Model predictive control design for reference tracking of a quadcopter

Mike de Pont
MSc Student S&C
4323955

Daniël van Hanswijk
MSc Student S&C
4351479

ABSTRACT

In this paper a model predictive controller has been designed which is applied on the dynamics of a quadrotor. The purpose of this work is to show the benefits of Model Predictive Control and to examine whether it is sufficient for trajectory tracking in $\mathcal{R}^3$.

The dynamics of the linearized and discretized drone are implemented in MATLAB and with the use of *cvx* several optimization techniques are being shown.

Various simulation scenarios are performed to evaluate the performance of the controller and to compare this performance with standard PID control. Simulation results show that this type of control is efficacious for reference tracking and is an improvement on maintaining a hovering altitude when compared to PID control.

Keywords

Autonomous flightpath tracking, Model Predictive Control, Quadcoper, MATLAB, CVX, Quadratic programming, PID Control

1. INTRODUCTION

A quadcopter is a multirotor helicopter that is lifted and propelled by four rotors. Interest as well as use have been increased because of their light weight, simple dynamics and ease of design [Sabatino 2015]. A good performing control design is essential to ensure a smooth flight. A current much often used control approach is the PID control. Where the PID control most often just regulates the hovering altitude of the quadrotor, a complex model predictive controller can be more widely used. This controller has the capability of working with constraints, predictive behavior and with multi inputs, multi outputs [Islam 2017].

2. SYSTEM DYNAMICS

In this section the dynamics of the quadcopter will be explained. A schematic view of the system is shown in figure 1. As the name suggests the quadcopter consists of four rotors which rotate all individually. Two of those rotate clockwise, while the other two rotate counter-clockwise. By changing the angular velocities of the individual rotors movements in 6 degrees can be realized. Translation in x, y and z direction, and rotation around its x, y and z axis, called roll, pitch and yaw respectively. The forces working on the quadcopter are the forces generated by the rotors, and the gravitation. The force generated by the rotors are used as control input to direct the movement of the drone.

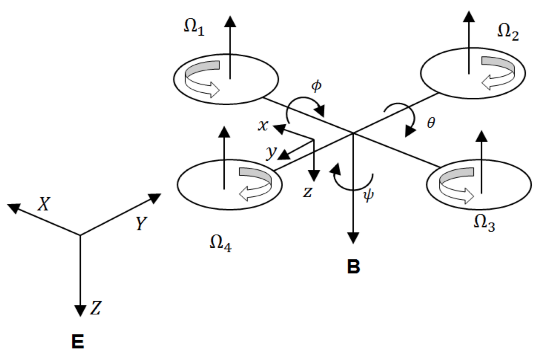


Figure 1: Dynamics of a Quadcopter [Islam 2017]

2.1 Nonlinear model

Following Euler's method the equations of motion in the body fixed frame $[xyz]$ are given by:

$$\sum F_x = 0 = m\ddot{x} \tag{2.1}$$
$$\sum F_y = 0 = m\ddot{y} \tag{2.2}$$
$$\sum F_z = -(F_1 + F_2 + F_3 + F_4) = m\ddot{z} \tag{2.3}$$

$$\sum M_x = (F_2 - F_4)l = I\dot{p} \tag{2.4}$$
$$\sum M_y = (F_1 - F_3)l = I\dot{q} \tag{2.5}$$
$$\sum M_z = K_m\left((\omega_1^2 + \omega_3^2) - (\omega_2^2 + \omega_4^2)\right) = I\dot{r} \tag{2.6}$$

Where $[\dot{p}, \dot{q}, \dot{r}]$ represents the angular accelerations in the body fixed frame. Note that the gravity force will be included in the earth fixed frame.

The equations of motion in the earth fixed frame are related with those of the body fixed frame by means of two transformation matrices.

The relation is described by equation 2.7 and 2.8.

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = \mathbf{T}_{\phi\theta\psi}\begin{bmatrix}p\\ q\\ r\end{bmatrix} \tag{2.7}$$

$$\sum F_{XYZ} = \mathbf{R}_{ZYX}\begin{bmatrix}F_x\\ F_y\\ F_z\end{bmatrix} + \begin{bmatrix}0\\ 0\\ F_g\end{bmatrix} \tag{2.8}$$

With $\mathbf{T}_{\phi}\theta\psi$ and $\mathbf{R}_{ZYX}$ as defined in equation 2.9 and 2.10.

$$\mathbf{T}_{\phi\theta\psi} = \begin{bmatrix}1 & \sin\phi\tan\theta & \cos\phi\tan\theta\\ 0 & \cos\phi & -\sin\phi\\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta\end{bmatrix} \tag{2.9}$$

$$\mathbf{R}_{ZYX} = \begin{bmatrix}c\psi c\theta & c\psi s\phi s\theta - c\phi s\psi & s\phi s\psi + c\phi c\psi s\theta\\ c\theta s\psi & c\phi c\psi + s\phi s\psi s\theta & c\phi s\psi s\theta - c\psi s\phi\\ -s\theta & c\theta s\phi & c\phi c\theta\end{bmatrix} \tag{2.10}$$

where

$$s(\cdot) = \sin(\cdot) \tag{2.11}$$

$$c(\cdot) = \cos(\cdot) \tag{2.12}$$

By combining equation 2.1 - 2.6, 2.7 and 2.8 the following nonlinear continuous time equation describes the dynamics of the drone in both frameworks.

$$\dot{X}_N = \begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\\ \dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\\ \ddot{x}\\ \ddot{y}\\ \ddot{z}\\ \dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\\ \dot{p} + \dot{r}\cos\phi\tan\theta + \dot{q}\sin\phi\tan\theta\\ \dot{q}\cos\phi - \dot{r}\sin\phi\\ \dot{r}\frac{\cos\phi}{\tan\theta} + \dot{q}\frac{\sin\phi}{\cos\theta}\\ \frac{(\sin(\phi)\sin(\psi)+\cos(\phi)\cos(\psi)\sin(\theta))*F_r}{m}\\ \frac{(-\sin(\phi)\cos(\psi)+\cos(\phi)\sin(\psi)\sin(\theta))*F_r}{m}\\ \frac{(\cos(\phi)\cos(\theta))*F_r - mg}{m}\\ \frac{(F_4-F_2)*l}{I}\\ \frac{(F_3-F_1)*l}{I}\\ \frac{K_m\left((\omega_1^2+\omega_3^2)-(\omega_2^2+\omega_4^2)\right)}{I}\end{bmatrix} \tag{2.13}$$

2.2 Linearization and discretization of system dynamics

Applying model predictive control to the full non-linear model is computational challenging. Therefore we simplify the model by means of linearization and discretization. The corresponding A and B matrices of the linear state space model are being found by taking the Jacobian of the nonlinear function describing the dynamics of the drone and linearizing around its hovering state (equation 2.14 and 2.15), that is a position in R3 with all angular velocities equal to zero and F_r compensating the gravitational force.

$$\begin{aligned}U_1 &\cong mg\\ \dot{p} \cong \dot{q} \cong \dot{r} &\cong 0\\ \sin\psi &\cong 0\\ \sin\theta &\cong \theta\\ \sin\phi &\cong \phi\end{aligned}$$

By forcing ψ to be zero we force the drone to track the trajectory by using θ and ϕ.

$$A = \left\{\frac{\partial f(X_N, U_N)}{\partial X_N}|X_N = X_h, U_N = U_h\right\} \tag{2.14}$$

$$B = \left\{\frac{\partial f\,(X_N, U_N)}{\partial U_N}|X_N = X_h, U_N = U_h\right\} \tag{2.15}$$

By taking the Jacobian and by separating the B matrix such that the input of the system is defined as the force of each rotor, we find the following state space description.

$$A = \begin{bmatrix}0_{3\times3} & 0_{3\times1} & 0_{3\times1} & 0_{3\times1} & I_{3\times3} & 0_{3\times3}\\ 0_{3\times3} & 0_{3\times1} & 0_{3\times1} & 0_{3\times1} & 0_{3\times3} & I_{3\times3}\\ 0_{1\times3} & 0 & g & 0 & 0_{1\times3} & 0_{1\times3}\\ 0_{1\times3} & -g & 0 & 0 & 0_{1\times3} & 0_{1\times3}\\ 0_{4\times3} & 0_{3\times1} & 0_{3\times1} & 0_{4\times1} & 0_{4\times3} & 0_{4\times3}\end{bmatrix} \tag{2.16}$$

$$B = \begin{bmatrix}0_{8\times1} & 0_{8\times1} & 0_{8\times1} & 0_{8\times1}\\ -\frac{1}{m} & -\frac{1}{m} & -\frac{1}{m} & -\frac{1}{m}\\ 0 & -\frac{l}{I} & 0 & \frac{l}{I}\\ \frac{l}{I} & 0 & -\frac{l}{I} & 0\\ -\frac{K_m}{K_f}\frac{l}{I} & \frac{K_m}{K_f}\frac{l}{I} & -\frac{K_m}{K_f}\frac{l}{I} & \frac{K_m}{K_f}\frac{l}{I}\end{bmatrix} \tag{2.17}$$

$$C = \begin{bmatrix}1 & 0 & 0 & 0 & 0 & 0 & 0_{1\times6}\\ 0 & 1 & 0 & 0 & 0 & 0 & 0_{1\times6}\\ 0 & 0 & 1 & 0 & 0 & 0 & 0_{1\times6}\\ 0 & 0 & 0 & 0 & 0 & 1 & 0_{1\times6}\end{bmatrix} \tag{2.18}$$

$$U = \begin{bmatrix}F_1 & F_2 & F_3 & F_4\end{bmatrix}^T \tag{2.19}$$

The system is then discretized using the MATLAB command *c2d*, such that the discrete state space system defined by

$$\delta x(k+1) = \tilde{A}\delta x(k) + \tilde{B}\delta u(k) \tag{2.20}$$

$$\delta y(k) = \tilde{C}\delta x(k) + w(k) \tag{2.21}$$

Where each $\delta x(k)$, $\delta u(k)$ and $\delta y(k)$ represent the value respective to its hovering position and $w(k)$ represents unknown measurement noise.

3. STATE OF THE ART CONTROL DESIGN

There are numerous control designs to control a quadrotor, however the most used control design of the moment is the PID control. These controllers are widely used for their ease and robustness

3.1 PID control

As the name suggests, this control mechanism consists of three parts: a proportional, an integral and a derivative term. PID control is the most common control algorithm used in industrial processes or, applied in quadcopters, in controlling the hovering altitude. This can be attributed to its functional simplicity and to its robust performance in a wide range of operating conditions. The output $u(t)$ of the PID can be formulated as in (3.1) [Ostojic et al. 2015]. Where K_p, K_i and K_d are the proportional, integral and derivative gains, respectively.

$$u(t) = K_p e(t) + K_i\int_0^t e(\tau)d\tau + K_d\frac{de(t)}{dt} \tag{3.1}$$

When a PID controller is used in a closed-loop feedback system these three coefficients have a direct influence on the error $e(t)$, between the reference, $r(t)$, and the output of the system, $y(t)$.

4. MODEL PREDICTIVE CONTROL DESIGN

Model predictive control is a powerful method in optimizing control sequences. The basic concept is that it uses a dynamical model to forecast system behavior [J.B. Rawlings 2009]. By optimizing this it is able to produce the best decision, that is the optimized control sequence.

4.1 Cost Function

The objective of the linear model predictive controller is to drive the state towards a specific reference trajectory. Therefore the stage cost is to be defined as:

$$l(x,u) = \sum_{k=1}^{N-1} \left\{ \Delta y(k)^T Q \Delta y(k) + \delta u(k)^T R \delta u(k) \right\} \tag{4.1}$$

With $\Delta y(k) = \delta y_r(k) - \delta y(k)$. For ease of notation we define

$$\Delta Y_{k+i} = \begin{bmatrix} \Delta y(k) \\ \Delta y(k+1) \\ \Delta y(k+2) \\ \vdots \\ \Delta y(k+N-1) \end{bmatrix}, \Delta U_{k+i} = \begin{bmatrix} \delta u(k) \\ \delta u(k+1) \\ \delta u(k+2) \\ \vdots \\ \delta u(k+N-1) \end{bmatrix} \tag{4.2}$$

In order to avoid having a terminal constraint $X(N) \in \mathcal{X}_f$ and to achieve infinite control that guarantees closed loop stability we implement the terminal cost function as given in equation 4.3.

$$V_f = \Delta y(N)^T Q_f \Delta y(N) \tag{4.3}$$

Where Q_f is the solution of the algebraic ricatti equation 4.4.

$$Q_f = A^T Q_f A - A^T Q_f B (B^T Q_f B + R)^{-1} B^T Q_f A + Q \tag{4.4}$$

The total value function is then defined as:

$$V^* = l(x,u) + V_f \tag{4.5}$$

4.2 Quadratic programming

In order to find the optimal control input sequence we simplify the objective function 4.5 into the following format.

$$\min_u u^T H u + h^T u + constant \tag{4.6}$$

such that

$$H = \bar{R} + S_o^T \bar{Q} S_o \tag{4.7}$$

$$h = x_k^T T_o \bar{Q} S_o - Y_r^T \bar{Q} S_o \tag{4.8}$$

Since the constant in 4.6 does not affect the minimization process, it has been left out. The matrices T_o and S_o are chosen such that both meet equation 4.9.

$$\Delta Y_{k+i} = T_o x_k + S_o \Delta U_{K+i} \tag{4.9}$$

The vector Y_r is defined as the reference trajectory at time stamp k. With the receding horizon implemented this comes down to equation 4.10.
The exact interpretation of $\bar{Q}$, $\bar{R}$, T_o and S_o is given in equation 4.11.

$$Y_r = \begin{bmatrix} X_r(k) \\ Y_r(k) \\ Z_r(k) \\ \psi_r(k) \\ X_r(k+1) \\ Y_r(k+1) \\ Z_r(k+1) \\ \psi_r(k+1) \\ \vdots \\ X_r(k+N-1) \\ Y_r(k+N-1) \\ Z_r(k+N-1) \\ \psi_r(k+N-1) \end{bmatrix} \tag{4.10}$$

$$\bar{Q} = \begin{bmatrix} Q & 0 & 0 & \cdots & 0 \\ 0 & Q & 0 & \cdots & 0 \\ 0 & 0 & Q & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & Q_f \end{bmatrix}, \bar{R} = \begin{bmatrix} R & 0 & 0 & \cdots & 0 \\ 0 & R & 0 & \cdots & 0 \\ 0 & 0 & R & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & R \end{bmatrix}$$

$$T_o = \begin{bmatrix} \tilde{C} \\ \tilde{C}\tilde{A} \\ \tilde{C}\tilde{A}^2 \\ \vdots \\ \tilde{C}\tilde{A}^{N-1} \end{bmatrix}, S_o = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ \tilde{C}\tilde{B} & 0 & 0 & \cdots & 0 \\ \tilde{C}\tilde{A}\tilde{B} & \tilde{C}\tilde{B} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \tilde{C}\tilde{A}^{N-1}\tilde{B} & \tilde{C}\tilde{A}^{N-2}\tilde{B} & \cdots & \tilde{C}\tilde{B} & 0 \end{bmatrix} \tag{4.11}$$

4.3 State and input constraints

To prevent that the matrix $\mathbf{T}_{\phi\theta\psi}$ in equation 2.9 will become singular, we define the following state constraints:

$$-\pi < \phi < \pi \tag{4.12}$$

$$-\frac{\pi}{2} < \theta < \frac{\pi}{2} \tag{4.13}$$

$$-\pi < \psi < \pi \tag{4.14}$$

The state constraints of 4.12 and 4.14 can be formulated in generalized form.

$$\underbrace{\begin{bmatrix} C_z \\ -C_z \end{bmatrix}}_{I_x} x_k < \underbrace{\begin{bmatrix} x_{ub} \\ -x_{lb} \end{bmatrix}}_{X_c} \tag{4.15}$$

Such that

$$C_z = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.16}$$

Since the continuous time system is linearized around its hovering position, which means that each rotor compensates a quarter of the gravity force of the drone, we define the lower input constraint to be $-\frac{m}{4g}$ and the upper input constraint $\frac{m}{4g}$ for each rotor. These constraints can be formulated in generalized form.

$$\underbrace{\begin{bmatrix} I_4 \\ -I_4 \end{bmatrix}}_{I_u} u_k < \underbrace{\begin{bmatrix} u_{ub} \\ -u_{lb} \end{bmatrix}}_{U_c} \tag{4.17}$$

The state and input constraints given in 4.15 and 4.17 are then easily transformed in the Model Predictive Control framework, such that the final minimization problem can be represented in the following way.

$$\min_{u} \quad u^T H u + h^T u \tag{4.18}$$

subject to

$$\begin{bmatrix} I_x & & & \\ & I_x & & \\ & & \ddots & \\ & & & I_x \end{bmatrix} \begin{bmatrix} x_k \\ x_{k+1} \\ \vdots \\ x_{k+N-1} \end{bmatrix} < \begin{bmatrix} X_c \\ X_c \\ \vdots \\ X_c \end{bmatrix} \tag{4.19}$$

$$\begin{bmatrix} I_u & & & \\ & I_u & & \\ & & \ddots & \\ & & & I_u \end{bmatrix} \begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+N-1} \end{bmatrix} < \begin{bmatrix} U_c \\ U_c \\ \vdots \\ U_c \end{bmatrix} \tag{4.20}$$

5. ASYMPTOTIC STABILITY

To prove the system is asymptotic stable, it needs to be shown the cost function, V^*, with optimal inputs, u^*, is a Lyapunov function. Following the MPC stability theorem ([Grammatico 2019]), the following inequalities must hold for the optimal value function to be a Lyapunov function, following lemma 2.14 in [J.B. Rawlings 2009].

$$\alpha_1(|x|) \le V_N^0(x) \le \alpha_2(|x|) \qquad \forall x \in \mathcal{X}_N \tag{5.1}$$

$$V_N^0(f(x, \kappa_N(x))) \le V_N^0(x) - \alpha_3(|x|) \qquad \forall x \in \mathcal{X}_N \tag{5.2}$$

The quadratic value function 4.5 is zero for $x = 0$ and $u = 0$ and strictly positive for $x \neq 0$ and $u \neq 0$. Besides that it can be seen in figure 2, where an horizon 10 steps ahead is used, that the inequality 5.2 holds, since the value function is non-increasing. Therefore the system is asymptotic stable.

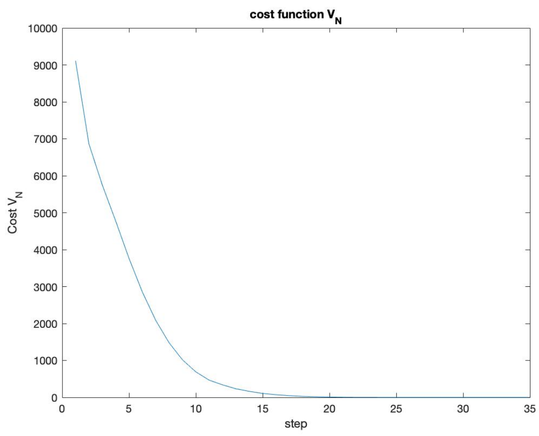


Figure 2: Optimal cost function, N $= 10$, $t_s = 0.2$

This asymptotic stability however depends on the size of N used and becomes unreliable as the dimension of the prediction horizon decreases. It is from figure 3 clearly visible that the value function is not non-increasing anymore for $N = 4$ and becomes unstable for $N = 3$. Therefore no statement can be made about asymptotic stability. Please note that minimum dimension of the horizon is dependent on the sampling time.

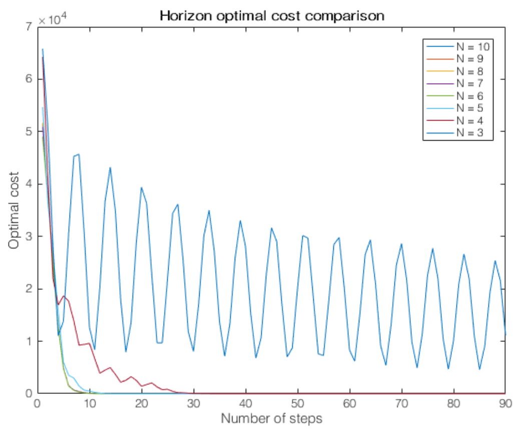


Figure 3: Optimal cost function, $t_s = 0.2$

6. NUMERICAL SIMULATION RESULTS

In this section multiple different situations are simulated. These situations are going to a predefined hovering altitude using a PID controller, going to a predefined hovering altitude using a MPC controller and finally tracking of a predefined path by using a MPC controller. The used parameters for the simulations are given in table 1. Please note that the values of K_m and K_f, the drones moment and thrust coefficient respectively, have been taken from [Islam 2017].

Table 1: Parameters

Parameter	Symbol	Value	Unit
Sample time	T_s	0.2	s
Simulation length	T	20	s
Mass drone	m	20	kg
Length arm	l	0.2	m
Gravity	g	9.81	$\frac{m}{s^2}$
Moment coefficient	K_m	$7.5e{-7}$	Nms^2
Thrust coefficient	K_f	$3.13e{-5}$	Ns^2
Moment of inertia	I	0.18	$\frac{m}{s^2}$
Horizon length	N	5	-
Proportional gain	K_p	20	-
Integrator gain	K_i	0	-
Derivative gain	K_d	20	-
Penalty weight	Q	$1e3 \times I_{12}$	-
Penalty weight	R	I_4	-

6.1 Results

In this section the results of various simulations are being shown. First a PID controller is used to maintain a certain altitude level and compared with a MPC controller whereafter the performance of the MPC will be tested on the basis of trajectory tracking. And at last different horizon lengths for the MPC are compared with each other.

6.1.1 Hovering altitude PID

Figure 4 shows the step response simulation of the drone controlled by the PID controller. From this figure we may conclude that a simple PID controller should be sufficient enough, however the biggest disadvantage of a PID controller is that it is designed such for a specific input. One of the requirements of this research is that it is important that the drone succeeds in tracking all kinds of trajectories, including different frequencies.

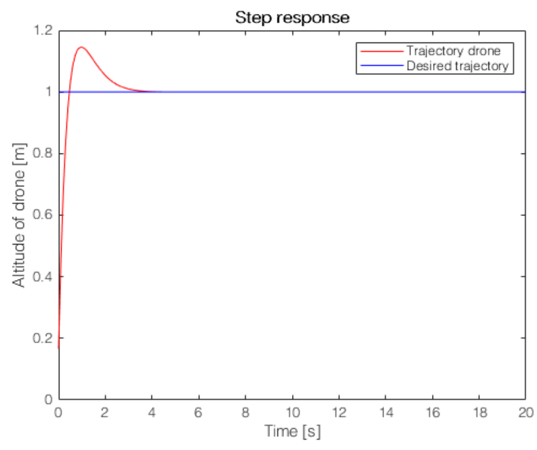


Figure 4: Step response altitude

Figure 5 therefore shows a major drawback for a classical PID-controller. It shows the altitude tracking of the drone following the reference path given in 6.1.

$$z_r = 1 + \sin(0.1k^2) \tag{6.1}$$

It is clearly visible that the closed loop system is less responsive at different kinds of frequencies. One way of solving this is with high gain feedback ($K_p = 1000$, $K_d = 1000$), however this will result in large input signals, which will violate the input constraints given in 4.17.

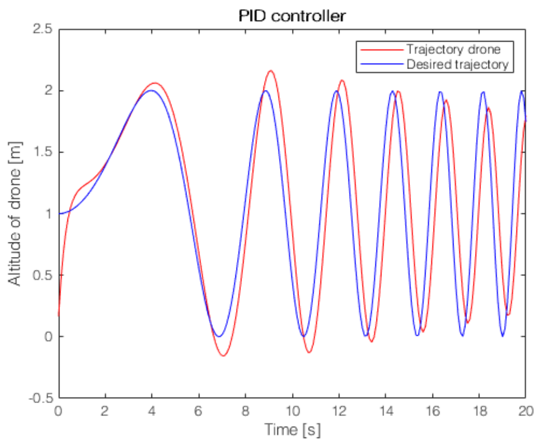


Figure 5: Trajectory tracking varying frequencies

6.1.2 *Hovering altitude MPC*

To compare the MPC with the PID controller the same procedure has been set up for the trajectory tracking. Figure 6 displays the step response of the closed loop system controlled by the MPC.

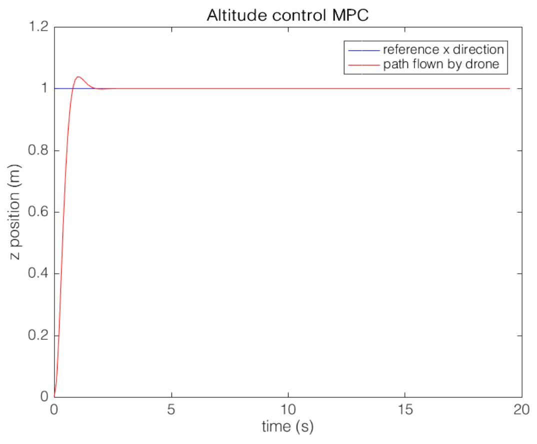


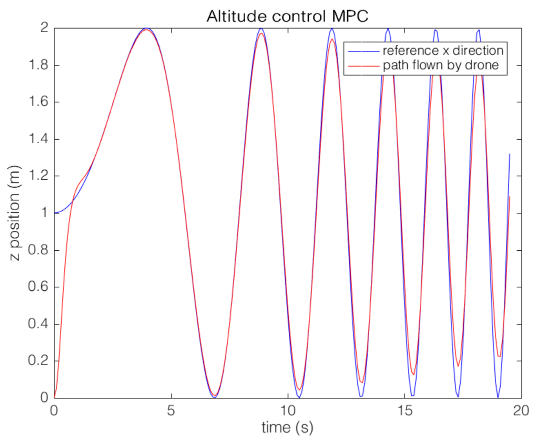
Figure 6: Step response of quadcopter with MPC

From the time domain comparison in 2 we can conclude that the Model Predictive Controller has a better performance. Although the PID controller performs better according to the rise time, does the MPC have better results looking at the settling time and overshoot. The latter properties are especially in trajectory tracking of great importance.

Table 2: Stepresponse information PID and MCP control

Property	PID	MPC
Rise time	0.3428 s	0.5042 s
Settling time	2.7084 s	1.3345 s
Overshoot	14.5639%	3.8020%

Figure 6.1.2 displays the trajectory tracking of the Model



Predictive Controller according to the reference signal given in 6.1. This figure compared with figure 5 we can conclude that with varying frequencies the MPC responds much faster than the PID-controller and is therefore preferable for trajectory tracking. This conclusion is substantiated by the root mean square error of both signals with respect to the reference signal, as can be seen in table 3, where it reduced the RMSE with 71%.

Table 3: Time domain comparison PID and MPC

	RMSE
PID	0.5185
MPC	0.1482

6.1.3 *Path trajectory MPC in $\mathcal{R}^3$*

To see how the model predictive controller responds to more complex trajectories a reference path has been made were,

$$x_r = 5\sin(k) \tag{6.2}$$
$$y_r = k\cos(k) \tag{6.3}$$
$$z_r = k \tag{6.4}$$

The results of the MPC control can be seen in figure 7.

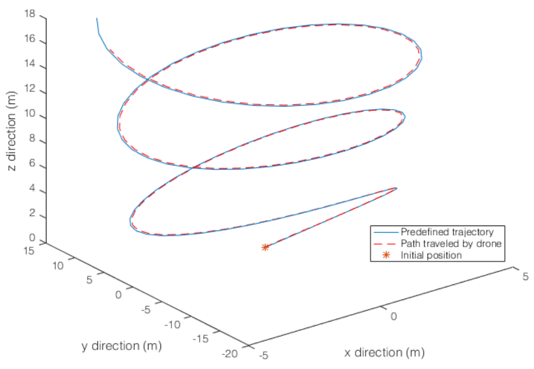


Figure 7: Path tracking in x, y and z direction

6.1.4 *MPC with disturbance rejection*

To examine the robustness of the designed controller a disturbance signal is excited on the system. Figure 8 illustrates the system response of two different disturbance signals, with different gains and directions. From this figure it hence can concluded that the closed loop remains stable and that the controller succeeds in achieving robustness in order to reject disturbances.

6.1.5 *Alternating horizon*

To examine the influence of the horizon length tests have been done varying horizon lengths. Figure 9 depicts result of the drone trajectory while alternating the horizon length. It is clearly visible that the first ≈ 7 seconds influence the Root Mean Square Error the most, since that is the maximum of the derivative of the desired trajectory. By investigating the RMSE of figure 10 (see table 4) we conclude that a horizon of $N = 6$ will give the best results. Note that $N = 3$ is let outside in table 4, since it is from figure 9 rather straightforward that $N = 3$ is not optimal.

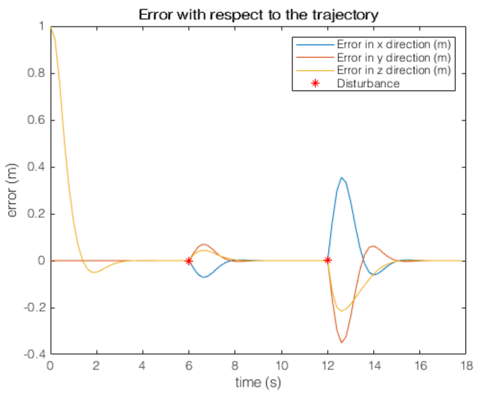


Figure 8: Disturbance rejection

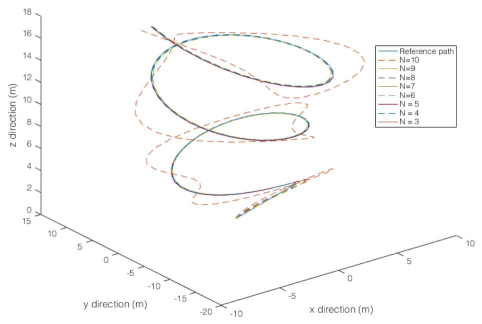


Figure 9: Trajectory tracking with various horizon lengths

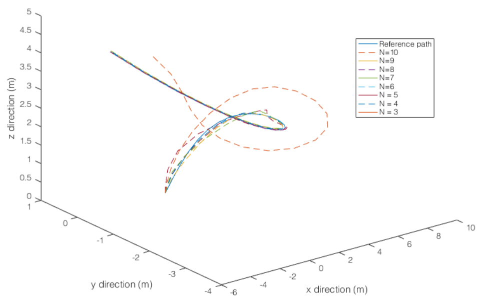


Figure 10: Trajectory tracking with various horizon lengths first 7 seconds

6.2 Discussion

The goal of this paper is to design a model predictive controller where performance is tested and globally com-

Table 4: RMSE per direction with different horizon lengths (N)

	Horizon length (N)						
	10	**9**	**8**	**7**	**6**	**5**	**4**
X dir.	0.608	0.597	0.587	0.578	0.570	0.577	0.619
Y dir.	0.092	0.092	0.091	0.086	0.088	0.097	0.076
Z dir.	0.116	0.113	0.112	0.109	0.110	0.097	0.089
Norm	0.626	0.614	0.605	0.595	$\underline{0.587}$	0.593	0.630

pared with different controllers. The results are all obtained with constant control parameters, e.g. PID gains, penalty weights and sample time. The PID controller mentioned in section 6.1.1 could be tuned such that it has better time response properties and follows the mentioned sinusoid in a more responsive manner, although this is not a realistic point of view since the drone should be capable of tracking all kinds of trajectories. This implies that by optimizing the controller for a certain trajectory, we decrease the performance of the controller for a complete different trajectory. That is why Model Predictive Control is such a favourable method of control, since it is less sensitive to different kinds of trajectories due to online computation of input sequences.

7. CONCLUSION

The results show a clear improvement in performance when comparing the PID and MPC in obtaining a predefined hovering altitude, where the settling time has been halved and the overshoot has been decreased from 14.5% to 3.8% using MPC instead of PID control. Testing more complex trajectories for the MPC resulted in relatively low errors. However the controller works accordingly, nothing can be said about the stability of the model predictive control for shorter prediction horizons (< 4, with $t_s = 0.1$) as for some moments $\dot{V}(x,u) \nless 0$.

Mike de Pont and Daniel van Hanswijk are masterstudents at TU Delft, The Netherlands. E-mail addresses:
m.h.m.depont@student.tudelft.nl
d.vanhanswijk@student.tudelft.nl

8. REFERENCES

[Grammatico 2019] Grammatico, S. (2019). *Lecture notes model predictive control.*

[Islam 2017] Islam, M. (2017). Dynamics and control of quadcopter using linear model predictive control approach. *IOP Conf. Ser.: Mater. Sci. Eng.*, *6*.

[J.B. Rawlings 2009] J.B. Rawlings, D. M. (2009). *Model predictive control - theory and design.*

[Ostojic et al. 2015] Ostojic, G., Stankovski, S., Tejic, B., Đukić, N., & Tegeltija, S. (2015, 01). Design, control and application of quadcopter. *International Journal of Industrial Engineering and Management*, *6*, 43-48.

[Sabatino 2015] Sabatino, F. (2015). Quadrotor control: modeling, nonlinear control design, and simulation.