, we discuss the external disturbances estimator employed to achieve offset-free trajectory tracking. The external disturbances $\boldsymbol{F}_{ext}$ is estimated by an augmented state Extended Kalman Filter (EKF) that includes external forces. The EKF is employing the same model used in the controller design, but it includes also vehicle heading angle ψ. In this way, the external forces will capture also modeling error achieving zero steady state tracking error Borrelli et al. (2015). The filter is employing the inner-loop attitude dynamics presented in Equation (4).

Table 1. RMSE of individual axes of LMPC and NMPC during hovering experiment with and without external disturbances. Error is in cm.

	LMPC	NMPC
without wind	$\left(0.9 \ 1.4 \ 0.7\right)$	$\left(1.0 \ 1.7 \ 0.5\right)$
with wind	$\left(1.6 \ 1.9 \ 1.1\right)$	$\left(1.2 \ 2.0 \ 0.9\right)$

7. IMPLEMENTATION AND RESULTS

In this section, we present the results of multiple experiments to evaluate the performance of the previously presented LMPC and NMPC.

7.1 System Description

The aforementioned LMPC and NMPC have been implemented and evaluated on Asctec NEO hexacopter equipped with an *Intel 3.1 GHz i7 Core* processor, 8 GB RAM. The MAV parameters are shown in Table 2. The on-board computer is running Robot Operating System (ROS). The vehicle state is estimated using an EKF as described in Lynen et al. (2013) by fusing vehicle Inertial Measurement Unit (IMU) with external motion capture system (Vicon).

7.2 Controllers Implementation

To implement the LMPC, an efficient QP solver using CVXGEN code generator framework by Mattingley and Boyd (2012) is employed to solve the optimization problem (9) every time step. While the NMPC is implemented by solving (15) every time step using an efficient solver generated using *ACADO* toolkit by Houska et al. (2011). A real time iteration scheme based on Gauss-Newton is employed to approximate the optimization problem and iteratively improve the solution during the runtime of the process. In this way, an improvement of the computation time is achieved. Both controllers are running at 100 Hz while internally the prediction is performed at $10Hz$, in this way we achieve longer prediction horizon with less computational efforts. The prediction horizon is chosen to be 20 steps resulting into 2 seconds prediction horizon. The penalties for both controllers are chosen such that the cost functions are identical, in this way we emphasis the benefits of full dynamics model over a linearized model. The terminal cost in both controllers is chosen to account for the infinite horizon cost.

7.3 Hovering Performance

The purpose of these experiments is to evaluate the hovering performance of the MAV in nominal conditions and under external wind disturbances. Figure 3 shows the x, y error under no external disturbances for LMPC and NMPC. Clearly the performance of both controllers is very comparable, and this result is expected since the LMPC is employing a model linearized around hovering condition. The RMSE is found to be 1.84 cm and 2.05 cm in the case of LMPC and NMPC respectively.

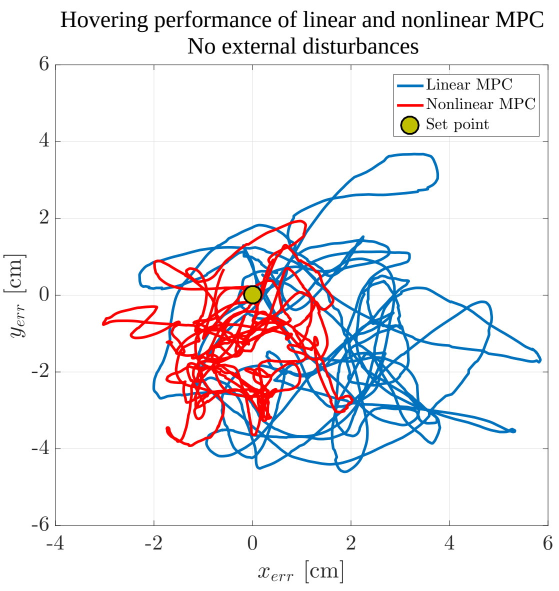

Fig. 3. $x - y$ error during hovering using LMPC and NMPC. No external wind is applied.

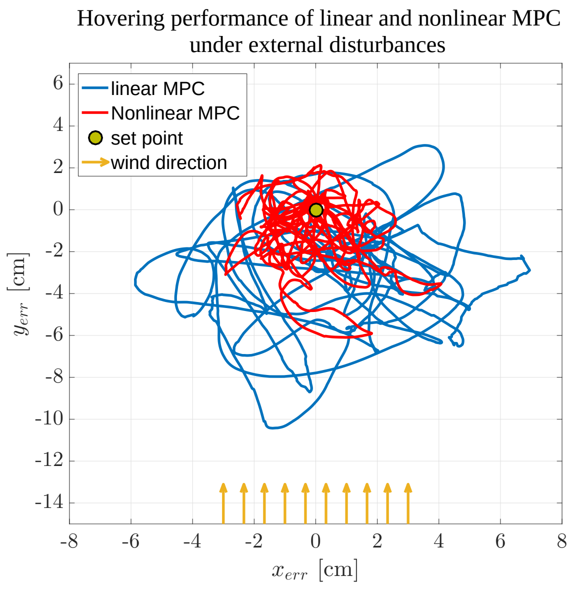

Fig. 4. $x-y$ error during hovering using LMPC and NMPC. An external wind of around 11 m/s is applied in the direction indicated by yellow arrows.

Under external wind of around 11 m/s, the $x - y$ error is shown in Figure 4. Both controllers achieve an RMSE of 2.7 cm and 2.5 cm in the case of LMPC and NMPC respectively. The RMSE is shown in Table 1.

7.4 Step Response

Even though the tracking error is weighted in the same way in both controllers, the NMPC outperforms the LMPC in the step response as can be seen in Figure 5. The main reason for the is the exploitation of the full system dynamics, in particular, the thrust command coupling with lateral motion. This leads to faster response with no noticeable overshoot. Figure 6 shows the thrust command during step response. The NMPC exploits the full thrust command between $T_{cmd,min}$ and $T_{cmd,max}$ while the LMPC is employing the thrust command in a very limited way, which is introduced by the nonlinear compensation shown in Equation (11).

To perform qualitative comparison, we compare the rise time and overshoot percentage. The rise time is around 1.6 s and 1.0 s for the LMPC and NMPC respectively. While the overshoot percentage is found to be 1.98 % for both controllers. This clearly highlights that the NMPC outperforms the LMPC in this case.

7.5 Aggressive Trajectory Tracking

In this experiment, we compare both controllers capabilities to track an aggressive polynomial trajectory under external disturbances. The trajectory tracking error of both controllers is shown in Figure 7, while the actual trajectory tracking is shown in Figure 8. The RMSE is found to be 10.8 cm and 7.1 cm in the case of LMPC and NMPC respectively.

To compare the computation effort required by each controller, in Figure 9 we show the time employed by the solver to find the control action. Clearly, the real time iteration scheme imple-

Table 2. NEO hexacopter parameters and control input constraints.

Parameter	Value
mass	3.42 kg
τ_ϕ	0.1901 sec
τ_θ	0.1721 sec
k_ϕ	0.91
k_θ	0.96
ϕ_{max}, θ_{max}	45 °
ϕ_{min}, θ_{min}	−45 °
$T_{cmd,max}$	40.3 N
$T_{cmd,min}$	13.5 N

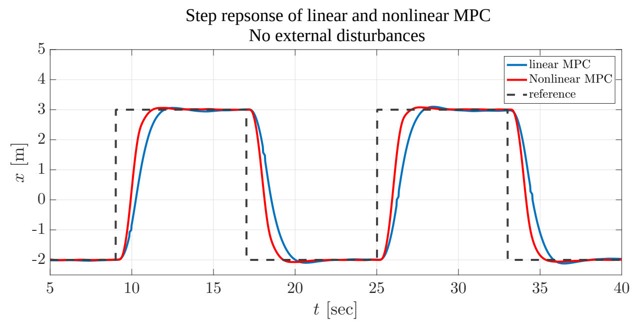

Fig. 5. Step response in x direction.

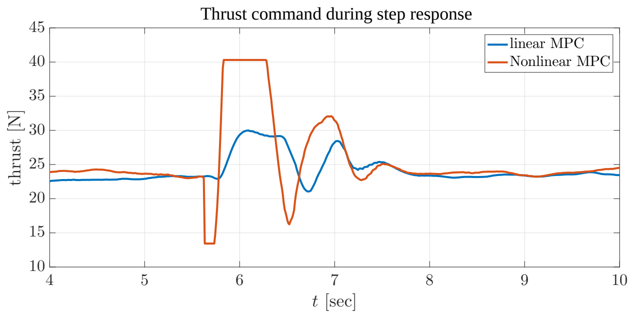

Fig. 6. Thrust command during step response.

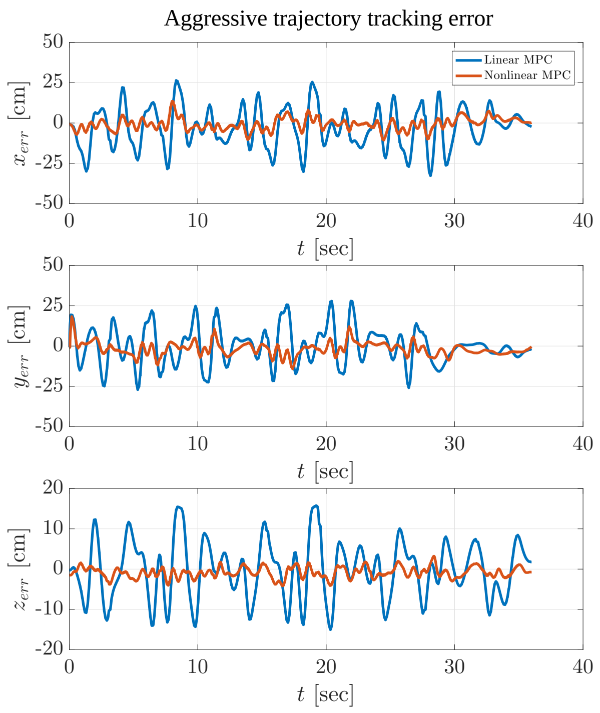

Fig. 7. Aggressive trajectory error plots under external wind disturbances. Wind speed is measured to be around 11 m/s.

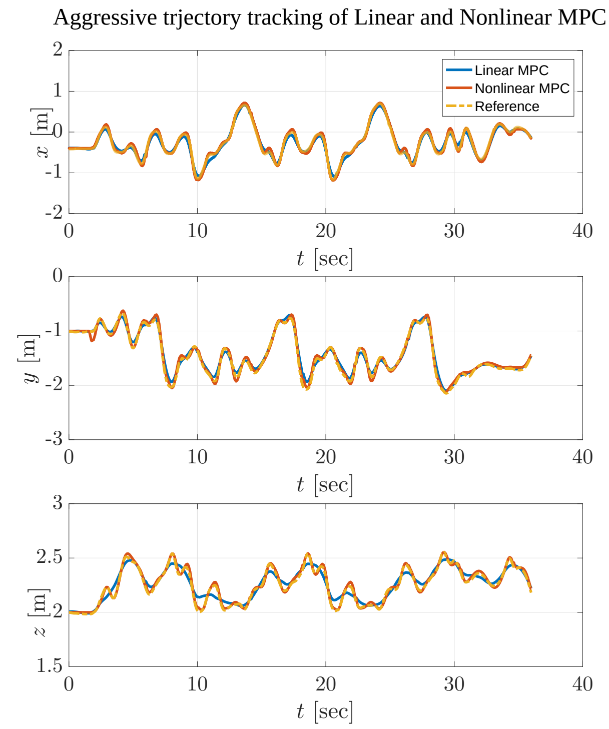

Fig. 8. Aggressive trajectory tracking plots under external wind disturbances.

mented in the NMPC greatly improves the solver speed, achieving an average solve time of 0.45 ms for the NMPC compared to 2.35 ms for the LMPC case, improving the computation effort by a factor of 5.

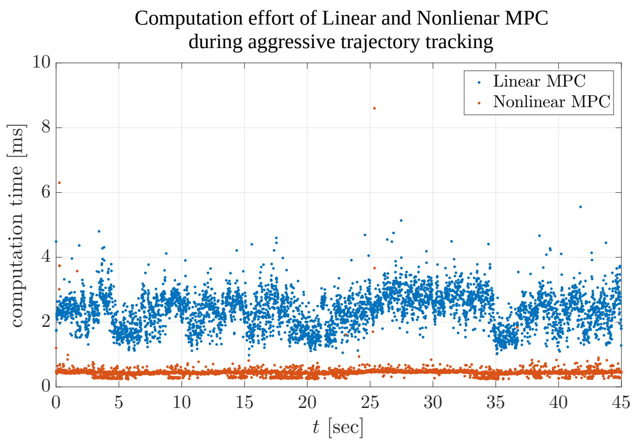

Fig. 9. Computation time comparison between LMPC and NMPC during aggressive trajectory tracking.

8. CONCLUSION

In this paper we presented a Linear and Nonlinear Model Predictive Controllers for trajectory tracking of MAV. Detailed comparison has been performed during hovering, step response and aggressive trajectory tracking under external disturbances. Both controllers showed comparable behavior while the NMPC showed a slightly better disturbance rejection capability, step response, tracking performance and computational effort. An open source implementation of these controllers can be found on Kamel (2016).

REFERENCES

Alexis, K., Papachristos, C., Siegwart, R., and Tzes, A. (2016). Robust model predictive flight control of unmanned rotorcrafts. *Journal of Intelligent & Robotic Systems*, 81(3), 443–469.

Bemporad, A., Pascucci, C., and Rocchi, C. (2009). Hierarchical and hybrid model predictive control of quadcopter air vehicles. {*IFAC*} *Proceedings Volumes*, 42(17), 14 – 19. doi: http://dx.doi.org/10.3182/20090916-3-ES-3003.00004. 3rd {IFAC} Conference on Analysis and Design of Hybrid Systems.

Berkenkamp, F. and Schoellig, A.P. (2015). Safe and robust learning control with Gaussian processes. In *Proc. of the European Control Conference (ECC)*, 2501–2506.

Bircher, A., Kamel, M., Alexis, K., Oleynikova, H., and Siegwart, R. (2016a). Receding horizon "next-best-view" planner for 3d exploration. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 1462–1468. doi:10.1109/ICRA.2016.7487281.

Bircher, A., Kamel, M., Alexis, K., Oleynikova, H., and Siegwart, R. (2016b). Receding horizon path planning for 3d exploration and surface inspection. *Autonomous Robots*, 1–16.

Blösch, M., Weiss, S., Scaramuzza, D., and Siegwart, R. (2010). Vision based mav navigation in unknown and unstructured environments. In *Robotics and automation (ICRA), 2010 IEEE international conference on*, 21–28. IEEE.

Borrelli, F., Bemporad, A., and Morari, M. (2015). *Predictive Control for Linear and Hybrid Systems*.

Bouabdallah, S., Noth, A., and Siegwart, R. (2004). Pid vs lq control techniques applied to an indoor micro quadrotor. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceed-*

ings. 2004 IEEE/RSJ International Conference on, volume 3, 2451–2456. IEEE.

Burri, M., Nikolic, J., Hürzeler, C., Caprari, G., and Siegwart, R. (2012). Aerial service robots for visual inspection of thermal power plant boiler systems. In *Applied Robotics for the Power Industry (CARPI), 2012 2nd International Conference on*, 70–75. IEEE.

Houska, B., Ferreau, H., and Diehl, M. (2011). ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization. *Optimal Control Applications and Methods*, 32(3), 298–312.

Kamel, M. (2016). Open source mav mpc controllers. URL `https://github.com/ethz-asl/mav_control_rw`.

Kamel, M., Alexis, K., Achtelik, M., and Siegwart, R. (2015). Fast nonlinear model predictive control for multicopter attitude tracking on so (3). In *Control Applications (CCA), 2015 IEEE Conference on*, 1160–1166. IEEE.

Lee, T., Leoky, M., and McClamroch, N.H. (2010). Geometric tracking control of a quadrotor uav on se (3). In *49th IEEE conference on decision and control (CDC)*, 5420–5425. IEEE.

Lynen, S., Achtelik, M.W., Weiss, S., Chli, M., and Siegwart, R. (2013). A robust and modular multi-sensor fusion approach applied to mav navigation. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, 3923–3929. IEEE.

Mahony, R., Kumar, V., and Corke, P. (2012). Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robotics Automation Magazine*, 19(3), 20–32. doi: 10.1109/MRA.2012.2206474.

Mattingley, J. and Boyd, S. (2012). Cvxgen: A code generator for embedded convex optimization. *Optimization and Engineering*, 13(1), 1–27.

Meier, L. (2016). Pixhawk autopilot. URL `https://pixhawk.org`. Accessed: 2016-11-08.

Mellinger, D., Michael, N., and Kumar, V. (2012). Trajectory generation and control for precise aggressive maneuvers with quadrotors. *The International Journal of Robotics Research*, 0278364911434236.

Oettershagen, P., Stastny, T., Mantel, T., Melzer, A., Rudin, K., Gohl, P., Agamennoni, G., Alexis, K., and Siegwart, R. (2016). Long-endurance sensing and mapping using a hand-launchable solar-powered uav. In *Field and Service Robotics*, 441–454. Springer.

Omari, S., Hua, M.D., Ducard, G., and Hamel, T. (2013). Nonlinear control of vtol uavs incorporating flapping dynamics. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2419–2425. doi:10.1109/IROS.2013.6696696.

Steich, K., Kamel, M., Beardsleys, P., Obrist, M.K., Siegwart, R., and Lachat, T. (2016). Tree cavity inspection using aerial robots. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.