A real-time model predictive position control with collision avoidance for commercial low-cost quadrotors*

Jan Dentler, Somasundar Kannan, Miguel Angel Olivares Mendez, Holger Voos[1]

***Abstract*— Unmanned aerial vehicles (*UAVs*) are the future technology for autonomous fast transportation of individual goods. They have the advantage of being small, fast and not to be limited to the local infrastructure. This is not only interesting for delivery of private consumption goods up to the doorstep, but also particularly for smart factories. One drawback of autonomous drone technology is the high development costs, that limit research and development to a small audience. This work is introducing a position control with collision avoidance as a first step to make low-cost drones more accessible to the execution of autonomous tasks. The paper introduces a semilinear state-space model for a commercial quadrotor and its adaptation to the commercially available *AR.Drone*[2] system. The position control introduced in this paper is a model predictive control (*MPC*) based on a condensed multiple-shooting continuation generalized minimal residual method (*CMSCGMRES*). The collision avoidance is implemented in the *MPC* based on a sigmoid function. The real-time applicability of the proposed methods is demonstrated in two experiments with a real *AR.Drone* quadrotor, adressing position tracking and collision avoidance. The experiments show the computational efficiency of the proposed control design with a measured maximum computation time of less than *2*ms.**

I. INTRODUCTION

The significance of unmanned aerial vehicles has been increasing over the last decades by more and more civil applications such as maintenance applications, transportation, as a toy, etc. The idea to use drones for fast transportation is particularly interesting for dynamically changing autonomous transportation tasks of small goods, as in smart factories or home delivery scenarios. As a further plus, no expensive infrastructure is needed. The main problem up to today is the high costs of professional drones that can be adapted for autonomous flights in urban areas. This is mainly caused by the ability to determine the precise position of the *UAV*s and its security measures, e.g. avoiding obstacles via vision sensors. As multi-rotor *UAV*s are typically capable of keeping a steady position, they are particularly interesting from the security point of view. The adaption of commercial low-cost *UAV*s for autonomous flying is therefore an important way to make this technology cheaper and therefore accessible. As a consequence, this work aims to autonomously control the position of an *AR.Drone* while avoiding obstacles. The *AR.Drone* comes with an inner controller that is stabilizing the attitude. Furthermore it can be controlled through velocity commands via WiFi which makes it easy to access with a ground control station. The *AR.Drone* is chosen, because these features are representative for most commercial low-cost quadrotor systems.

Today there is a wide variety of quadrotor models and control strategies. An overview of different designs and quadrotor models is given in the thesis [13]. A widely applied control algorithm is a position control approach realized as a hover *PID* controller which is presented in detail in [8]. This control strategy separates forward, sideward, upward and heading by linearization of the system dynamics. The four channels are then controlled independantly with PID controllers. The influence of aerial disturbances on the control behavior is analyzed in [16]. [9] describes the construction of a mathematical drone model in a detailed and comprehensive way. Furthermore a state estimator and a vision-based *PID* position control are derived. More detailed information of vision based quadrotor control strategies are given by [11]. A vision based fuzzy position control is presented in [15]. In [10] a backstepping controller is introduced. In [14] the authors discuss a multiple surfaces sliding mode control for quadrotors. A model reference adaptive control concept for quadrotors is presented in [12]. It is based on the adaptation of controller parameters based on linearized system dynamics. This allows the controller to act in a bigger trust region than conventional hover control.

An important project for quadrotor design and control was the OS4 Project at the Swiss Federal Institute of Technology. In the context of this project [2] is introducing a full nonlinear quadrotor model. Based on the stabilization proof via Lyapunov function, a hover *PID*-controller is derived to stabilize the system around its nominal state. The disadvantage of this controller is its poor performance regarding disturbance. A quadrotor *LQR*-controller was developed in [3] and compared to the classical *PID* hover control. In the experiment the *LQR* approach showed a less dynamic behavior plus a steady-state error. The author came to the conclusion that even if the *LQR* was expected to show better results, the *PID* controller showed better performance in the experiments and is therefore preferred. In [4] a backstepping control is developed for the linear translational subsystem and a sliding mode controller is derived for the attitude subsystem of a quadrotor. The presented controlled system shows a strong resistance towards disturbance, but introduces

*This work was supported by FNR "Fonds national de la Recherche" (Luxembourg) through AFR "Aides à la Formation-Recherche" Ph.D. grant scheme No. 9312118.

[1] Jan Dentler, Dr. Somasundar Kannan, Dr. Miguel Angel Olivares-Mendez and Prof. Dr.-Ing Holger Voos are with Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg, L-1359 Luxembourg, Luxembourg jan.dentler@uni.lu, somasundar.kannan@uni.lu, miguel.olivaresmendez@uni.lu, holger.voos@uni.lu

high frequencies into the system controls, that are causing sensor drift. More detailed information is given in [5]. Based on this, [6] is presenting an integral backstepping controller as further development. The result is a control law for the full state model.

With the increasing development of fast computers, model predictive control approaches have become real-time applicable for fast systems. A *LQR* control, as e.g. presented in [3], is equal to the analytical solution for an unconstrained linear model predictive control problem and therefore familiar. The big advantage of the model predictive approach is the possible usage of constraints, which is particularly interesting in this work for implementing security measures. A *MPC* example with state constraints for quadrotors is simulated in [18]. Details about collision avoidance for quadrotors and various *MPC* methods for multiple drone coordination are given in [17]. The most essential obstacle to apply *MPC* on quadrotor systems, is the high computational burden plus the quadrotor nonlinearity. Examples for especially fast nonlinear *MPC* algorithms are Gauß-Newton methods [21] as implemented in *ACADO* [19], or gradient methods as implemented in *GRAMPC* [20]. Another particularly fast method is the continuation generalized minimal residual method *CGMRES* method as given in [23],[24],[25],[26],[27]. An adaption of *CGMRES* for cooperative control tasks in real-time is discussed in [7]. For this work, a condensed multiple shooting *CGMRES* derivative has been applied, which offers a compromise of higher numerical stability with fast computation. This condensed multiple-shooting continuation generalized minimal residual method *CMSCGMRES* was developed under supervision of Prof. Dr. Toshiyuki Ohtsuka and is presented in [28],[29],[30].

The first contribution of the presented paper is the introduction of a semilinear analytical quadrotor model based on velocity controls as described in section II. Second, the proposed model is adapted to the commercial *UAV AR.Drone* by identification in section II-A. Third, section III is proposing the condensed multiple shooting continuation generalized minimal residual method (*CMSCGMRES*) [28] as a real-time applicable nonlinear model predictive control approach for *UAV* applications. The advantages of *CMSCGMRES* are the handling of nonlinearities and its very low computation time combined with the possibility to introduce constraints. In section III-A the proposed algorithm is experimentally validated on a real *AR.Drone* under use of the presented model. The final contribution is the combination of *CMSCGMRES* with a sigmoid collision avoidance (CA) method, as described in section IV. The resulting performance with a real *AR.Drone* is shown in section IV-A. In section V the conclusion and further developments are discussed.

II. Semilinear hover control model

In typical quadrotor applications, the *UAV* is internally controlled by a hover controller as introduced in [8]. This means it is driven around its nominal state, the hovering position. Accordingly the controller reference is representing desired velocity in forward direction u_f, sidewards u_s, upwards u_z and the angular velocity around the z axis u_Ψ. Standard *PID* hover controllers are based on a linearization in the hover position which is equal to a small angle approximation of roll $\mathbf{\Phi}$ and pitch Θ. The forward and sideward movement of the quadrotor is therefore considered to be in the *xy*-plane and can be mapped to the global coordinate system via a Ψ rotation. This can be expressed in the state space with the state vector

$$\mathbf{x}(t) = \left[x_{\mathscr{W}}(t), y_{\mathscr{W}}(t), z_{\mathscr{W}}(t), \Psi_{\mathscr{W}}(t), \dot{x}_{\mathscr{V}}(t), \dot{y}_{\mathscr{V}}(t)\right]^{\top}, \quad (1)$$

which yields to a quasi-linear state-space model

$$\begin{bmatrix} \dot{x}_{\mathscr{W}}(t) \\ \dot{y}_{\mathscr{W}}(t) \\ \dot{z}_{\mathscr{W}}(t) \\ \dot{\Psi}_{\mathscr{W}}(t) \\ \ddot{x}_{\mathscr{V}}(t) \\ \ddot{y}_{\mathscr{V}}(t) \end{bmatrix} = \begin{bmatrix} \dot{x}_{\mathscr{V}}(t)\cos(\Psi) - \dot{y}_{\mathscr{V}}\sin(\Psi) \\ \dot{x}_{\mathscr{V}}(t)\sin(\Psi) + \dot{y}_{\mathscr{V}}\cos(\Psi) \\ a_z \cdot z(t) + b_z \cdot u_z(t) \\ a_\Psi \cdot \Psi(t) + b_\Psi \cdot u_\Psi(t) \\ a_f \cdot \dot{x}_{\mathscr{V}}(t) + b_f \cdot u_f(t) \\ a_s \cdot \dot{y}_{\mathscr{V}}(t) - b_s \cdot u_s(t) \end{bmatrix} \begin{matrix} \left.\vphantom{\begin{matrix} a \\ a \end{matrix}}\right\}\text{Map: } \mathscr{V} \to \mathscr{W} \\ \left.\vphantom{\begin{matrix} a \\ a \\ a \\ a \end{matrix}}\right\}\text{Linear model.} \end{matrix} \quad (2)$$

$\mathscr{W}$ is referring to the global world coordinate frame, while index $\mathscr{V}$ indicates the vehicle coordinate frame. The origin of $\mathscr{V}$ is equivalent with the origin of the quadrotor. The pose of the coordinate frames can be seen in Fig. 1, where pitch and roll are considered to be neglectable. $x_{\mathscr{V}}, y_{\mathscr{V}}$ are therefore lying in the $x_{\mathscr{W}}y_{\mathscr{W}}$-plane, where $x_{\mathscr{V}}$ is pointing to the front of the *UAV* and $y_{\mathscr{V}}$ to the left. The z-axis is aligned with the world coordinate frame z . As shown in (2) the state-space model consists of a linearization around the hover position in the vehicle-frame, as well as a nonlinear mapping of the x, y vehicle frame components to the world coordinates. Using the states $x_{\mathscr{W}}, y_{\mathscr{W}}, z_{\mathscr{W}}, \psi_{\mathscr{W}}$ has the advantage of being able to calculate the error to a world coordinate reference position.

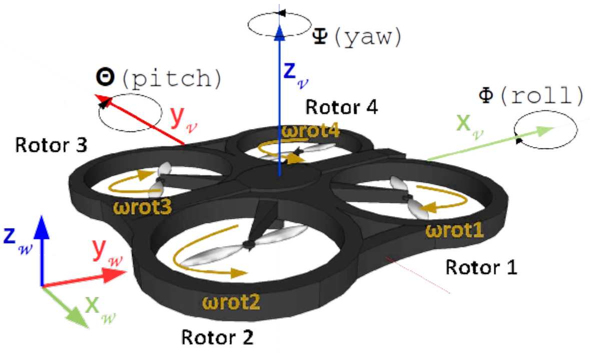

Fig. 1: Coordinate frame definition

A. Parameter identification

The used case considered for this paper is the commercial quadrotor system *Parrot AR.Drone 2.0*. To identify the linear model parameters $a_z, a_\Psi, a_f, a_s, b_z, b_\Psi, b_f, b_s$ of (2), the step response of the system is approximated with the given model. The measurement of the quadrotor position is realized with the motion capture system *OPTITRACK*[1]. The quadrotor velocities are determined by a finite difference of the position and subsequent filtering. Finally the velocities are mapped to the vehicle coordinate frame via the inverse mapping of (2).

[1] Copyright 2016 NaturalPoint, Inc. All rights reserved.

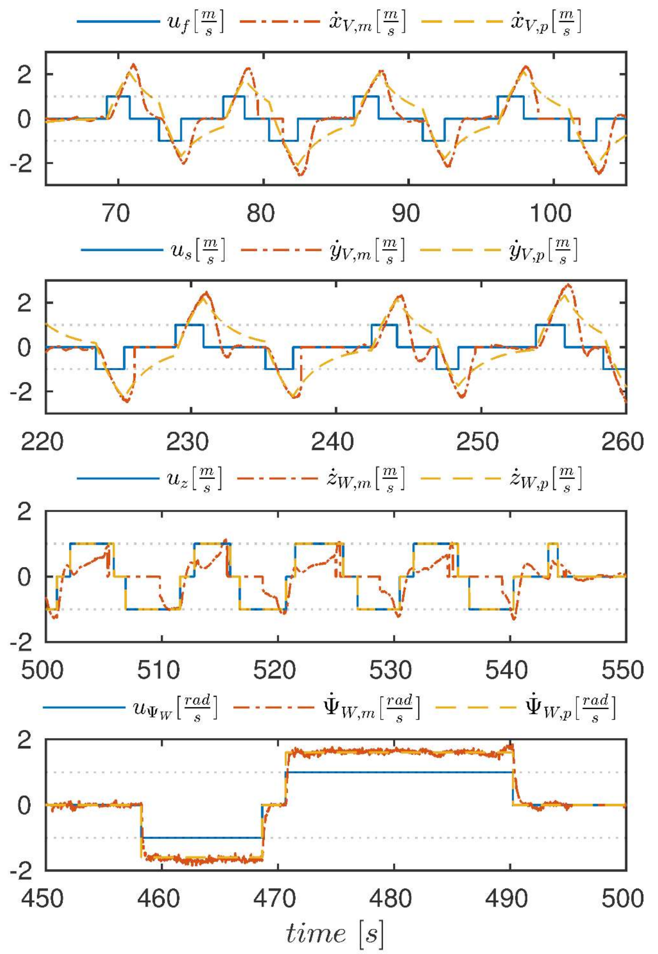

Fig. 2: *AR.Drone* identification: Top to bottom: forward, sideward, upward and yaw channel.

Figure 2 is showing the identification signals, with the given control inputs, the processed (filtered) measurement response (index m) and the model prediction (index p) for the chosen model parameters $a_f = a_s = -0.5092, b_f = b_s = 1.458, a_z = 0, b_z = 1, a_\Psi = 0, b_\Psi = 1.6$. Considering $\dot{x}_{\mathscr{V}}$ and $\dot{y}_{\mathscr{V}}$ in (2), the chosen model parameters yield to a tangential approximation of the real systems behavior in the points of control changes, as we linearize around the nominal state (where the controls are 0). In the experiments, the *Parrot AR.Drone 2.0* has shown an indeterministic behavior in the z axis. This is expected to be caused by the quality of the ultrasonic sensor data that is used to estimate the altitude for the internal controller. The final choice of using a simple integrator with $b_z = 1$ is representing the idea of a system that reaches the reference in the z-axis very fast. Also $\Psi_{\mathscr{W}}$ shows integration behavior under the typical limitation of $-\pi < z_{\mathscr{W}} \leq \pi$. Finally the *AR.Drone* model yields to

$$\begin{bmatrix} \dot{x}_{\mathscr{W}}(t) \\ \dot{y}_{\mathscr{W}}(t) \\ \dot{z}_{\mathscr{W}}(t) \\ \dot{\Psi}_{\mathscr{W}}(t) \\ \ddot{x}_{\mathscr{V}}(t) \\ \ddot{y}_{\mathscr{V}}(t) \end{bmatrix} = \begin{bmatrix} \dot{x}_{\mathscr{V}}(t)\cos(\Psi) - \dot{y}_{\mathscr{V}}\sin(\Psi) \\ \dot{x}_{\mathscr{V}}(t)\sin(\Psi) + \dot{y}_{\mathscr{V}}\cos(\Psi) \\ 1 \cdot u_z(t) \\ 0 \cdot \Psi(t) + 1.6 \cdot u_\Psi(t) \\ -0.5092 \cdot \dot{x}_{\mathscr{V}}(t) + 1.458 \cdot u_f(t) \\ -0.5092 \cdot \dot{y}_{\mathscr{V}}(t) + 1.458 \cdot u_s(t) \end{bmatrix}. \tag{3}$$

III. MODEL PREDICTIVE POSITION CONTROL

More advanced commercial *UAV* applications face the problem of navigation in urban environments. The difficulty of urban navigation is to autonomously take into account the movement constraints, that are represented by obstacles or prohibited areas. An advanced control method to handle such constraints is *MPC*. *MPC* is calculating optimal controls online to minimize a given cost function within a receding horizon. Combined with constraints and boundary values, this represents an optimal control problem *OCP*. With state vector $\mathbf{x} = [x_{\mathscr{W}}, y_{\mathscr{W}}, z_{\mathscr{W}}, \Psi_{\mathscr{W}}, \dot{x}_{\mathscr{V}}, \dot{y}_{\mathscr{V}}]$ and control vector $\mathbf{u} = [u_f, u_s, u_z, u_\Psi]$, the *OCP* for a position control with the nonlinear quadrotor dynamics (3) can be formulated as

$$\min_{\mathbf{u}} J = \int_{t_0}^{t_f} (\mathbf{x}^* - \mathbf{x})^\top \mathbf{Q} (\mathbf{x}^* - \mathbf{x}) + \mathbf{u}^\top \mathbf{R} \mathbf{u} \ \mathrm{d}\tau \tag{4}$$

s.t.

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{x}_{\mathscr{V}}(t)\cos(\Psi) - \dot{y}_{\mathscr{V}}\sin(\Psi) \\ \dot{x}_{\mathscr{V}}(t)\sin(\Psi) + \dot{y}_{\mathscr{V}}\cos(\Psi) \\ 1 \cdot u_z(t) \\ 0 \cdot \Psi(t) + 1.6 \cdot u_\Psi(t) \\ -0.5092 \cdot \dot{x}_{\mathscr{V}}(t) + 1.458 \cdot u_f(t) \\ -0.5092 \cdot \dot{y}_{\mathscr{V}}(t) + 1.458 \cdot u_s(t) \end{bmatrix} \tag{5}$$

$$c \leq (u - \bar{u})^2 - (u_{max} - \bar{u})^2 : \forall u : u_{max} = 1 \vee \bar{u} = 0 \tag{6}$$

$$\mathbf{x}(0) = [0,0,0,0,0,0] \tag{7}$$

$$\mathbf{Q} = \mathscr{D}\{[1,1,1,10,0,0]\}, \qquad \mathbf{R} = \mathscr{D}\{[1.3,1.3,3.0,1.1]\}. \tag{8}$$

The cost function (4) is defined to become minimal for reaching the target state $\mathbf{x} = \mathbf{x}^*$ and minimal input $\mathbf{u} = \mathbf{0}$ subject to the given input constraints (6) that limit $-1 \leq u_f, u_s, u_z, u_\Psi \leq 1$ and the initial state (7). To track the position, $\mathbf{Q}$ is defined as diagonal matrix $\mathscr{D}$ with the entries as given in (8). There are just relevant penalty entries for the first 4 elements which penalize an error in $x_{\mathscr{W}}, y_{\mathscr{W}}, z_{\mathscr{W}}, \Psi_{\mathscr{W}}$. Accordingly $\mathbf{R}$ in (8) is representing the diagonal control penalty matrix to minimize the control action and therefore the energy consumption. The values of $\mathbf{R}, \mathbf{Q}$ are chosen experimentally to achieve a smooth and fast flight behavior.

To create a feedback of the system, after the solving of the *OCP* and applying the control, the horizon $[t_0, t_f]$ is shifted and now solved in respect to the new initial value provided by measurements of the real system. Accordingly the *OCP* (4)-(7) has to be solved at each control update interval, which is challenging for fast systems like quadrotors. As discussed in the introduction, within this work a fast nonlinear *MPC* solver *CMSCGMRES* is used. This is a condensed multiple-shooting version of the *CGMRES* by Ohtsuka [23] that represents a newton-type method under use of Hessian approximation via forward difference method. Algorithmic details are given in [28],[29] and [30]. *CMSCGMRES* is capable of nonlinear optimization and offers low computation times, as well as constraint handling. This makes it particularly suited for *UAV* applications. The reason for using a nonlinear *MPC* solver to control a semilinear system is the possibility to directly extend the results to more nonlinear systems, e.g. nonlinear models, a drone with a manipulator, etc. Furthermore the nonlinear mapping between vehicle and coordinate frame can be directly calculated within the optimization. The performance of *CMSCGMRES* with the proposed *AR.Drone* model is evaluated experimentally in the following section.

A. POSITION CONTROL EXPERIMENT

To validate the proposed *CMSCGMRES* position control, introduced in section III, the quadrotor is moved in a square in the *xy*-plane in 5 phases. Phase 1 to 5 are initiated by a change of the tracked state as given in table I. The reference states represent corners of the square, starting from its initial position. Phase 5 is moving the drone subsequently to the center. The behavior of the *AR.Drone* in phase 1 to 5 is illustrated simplified in Fig. 3. As expected the *AR.Drone* is executing a square movement from phase 1 to 5. The corresponding trajectory plots are given in Fig. 5. The movement between the corner point is shown transparent in Fig. 3 for means of visualization.

Phase	Time	State	Task
Initial:	$\mathbf{x}^*(0\text{s})$	$=[-1,\ 1,1,0,0,0]^\top$	Keep initial position
1:	$\mathbf{x}^*(8.4\text{s})$	$=[-1,-1,1,0,0,0]^\top$	Reach point in square
2:	$\mathbf{x}^*(13.4\text{s})$	$=[\ 1,-1,1,0,0,0]^\top$	Reach point in square
3:	$\mathbf{x}^*(18.4\text{s})$	$=[\ 1,\ 1,1,0,0,0]^\top$	Reach point in square
4:	$\mathbf{x}^*(23.4\text{s})$	$=[\ 1,-1,1,0,0,0]^\top$	Reach point in square
5,6,7:	$\mathbf{x}^*(28.4\text{s})$	$=[\ 0,\ 0,1,0,0,0]^\top$	Reach center/Disturbance

TABLE I: Tracked points of the position control experiment

To validate the stability in case of more complex disturbances, an impulse is applied on the *AR.Drone* around the *z*-axis in phase 6 and 7. The corresponding drone behavior is shown simplified in Fig. 4. As desired, the drone is stabilizing Ψ and the position in the resulting compensation circle. Within Fig. 4 the drone transparency is decreased with the proceeding time to demonstrate the speed of the system.

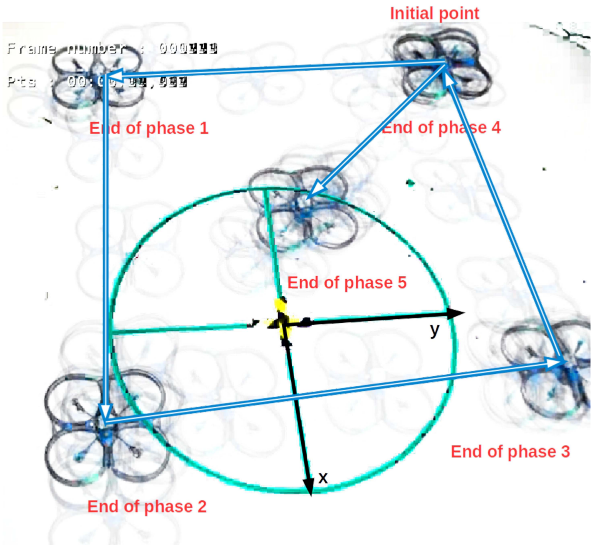

Fig. 3: *AR.Drone* position control:

The system trajectory is given in Fig. 5. The plots are showing forward, sideward, upward and yaw channel and the computation time for a control update interval of $\Delta t = 0.1$s.

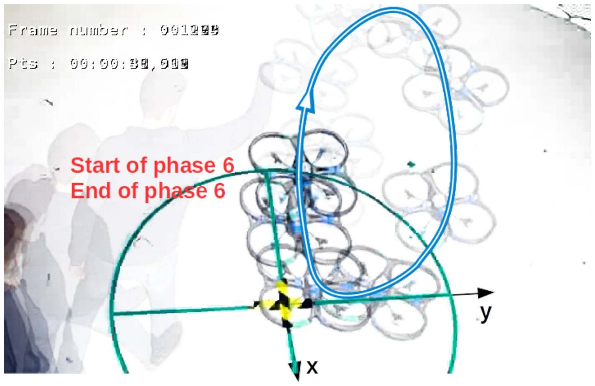

Fig. 4: *AR.Drone* stabilization under disturbance:

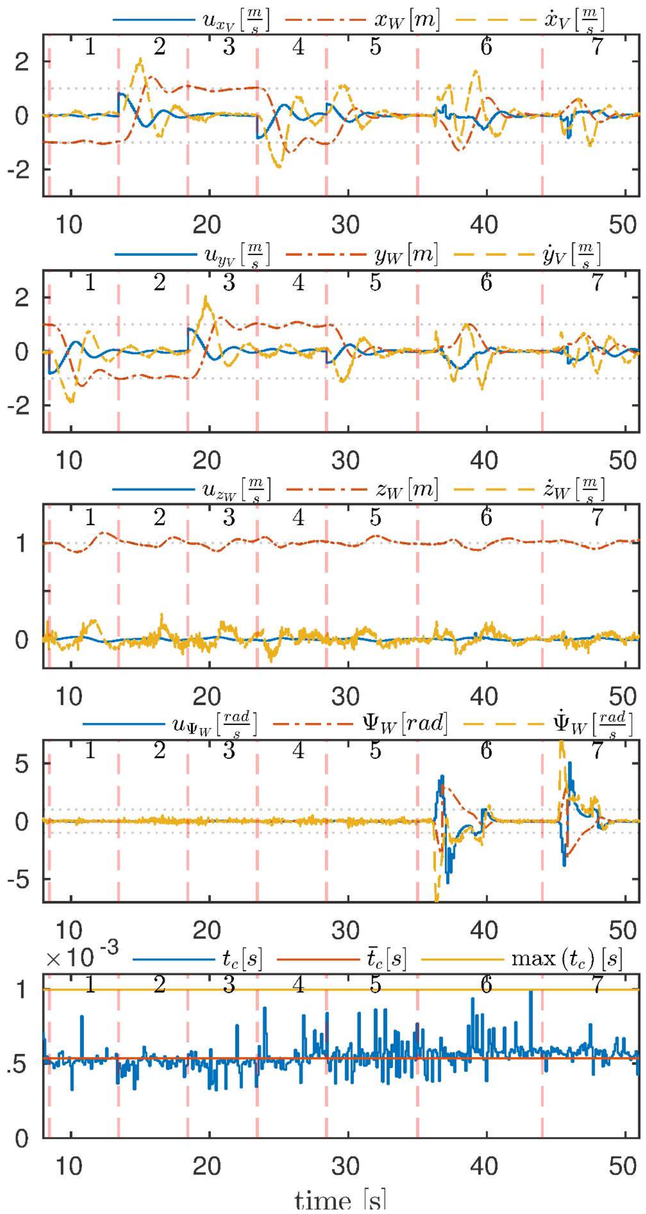

Fig. 5: *AR.Drone* position control trajectory:
Top to bottom: forward, sideward, upward and yaw channel, computation time with $max(t_c) \approx 1$ms.

The *CMSCGMRES* solver is parametrized with a horizon time $T = 1$s, $nhor = 10$ samples per horizon, the forward difference approximation interval $h = 1$ms, a maximum of $it_{max} = 10$ iterations and a continuation parameter of $\alpha = 10$. Furthermore the control limitation constraints (6) are treated with the *CGMRES* package internal interior point constraint handling method. This method is using slack variables to model inequality constraints and slack variable penalties in the cost function (details in [26]). For this experiment all slack penalties are set to $r_{slack} = 0.001$. As can be seen in phase 6 and 7, this constraint handling technique has to be adapted for high disturbances as $|u_\Psi| > 1$ by e.g. using higher r_{slack}. As $\Psi \approx 0$ up to phase 5, the vehicle and world frame can be considered to be aligned. This allows to directly relate vehicle frame velocity to world frame position. The system response to position changes shows a typical damping of $D \approx \sqrt(2)$ which is equivalent to a minimization of the integrated position error. This matches the definition of dominating position tracking in the cost function (4). Therefore the *CMSCGMRES* approach is validated for the developed system model (3).

The advantage of the proposed *MPC* solver combined with the used model is the low computation time of $max(t_c) \approx 1$ms as shown in the bottom of Fig. 5. Furthermore the computation time is not changing particularly in case of disturbances as with comparable gradient method approaches. The video of the experiment can be downloaded via [1].

IV. MODEL PREDICTIVE COLLISION AVOIDANCE

Avoiding collisions is essential for security reasons. A typical collision avoidance (*CA*) therefore keeps the *UAV* in a desired distance d_{des} from the quadrotor postion $\vec{x}_q$ to an obstacle position $\vec{x}_O$. The problem can be formulated as inequality (9) which can be translated with $c \leq 0$ into a constraint (10). To avoid the expensive square root computation in (10), it is advantageous to use the quadratic form (11) instead. (11) represents equivalent roots to (10) as d_{des} is always positive.

$$d_{des} \leq \sqrt{(\vec{x}_O - \vec{x}_q)^\top (\vec{x}_O - \vec{x}_q)} \quad (9)$$

$$c \leq d_{des} - \sqrt{(\vec{x}_O - \vec{x}_q)^\top (\vec{x}_O - \vec{x}_q)} \quad (10)$$

$$c \leq d_{des}^2 - (\vec{x}_O - \vec{x}_q)^\top (\vec{x}_O - \vec{x}_q) \quad (11)$$

As *CMSCGMRES* is based on the solution of *OCP* (4)-(8) optimality conditions, the optimality of the inequality constraints has to be solved at each time instance, which leads to a higher computation time. A different approach is to approximate the switching behavior of an inequality constraint with a sigmoid function

$$sig(x) = \frac{b}{1 + e^{-ax}}. \quad (12)$$

Parameter b in (13) is determining the maximum value of the sigmoid and a is affecting the sharpness of the switching behavior. Inserting the right hand side of constraint (11) into (12) leads to an approximation of (11) by an additional *OCP* cost term J_{CA} to (4).

$$J_{CA} = \frac{b}{1 + e^{-a\left(d_{des}^2 - (\vec{x}_O - \vec{x}_q)^\top (\vec{x}_O - \vec{x}_q)\right)}} \quad (13)$$

The parameter b, J_{CA} can be adapted to the other system costs (4). Accordingly b has to be chosen big enough to have dominating costs J_{CA} to ensure that *CA* is prioritized in relation to trajectory tracking. Fig. 6 is showing the influence of the parameter a which is affecting the sharpness of the switching behavior. For increasing a values (Fig. 6 from left to right), the switching behavior is approximated better, but the system becomes more ill-conditioned and therefore more difficult to solve. d is representing the distance $|\vec{x}_O - \vec{x}_q|$ and d_{des} the desired distance that should be kept. If the *UAV* is in the prohibited area $d < d_{des}$ (top area in Fig. 6), the system function is dominated by (13) which means, that the solver tries to preferably minimize J_{CA} (13) and therefore increases the distance d to the obstacle. The validation of the proposed sigmoid *CA* is shown in the following section IV-A.

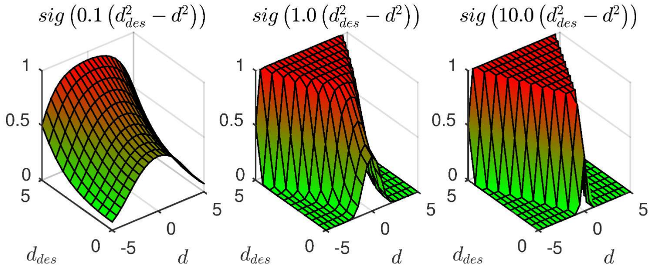

Fig. 6: Sigmoid distance cost function J_{CA} with $b = 1$

A. COLLISION AVOIDANCE EXPERIMENT

For the experimental validation J_{CA} (13) is parametrized with $a = 6$ and $b = 3$ which have been chosen experimentally. Furthermore to show an avoidance more dominant in the xy-plane, the z-axis tracking penalty is increased to $q_z = 3$. In the experiment the quadrotor is tracking a position on the opposite side of an obstacle. Accordingly the *CA* is forcing the *AR.Drone* from the direct connection trajectory onto a curve that surrounds the given obstacle. Fig. 7 is showing an example of such an *CA*-movement.

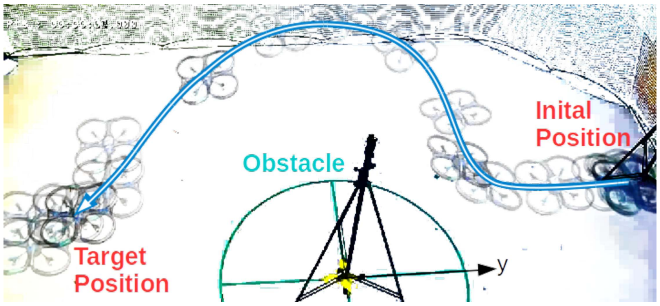

Fig. 7: *AR.Drone* collision avoidance: The trajectory of the quadrotor is deviated by an obstacle, depicted as stand in the center point of the circle. The circle radius of $r = 1$m is illustrating the keep out area. As desired, the drone trajectory is subject to $\|\vec{x}_O - \vec{x}_q\|_2 \geq d_{des} = 1$.

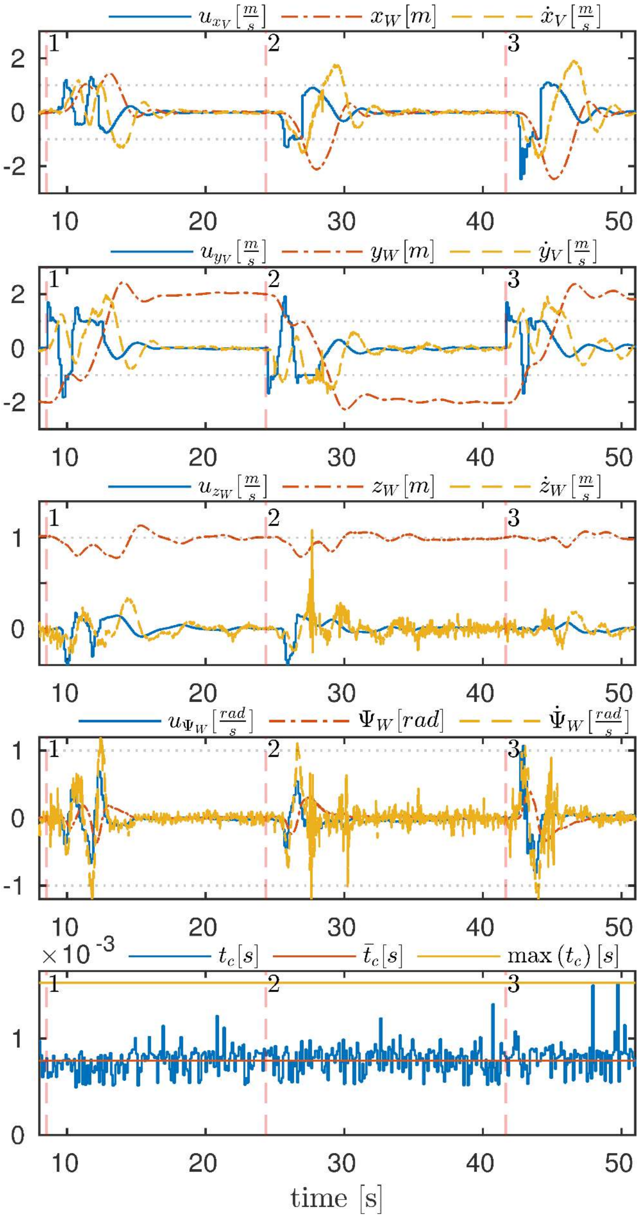

Fig. 8: *AR.Drone* position control collision avoidance

The corresponding trajectories for multiple trajectory changes are depicted in Fig. 8. Each of the three depicted section is representing a change in the target position. As the main displacement is in *y* direction, the time difference between the *x* and *y* action is caused by first just moving in *y*-direction until the *CA* sphere would be violated. Then the *x*-axis is deployed to initiate the *CA* curve. When reaching the Obstacle avoidance sphere the quadrotor is pushed away from the sphere which leads to an oscillating movement, until a feasible path is found. These oscillations are caused by a wrong prediction due to the model missmatch, and can be treated by using a more precise model, further smoothening the sigmoid and reducing the control action. As for the position control of section III, the advantage of the combination of sigmoid collision avoidance with *CMSCGMRES* is the low computation time of $max(t_c) = 1.6$ms. For this scenario the computation time is significantly higher than without collision avoidance due to the implementation. The position of the obstacle is not directly given to the cost function. Instead they are implemented as additional states to be also able to provide obstacle dynamics in a future step that allows more efficient avoidance of moving obstacles. According to the larger state vector, also the computation time is increased. To sum up, Fig. 7 and 8 are validating the efficiency of the proposed combination of *CMSCGMRES* with a sigmoid *CA*, as the obstacle is avoided and a very low computation time is achieved. The video footage of the experiment is accessible online [1].

V. CONCLUSION AND FURTHER WORK

The presented work is proposing a *CMSCGMRES* control approach for commercial low-cost multi-rotor systems, to control the global *UAV* position while avoiding obstacles. For this purpose, a semilinear *UAV* state space model is presented. Its parameters are identified by a simple step response analysis, as shown for the *AR.Drone*. The given model suits most commercial multi-rotor systems with an internal controller that receives velocity commands. To use the *MPC* only for the outer position control loop is advantageous, as the *UAV* internal controller can typically not be switched off and furthermore provides basic attitude stability. For the position control, this work is proposing the *CMSCGMRES* algorithm, as it represents a *MPC* for nonlinear systems with very low computation times combined with the ability to handle constraints. This also allows an extension to more nonlinear models, respective systems in future.

Section III-A shows a real *AR.Drone* position control scenario [1], which validates the stability of the proposed algorithm and its efficiency with a maximum computation time of $max(t_c) \approx 1$ms. The resulting position trajectory shows a typical damping of $D \approx \sqrt{2}$ which validates the domination of the position tracking in the *MPC* cost function. The position control is also stabilizing the *AR.Drone* under more complex disturbances (involving Ψ), with the remark that the input limitation constraint handling of u_Ψ has to be adjusted accordingly.

To be able to avoid obstacles, section IV presents a collision avoidance, that models the inequality constraint of keeping a minimum distance to an obstacle with a sigmoid function. The corresponding measurements with the real *AR.Drone* validate this approach [1]. Furthermore they state the computational efficiency in combination with *CMSCGMRES* with a maximum computation time of $max(t_c) \approx 1.6$ms. For the real *AR.Drone CA* experiment, the computational load on a standard computer is with $\frac{max(t_c)}{\Delta t} \approx \frac{1.6\text{ms}}{100\text{ms}} \approx 1.6\%$ very low. As this paper focuses on the control and not the sensing part of the quadrotor, all states in the experiments are measurements via a motion capture system. The low computational load of the proposed controller aims to allow the execution of other computationally expensive algorithms besides.

This refers e.g. to simultaneous localization and mapping (*SLAM*) or obstacle detection via vision, to substitute the motion capture system with onboard sensors in a further step.

Future work will also include the introduction of different constraint handling techniques for the *CMSCGMRES* solver. Subject to these, the stability of the proposed algorithm shall be proven analytically based on [24], [21], [22]. For the application of multiple quadrotors within limited space e.g. smart factories, multi *UAV* control is crucial. This will be addressed by first developing central *MPC* solutions which will then be decentralized in the final step. The presented work will be continued to automatize commercial *UAV* systems, to further reduce development costs for autonomous *UAV* systems and to open this field of research and development to a wider audience in future.

References

[1] Jan Dentler, Videos of AR.Drone CMSCGMRES experiments https://www.dropbox.com/sh/3rsnvr7s8j65id6/AACHvXm0NIxRmpTzIVNSRtkma?dl=0, updated 14.04.2016, Luxembourg, 2016

[2] S. Bouabdallah, P. Murrieri and R. Siegwart Design and control of an indoor micro quadrotor, in Robotics and Automation, 2004. Proceedings. ICRA ’04. 2004 IEEE International Conference on, 2004, pp. 4393-4398 Vol.5. doi: 10.1109/ROBOT.2004.1302409 http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1302409&isnumber=28923

[3] Bouabdallah, S.; Noth, A.; Siegwart, R., PID vs LQ control techniques applied to an indoor micro quadrotor, in Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on , vol.3, no., pp.2451-2456 vol.3, 28 Sept.-2 Oct. 2004, DOI: 10.1109/IROS.2004.1389776, http://ieeexplore.ieee.org.proxy.bnl.lu/stamp/stamp.jsp?tp=&arnumber=1389776&isnumber=30277

[4] S. Bouabdallah and R. Siegwart, Proceedings - IEEE International Conference on Robotics and Automation, ISBN: 078038914X, pp. 2247-2252, 2005

[5] S. Bouabdallah and R. Siegwart, Advances in Unmanned Aerial Vehicles - Design and control of a miniature quadrotor, ISBN: 9781402061134, Springer Netherlands, Intelligent Systems, Control and Automation: Science and Engineering, url = http://link.springer.com/chapter/10.1007/978-1-4020-6114-1_6, volume 33, pp. 171-210, 2007

[6] S. Bouabdallah and R. Siegwart, Full control of a quadrotor, in Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on, DOI: 10.1109/IROS.2007.4399042, pp. 153-158, 2007

[7] J. Dentler, S. Kannan, M. A. Olivares Mendez, H. Voos A modularization approach for nonlinear model predictive control of distributed fast systems, Control and Automation (MED), 2016 24th Mediterranean Conference on, Athens, 2016

[8] P. Corke, Robotics, Vision and Control, ISBN: 9783642201431, edition 73, pp. 572,, 2013

[9] Rw. Beard, Quadrotor dynamics and control, Brigham Young University, pp. 1-47, 2008

[10] T. Hamel and R. Mahony and R. Lozano and J. Ostrowski, Dynamic modelling and configuration stabilisation for an X4-flyer, IFAC World Congress, pp. 200-212, volume 1, 2002

[11] García and Dzul and Lozano and Pégard, Quad Rotorcraft Control, Vision-Based Hovering and Navigation, DOI: 10.1007/978-1-4471-4399-4, ISBN: 978-1-4471-4399-4, ISSN: 1430-9491, pp. 179, 2013

[12] J. M. Selfridge and G. Tao, A multivariable adaptive controller for a quadrotor with guaranteed matching conditions, Proceedings of the American Control Conference, DOI: 10.1109/ACC.2014.6859355, ISBN: 9781479932726, ISSN: 07431619, pp. 26-31, 2014

[13] P. E. I. Pounds, Design, Construction and Control of a Large Quadrotor Micro Air Vehicle, PhD thesis, The Australian National University, 2007

[14] N. Shakev and A.V. Topalov and K. Shiev and O. Kaynak, Stabilizing multiple sliding surface control of quad-rotor rotorcraft, in Control Conference (ASCC), 2013 9th Asian , vol., no., pp.1-6, 23-26 June 2013, DOI: 10.1109/ASCC.2013.6606278, http://ieeexplore.ieee.org.proxy.bnl.lu/stamp/stamp.jsp?tp=&arnumber=6606278&isnumber=6605987

[15] M.A. Olivares-Mendez and S. Kannan and H. Voos, Vision Based Fuzzy Control Approaches for Unmanned Aerial Vehicles, in Proceedings of 16th World Congress of the International Fuzzy Systems Association (IFSA) 9th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT), pp. 711-718, 2015

[16] M.K. Joyo and D. Hazry and S. Faiz Ahmed and M.H. Tanveer and F.A. Warsi and A.T. Hussain, Altitude and horizontal motion control of quadrotor UAV in the presence of air turbulence, in Systems, Process & Control (ICSPC), 2013 IEEE Conference on , vol., no., pp.16-20, 13-15 Dec. 2013, DOI: 10.1109/SPC.2013.6735095, http://ieeexplore.ieee.org.proxy.bnl.lu/stamp/stamp.jsp?tp=&arnumber=6735095&isnumber=6735086

[17] S. Bertrand, J. Marzat, H. Piet-Lahanier, A. Kahn, Y. Rochefort, MPC Strategies for Cooperative Guidance of Autonomous Vehicles, in: Aerospace Lab Journal 2014, Issue 8, pp. 1-18.

[18] R.V. Lopes and P. Santana, Model Predictive Control applied to tracking and attitude stabilization of a VTOL quadrotor aircraft, in 21st International Congress of Mechanical Engineering 2011, Brazil, pp. 176-185

[19] D. Ariens and M. Diehl and H. J. Ferreau and B. Houska and F. Logist and R. Quirynen and M. Vukov, ACADO Toolkit User’s Manual, 2015 http://acado.github.io, Accessed: 2015-09-11

[20] K. Graichen and T. Utz, GRAMPC documentation, 2014, https://www.uni-ulm.de/in/mrm/forschung/regelung-und-optimierung/grampc.html, Accessed: 2015-04-20

[21] M. Diehl and H. Ferreau and N. Haverbeke, Efficient Numerical Methods for Nonlinear MPC and Moving Horizon Estimation, in Nonlinear Model Predictive Control, Magni, Lalo and Raimondo, DavideMartino and Allgöwer, Frank, volume 384, 2009, ISBN: 978-3-642-01093-4 pp. 391-417

[22] M.Diehl, R. Findeisen F. Allgwer, A Stabilizing Real-Time Implementation of Nonlinear Model Predictive Control in Real-Time PDE-Constrained Optimization, ISBN: 978-0-89871-621-4, eISBN: 978-0-89871-893-5 DOI: http://dx.doi.org/10.1137/1.9780898718935.ch2

[23] T. Ohtsuka, symlab: cgmres source code, http://www.symlab.sys.i.kyoto-u.ac.jp/~ohtsuka/code/index.htm, Accessed: 2015-09-4

[24] T. Ohtsuka, A continuation/GMRES method for fast computation of nonlinear receding horizon control, in Automatica, volume 40, 2004, pp. 563-574

[25] H. Seguchi and T. Ohtsuka, Nonlinear receding horizon control of an RC hovercraft, in Proceedings of the International Conference on Control Applications, volume 2, 2002, pp. 1076-1081

[26] H. Seguchi and T. Ohtsuka, Nonlinear receding horizon control of an underactuated hovercraft, in International journal of robust and nonlinear control, volume 13, 2003, pp. 381-398

[27] Y. Soneda and T. Ohtsuka, Nonlinear moving horizon state estimation for a hovercraft with continuation/GMRES method, in Control Applications, 2002. Proceedings of the 2002 International Conference on, volume 2, 2002, pp. 1088-1093

[28] Y. Shimizu, T. Ohtsuka and M. Diehl, Nonlinear receding horizon control of an underactuated hovercraft with a multiple-shooting-based algorithm, 2006 IEEE Conference on Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, Munich, 2006, pp. 603-607, doi: 10.1109/CACSD-CCA-ISIC.2006.4776714 http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4776714&isnumber=4776587

[29] Y. Shimizu and T. Ohtsuka and M. Diehl, A real-time algorithm for nonlinear receding horizon control using multiple shooting and continuation/Krylov method, in International Journal of Robust and Nonlinear Control, volume 19, 209, pp. 919-936

[30] Y. Shimizu, T. and Ohtsuka, A real-time algorithm for nonlinear receding horizon control of descriptor systems in Proceedings of SICE Annual Conference 2010, 2010, pp. 219-222