2013 European Control Conference (ECC)
July 17-19, 2013, Zürich, Switzerland.

Fast Model Predictive Control of Miniature Helicopters

Konstantin Kunz, Stephan M. Huck, and Tyler H. Summers

Abstract— **Model Predictive Control (MPC) is a well-developed and widely-used control design method, in which the control input is computed by solving an optimization problem at every sampling period. Traditionally, MPC has been associated with control of slow processes, with sampling times in the seconds/minutes/hours range, because an optimization problem must be solved online. However, dramatic increases in computing power and recent developments in code generation for convex optimization, which tailor to specific optimization problem structure, are allowing the use of MPC in fast processes, with sampling times in the millisecond range. In this paper, a MPC control design for a miniature remote-controlled coaxial helicopter is developed and experimentally validated. The nonlinear dynamic behavior of the helicopter was identified, simplified and approximated by a Linear Time Varying (LTV) model. A custom convex optimization solver was generated for the specific MPC problem structure and integrated into a controller, which was tested in simulation and implemented on a hardware testbed. A performance analysis shows that the MPC approach performs better than a tuned Proportional Integral Differential (PID) controller.**

I. Introduction

Model Predictive Control (MPC) [4] is a well-developed and widely-used control design method partly due to its ability to handle multivariable systems with state and input constraints. At each sampling instant, a model of the system is used to optimize the predicted behavior of the system over a finite horizon. The first input of the solution to this open-loop optimization is applied to the system, and the optimization is repeated whenever new measurements are received. This optimization problem is convex when the dynamic model is linear and the stage cost function and constraints are convex. Traditionally, MPC has been associated with control of slow processes, with sampling times in the seconds/minutes/hours range, because an optimization problem must be solved online. However, dramatic increases in computing power and recent developments in code generation for convex optimization are allowing the use of MPC in fast processes.

Fast model predictive control methods fall into two categories: (1) so-called explicit MPC and (2) MPC with online optimization. In explicit MPC, the control law can be explicitly computed offline when the dynamics are linear, the stage cost is convex quadratic or linear, and the constraints are polytopes [3] [2], but this method is limited to very small problems due to the complexity of the explicit solution.

The authors are with the Automatic Control Laboratory, Department of Information Technology and Electrical Engineering, ETH Zurich, Zurich 8092, Switzerland. Email: kkunz@ee.ethz.ch {huck,tsummers}@control.ee.ethz.ch. This research was partially supported by the NanoTera-SSSCT grant under the project NetCam

When an optimization problem must be solved online, recent developments in fast convex optimization algorithms and code generators, which exploit specific problem structure, are enabling the use of MPC for systems with fast sampling times [14], [7], down to the millisecond range, depending on problem size and processor capability.

There has been significant recent interest from various fields on autonomous helicopters due to a wide variety of possible civilian and military applications ranging from aerial imagery to search and rescue. In particular, various advanced control methodologies have been proposed, including some limited work on MPC. Liu at al developed a nonlinear explicit MPC approach for the automatic control of small scale helicopters [13]. An approximate solution of the nonlinear MPC optimization problem is computed offline and applied to the helicopter online by function evaluation. A disturbance estimator is implemented to reduce errors resulting from model mismatch, wind gusts, etc. Experimental results include trajectory tracking. Zhou at al [21] utilize a hierarchical inner-outer loop control scheme for helicopter control. The outer loop consists of a MPC controller that generates the desired pitch and roll angles from a precomputed trajectory and the desired main rotor thrust. The inner loop controls the attitude angles with a back-stepping algorithm. However, the method was only tested in simulation and not experimentally validated. Joelianto at al [12] present a sliding mode MPC in which the helicopter model changes during control in order to approximate the helicopter behavior better for different flight regimes; however, their method was also not experimentally validated. Du et al. [8] present a hierarchical inner-outer loop controller, with a PID controller in the outer loop and a MPC controller in the inner loop to stabilize desired attitude angles. Experimental results are limited to hover, which had moderate errors (100 mm) in horizontal dimensions.

The main contribution of the present paper is the development and experimental validation of an online model predictive control design using a full 10-dimensional miniature autonomous helicopter model for both hover and trajectory tracking. This work is part of the ETH Zurich RCopterX project (http://www.rcopterx.ethz.ch), which studies advanced control and estimation techniques for small autonomous helicopters. The nonlinear dynamic behavior of the helicopter was identified, simplified and approximated by a Linear Time Varying (LTV) model. Feasible desired trajectories are generated based on differential flatness of the model. A custom convex optimization solver was generated for the specific MPC problem structure and integrated into a controller, which was tested in simulation and implemented on a hardware testbed for both hover and trajectory tracking.

978-3-033-03962-9/©2013 EUCA

A performance analysis shows that the MPC approach performs better than a tuned Proportional Integral Differential (PID) controller.

The rest of the paper is organized as follow. In Section II, the nonlinear continuous-time helicopter model is presented. The flat output formulation of this model, that is necessary to generate feasible trajectories, will be introduced in a next step. Next, a Linear Time Varying (LTV) approximation of the continuous-time model is introduced that is used in the formulation of the MPC optimization problem, presented next. Section III presents simulation results, experimental validation on a hardware testbed, and an experimental performance comparison to a tuned PID controller. Finally, Section IV provides concluding remarks.

II. Fast Model Predictive Control

A. Helicopter Model

The helicopter considered in this work is a small-scale remote controlled coaxial helicopter. The dynamics can be described by the Newton-Euler laws of mechanics for single rigid bodies. A full state description can be found for example in [17]. However, observations of the dynamics of the helicopter suggest that a simplified version can be used. By assuming that the pitch or roll inputs directly act on the translational accelerations, the pitch and roll angular states of the helicopter body can be omitted. The resulting simplified helicopter model is described by

$$\begin{aligned}
\dot{x}_I &= \cos(\Psi)\dot{x}_B - \sin(\Psi)\dot{y}_B, \quad \dot{y}_I = \sin(\Psi)\dot{x}_B + \cos(\Psi)\dot{y}_B \\
\dot{z}_I &= \dot{z}_B, \quad \dot{\Psi} = \dot{\Psi} \\
\ddot{x}_B &= b_x u_x + k_x \dot{x}_B + \dot{\Psi}\dot{y}_B, \quad \ddot{y}_B = b_y u_y + k_y \dot{y}_B - \dot{\Psi}\dot{x}_B \\
\ddot{z}_B &= b_z u_z - g, \quad \ddot{\Psi} = b_\Psi u_\Psi + k_\Psi \dot{\Psi} \\
\dot{x}_{int} &= k_i(x_I - x_{ref}), \quad \dot{y}_{int} = k_i(y_I - y_{ref}),
\end{aligned} \tag{1}$$

where x_I, y_I, z_I denote the position of the helicopter in the inertial frame, $\dot{x}_B$, $\dot{y}_B$, $\dot{z}_B$ the velocities of the helicopter in the body frame, Ψ the yaw heading angle and $\dot{\Psi}$ its rotational velocity in yaw. The control inputs for pitch, roll, thrust and yaw are denoted by u_x, u_y, u_z, u_Ψ. The model is augmented by two integral states x_{int}, y_{int} to reduce steady-state error in the respective dimension. The inertial frame of reference is fixed to the measurement space with a North-West-Up configuration. The body frame coordinate system is fixed to the helicopter Center of Gravity (CoG). The x-axis points towards the nose, the y-axis towards the port side and the z-axis along the rotor axis. The identification of the model parameters $b_x, b_y, b_z, b_\Psi, k_x, k_y, k_\Psi$ and k_i is described later in Section III.

B. Trajectory Generation

The aim of our approach is to design a MPC controller that is able to steer the helicopter along trajectories, following both position and orientation. Here we are interested in predefined trajectories which are computed offline. Consider a continuous-time reference trajectory $[\mathbf{x_r}(t), \mathbf{u_r}(t)]$ describing the states and inputs of a system over time. The objective of the controller is to track this trajectory as close as possible. However, this is only achievable if the generated trajectory is feasible with respect to the dynamics, i.e., feasible trajectories are trajectories $[\mathbf{x_r}(t), \mathbf{u_r}(t)]$ that satisfy the differential equation of the process dynamics:

$$\begin{aligned}
\dot{\mathbf{x}}_\mathbf{r}(t) &= \mathbf{f}(\mathbf{x_r}(t), \mathbf{u_r}(t)) \\
\mathbf{x_r}(t) &\in \mathcal{X}, \quad \mathbf{u_r}(t) \in \mathcal{U}.
\end{aligned}$$

The function $f(\cdot,\cdot)$ describes the previously identified helicopter dynamics (1) and $\mathcal{X}$ and $\mathcal{U}$ are constraint sets for the states and inputs, respectively.

Generating such trajectories for nonlinear systems is a nontrivial task. Murray [16] shows that the generation of such feasible trajectories is possible if the system is differentially flat [11]. The general idea is to find mapping between the original states and so called *flat outputs* such that all states and inputs can be determined from these outputs without integration. Although there is no general rule to derive these outputs, it turns out that this approach can be applied to the nonlinear helicopter model used in our application. Consider the state vector of the helicopter model without integral states

$$\mathbf{x} = \begin{bmatrix} x_I & y_I & z_I & \Psi & \dot{x}_B & \dot{y}_B & \dot{z}_B & \dot{\Psi} \end{bmatrix}^T \tag{2}$$

and the input vector $\mathbf{u} = \begin{bmatrix} u_x & u_y & u_z & u_\Psi \end{bmatrix}^T$. Note that the integral states are omitted because the reference value for integral state is always zero. We select four flat outputs z_i given by

$$z_1 = x_I, \quad z_2 = y_I, \quad z_3 = z_I, \quad z_4 = \Psi. \tag{3}$$

Expressing the remaining states by the flat outputs and their derivatives yields

$$\begin{aligned}
\dot{x}_B &= \cos(z_4)\dot{z}_1 + \sin(z_4)\dot{z}_2 \\
\dot{y}_B &= -\sin(z_4)\dot{z}_1 + \cos(z_4)\dot{z}_2 \\
\dot{z}_B &= \dot{z}_3, \quad \dot{\Psi} = \dot{z}_4.
\end{aligned} \tag{4}$$

The next step is to describe the inputs as functions of the flat outputs and their derivatives:

$$u_z = \frac{1}{b_z}(\ddot{z} - g) = \frac{1}{b_z}(\ddot{z}_3 - g) \tag{5}$$

$$u_\Psi = \frac{1}{b_\Psi}(\ddot{\Psi} - k_\Psi \dot{\Psi}) = \frac{1}{b_\Psi}(\ddot{z}_4 - k_\Psi \dot{z}_4) \tag{6}$$

$$\begin{aligned}
u_x &= \frac{1}{b_x}(\ddot{x}_B - k_x \dot{x}_B - \dot{\Psi}\dot{y}_B) \\
&= \frac{1}{b_x}(\cos(z_4)[\ddot{z}_1 - k_x \dot{z}_1] + \sin(z_4)[\ddot{z}_2 - k_x \dot{z}_2])
\end{aligned} \tag{7}$$

$$\begin{aligned}
u_y &= \frac{1}{b_y}(\ddot{y}_B - k_y \dot{y}_B + \dot{\Psi}\dot{x}_B) \\
&= \frac{1}{b_y}(\cos(z_4)[\ddot{z}_2 - k_y \dot{z}_2] + \sin(z_4)[-\ddot{z}_1 + k_y \dot{z}_1]).
\end{aligned} \tag{8}$$

Finally, the functions for the states and inputs can be rewritten in compact matrix form as

$$\mathbf{x} = \mathbf{h}(\mathbf{z}, \dot{\mathbf{z}}), \quad \mathbf{u} = \mathbf{i}(\mathbf{z}, \dot{\mathbf{z}}, \ddot{\mathbf{z}}). \tag{9}$$

We can now generate smooth point-to-point trajectories for the flat outputs according to the method presented in [16].

The states and input of the reference trajectory can now be computed by a simple evaluation the functions (9).

C. Linear Time Varying Model

In general a MPC optimization problem requires a discrete-time representation of the plant to compute the necessary state predictions. In fast MPC applications, the model must be linear so that the problem can be cast as a convex quadratic program and solved by available fast convex optimizers obtained from generators like CVXGEN [14] or FORCES [7]. However, the helicopter model described in Section II-A is nonlinear. Therefore a suitable linear discrete-time approximation of the nonlinear model over the MPC prediction horizon is needed. A simple linearization around the initial state is not sufficient since the helicopter state may change significantly over time along a desired reference trajectory.

Falcone et al. show that nonlinear systems can be approximated by a Linear Time Varying (LTV) model [10]. We will briefly restate the basic theory and include two modifications of the original derivation. The first one addresses the fact that our initial model is a continuous-time nonlinear model whereas Falcone begins with a discrete-time nonlinear model. Hence the linearization step for the LTV approximation is followed by a discretization step. Second, we consider a non-constant input trajectory for the linearization step. By including more knowledge about actual possible inputs to the system a better approximation of the system dynamics can be obtained. Further, based on this variable input sequence, we extend the LTV approximation by introducing a "warm-start"-like method for model linearization.

Consider the following nonlinear continuous-time system

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \tag{10}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state vector, $\mathbf{u} \in \mathbb{R}^m$ is the input vector. The function $f(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ describes the state dynamics. Let $\mathbf{u}$ be a piecewise constant function in time given by

$$\begin{aligned} \mathbf{u}(t) \quad &= \mathbf{u}_\mathrm{k} \\ \text{with} \quad &\mathrm{k}\mathrm{T_s} \leq t < (\mathrm{k}+1)\mathrm{T_s}, \quad \mathrm{k} \in \mathbb{Z}^+, \end{aligned} \tag{11}$$

where $\mathrm{T_s}$ is the sampling period of the system. We will refer to $\hat{\mathbf{u}}(t)$ as the nominal inputs and $\hat{\mathbf{x}}(t)$ as the nominal state trajectory generated by applying the piecewise constant inputs $\hat{\mathbf{u}}(t)$ to the nonlinear system (10). Starting at some time $t_0 = \mathrm{k_0 T_s}$, $\mathrm{k_0} \in \mathbb{Z}^+$ with initial condition $\mathbf{x}(\mathrm{k_0 T_s}) = \mathbf{x_0}$, the nominal state and input trajectories are sampled with

$$\hat{\mathbf{x}}_\mathrm{k} = \hat{\mathbf{x}}(\mathrm{kT_s}), \quad \mathrm{k} = \{\mathrm{k_0}, ..., \mathrm{k_0}+p+1\} \tag{12}$$

$$\hat{\mathbf{u}}_\mathrm{k} = \hat{\mathbf{u}}(\mathrm{kT_s}), \quad \mathrm{k} = \{\mathrm{k_0}, ..., \mathrm{k_0}+p\}. \tag{13}$$

Note already that the states trajectory needs to be sampled $p+1$ times. This is requires to generate a LTV approximation for a horizon of p steps as can be seen later from (21).

The system (10) is then linearized at p time instances of the nominal trajectory leading to the following set of linearized dynamics

$$\delta\dot{\mathbf{x}}(t) = \mathbf{A}_\mathrm{k}\delta\mathbf{x}(t) + \mathbf{B}_\mathrm{k}\delta\mathbf{u}(t) \tag{14}$$

$$\mathbf{A}_\mathrm{k} = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_\mathrm{k}, \hat{\mathbf{u}}_\mathrm{k}}, \quad \mathbf{B}_\mathrm{k} = \left.\frac{\partial f}{\partial \mathbf{u}}\right|_{\hat{\mathbf{x}}_\mathrm{k}, \hat{\mathbf{u}}_\mathrm{k}} \tag{15}$$

$$\delta\mathbf{x}(t) = \mathbf{x}(t) - \hat{\mathbf{x}}_\mathrm{k}, \quad \delta\mathbf{u}(t) = \mathbf{u}(t) - \hat{\mathbf{u}}_\mathrm{k} \tag{16}$$

with $\mathrm{kT_s} \leq t < (\mathrm{k}+1)\mathrm{T_s}, \quad \mathrm{k} = \{\mathrm{k_0}, ..., \mathrm{k_0}+p\}$. The discretized dynamics of the continuous time linearizations (14) are

$$\delta\mathbf{x}_{\mathrm{k}+1} = \mathbf{A}_{\mathrm{d,k}}\delta\mathbf{x}_\mathrm{k} + \mathbf{B}_{\mathrm{d,k}}\delta\mathbf{u}_\mathrm{k}, \tag{17}$$

where $A_{d,k}$ and $B_{d,k}$ are Euler discretization of A_k, B_k and

$$\delta\mathbf{x}_\mathrm{k} = \mathbf{x}(\mathrm{kT_s}) - \hat{\mathbf{x}}_\mathrm{k} = \mathbf{x}_\mathrm{k} - \hat{\mathbf{x}}_\mathrm{k} \tag{18}$$

$$\delta\mathbf{u}_\mathrm{k} = \mathbf{u}(\mathrm{kT_s}) - \hat{\mathbf{u}}_\mathrm{k} = \mathbf{u}_\mathrm{k} - \hat{\mathbf{u}}_\mathrm{k} \tag{19}$$

$$\mathrm{k} = \{\mathrm{k_0}, ..., \mathrm{k_0}+p\}. \tag{20}$$

The discretized system (17) is written in a δ formulation, but can also be stated as a full state model

$$\mathbf{x}_{\mathrm{k}+1} - \hat{\mathbf{x}}_{\mathrm{k}+1} = \mathbf{A}_{\mathrm{d,k}}(\mathbf{x}_\mathrm{k} - \hat{\mathbf{x}}_\mathrm{k}) + \mathbf{B}_{\mathrm{d,k}}(\mathbf{u}_\mathrm{k} - \hat{\mathbf{u}}_\mathrm{k}) \tag{21}$$

$$\Leftrightarrow \mathbf{x}_{\mathrm{k}+1} = \mathbf{A}_{\mathrm{d,k}}\mathbf{x}_\mathrm{k} + \mathbf{B}_{\mathrm{d,k}}\mathbf{u}_\mathrm{k} + \hat{\mathbf{x}}_{\mathrm{k}+1} - \mathbf{A}_\mathrm{k}\hat{\mathbf{x}}_\mathrm{k} - \mathbf{B}_\mathrm{k}\hat{\mathbf{u}}_\mathrm{k}$$

$$= \mathbf{A}_{\mathrm{d,k}}\mathbf{x}_\mathrm{k} + \mathbf{B}_{\mathrm{d,k}}\mathbf{u}_\mathrm{k} + \mathbf{d}_\mathrm{k} \tag{22}$$

$$\mathrm{k} = \{\mathrm{k_0} \ldots \mathrm{k_0}+p\}. \tag{23}$$

This discrete-time linear approximation (21)-(23) of the nonlinear continuous-time system along the trajectory $\hat{\mathbf{x}}_\mathrm{k}$ and $\hat{\mathbf{u}}_\mathrm{k}$ is called the Linear Time Varying (LTV) model of (10) for a time period of $t \in [\mathrm{k_0}T_s \ldots (\mathrm{k_0}+p)\mathrm{T_s}]$.

The MPC controller described in Section II-D solves a problem at every sampling time $t \geq t_0$ for a prediction horizon of p steps. Since we chose to use a non-constant input for the LTV linearization, an input trajectory needs to be selected. For the initial LTV-approximation at time t_0, the nominal input trajectory (13) can be obtained by sampling the input trajectory generated by (9). In the consecutive approximations one can make use of the previously computed control sequences of the MPC optimization. The control trajectory at time t that minimizes the stage cost function over the horizon p is

$$\mathbf{U}^*_{t\to t+p|t} = \left[\mathbf{u}^*_{t|t}, \ldots, \mathbf{u}^*_{t+p|t}\right], \tag{24}$$

where the $u^*_{i|t}$ denotes the optimized input for the i-th step of the prediction horizon computed at time instance t. From this control sequence only the first control input $\mathbf{u}^*_{t,t}$ is applied to the system. The other control inputs will be used as nominal input trajectory $\hat{\mathbf{U}}_{t+1\to t+1+p}$ for the next MPC problem at time $t+1$. This procedure "warm-starts" the LTV model approximation. The nominal input trajectory for the next problem will be

$$\hat{\mathbf{U}}_{t+1\to t+1+p|t+1} = \left[\hat{\mathbf{u}}_{t+1|t+1} \quad \ldots \quad \hat{\mathbf{u}}_{t+p|t+1} \quad \hat{\mathbf{u}}_{t+1+p|t+1}\right]$$

$$= \left[\mathbf{u}^*_{t+1|t} \quad \ldots \quad \mathbf{u}^*_{t+p|t} \quad \mathbf{u}^*_{t+p|t}\right]. \tag{25}$$

In (25) the p-th value for the nominal input is not available from the preceding optimization step, hence we chose set it to the previous input value as depicted in Figure 1.

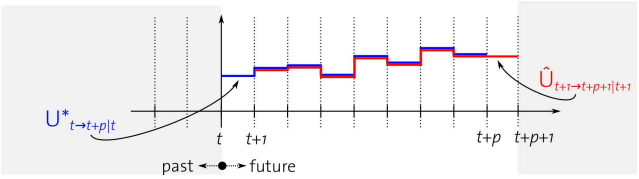

Fig. 1. The "warm-start" of the LTV nominal input trajectory.

D. MPC Problem Formulation

The Model Predictive Control problem with the LTV model approximation of the helicopter dynamics is formulated as the optimization problem (26), which is solved at every time instance t.

$$\begin{aligned}
\underset{\mathbf{U}_{t\to t+p|t}}{\text{minimize}} \quad & J_t(\mathbf{X}_{t\to t+p+1|t}, \mathbf{U}_{t\to t+p|t}, \dots \\
& \mathbf{X}_{\text{ref},t\to t+p+1}, \mathbf{U}_{\text{ref},t\to t+p}) \\
& = \sum_{k=t}^{t+p} (\Delta\mathbf{x}_{\text{k}}^T \mathbf{Q} \Delta\mathbf{x}_{\text{k}} + \Delta\mathbf{u}_{\text{k}}^T \mathbf{R} \Delta\mathbf{x}_{\text{k}}) \\
& \quad + \Delta\mathbf{x}_{t+p+1}^T \mathbf{Q}_f \Delta\mathbf{x}_{t+p+1} \\
\text{subject to} \quad & \mathbf{x}_{\text{k}+1|t} = \mathbf{A}_{\text{d,k}|t}\mathbf{x}_{\text{k}|t} + \mathbf{B}_{\text{d,k}|t}\mathbf{u}_{\text{k}|t} + \mathbf{d}_{\text{k}|t} \\
& \mathbf{d}_{\text{k}|t} = \hat{\mathbf{x}}_{\text{k}+1|t} - \mathbf{A}_{\text{d,k}|t}\hat{\mathbf{x}}_{\text{k}|t} - \mathbf{B}_{\text{d,k}|t}\hat{\mathbf{u}}_{\text{k}|t} \\
& \mathbf{u}_{\text{k}|t} \in \mathcal{U}_k, \quad \text{k} = t, \dots, t+p \\
& \mathbf{x}_{\text{k}|t} \in \mathcal{X}_k, \quad \text{k} = t, \dots, t+p+1,
\end{aligned} \tag{26}$$

with the predicted state trajectory $\mathbf{X}_{t\to t+p+1|t} = \left[\mathbf{x}_{t|t}, \dots, \mathbf{x}_{t+p+1|t}\right]$, the planned input trajectory $\mathbf{U}_{t\to t+p|t} = \left[\mathbf{u}_{t|t}, \dots, \mathbf{u}_{t+p|t}\right]$, the state reference trajectory $\mathbf{X}_{\text{ref},t\to t+p+1} = \left[\mathbf{x}_{\text{ref},t}, \dots, \mathbf{x}_{\text{ref},t+p+1}\right]$ and the input reference trajectory $\mathbf{U}_{\text{ref},t\to t+p+1} = \left[\mathbf{u}_{\text{ref},t}, \dots, \mathbf{u}_{\text{ref},t+p}\right]$. Note that $\Delta\mathbf{x}_{\text{k}} = \mathbf{x}_{\text{k}|t} - \mathbf{x}_{\text{ref,k}}$, $\Delta\mathbf{u}_{\text{k}} = \mathbf{u}_{\text{k}|t} - \mathbf{u}_{\text{ref,k}}$ are the differences between the predicted states and the reference trajectory. The states and inputs are constrained to lie in the convex sets $\mathcal{X}_k$ and $\mathcal{U}_k$, respectively. The state error weighting matrix is $\mathbf{Q} \geq \mathbf{0}$, the input error weighting matrix is $\mathbf{R} > \mathbf{0}$, and the weight on the final state error is $\mathbf{Q}_f \geq \mathbf{0}$. The weighting matrices are positive semi-definite so that the stage cost is convex.

In general a terminal cost $\mathbf{Q}_f$ and a terminal set $\Delta x_{t+p+1} \in \mathcal{X}_{p+1}$ can be chosen to guarantee asymptotic stability. Mayne et al. [15] show that choosing a final cost function $\Delta x_{t+p+1}^T \mathbf{Q}_f \Delta x_{t+p+1}$ that is a good approximation of the infinite horizon cost function $J_{0\to\text{inf}}$ is essential for asymptotic stability. It can be chosen e.g. as the control Lyapunov function satisfying the Lyapunov equation of the linearized system as shown in [5]. Chen et al. [5] suggest choosing $\mathcal{X}_{p+1}$ to be a sufficiently small level set of the terminal cost function to guarantee stability.

The presented MPC problem formulation (26) is used as structure to generate a fast optimization problem solver.

III. Results

The results achieved by the fast MPC approach for miniature helicopters are now presented. The controller was tested in simulation and experimentally validated on a testbed.

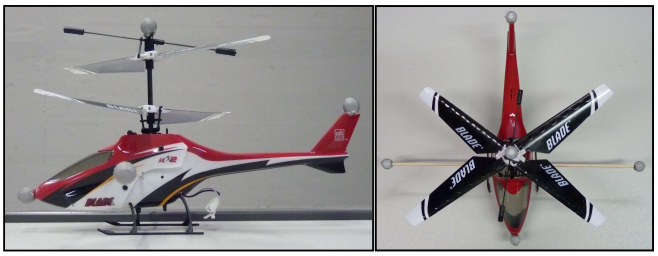

Fig. 2. Blade mCX2 micro-coaxial helicopters

Experimental Setting

The helicopter used in this application is a 28-gram Blade mCX2 micro-coaxial helicopter as shown in Figure 2. It is approximately 20 cm long with a main rotor diameter of ca. 19 cm. The experimental flight area for the helicopter is equipped with a 3D-motion tracking system. It consists of four Vicon Bonita Cameras [19] using infrared diodes that emit light which is reflected by special passive reflectors placed onto the helicopter (see Figure 2). The data of the four cameras is evaluated by the Vicon tracker software [20] in order to compute the position and orientation of the helicopter in the room. This data is input to the Coaga custom helicopter control software [1] framework. Coaga is a Object-Oriented control framework in C++ developed by the RCopterX project group at ETH Zurich [9]. It runs on a stand-alone Desktop PC allowing fast computations and enough resources to perform various control schemes. Control algorithms, such as the fast MPC algorithm, can easily be integrated into the Coaga framework. Coaga includes a Kalman filter for velocity estimations as well as the communication modules necessary to send the control signals to the helicopter via a remote control. The testbed sampling period is 20 ms due to communication channel limitations imposed by the helicopter remote control.

Model Identification

The model linearizations $\mathbf{A}_{\text{k}}$ and $\mathbf{B}_{\text{k}}$ used for the LTV model (14) are further simplified to avoid an unwanted actuation of the yaw input. The linearization of the dynamic equations for $\ddot{x}_B$, $\ddot{y}_B$, x_i and y_i in (1) lead to a cross-coupling of the x and y components via the rotational states Ψ and $\dot{\Psi}$. We observed experimentally that the helicopter model did not predict the actual behavior of the system well. When the coupling terms were removed, the resulting closed loop motion of the system was significantly improved. Therefore we used the simplified version for all of our experimental implementations.

The parameters of the model were identified by fitting the suggested acceleration functions (1) to measured input step responses of the helicopter which were obtained by Daellenbach and Semeraro [6]. We used a Least-Squares method to fit the parameters to the data and subsequently

tuned them by hand. The parameters obtained are:

$$\begin{aligned} b_x &= 2.0, \quad k_x = -0.5, \\ b_y &= 2.1, \quad k_y = -0.4, \\ b_\Psi &= 111.0, \quad k_\Psi = -5.0, \quad b_z = 18. \end{aligned} \tag{27}$$

The integral parameter $k_i = 2$ was chosen for both integral states and obtained by manual tuning.

MPC Controller Settings

The states are constrained by their physical limits that were determined in [6]. In our case, upper and lower limits were given only for the velocities $[\dot{x}_B, \dot{y}_B, \dot{z}_B, \dot{\Psi}]$ by

$$\begin{aligned} \overline{\mathbf{x}} &= [3\frac{\text{m}}{\text{s}},\ 3\frac{\text{m}}{\text{s}},\ 2\frac{\text{m}}{\text{s}}, 25\frac{\text{rad}}{\text{s}}]^T \\ \underline{\mathbf{x}} &= [-3\frac{\text{m}}{\text{s}},\ -3\frac{\text{m}}{\text{s}},\ -2\frac{\text{m}}{\text{s}}, -25\frac{\text{rad}}{\text{s}}]^T. \end{aligned} \tag{28}$$

For the inputs the box constraints were derived from the actuator control limits and stability considerations as

$$\overline{\mathbf{u}} = [1,\ 1,\ 0.4,\ 1]^T, \quad \underline{\mathbf{u}} = [-1,\ -1,\ -0.4,\ 0]^T. \tag{29}$$

The state and input weighting matrices used in the implementation are

$$\begin{aligned} \mathbf{Q} &= \mathbf{Q}_f = diag(50; 50; 5; 10; 3; 3; 1; 2; 15; 15) \\ \mathbf{R} &= diag(2; 2; 2; 2), \end{aligned} \tag{30}$$

where $diag(\cdot)$ denotes the entries on the diagonal of a matrix with appropriate dimension. The convex optimization solver used both in simulation and on the real system was generated with CVXGEN [14]. The MPC control prediction horizon was chosen to be $p = 18$. The solve times of the generated solver when used in the testbed varied between 5 ms and 7 ms depending on the problem data. This is a well lower than the system sampling period of 20 ms.

Simulation

The fast MPC controller was tested in simulation prior to being applied to the helicopter in the testbed. Figure 3 shows the helicopter position and heading while tracking a circle trajectory. The helicopter tracks the heading angle very well and the deviations in x_I and y_I are also very small (approximately 2.5 cm).

Hardware implementation

The MPC controller can stabilize the helicopter in a hover flight as can be seen in Figure 4. The states x_I and y_I are shown with their inputs u_x and u_y respectively. The largest deviation in x_I is approximately 5.5 cm and in y_I it is 4 cm. The control signals have an offset because of the integral action due to a constant disturbance.

As already shown in simulation the fast MPC controller can track trajectories such as circles. The trajectory tracking performance of the MPC controller on the testbed can be seen in Figure 5. The horizontal position as well as heading are tracked very well. The maximum deviation in x_I is approximately 4.0 cm and in x_I approximately 8.5 cm. Several other trajectories have been tracked successfully such as pirouettes, ferris-wheels and pirouettes during circle flight.

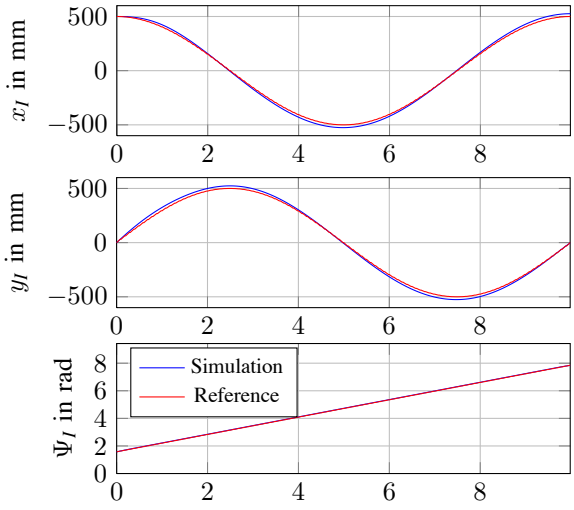

Fig. 3. Simulation of the helicopter tracking a circle trajectory while being controlled by the fast MPC controller.

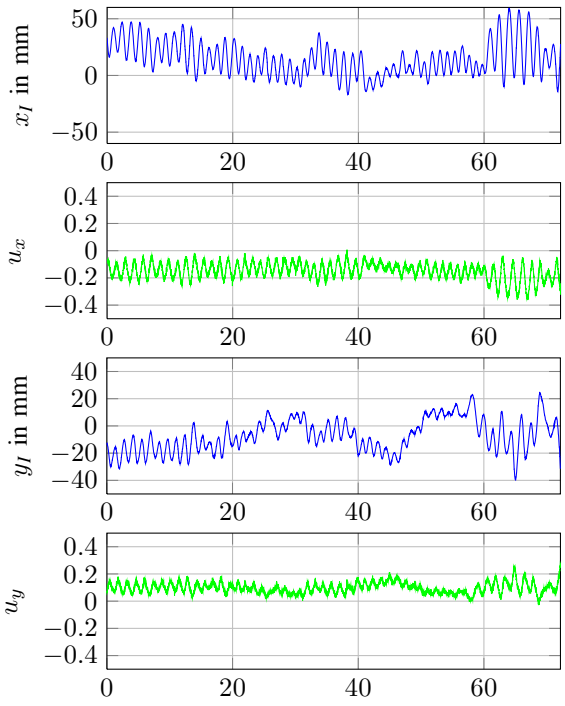

Fig. 4. The horizontal helicopter CoG position during hover flight in the testbed.

Comparing Controller Performance

We compared the performance of the MPC controller to a tuned Proportional-Integral-Differential (PID) controller also implemented on the testbed. Absolute performance comparison is of course difficult because the performance of each method depends highly on adjustable parameters (stage cost weights for MPC and gains for PID). Nevertheless, we attempted to tune both controllers to achieve good performance. The PID controller was tuned by applying the Ziegler-Nichols method and subsequent manual fine-tuning. The MPC controller weights were tuned by hand. The

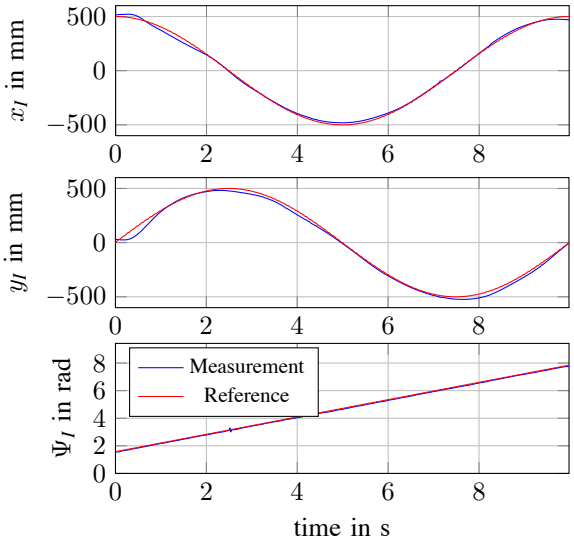

Fig. 5. The horizontal helicopter CoG position and the heading angle during trajectory tracking of a circle on the testbed.

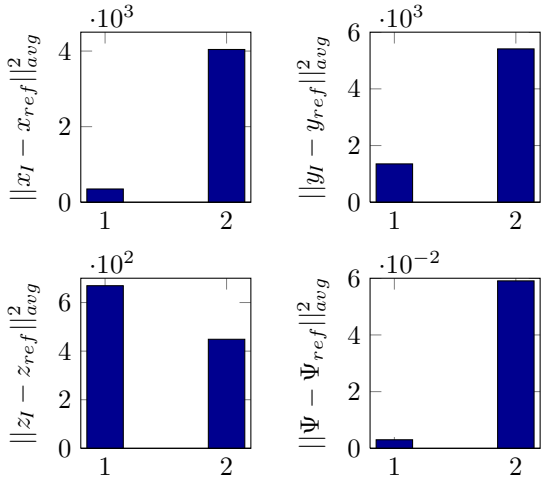

Fig. 6. The average tracking error of the (1) MPC and (2) PID controller flying a circle trajectory in position and heading.

tracking performance is quantified by the average squared-error of the individual states over multiple experiments. The MPC controller performs significantly better in x_I, y_I, and Ψ_I (though the Ψ_I errors are negligible in both cases). In z_I the average PID tracking error is smaller than that of the MPC controller because there is no integral state in z included into the model that could compensate for a constant model mismatch.

IV. Conclusion

In conclusion, we developed and experimentally validated an online model predictive controller for a miniature autonomous helicopter in both hover and trajectory tracking. A nonlinear helicopter model was identified and used to generate feasible trajectories for the helicopter by finding flat outputs of the model. The MPC problem was formulated using a LTV model approximation of the nonlinear model. The controller was tested in simulation and implemented on the helicopter testbed. The performance of both hover and trajectory tracking are very good, and compared favorably to a tuned PID controller. Future work includes identifying and utilizing more detailed models and working with multiple helicopters simultaneously, where one could consider using a distributed MPC scheme such as that described in [18].

References

[1] J. Amstuz, R. Bernhard, T. Wellerdieck, and F. Wermelinger. Realisation of multi-agent support. Semester Thesis, Automatic Control Lab ETH Zurich, 2012.

[2] A. Bemporad, F. Borrelli., M. Morari, et al. Model predictive control based on linear programming˜ the explicit solution. *IEEE Transactions on Automatic Control*, 47(12):1974–1985, 2002.

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

[4] E.F. Camacho and C. Bordons. *Model Predictive Control*. Springer, 2nd edition, 2007.

[5] H. Chen and F. Allgoewer. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica*, 34(10):1205 – 1217, 1998.

[6] D. Daellenbach and V. Semeraro. System identification design. Semester Thesis, Automatic Control Lab ETH Zurich, 2012.

[7] A. Domahidi, A. Zgraggen, M.N. Zeilinger, M. Morari, and C.N. Jones. Efficient Interior Point Methods for Multistage Problems Arising in Receding Horizon Control. In *IEEE Conference on Decision and Control*, Grand Wailea Maui, HI, USA, December 2012.

[8] J. Du, K. Kondak, M. Bernardand Y. Zhang, T. Lu, and G. Hommel. Model predictive control for a small scale unmanned helicopter. *International Journal of Advanced Robotic Systems*, 2008.

[9] S.M. Huck et al. Rcopterx project. Website, 2012. Available online at http://www.rcopterx.ethz.ch/; visited on October 4th 2012.

[10] P. Falcone, M. Tufo, F. Borrelli, J. Asgari, and H.E. Tseng. A linear time varying model predictive control approach to the integrated vehicle dynamics control problem in autonomous systems. In *Decision and Control, 2007 46th IEEE Conference on*, pages 2980 –2985, dec. 2007.

[11] M. Fliess, J. Levine, P. Martin, and Rouchon P. On differentially flat nonlinear systems. *Comptes rendus de l'Acadmie des sciences. Srie 1, Mathmatique*, 1992.

[12] E. Joelianto, E.M. Sumarjono, A. Budiyono, and D.R. Penggalih. Model predictive control for autonomous unmanned helicopters. *Aircraft Engineering and Aerospace Technology*, 83:375 – 387, 2011.

[13] C. Liu, W.-H. Chen, and J. Andrews. Tracking control of small-scale helicopters using explicit nonlinear mpc augmented with disturbance observers. *Control Engineering Practice*, 20(3):258 – 268, 2012.

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

[15] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789 – 814, 2000.

[16] R.M. Murray. *Optimization Based Control*, chapter Trajectory Generation and Tracking. 2010.

[17] D. Schafroth, C. Bermes, S. Bouabdallah, and R. Siegwart. Modeling, system identification and robust control of a coaxial micro helicopter. *Control Engineering Practice*, 18(7):700 – 711, 2010.

[18] T. H Summers and J. Lygeros. Distributed model predictive consensus via the alternating direction method of multipliers. In *50th Allerton Conference on Communication, Control, and Computation*, Monticello, IL, USA, 2012.

[19] Vicon Motion Systems. Bonita camera specs. Website, 2012. Available online at http://www.vicon.com/products/bonita-features.html; visited on September 2th 2012.

[20] Vicon Motion Systems. Vicon tracker. Website, 2012. Available online at http://www.vicon.com/products/vicontracker.html; visited on September 2th 2012.

[21] H. Zhou, H. Pei, and Y. He. Hierarchical control of a small unmanned helicopter using mpc and backstepping. In *30th Chinese Control Conference (CCC)*, pages 147 –151, july 2011.