Hierarchical and Hybrid Model Predictive Control of Quadcopter Air Vehicles

A. Bemporad * **C.A. Pascucci** * **C. Rocchi** *

* *Department of Information Engineering, University of Siena, Italy*
(e-mail: {bemporad,pascuccica,rocchicla}@dii.unisi.it)

Abstract: This paper proposes a hierachical hybrid MPC approach to design feedback control functions for stabilization and autonomous navigation of unmanned air vehicles. After formulating the nonlinear dynamical equations of a "quadcopter" air vehicle, a linear MPC controller is designed to stabilize the vehicle around commanded desired set-points. These are generated at a slower sampling rate by a hybrid MPC controller at the upper control layer, based on a hybrid dynamical model of the UAV and of its surrounding environment, with the overall goal of controlling the vehicle to a target set-point while avoiding obstacles. The performance of the complete hierarchical control scheme is assessed through simulations and visualization in a virtual 3D environment, showing the ability of linear MPC to handle the strong couplings among the dynamical variables of the quadcopter under various torque and angle/position constraints, and the flexibility of hybrid MPC in planning the desired trajectory on-line.

Keywords: Model predictive control, hierarchical control, mixed logical dynamical systems, unmanned air vehicles, obstacle avoidance

1. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) have emerged over the last few years as a very active research area, due to the large variety of missions where they can replace humans in accomplishing complex exploration tasks, especially in dangerous environments, for both military and civilian applications. Several different types of UAVs are available and have been investigated, where each one is most suitable for a particular purpose. Gliders and planes are energy efficient and can be used in long-range applications, but in general they need wide operating spaces. Small helicopters, and in particular *quadcopters*, require more energy for navigation but can operate in limited workspaces, as they can take off and land vertically, and easily hover above targets. An example of quadcopter is depicted in Figure 1.

The ability to navigate in smaller workspaces is paid by an increased complexity in controlling the vehicle (Altug et al., 2005), mainly due to the highly nonlinear and coupled dynamics, and to limitations on actuators and pitch/roll angles. Besides classical PID control strategies, a few other approaches were proposed in the literature for stabilization of quadcopters: nonlinear control (Castillo et al., 2004), LQR (Kivrak, 2006), visual feedback (Altug et al., 2005), and H_∞ (Chen and Huzmezan, 2003). Because of stringent hard constraints on rotor torques and soft constraints on pitch and roll angles (and altitude), in this paper we propose linear Model Predictive Control (MPC) for quadcopter stabilization and offset-free tracking of desired orientations, positions and velocities. MPC is in fact particularly suitable for control of multivariable systems governed by constrained dynamics, as it allows to operate closer to the boundaries imposed by hard constraints, compared to more classical control schemes (Maciejowski, 2002).

Fig. 1. Example of quadcopter: the Silverlit Toys X-UFO

In the context of UAVs, MPC techniques have been mainly applied for formation flight (Dunbar and Murray, 2002; Dunbar, 2001), (Borrelli et al., 2005), (Li and Cassandras, 2006). This represents a higher level of control for autonomous navigation, where reference trajectories are commanded to stabilization algorithms placed at the lower-level. In the context of path planning for obstacles avoidance, several other solutions have been proposed in the literature, such as potential fields (Chuang, 1998), A^* with visibility graphs (Hoffmann et al., 2008; Latombe, 1991), nonlinear trajectory generation (see e.g. the NTG software package developed at Caltech (Dunbar and Murray, 2002), and mixed-integer linear programming (MILP) (Richards and How, 2002). In particular the latter has shown the great flexibility of on-line mixed-integer optimization in real-time trajectory planning of aircrafts, as also reported in (Pallottino et al., 2002) where on-line MILP techniques were proved very effective in handling multiaircraft conflict avoidance problems.

⋆ This work was partially supported by the European Commission under project "WIDE – Decentralized and Wireless Control of Large-scale Systems", contract number FP7-IST-224168.

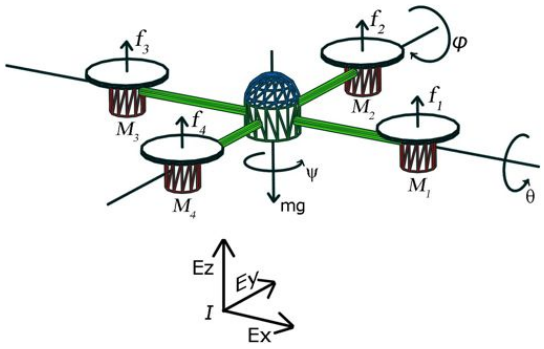

Fig. 2. Quadcopter model

In this paper we propose a two-layer MPC approach to quadcopter stabilization and on-line trajectory generation for autonomous navigation with obstacle avoidance. A linear constrained MPC controller with integral action takes care of the stabilization task. At a higher hierarchical level and lower sampling rate, a hybrid MPC controller generates on line the path to follow to reach a given target position/orientation while avoiding obstacles. Such a hierachical MPC setup can be easily extended to cope with missions involving multiple cooperating (or competing) UAVs.

The paper is organized as follows. In Section 2 we describe a nonlinear model of the quadcopter, whose linearization provides the prediction model for linear MPC design in Section 3 for stabilization under constraints and tracking of generic trajectories. The performance of linear MPC stabilization is assessed in simulation using the FlightGear virtual environment. Then, the higher-level hybrid MPC controller is described in Section 4 for obstacle avoidance. Finally, some conclusions are drawn in Section 5.

2. NONLINEAR QUADCOPTER DYNAMICS

A quadcopter air vehicle can move freely in the three-dimensional space, therefore six degrees of freedom are needed to describe its dynamics, as depicted in Figure 2. We denote by x, y, z the position of the vehicle and by θ, ϕ, ψ its rotations around the Cartesian axes, relative to the "world" frame I. In particular, x and y are the coordinates in the horizontal plane, z is the vertical position, ψ is the yaw angle (rotation around the z-axis), θ is the pitch angle (rotation around the x-axis), and ϕ is the roll angle (rotation around the y-axis). The dynamical model adopted in this paper is mainly based on the model proposed in (Castillo et al., 2004), slightly modified to take into account realistic friction effects.

As described in Figure 2, each of the four motors M_1, M_2, M_3, M_4 generate four thrust forces f_1, f_2, f_3, f_4, and four torques τ_1, τ_2, τ_3, τ_4, respectively, which are adjusted by manipulating the voltages applied to the motors. As a result, three torques τ_θ, τ_ϕ, τ_ψ around their corresponding axes are generated

$$\tau_\theta = (f_2 - f_4)l \tag{1a}$$

$$\tau_\phi = (f_3 - f_1)l \tag{1b}$$

$$\tau_\psi = \sum_{i=1}^{4} \tau_i \tag{1c}$$

where l is the distance between each motor and the center of gravity of the vehicle, as well as a total force $u = f_1 + f_2 + f_3 + f_4$, that allow changing the position and orientation coordinates of the quadcopter freely in the three-dimensional space.

By letting $\tau = [\tau_\theta \ \tau_\phi \ \tau_\psi]'$ be the overall torque vector, $\eta = [\theta \ \phi \ \psi]'$ the angular displacement vector, and J the inertia matrix, the rotational dynamics of the quadcopter are described by

$$\tau = J\ddot{\eta} + J\dot{\eta} - \frac{1}{2}\frac{\partial}{\partial \eta}(\dot{\eta}' J \dot{\eta}) \tag{2}$$

which can be rewritten as

$$\ddot{\eta} = \tilde{\tau} \tag{3}$$

where $\tilde{\tau} = [\tilde{\tau}_\theta \ \tilde{\tau}_\phi \ \tilde{\tau}_\psi]'$ is treated as a new command input. Through rotational transformations between the frame I and the one placed in the center of mass of the quadcopter, we obtain the dynamic equations

$$m\ddot{x} = -u\sin\theta - \beta\dot{x} \tag{4a}$$

$$m\ddot{y} = u\cos\theta\sin\phi - \beta\dot{y} \tag{4b}$$

$$m\ddot{z} = u\cos\theta\cos\phi - mg - \beta\dot{z} \tag{4c}$$

$$\ddot{\theta} = \tilde{\tau}_\theta \tag{4d}$$

$$\ddot{\phi} = \tilde{\tau}_\phi \tag{4e}$$

$$\ddot{\psi} = \tilde{\tau}_\psi \tag{4f}$$

where the damping factor β takes into account realistic friction effects. The nonlinear dynamical model has twelve states (six positions and six velocities) and four inputs (one total force, three torques), largely coupled through the nonlinear relations in (4). In the next section we propose a linear MPC approach to cope with the stabilization of such a multivariable dynamics under constraints on input and state variables.

3. LINEAR MPC FOR STABILIZATION

In order to design a linear MPC controller for the quadcopter air vehicle, we first linearize the nonlinear dynamical model (4) around an equilibrium condition of hovering. The resulting linear continuous-time state-space system is converted to discrete-time with sampling time T_s

$$\begin{cases} \xi_L(k+1) = A\xi_L(k) + Bu_L(k) \\ y_L(k) = \xi_L(k) \end{cases} \tag{5}$$

where $\xi_L(k) = [\theta, \phi, \psi, x, y, z, \dot{\theta}, \dot{\phi}, \dot{\psi}, \dot{x}, \dot{y}, \dot{z}, z_{\mathrm{I}}]' \in \mathbb{R}^{13}$ is the state vector, $u_L(k) = [u, \tilde{\tau}_\theta, \tilde{\tau}_\phi, \tilde{\tau}_\psi]' \in \mathbb{R}^4$ is the input vector, $y_L(k) \in \mathbb{R}^{13}$ is the output vector (that we assume completely measured or estimated), and A, B, C, D are matrices of suitable dimensions obtained by the linearization process. The additional state, $z_{\mathrm{I}} = \int(z - z_d)$ is included to provide integral action on the altitude z, so that offset-free tracking of the desired setpoint z_d is guaranteed in steady-state. The integral action is mainly due to counteract effect of the gravity force acting against the force developed by the collective input u.

The linear MPC formulation of the Model Predictive Control Toolbox for Matlab (Bemporad et al., 2004b) is used here, where the MPC control action at time k is obtained by solving the optimization problem

$$\min_{\substack{\Delta u_L(k|k) \\ \vdots \\ \Delta u_L(m-1+k|k) \\ \epsilon}} \sum_{i=0}^{N_L-1} \left(\sum_{j=1}^{n_y} |w_j^y \left[y_{Lj}(k+i+1|k) - r_{Lj}(k+i+1) \right]|^2 + \sum_{j=1}^{n_u} |w_j^{\Delta u} \Delta u_{Lj}(k+i|k)|^2 + \rho_\epsilon \epsilon^2 \right) \tag{6a}$$

subject to

$$\begin{aligned} & u_{Lj}^{\min} \le u_{Lj}(k+i|k) \le u_{Lj}^{\max}, \ j = 1, \dots, n_u \\ & y_{Lj}^{\min} - \epsilon V_{Lj}^{\mathrm{y,min}} \le y_{Lj}(k+i+1|k) \le y_{Lj}^{\max} + \epsilon V_{Lj}^{\mathrm{y,max}} \\ & \qquad j = 1, \dots, n_y \\ & \Delta u(k+h|k) = 0, \ h = N_{Lu}, \dots, N_L \\ & \epsilon \ge 0 \end{aligned} \tag{6b}$$

for all $i = 0, \dots, N_L - 1$, with respect to the sequence of input increments $\{\Delta u(k|k), \dots, \Delta u(N_{Lu} - 1 + k|k)\}$ and the slack variable ϵ. In (6a) the subscript "$()_j$" denotes the jth component of a vector, "$(k+i|k)$" denotes the value predicted for time $k+i$ based on the information available at time k, $r_L(k)$ is the current sample of the output reference, $V^{\mathrm{y,min}}$, $V^{\mathrm{y,max}}$ are constant vectors with nonnegative entries which represent the designer's concern for relaxing the corresponding output constraint, $n_y = 13$ is the number of outputs, $n_u = 4$ is the number of inputs. The linear MPC controller sets $u(k) = u(k-1) + \Delta u^*(k|k)$, where $\Delta u^*(k|k)$ is the first element of the optimal sequence.

3.1 Linear MPC tuning and validation

The linear MPC controller is tuned according to the following setup. Regarding input variables, we set $u_{Lj}^{\min} = -6$ Nm, $u_{Lj}^{\max} = 6$ Nm, $j = 1, 2, 3$, $u_{L4}^{\min} = -6$ N, $u_{L4}^{\max} = 6$ N, $w_{i,j}^{\Delta u} = 0.1$, $\forall j = 1, \dots, 4$, $i = 0, \dots, N_L - 1$. For output variables we set a lower bound $y_{L6}^{\min} = 0$ on altitude z, and upper and lower bounds $y_{L1-2}^{\max} = -y_{L1-2}^{\min} = \frac{\pi}{12}$ on pitch θ and roll ϕ angles. The output weights are $w_j^y = 1$, $j \in \{4, 5, 11, 12\}$, on x, y, $\dot{x}$, $\dot{y}$, respectively, and $w_j^y = 10$ on the remaining output variables. The chosen set of weights ensures a good trade-off between fast system response and actuation energy. The prediction horizon is $N_L = 20$, the control horizon is $N_{Lu} = 3$, which, together with the choice of weights, allow obtaining a good compromise between tracking performance, robustness, and limited computational complexity. The sampling time of the controller is $T_s = \frac{1}{14}$ s. The remaining parameters $V^{\mathrm{y,min}}$, $V^{\mathrm{y,max}}$, ρ_ϵ are defaulted by the Model Predictive Control Toolbox (Bemporad et al., 2004a).

The closed-loop performance is tested by simulating the nonlinear quadcopter model (4) under the effect of the linear MPC controller (6) using Simulink and the Model Predictive Control Toolbox. Additional blocks were designed to generate reference signals from either the computer keyboard or an external four-axes joystick. Moreover, the numerical signals are connected to an animation block based on FlightGear (Basler et al., 2008) for 3D visual inspection on the regulation performance, connected through a graphical user interface designed for easy human-machine interaction. FlightGear allows one to very intuitively move the quadcopter around the virtual world to any desired direction by sending desired set-point commands directly from the keyboard or joystick, and assess closed-loop performance visually. An animation movie can be retrieved at `http://www.dii.unisi.it/hybrid/aerospace/quadcopter`. Due to different angle representation systems, the following coordinate transformation

$$\begin{bmatrix} \theta' \\ \phi' \\ \psi' \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \theta \\ \phi \\ \psi \end{bmatrix}$$

is used to map the signals (θ, ϕ, ψ) generated by model (4) into the coordinates (θ', ϕ', ψ') defining the angle coordinates of the quadcopter in the virtual environment:

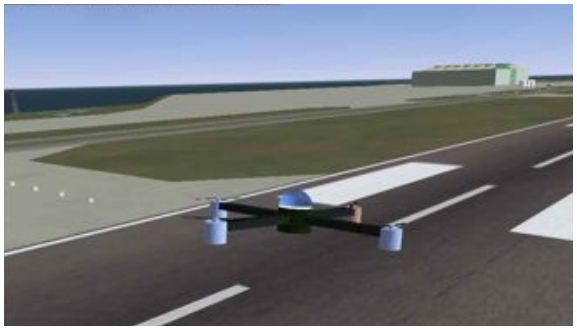

Fig. 3. Visualization of quadcopter dynamics in FlightGear (chase view)

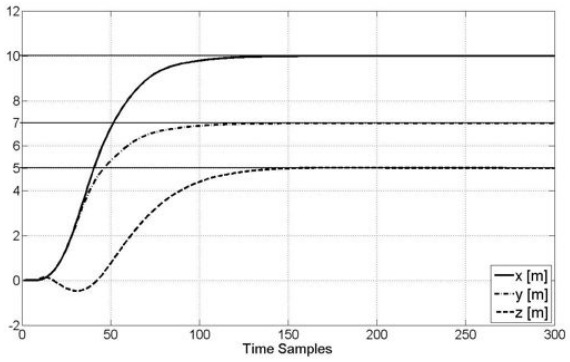

Fig. 4. Linear MPC for tracking generic position references

Figure 4 shows simulation results obtained by tracking a generic reference signal. The corresponding input plots are shown in Figure 5, where the thick solid line represents the collective activity u. Note that u is nonzero at steady-state due to the task of keeping the quadcopter in hovering. The other lines represent, respectively, the actuations on pitch $\tilde{\tau}_\theta$, roll $\tilde{\tau}_\phi$, and yaw $\tilde{\tau}_\psi$ angles.

Note that because of the constraints imposed on θ and ϕ, the nonlinear dynamics of the vehicle is maintained close enough to the linearized model used in the MPC design. As performance is rather satisfactory, the possible use of multiple MPC controllers based on models linearized at different conditions has not been explored here.

The results were obtained on a Core 2 Duo running Matlab R2008a and the MPC Toolbox under MS Windows. The average CPU time to compute the MPC control action is about 13 ms per time step, which is about 1/6 of

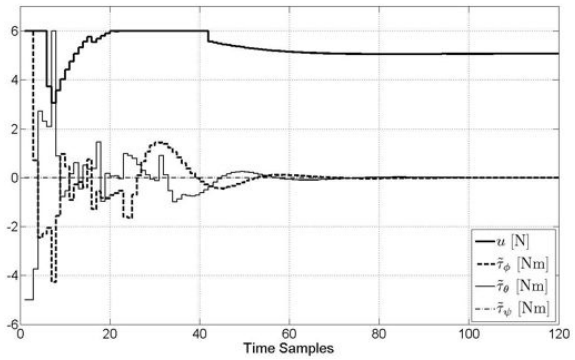

Fig. 5. Linear MPC: commanded inputs

the sampling time T_s. No attempt was done to speed up computations, such as using fast on-line MPC implementations (Wang and Boyd, 2008) or explicit off-line MPC solutions (Bemporad et al., 2002).

4. HYBRID MPC FOR OBSTACLE AVOIDANCE

The linear MPC design developed in Section 3 is quite satisfactory in making the quadcopter air vehicle track references on position/orientation and velocity, despite the strong coupling among states and the presence of constraints (6b).

As shown in Figure 6, we extend the linear MPC design with a hierarchical architecture where at the top layer a hybrid MPC controller commands set-points r_L on positions, angles, and velocities in order to accomplish the main mission, namely reach a given target position (x_t, y_t, z_t) while avoiding obstacles [1]. We assume that target and obstacle positions may be time-varying and not known in advance, so that off-line (optimal) planning would not be possible. Hence, we still rely on model predictive control ideas to generate the set-points (x_d, y_d, z_d) to the linear MPC controller in real-time. The proposed approach consists of constructing an abstract hybrid model of the controlled air vehicle and of the surrounding obstacles, and then use a hybrid MPC strategy for on-line generation of the desired positions (x_d, y_d, z_d) (the remaining output set points in vector r_L are all zero).

The simulation results of Section 3 show that the closed-loop dynamics composed by the quadcopter and the linear MPC controller can be very roughly approximated as a 3-by-3 diagonal linear dynamical system, whose inputs are (x_d, y_d, z_d) and whose outputs are (x, y, z). Accordingly, the dynamics is formulated in discrete-time as

$$\begin{cases} x(k+1) = \alpha_{1x}x(k) + \beta_{1x}(x_d(k) + \Delta x_d(k)) \\ y(k+1) = \alpha_{1y}y(k) + \beta_{1y}(y_d(k) + \Delta y_d(k)) \\ z(k+1) = \alpha_{1z}z(k) + \beta_{1z}(z_d(k) + \Delta z_d(k)) \end{cases} \quad (7)$$

where $\Delta x_d(k)$ is the increment of desired x-position commanded at time $kT_{\rm hyb}$, and $T_{\rm hyb} > T_s$ is the sampling time of the hybrid MPC controller.

Input increments $\Delta x_d(k)$, $\Delta y_d(k)$, $\Delta z_d(k)$ are upper and lower bounded by a quantity Δ

[1] The formulation of this paper can be immediately extended to include the yaw angle ψ as an additional controlled variable. Note that ψ is dynamically decoupled in steady-state from (x, y, z).

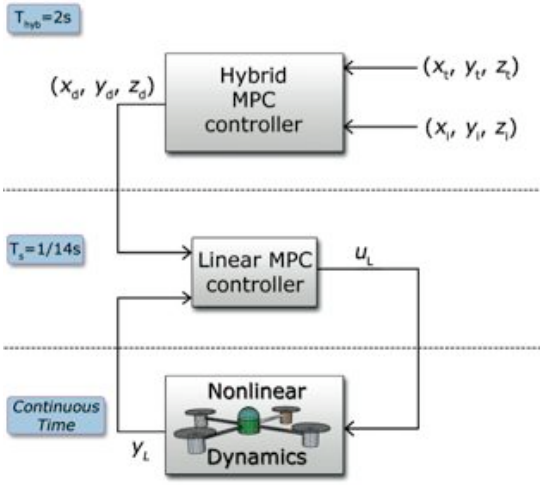

Fig. 6. Hierarchical structure

$$-\Delta \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \leq \begin{bmatrix} \Delta x_d(k) \\ \Delta y_d(k) \\ \Delta z_d(k) \end{bmatrix} \leq \Delta \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad (8)$$

Constraint (8) is a tuning knob of the hybrid MPC controller, as it allows one to directly control the speed of maneuver of the quadcopter by imposing constraints on the reference derivatives $\left\| \begin{bmatrix} \dot{x}_d(t) \\ \dot{y}_d(t) \\ \dot{z}_d(t) \end{bmatrix} \right\|_\infty \leq \Delta \cdot T_{\rm hyb}$.

Obstacles are modeled as polyhedral sets. For minimizing complexity, the ith obstacle, $i = 1, \ldots, M$, is modeled as the tetrahedron

$$A_{\rm obs} k_i \begin{bmatrix} x(k)-x_i(k) \\ y(k)-y_i(k) \\ z(k)-z_i(k) \end{bmatrix} \leq B_{\rm obs} \quad (9)$$

where $A_{\rm obs} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \leq B_{\rm obs}$ is a fixed hyperplane representation of a reference tetrahedron, k_i is a fixed scaling factor, and $\begin{bmatrix} x_i(k) \\ y_i(k) \\ z_i(k) \end{bmatrix}$ is a reference point of the obstacle. In this paper we choose $A_{\rm obs}$, $B_{\rm obs}$ such that the corresponding polyhedron is the convex hull of vectors $\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 5/2 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 5/2 \\ 0 \end{bmatrix}, \begin{bmatrix} 5/6 \\ 5/6 \\ 5/2 \end{bmatrix}$, which makes the reference point $\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$ its vertex with smallest coordinates.

Equation (9) can be rewritten as

$$A_{\rm obs} k_i \begin{bmatrix} x(k) \\ y(k) \\ z(k) \end{bmatrix} \leq C_{\rm obs}(k) \quad (10)$$

where $C_{\rm obs}(k) = B_{\rm obs} + A_{\rm obs} k_i \begin{bmatrix} x_i(k) \\ y_i(k) \\ z_i(k) \end{bmatrix} \in \mathbb{R}^4$ is a quantity that may vary in real-time. Here, it is modeled *in prediction* through the constant dynamics

$$C_{\rm obs}(k+1) = C_{\rm obs}(k) \quad (11)$$

Finally, to represent the obstacle avoidance constraint, define the following binary variables $\delta_{ij} \in \{0, 1\}$, $i = 1, \ldots, M$, $j = 1, \ldots, 4$

$$[\delta_{ij}(k) = 1] \leftrightarrow [A^j_{\rm obs} k_i \begin{bmatrix} x(k) \\ y(k) \\ z(k) \end{bmatrix} \leq C^j_{\rm obs}(k)] \quad (12)$$

where ^{j} denotes the jth row (component) of a matrix (vector). The following logical constraints

$$\bigvee_{j=1}^{4} \neg\delta_{ij}(k) = \text{TRUE}, \ \forall \text{i} = 1, \ldots, \text{M} \qquad (13)$$

impose that at least one linear inequality in (10) is violated, therefore enforcing the quadcopter position $\begin{bmatrix} x(k) \\ y(k) \\ z(k) \end{bmatrix}$ to lie outside each obstacle.

The sampling time T_{hyb} must be chosen large enough to neglect fast transient dynamics, so that the lower and upper MPC designs can be conveniently decoupled. On the other hand, the obstacle avoidance constraint (13) is only imposed at multiples of T_{hyb}, and hence an excessively large sampling time may lead to trajectories that go through the obstacles during intersampling intervals.

The overall hybrid dynamical model is obtained by collecting (7), (11), (12), (13). These are modeled through the modeling language HYSDEL (Torrisi and Bemporad, 2004) and converted automatically by the Hybrid Toolbox (Bemporad, 2003) into a mixed logical dynamical (MLD) system form (Bemporad and Morari, 1999)

$$\xi_H(k+1) = A\xi_H(k) + B_1\Delta u_H(k) \qquad (14a)$$
$$y_H(k) = C\xi_H(k) \qquad (14b)$$
$$E_2\delta(k) \leq E_1\Delta u_H(k) + E_4\xi_H(k) + E_5, \qquad (14c)$$

where $\xi_H(k) = [x(k)\ y(k)\ z(k)\ C_1'(k)\ \ldots C_M'(k)\ x_d(k-1)\ y_d(k-1)\ z_d(k-1)]' \in \mathbb{R}^{6+4M}$ is the state vector, $\Delta u_H(k) = [\Delta x_d(k)\ \Delta y_d(k)\ \Delta z_d(k)]' \in \mathbb{R}^3$ is the input vector, $y_H(k) = [x(k)\ y(k)\ z(k)] \in \mathbb{R}^3$ is the output vector, and $\delta(k) \in \{0,1\}^{4M}$ is the vector of auxiliary binary variables defined in (12). The inequalities (14c) include a big-M representation (Bemporad and Morari, 1999) of (12) and a polyhedral inequality representation of (13). Matrices A, B_1, C, E_1, E_2, E_4, E_5 have suitable dimensions and are generated by the HYSDEL compiler. In order to design a hybrid MPC controller, consider the finite-time optimal control problem

$$\min_{\{\Delta u(k)\}_{k=0}^{N_H-1}} \sum_{j=0}^{N_H-1} (y_H(k+j+1) - y_{Ht})'Q(y_H(k+j+1) - y_{Ht}) + \Delta u_H'(k+j)R\Delta u_H(k+j) \qquad (15a)$$
$$\text{s.t. MLD dynamics (14)} \qquad (15b)$$
$$\text{constraints (8)} \qquad (15c)$$

where N_H is the prediction horizon, $y_{Ht} = [x_t\ y_t\ z_t]'$ is the target destination, and $Q \geq 0$ and $R > 0$ are weight matrices of $\mathbb{R}^{3\times 3}$.

The MLD hybrid dynamics (14) has the advantage of making the optimal control problem (15) solvable by mixed-integer quadratic programming (MIQP). At each sample step k, given the current reference values $y_H(k)$ and the current state $\xi(t)$, Problem (15) is solved to get the first optimal input sample $\Delta u_H^*(k)$, which is commanded as the increment of desired set-point (x_d, y_d, z_d) to the linear MPC controller at the lower hierarchical level.

4.1 Simulation results

To test the behavior of the overall system we cascade the linear MPC controller described in Section 3 with the hybrid MPC controller designed in this section, according to the hierarchical scheme of Figure 6. The simulation

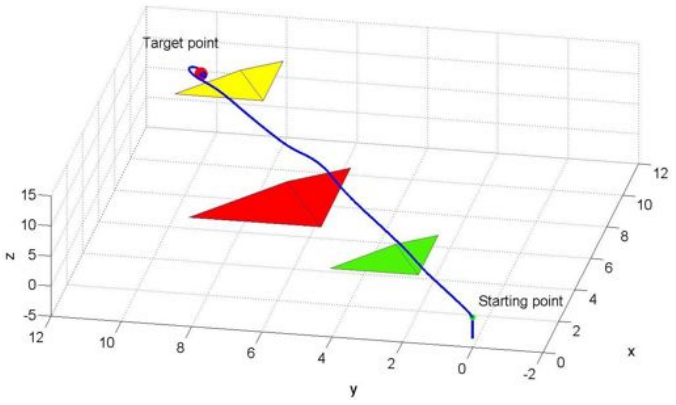

Fig. 7. Obstacle avoidance maneuvers commanded by the hybrid MPC controller

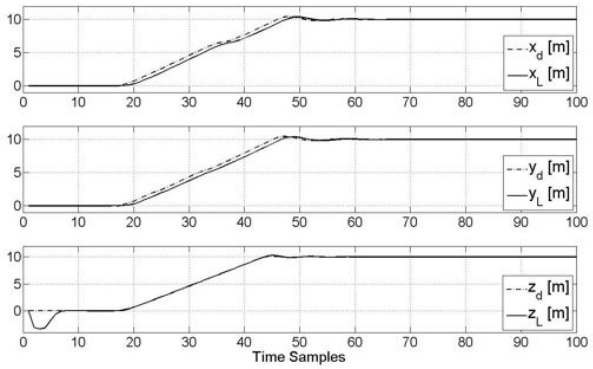

Fig. 8. Position and reference signals

consists of avoiding three obstacles (tetrahedra) of different dimensions, placed along the path between the quadcopter and the target point (see Figure 7). The following parameters are employed: $\alpha_{1x} = \alpha_{1y} = \alpha_{1z} = 0.6$, $\beta_{1x} = \beta_{1y} = \beta_{1z} = 0.4$ for the approximation of the lower level dynamics; N_H=10 (prediction horizon), T_{hyb}=2 s, and $\Delta = \frac{1}{5}T_{hyb}$; $k_1 = k_3 = 1$, $k_2 = \frac{2}{3}$ are the scaling factors for the tetrahedra; $Q = I_{3\times 3}$ and $R = 0.1 \cdot I_{3\times 3}$ are the weight matrices; the initial position is $x(0) = y(0) = z(0)$=0 and the target point is located at $x_t = y_t = z_t$=10.

The overall performance is quite satisfactory: The trajectory generated on-line circumvents the obstacles without collisions (see Figure 7), and finally the quadcopter settles at the target point (see Figure 8). Note that in the first transient the altitude z first drops by about 3 m due to the effect of gravity.

In the simulations we assumed that the positions of the obstacles were known at each sample step. In more realistic applications with several obstacles it may be enough to only know the locations of the obstacles which are closest to the vehicle, for a threefold reason. First, because of the finite-horizon formulation, remote obstacles will not affect the optimal solution, and may be safely ignored to limit the complexity of the optimization model. Second, because of the receding horizon mechanism, the optimal plan is continuously updated, which allows one to change the maneuvers to avoid new obstacles. Third, in a practical context obstacles may be moving in space, and since such dynamics is not modeled here, taking into account remote

obstacles in their current position has a weak significance. The number M of obstacles to be taken into account in the hybrid model clearly depends on the density of the obstacles and the speed of the vehicle. Note that, depending on the sensor system on board of the vehicle, it may be even impossible to measure the position of remote obstacles.

The average CPU time to compute the hybrid MPC action for set-point generation is about 17 ms per time step on the same platform used for linear MPC, using the mixed-integer quadratic programming solver of CPLEX 11.2 (ILOG, Inc., 2008).

5. CONCLUSIONS

In this paper we have shown how model predictive control strategies can be employed for autonomous navigation of unmanned aerial vehicles such as quadcopters. Due to the multivariable and constrained dynamics of such UAVs, a linear MPC design takes care of the nontrivial task of stabilization of the vehicle. Then, a hybrid MPC "pilot" takes care of generating the path to follow in real-time to reach a given target while avoiding obstacles. Compared to off-line planning methods, such a feature of on-line generation of the 3D path to follow is particularly appealing in realistic scenarios where the positions of the target and of the obstacles are not known in advance, but rather acquired (and possibly time-varying) during operations. Moreover, compared to on-line planning methods like potential fields, hybrid MPC offers much more flexibility in specifying constraints, prioritization of tasks, etc. in a very direct and natural way.

The results of this paper can be extended in many ways. First, further improvements on stabilization (such as faster response) could be achieved by extending the nonlinear model with the inclusion of motor/blade dynamics, therefore treating motor voltages as command inputs. Second, the approach can be immediately extended to coordination of multiple UAVs, where each of them is stabilized by a local linear MPC controller (or any other type of stabilizing controller), and decentralized hybrid MPC is used at the tactical level to accomplish the mission. Due to the flexibility of specifying set-points and constraints of hybrid MPC, several different mission scenarios can be easily formulated and solved.

REFERENCES

Altug, E., Ostrowski, J., and Taylor, C. (2005). Control of a quadrotor helicopter using dual camera visual feedback. *The International Journal of Robotics Research*, 24(5), 329.

Basler, M., Spott, M., et al. (2008). *The FlightGear Manual*. `http://www.flightgear.org/Docs/getstart/getstart.html`.

Bemporad, A. (2003). *Hybrid Toolbox – User's Guide*. `http://www.dii.unisi.it/hybrid/toolbox`.

Bemporad, A. and Morari, M. (1999). Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3), 407–427.

Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1), 3–20.

Bemporad, A., Morari, M., and Ricker, N. (2004a). *Model Predictive Control Toolbox for Matlab – User's Guide*. The Mathworks, Inc. `http://www.mathworks.com/access/helpdesk/help/toolbox/mpc/`.

Bemporad, A., Ricker, N., and Owen, J. (2004b). Model predictive control – New tools for design and evaluation. 5622–5627. Boston, MA.

Borrelli, F., Keviczky, T., Fregene, K., and Balas, G. (2005). Decentralized receding horizon control of cooperative vechicle formations. In *Proc. 44th IEEE Conf. on Decision and Control and European Control Conf.*, 3955–3960. Sevilla, Spain.

Castillo, P., Dzul, A., and Lozano, R. (2004). Real-time stabilization and tracking of a four-rotor mini rotorcraft. *IEEE Transactions on Control Systems Technology*, 12(4), 510–516.

Chen, M. and Huzmezan, M. (2003). A combined mbpc/2 dof h8 controller for a quad rotor UAV. In *AIAA Atmospheric Flight Mechanics Conference and Exhibit*.

Chuang, J. (1998). Potential-based modeling of three-dimensional workspace for obstacle avoidance. *IEEE Transactions on Robotics and Automation*, 14(5), 778–785.

Dunbar, W. (2001). Model predictive control: extension to coordinated multi-vehicle formations and real-time implementation. Technical report, CDS Technical Memo CIT-CDS 01-016, California Institute of Technology, Pasadena, CA 91125.

Dunbar, W. and Murray, R. (2002). Model predictive control of coordinated multi-vehicle formations. In *IEEE Conference on Decision and Control*, volume 4, 4631–4636.

Hoffmann, G., Waslander, S., and Tomlin, C. (2008). Quadrotor helicopter trajectory tracking control. In *Proc. AIAA Guidance, Navigation, and Control Conf., Honolulu, HI*.

ILOG, Inc. (2008). *CPLEX 11.2 User Manual*. Gentilly Cedex, France.

Kivrak, A. (2006). Design of control systems for a quadrotor flight vehicle equipped with inertial sensors.

Latombe, J. (1991). *Robot motion planning*. Kluwer academic publishers.

Li, W. and Cassandras, C. (2006). Centralized and distributed cooperative receding horizon control of autonomous vehicle missions. *Mathematical and computer modelling*, 43(9-10), 1208–1228.

Maciejowski, J. (2002). *Predictive control with constraints*. Prentice Hall.

Pallottino, L., Feron, E., and Bicchi, A. (2002). Conflict resolution problems for air traffic management systems-solved with mixed integer programming. *IEEE Transactions on Intelligent Transportation Systems*, 3(1), 3–11.

Richards, A. and How, J. (2002). Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In *American Control Conference, 2002. Proceedings of the 2002*, volume 3.

Torrisi, F. and Bemporad, A. (2004). HYSDEL — A tool for generating computational hybrid models. *IEEE Trans. Contr. Systems Technology*, 12(2), 235–249.

Wang, W. and Boyd, S. (2008). Fast model predictive control using online optimization. 6974–6979.