Fast Nonlinear Model Predictive Control for Very-Small Aerial Vehicles

Tiago P. Nascimento, Member, IEEE, and Martin Saska, Member, IEEE

Abstract—Highly dynamic systems such as Micro Multirotor Aerial Vehicles (Micro-MAVs) require control approaches that enable safe operation where extreme limitations in embedded systems, such as energy, processing capability and memory, are present. Nonlinear model predictive control (NMPC) approaches can respect operational constraints in a safe manner. However, they are typically challenging to implement using embedded computers on-board of Micro-MAVs. Implementations of classic NMPC approaches rely on high-performance computers. In this work, we propose a fast nonlinear model predictive control approach that ensures the stabilization and control of Micro Multirotor Aerial Vehicles (Micro-MAVs). This aerial robotic system uses a low processing power board that relies solely on on-board sensors to localize itself, which makes it suitable for experiments in GPS-denied environments. The proposed approach has been verified in numerical simulations using processing capabilities that are available on Micro-MAVs.

I. INTRODUCTION

Unmanned multirotor aerial vehicles (MAVs) have been studied extensively in recent years [1], [2]. The mobility and capacity of the MAV to perform tasks autonomously are the main reasons for the high number of studies in this area. There is a need to test these vehicles regarding limitations such as low computational power and underactuation [3], making MAVs ideal testbeds to deal with the problem of controlling highly dynamic mechanical systems. Within the control loop of these control systems, there are the attitude controller and the position controller.

The inner loop controller, which points the MAV in the desired direction, is the attitude control. Due to the under-actuation characteristics of the MAV, the attitude controller is responsible for the orientation control of the MAV on all three Euler angles (roll $\phi$, pitch $\theta$ and yaw $\psi$). It is also responsible for controlling the thrust of the MAV ($-F_T$). The output of the attitude control is sent to a mixing-of-motors algorithm, which enables the adaptability of the control system for N number of motors. Furthermore, some researchers have proposed fault-tolerant control approaches [4], [5]. Fault-tolerant control approaches usually act on the control signals and their respective number of motors in the attitude control.

Position control is placed in the outer loop of the control system and has as an output the three Euler angles as references to the attitude controller. It is the MAV controller that enables it to move in a desired trajectory [6]. It is used, among other reasons, to track a desired trajectory. Trajectory tracking problems are usually solved by designing controllers that ensures the feasibility of the tracking of a predetermined trajectories by the vehicles. While tracking the trajectory, the controller minimizes the trajectory error. The drawback of this approach is that the vehicle dynamics exhibit complex nonlinearities and significant uncertainties.

The controller attempts to make the outputs catch up with the time-parameterized desired outputs. This may lead to closed-loop performance difficulties and to excessively large control signals. One way to deal with this problem is by using Nonlinear Model Predictive Controllers (NMPC). In classic NMPC, more complex modeling increases the computational cost. In contrast, robust NMPC approaches usually use learning-based techniques. These robust approaches either need an off-line training phase or increase the computational cost.

A review describing nonlinear model predictive controllers applied to trajectory tracking is provided by Nascimento et al. [7]. In this survey, we focus on nonholonomic mobile robots with respect to solving two problems: feedback stabilization, and motion planning control systems when using model predictive control (MPC) approaches. The MPC problem has its origin in the late seventies [8]. According to Findeisen and Allgöwer [9], the model predictive control problem is formulated as solving, on-line, a finite horizon open-loop optimal control problem subject to system dynamics and constraints involving states and controls. This on-line problem-solving characteristic of NMPC controllers brings up another issue when its application in embedded systems is taken into account: the fast-tracking solution. High-speed and low-power/memory implementations of nonlinear MPC (NMPC) are relatively unexplored but critical problems. Since the underlying optimal control problem in NMPC is typically not convex, local solutions via Newton-type optimization algorithms such as sequential quadratic programming (SQP) are effective [10].

To the best of our knowledge, very few embedded NMPC implementations have exhibited the potential for operations under severe resource constraints, e.g., a need for ultra-fast (in the order of nanoseconds) control updates or very low-memory implementation. Thus, our main contribution is to propose a nonlinear model predictive controller approach that can be embedded into a low-processing board with limited memory, and that is able to solve the trajectory tracking
problem of a Micro-MAV, while increasing the controller accuracy in a desired path. This is achieved by proposing modified optimized cost function, MAV model and optimizer in the formulation of a fast NMPC that minimizes the distance between the robot position and the desired global path coordinate.

II. NONLINEAR MODEL PREDICTIVE CONTROL

Our proposed control approach assumes that each reference coordinate sent by the path planner is the coordinate vector of a tracked virtual target. Our proposed NMPC possesses modifications to increase the controller’s nonlinearities, and the controller’s accuracy (i.e. it is possible to apply the cost function penalization weights in different trajectories maintaining a low steady-state error without retuning the controller). Therefore, to summarize our approach, we have proposed an NMPC that:

- uses a simplified dynamical model of the MAV to predict its behavior;
- tracks the Manhattan distance between the reference trajectory coordinate and the robot. Differing from some works, we do not use different penalization weights ($\lambda_t$) for different trajectories [11]. This is achieved due to the loss of dependency on the trajectory equations;
- handles the variation on the control outputs by using the 1-norm in the control effort penalization term;
- uses RPROP algorithm to deal with the nonlinearities introduced by the use of the 1-norm and as a fast convergent optimizer;
- has all its parameters optimized to achieve fast convergence and a lower computational cost controller that can be embedded in the robot’s board.

In contrast to the classical NMPC approach, some works do not present a sub-block to plan the trajectory from the robot’s current coordinate to the $N_p$-ahead coordinate to better track the trajectory error [12], [13]. Thus, with the position given by the observation of the target, this sub-block has lost its importance. The proposed NMPC is divided into three sub-blocks:

- **Optimizer** - the Resilient Propagation (RPROP) algorithm [14];
- **Predictor** - a dynamic MAV motion prediction model, and a target prediction model.
- **Cost Function (C.F.)** - a norm-1 based cost function.

Our proposed nonlinear model predictive controller (NMPC) is presented in Fig. 1. At an instant $k$, the MAV sends its pose (state) $\hat{R}_k = \begin{bmatrix} x_k & y_k & z_k & \phi_k & \theta_k & \psi_k \end{bmatrix}^T$ to the NMPC to be used by the predictor sub-block. The predictor sub-block also receives the pose of the target in the world frame $T_p$. The desired reference angles in the world frame $T_v$ are also received, and are used as an initial guess in each control loop. This sub-block predicts the robot state ($\hat{R}_k$) evolution for $N_p$ steps (prediction horizons) through the prediction model sub-block. For each prediction, it calculates a cost value through the cost function (C.F) sub-block $J_{k+j,k}$ with $j = 1,...,N_p$, and sends it to the optimizer sub-block. The optimizer sub-block then performs its calculations and provides to the prediction sub-block the control input $\hat{u}_{k+1,k}$ in a limited control horizon $i = 0,1,...,N_c$. To minimize the cost function, the predictor sub-block of the NMPC estimates the evolution of the MAV behavior, as well as the behavior of the target. This iterative minimization process is repeated in a cyclic fashion. Finally, the control output in the first step $u_{k,k} = u_k^* = \begin{bmatrix} \phi_k^* & \theta_k^* \end{bmatrix}^T$ is sent to the robot.

A. Micro Aerial Vehicle Modeling

The prediction model is essential to predict the behavior of the system, taking the global given positions as positions of a virtual target. In this section we present the model of a MAV. In this work we present the model of a quadrotor, a MAV with four propellers, as shown in Fig. 2. We will follow the same notation presented by L’Afflitto et al. [3].

In this model, let consider $\mathbb{R}$ as a set of real numbers (where $\mathbb{R}^n$ and $\mathbb{R}^{n \times m}$ are the $n \times 1$ real column vectors and the $n \times m$ matrices, respectively). We assume also that $\mathbb{I} = [O; X, Y, Z]$ is the orthogonal, inertial world reference frame, with some origin $O$. We also consider that $\mathbb{J} = A: x(t), y(t), z(t), t \geq t_0$, is the robot’s robot frame, which is orthogonal and centered at some point $A$. Let us assume that the MAV’s center of mass is a point $C$ w.r.t. $A$, which in turn is where the robot’s reference frame is centered. Therefore, the position of the quadrotor is given by $r_C : [t_0, \infty) \rightarrow \mathbb{R}^3$. This is assumed because it is not easy to identify the MAV center of mass. The center of mass can vary in time in a generalized model, either when the robot is caring a load or when the center of mass moves due to the displacement of a part of the MAV. Thus, with $a, b \in \mathbb{R}^3$, where $a = [a_1, a_2, a_3]^T$, the cross product of $a$ and $b$ is $a \times b$, in which

$$a \times b = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}.$$  

Note now that $\mathbb{I}$ and $\mathbb{J}$, the two reference frames, form two orthogonal bases in $\mathbb{R}^3$. A vector $a \in \mathbb{R}^3$ is denoted by $a$ if it is expressed in $\mathbb{I}$, but if $a$ is expressed in $\mathbb{J}$, then no superscript will be used. As shown in Fig. 2, $\mathbb{I}F_g = mg$ gives the weight of the MAV of mass $m$, where $g$ is the gravitational acceleration. Finally, the matrix of inertia of each propeller w.r.t. $A$ is $I_p \in \mathbb{R}^{3 \times 3}$, and the matrix of inertia is $I \in \mathbb{R}^{3 \times 3}$. The position of point $A$ w.r.t. the origin of the inertial reference frame $\mathbb{J}$ is $r_A : [t_0, \infty) \rightarrow \mathbb{R}^3$, and the velocity of $A$ w.r.t. $\mathbb{I}$ is $v_A : [t_0, \infty) \rightarrow \mathbb{R}^3$.

Therefore, the attitude of the MAV frame $\mathbb{I}$ with respect to the world frame $\mathbb{I}$ is captured by the Euler angles (roll, pitch and yaw), using the rotation 3-2-1 [3]. We also denote by $\phi, \psi : [t_0, \infty) \rightarrow [0,2\pi]$ the roll and yaw angles, respectively, $\theta : [t_0, \infty) \rightarrow (-\pi/2, \pi/2)$ the pitch angle, and the angular velocity of $\mathbb{J}$ with respect to $\mathbb{I}$ by

524
\[ R \subseteq \{0, 2\pi\} \times (\frac{-\pi}{2}, \frac{\pi}{2}) \times (\frac{-\pi}{2}, \frac{\pi}{2}) \]

The force along the Z-axis of the body frame of the MAV.

Furthermore, Th. 1.7 in [15] gives the rotational kinematics as follows:

\[
\begin{bmatrix}
\dot{\phi}(t) \\
\dot{\theta}(t) \\
\dot{\psi}(t)
\end{bmatrix} = \Gamma(\phi(t), \theta(t)) \Omega_A(t),
\begin{bmatrix}
\phi(t_0) \\
\theta(t_0) \\
\psi(t_0)
\end{bmatrix} = \begin{bmatrix}
\phi_0 \\
\theta_0 \\
\psi_0
\end{bmatrix},
\tag{4}
\]

where \( \Gamma(\phi(t), \theta(t)) \) is invertible, since \( \theta \in (-\frac{\pi}{2}, \frac{\pi}{2}) \) (of [15], pp. 18-19), and

\[
\Gamma(\phi(t), \theta(t)) \triangleq \begin{bmatrix}
1 & \sin \phi \tan \theta & \cos \phi \tan \theta \\
0 & \cos \phi & -\sin \phi \\
0 & \sin \phi \sec \theta & \cos \phi \sec \theta
\end{bmatrix},
\tag{5}
\]

\((\phi, \theta) \in [0, 2\pi) \times (-\frac{\pi}{2}, \frac{\pi}{2}) \)

For the purpose of simplification, works often assume that the MAV has a constant mass over time [16]. Such realistic cases can be found when the MAV is not deploying any payload and is battery-operated. This is the case of micro-MAVs. According to Mohammadi and L’Affitto [16], considering that the mass of the MAV is constant, this will give us the translational dynamics as follows:

\[
F_g(\phi(t), \theta(t)) = \mathbf{m} \ddot{v}_A(t) + \mathbf{r}_C(t) + \mathbf{r}_r(t) + \mathbf{r}_D(t) + \mathbf{r}_L(t),
\]

\[
F_g(\phi(0), \theta(0)) = \mathbf{m} \ddot{v}_A(0) + \mathbf{r}_C(0) + \mathbf{r}_r(0) + \mathbf{r}_D(0) + \mathbf{r}_L(0),
\tag{6}
\]

where \( F_T(t) = [0, 0, u_1(t)]^T \) denoting the thrust force \(^1\), and where the weight of the quadrotor is \( F_g(\phi(t), \theta(t)) \), and is given by

\[ F_g(\phi(t), \theta(t)) = \mathbf{m} \mathbf{g} \mathbf{e}_z \sin \phi \cos \theta, \]

\[ F_g(\phi(0), \theta(0)) = \mathbf{m} \mathbf{g} \mathbf{e}_z \sin \phi_0 \cos \theta_0. \]

\(^1\)The force along the Z-axis of the body frame of the MAV.
\[
\begin{align*}
F_g(\phi, \theta) &= mg[-\sin \theta, \cos \theta \sin \phi, \cos \theta \cos \phi]^T \\
(\phi, \theta) &\in [0, 2\pi) \times (-\frac{\pi}{2}, \frac{\pi}{2}),
\end{align*}
\]
where the aerodynamic forces acting on the MAV are \( F: [t_0, \infty) \rightarrow \mathbb{R}^3. \)

Let \( r_{Ax} = [r_{x_k}, r_{y_k}, r_{z_k}]^T \) and \( r_{Dx} = [r_{x_k}, r_{y_k}]^T, \) \( k \geq k_0. \) According to [17], let us also assume that \( \phi_k, k \geq k_0, \) and \( \theta_k \) are sufficiently small, and let us set \( r_{C_k} = 0 \) and neglect the aerodynamic forces. In this case, from equation (6) we have that
\[
\begin{bmatrix}
\dot{r}_{x_k} \\
\dot{r}_{y_k}
\end{bmatrix} = \begin{bmatrix}
\frac{2}{m} \sin \psi_k & -\cos \psi_k \\
\cos \psi_k & \sin \psi_k
\end{bmatrix} \begin{bmatrix}
\dot{\phi}_k \\
\dot{\theta}_k
\end{bmatrix}.
\]

Through equation (8) we have that
\[
\begin{bmatrix}
\dot{r}_{x_k} \\
\dot{r}_{y_k}
\end{bmatrix} = \begin{bmatrix}
\dot{r}_{x_{k_0}} \\
\dot{r}_{y_{k_0}}
\end{bmatrix} + \tau \cdot \begin{bmatrix}
\dot{r}_{x_k} \\
\dot{r}_{y_k}
\end{bmatrix} + \frac{\tau}{2} \begin{bmatrix}
\dot{r}_{x_k} \\
\dot{r}_{y_k}
\end{bmatrix}.
\]

Finally, the evolution of the trajectory has to be predicted as well. To do so, we assume that the virtual target travels at a constant velocity. Thus, the position \( iT_p_k \) and velocity \( iT_v_k \) of the virtual target (the given positions and velocities of the trajectory) in the world frame at an instant \( k \) is defined as
\[
\begin{align*}
iT_p_k &= [x_{ref_k}, y_{ref_k}]^T, \\
iT_v_k &= [v_{x_{ref_k}}, v_{y_{ref_k}}]^T,
\end{align*}
\]
where \( x_{ref_k} = x_{ref_k-1} + \tau (v_{x_{ref_k}}), \quad y_{ref_k} = y_{ref_k-1} + \tau (v_{y_{ref_k}}). \)

B. The Cost Function

Besides the prediction model and the optimization algorithm, a key point in our proposal is the cost function. In our proposed approach we use a modified cost function to minimize the distance between the robot position and the reference coordinates. This modification increases the nonlinearity of the system. To maintain stabilization we use the 1-norm in the control effort term of the modified cost function [8].

The final cost function is then the sum of two main terms. The first term (12a) is the penalization of the Manhattan distance between the reference trajectory coordinate and the robot. The second term (12b) penalizes the control effort. In this last function, the variation in the output control signal is penalized instead of its absolute value. Penalizing the output control signal would create a steady-state error at non-zero velocities. The final cost function (12) is a composition of only two terms. Taking into account the two terms previously described with their weights, the resulting cost function is as follows:

\[
\begin{align*}
J(\tilde{R}_p, U) &= \sum_{i=1}^{N_p} \lambda_1 |r_{Dk+i} - 1 T_p_k| + \\
\sum_{i=1}^{N_c} \lambda_2 |\Delta U_{k+i-1}|,
\end{align*}
\]

where \( \text{abs}(\cdot) \) denotes the 1-norm for vector arguments and the absolute value for scalars.

III. RESULTS

Several simulations were performed to validate the modified NMPC controller. However, some comments should be made here:

- In our simulations, we used a Micro multirotor aerial vehicle (Micro-MAV) model (Figure 2) using MatLab/Simulink software. The parameters we use are provided by Andrade [18] in the MatLab model. Their system considers all modules of the real MAV, including but not limited to trajectory generation, sensor fusion, optical flow, a nonlinear model of the MAV, a PD control block (based on the work of Pounds et al. [19]) and a log block. Furthermore, the simulation used the standard Bogacki-Shampine method as numerical solver;
- In our simulations, the desired angular speed is 0.25 rad/s;
- The comparison was performed with the PD position controller based on Pounds et al. [19] with \( K_p = 0.24 \) and \( K_d = 0.1; \)
- For our proposed modified NMPC we obtained the controller’s gains through the tuning method by Nascimento et al. [20], as seen in Table I. Nevertheless, due to the high nonlinearity, we cannot guarantee the optimality of the controller’s gains [8].

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Simulation Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \lambda_1 )</td>
<td>2250</td>
</tr>
<tr>
<td>( \lambda_2 )</td>
<td>1</td>
</tr>
<tr>
<td>( N_p )</td>
<td>10</td>
</tr>
<tr>
<td>( N_c )</td>
<td>2</td>
</tr>
<tr>
<td>( I_{max} )</td>
<td>5</td>
</tr>
<tr>
<td>( c )</td>
<td>0.05</td>
</tr>
<tr>
<td>( U_{step} )</td>
<td>0.01</td>
</tr>
<tr>
<td>( \eta^+ )</td>
<td>1.2</td>
</tr>
<tr>
<td>( \eta^- )</td>
<td>0.8</td>
</tr>
<tr>
<td>( \Delta_{max} )</td>
<td>0.1</td>
</tr>
<tr>
<td>( \Delta_{min} )</td>
<td>0.000001</td>
</tr>
</tbody>
</table>

We performed these simulations to observe two features of our proposed approach. The first feature is the feasibility of controlling a Micro-MAV with the embedding constraints imposed by the simulation model of the real controller board. The second feature is to show that this controller, although having a higher computational cost than a PD controller, achieves a better result in trajectory tracking. The simulations
Fig. 3: Simulation using the Parrot Mambo Micro Multirotor Aerial Vehicle performing a figure-of-eight trajectory.

that were performed have a simulation time of 100 seconds. The trajectory is a figure-of-eight shaped trajectory with
radius = 2m and total length = 4m. The robot starts from the initial pose $X_0 = [0, 0, 0]^T$ (m, m, rad). Under tracking, the update time step of the Modified NMPC controller for the Micro-MAV Parrot Mambo was 0.01 seconds, with number of maximum iterations $I_{\text{max}} = 5$, prediction horizon $N_p = 10$ and control horizon $N_c = 2$.

Simulations with the figure-of-eight shaped trajectory were performed comparing the results from two different approaches: the PD Controller by Pounds et al. [19], and our proposed NMPC. Figure 3 presents the simulation results from both control algorithms performing a figure-of-eight shaped trajectory. These simulations demonstrate the efficiency of our approach in comparison with the PD controller by analyzing the error plots in this figure, which presents the trajectory tracking error by comparing the performance of the two controllers over time. Note that the trajectory obtained using our approach converges rapidly to the steady-state point and tries to maintain the steady-state error at a minimum.

The efficiency of our approach can be analyzed through the controller’s output signals and the error in XY position (both also in Fig. 3). In addition, Table II presents a comparison of the two controllers through four indexes: IAE (Integral Absolute Error), ISE (Integral Square Error), ITSE (Integral Time-weighted Squared Error) and ITAE (Integral Time-weighted Absolute Error). This comparison demonstrates the efficiency of our approach by showing that it produced lower index values. In these simulations, our approach achieve better results than the PD controller, with an average improvement of 3.3 times (IAE), 9.1 times (ISE), 12.0 times (ITSE) and 3.7 times (ITAE).

<table>
<thead>
<tr>
<th>Controller</th>
<th>IAE</th>
<th>ISE</th>
<th>ITSE</th>
<th>ITAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PD</td>
<td>63.02</td>
<td>47.86</td>
<td>2.46e-3</td>
<td>3.20e-8</td>
</tr>
<tr>
<td>NMPC</td>
<td>19.11</td>
<td>5.27</td>
<td>204.23</td>
<td>862.77</td>
</tr>
</tbody>
</table>

IV. CONCLUSION

In this work, we have proposed a fast nonlinear model predictive control approach that ensures the stabilization and control of Micro Multirotor Aerial Vehicles (Micro-MAVs). Like their bigger counterpart, these aerial robotic systems rely solely on on-board sensors to localize themselves, which makes them suitable for experiments in GPS-denied environments. However, the Micro-MAV robots use low processing power boards, which causes an issue on implementing control approaches with high computational cost. To prove the efficiency and the feasibility of our approach, we performed a simple comparison with a PD controller to show the difference in performance. This comparison was performed in several numerical simulations and statistically verified using performance indices. The controller was able to track a given path and out-stay disturbances while hovering or performing the desired trajectories.