Reactive Navigation of an Unmanned Aerial Vehicle With Perception-Based Obstacle Avoidance Constraints

Björn Lindqvist, Sina Sharif Mansouri, *Member, IEEE*, Jakub Haluška, and George Nikolakopoulos, *Member, IEEE*

***Abstract*— In this article, we propose a reactive constrained navigation scheme, with embedded obstacles avoidance for an unmanned aerial vehicle (UAV), for enabling navigation in obstacle-dense environments. The proposed navigation architecture is based on a nonlinear model predictive controller (NMPC) and utilizes an onboard 2-D LiDAR to detect obstacles and translate online the key geometric information of the environment into parametric constraints for the NMPC that constrain the available position space for the UAV. This article focuses also on the real-world implementation and experimental validation of the proposed reactive navigation scheme, and it is applied in multiple challenging laboratory experiments, where we also conduct comparisons with relevant methods of reactive obstacle avoidance. The solver utilized in the proposed approach is the optimization engine (OpEn) and the proximal averaged Newton for optimal control (PANOC) algorithm, where a penalty method is applied to properly consider obstacles and input constraints during the navigation task. The proposed novel scheme allows for fast solutions while using limited onboard computational power, which is a required feature for the overall closed-loop performance of a UAV and is applied in multiple real-time scenarios. The combination of built-in obstacle avoidance and real-time applicability makes the proposed reactive constrained navigation scheme an elegant framework for UAVs that is able to perform fast nonlinear control, local path planning, and obstacle avoidance, all embedded in the control layer.**

***Index Terms*— Model predictive control (MPC), obstacle avoidance, path planning, reactive navigation, unmanned aerial vehicles (UAVs).**

I. Introduction

A. Applications and Problem Statement

DUE to the latest massive technological improvements in computation power and smart systems, the unmanned aerial vehicles (UAVs) have turned out to be the ultimate all-purpose tool for inspection and exploration tasks. Nowadays, UAVs demonstrate their promising capabilities in application domains, such as wind-turbine inspection [1] and underground mine navigation [2], search and rescue missions [3], including the delivery of first-aid or defibrillators in case of an accident [4], and so on. The main inspiration for this work was related to subterranean applications in the context of the DARPA Subterranean Challenge [5], [6] as part of Team CoSTAR [7] and the Nebula autonomy developments [8]–[10].

Because of the agility of the UAVs, these platforms can access hard-to-reach places, navigate through constrained spaces, and ignore any issues related to the ground terrain. The disadvantages of a UAV platform lie in the fast run-time requirements to maintain flight stability and the fragility of the platform, where an environmental interaction often leads to a crash. To overcome these problems, effective navigation and obstacle avoidance frameworks are needed with the capability to operate during high run-time requirements while considering the nonlinear dynamics of the UAV.

As UAVs start appearing in more and more applications, the need for stable and intelligent control algorithms is increasing as well. Moving from a human-controlled or human-aided UAV to a fully autonomous operation, it requires the controller to naturally and in real time interact with the environment around the UAV for obstacle avoidance and fast path planning and replanning. Thus, this article proposes a totally novel reactive local path planning and obstacle avoidance scheme by utilizing a nonlinear model predictive controller (NMPC) and related environmental information from the 2-D LiDAR measurements that aim for a fully autonomous online navigation through constrained environments, which can be applied in complex exploration or agile inspection tasks.

B. Background and Motivation

The problem of path planning is fundamental in robotics, and as such, it has been very well studied [11]. The UAV brings its own set of challenges to the path planning problem, namely, fast run-time requirements, sensitivity to collisions, and nonlinear system dynamics, while many approaches have been successfully applied in real-life use cases [12]. In general, for the problem of path planning in a constrained environment, the planners can be divided into two generic categories, namely, reactive online path planners and global path planners, often based on occupancy maps.

Manuscript received 23 May 2021; accepted 4 October 2021. Date of publication 23 November 2021; date of current version 18 August 2022. This work was supported by the European Union's Horizon 2020 Research and Innovation Programme (illuMINEation) under Grant 869379. Recommended by Associate Editor G. Notarstefano. *(Corresponding author: Björn Lindqvist.)*

The authors are with the Department of Computer, Electrical and Space Engineering, Robotics and Artificial Intelligence Team, Luleå University of Technology, 97187 Luleå, Sweden (e-mail: bjolin@ltu.se; sinsha@ltu.se; jakhal@ltu.se; geonik@ltu.se).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCST.2021.3124820.

Digital Object Identifier 10.1109/TCST.2021.3124820

The nominal global planners are the methods based on Dijkstra's algorithm and the rapidly exploring random tree (RRT) algorithms [13], with more modern versions, such as jump-point search [14] and RRT^x [15], respectively, both capable of a quick replanning. Despite this, global planners often suffer from a high computation time, drifts in localization and mapping, and generally being dependent on an occupancy map of the surrounding environment. As such, global planners, while necessary for many missions and tasks, require to be paired with a reactive planner, running online, and paired with the control layer to guarantee no collisions with the environment. For example, in [16], an RRT-based global planner is paired with an artificial potential field (APF) for UAV navigation. Due to multiple benefits, the APF [17] is the current most common approach utilized for reactive obstacle avoidance and has been used for the UAV application use cases for both mapping [18] and exploration tasks [19], as an extra reactive control layer for obstacle avoidance.

There are also reactive, or local planning, approaches to occupancy map-based planning [20], [21], where by smart partitioning of the map, the computation time can be greatly reduced. Several methods that link the visual information from depth or monocular cameras to the optimization problems have also been attempted [22], [23].

Another optimization-based method of path planning is model predictive control (MPC) [24], where predicted future states are directly linked to a dynamic model of the system and as such described by a series of optimized control inputs, acting on the model within the dynamic constraints. With the gaining popularity of NMPCs and with more powerful computation and smarter optimization algorithms, such planners are a perfect fit for the nonlinear and dynamically constrained UAV system. Thus, if obstacle avoidance is included in an NMPC scheme, the result is a reactive planner that completely considers the dynamics and constraints of the system.

This approach has gained attention in the latest years. In [25], an NMPC scheme was developed for an autonomous car with integrated obstacle avoidance of elliptical obstacles based on the generalized minimal residual (GMRES) method. In [26], a robust MPC is developed, which is in addition also capable of dealing with uncertain moving obstacles of general shapes, evaluated in simulation for an autonomous car. In [27], the proximal averaged newton for optimal control (PANOC) algorithm [28] was applied for this purpose, using a ground robot, by considering nonlinear constraints to limit the available position space of the robot. The benefit of this type of method is the combination of control, local path planning, and obstacle avoidance into one control layer based on nonlinear optimization. This method has also been applied to a UAV [29], where the UAV successfully avoided a predefined cylindrical obstacle in a laboratory environment based on nonlinear constraints. In [30], more complex obstacle geometries, and multiple obstacles, are considered for the UAV navigation and a penalty method [31], [32] is applied for the consideration of constraints, while in [33], a constrained NMPC was used to keep a safety distance from tunnel walls in a subterranean application.

Using the optimization engine (OpEn) [34], [35] and the PANOC algorithm, this article aims to extend the results of this line of approaches to fully autonomous and reactive obstacle avoidance and path planning scheme, where onboard 2-D LiDAR data are used to form parametric constraints, based on a geometric approximation of the surrounding environment, to guarantee collision-free navigation for multiobstacle constrained environments while keeping the solver time low enough for stable control of the system and relying only on limited onboard computation. Extending this obstacle avoidance approach to real-time and real-life experiments, using onboard sensors for obstacle detection, links the nonlinear dynamics directly to the local path planning and obstacle avoidance problem in a realistic context.

C. Contributions

Based on the aforementioned state of the art, this article focuses on the missing key ingredient in previous NMPC-based obstacle avoidance schemes, namely, the fundamental seamless and functional link between the NMPC and the environment awareness, with an extended experimental verification of the overall concept and proper validation of the combined proposed architecture that could stand as a stable and properly functional navigation framework. Toward this direction, the underlying work on the NMPC scheme is a natural continuation in the form of a real-life implementation/experimentation of the theoretical framework proposed in [27], [29], and [35], developed in the context of real application scenarios. This is one of the major contributions of this article, since all previous works, to the best of authors' knowledge, rely either on simulations [25], [26], [30], [36] or when experiments are performed rely on predefined obstacles [27], [29] or motion-capture system to track obstacles [37], [38]. This limits the real-world impact of NMPC-based obstacle avoidance in the robotics context and we show, for the first time, in multiple challenging laboratory experiments the application of an NMPC-based obstacle avoidance scheme in real-time, real-world scenarios for a UAV, relying only on onboard computation and obstacle detection using onboard 2-D LiDAR, forming a completely novel navigation scheme for constrained environments that constitute the second major contribution. In this framework, the NMPC commands roll, pitch, and thrust references at a high frequency and as such merges set-point tracking with the obstacle avoidance problem. As it will be experimentally demonstrated, the novel proposed reactive scheme is computationally efficient and fast enough to satisfy the run-time requirements of the UAV platform, using limited computation power, while maintaining a safe distance from any obstacle and performing smooth and efficient obstacle avoidance maneuvers. In addition, we offer a comparison with two APF avoidance formulations and shall show that the NMPC-based avoidance outperforms the APF in every considered aspect.

D. Outline

The rest of this article is structured as follows. Initially, the dynamic model of the UAV is presented in Section II-A,

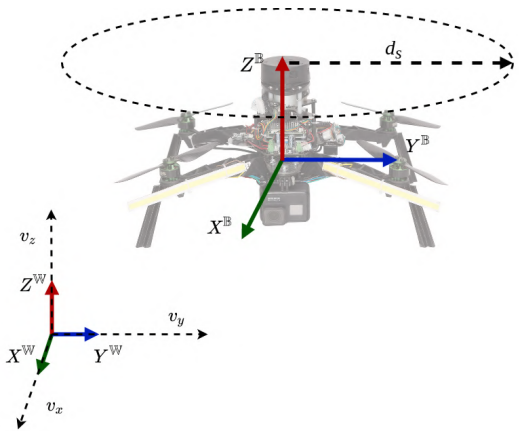

Fig. 1. Utilized coordinate frames, where $\mathbb{W}$ and $\mathbb{B}$ denote the world and body coordinate frames, respectively. In addition, the safety distance around the UAV is denoted d_s.

followed by the presentation of the corresponding cost function and the formulation of the obstacle constraints in Sections II-B and II-C, respectively. The proposed optimization is presented in Section II-D, the two comparison APF formulations are presented in Section II-E, and the experimental setup and tuning is presented in Section III. Multiple experimental scenarios, with related results that display the efficiency of the proposed framework, are presented in Section IV, with an additional corresponding comparison and discussion. In Section V, we discuss the discovered limitation of the framework and offer directions for future works and additions to the method. Finally, Section VI concludes this article by summarizing the findings.

II. Methodology

A. System Dynamics

The UAV coordinate systems are shown in Fig. 1, where $(x^{\mathbb{B}}, y^{\mathbb{B}}, z^{\mathbb{B}})$ denote the body-fixed coordinate system, while $(x^{\mathbb{W}}, y^{\mathbb{W}}, z^{\mathbb{W}})$ denote the global coordinate system. In this article, the states are defined in a yaw-compensated global frame of reference. The six degrees of freedom (DoF) UAV is defined by the set of equations (1). The same model has been successfully used in previous works, such as in [29], [30], and [33]

$$\dot{p}(t) = v(t) \tag{1a}$$

$$\dot{v}(t) = R(\phi,\theta)\begin{bmatrix}0\\0\\T_{\text{ref}}\end{bmatrix} + \begin{bmatrix}0\\0\\-g\end{bmatrix} - \begin{bmatrix}A_x & 0 & 0\\0 & A_y & 0\\0 & 0 & A_z\end{bmatrix} v(t) \tag{1b}$$

$$\dot{\phi}(t) = 1/\tau_\phi\big(K_\phi \phi_{\text{ref}}(t) - \phi(t)\big) \tag{1c}$$

$$\dot{\theta}(t) = 1/\tau_\theta(K_\theta \theta_{\text{ref}}(t) - \theta(t)) \tag{1d}$$

where $p = [p_x, p_y, p_z]^\top$ is the position, $v = [v_x, v_y, v_z]^\top$ is the linear velocity in the global frame of reference, and ϕ and $\theta \in [-\pi, \pi]$ are the roll and pitch angles along the $x^{\mathbb{W}}$- and $y^{\mathbb{W}}$-axes, respectively. Moreover, $R(\phi(t), \theta(t)) \in \text{SO}(3)$ is a rotation matrix that describes the attitude in the Euler form, with $\phi_{\text{ref}} \in \mathbb{R}$, $\theta_{\text{ref}} \in \mathbb{R}$, and $T_{\text{ref}} \geq 0$ to be the references in roll, pitch, and total mass-less thrust generated by the four rotors, respectively. The above model assumes that the acceleration depends only on the magnitude and angle of the thrust vector, produced by the motors, as well as the linear damping terms A_x, A_y, $A_z \in \mathbb{R}$ and the gravitational acceleration g.

The attitude terms are modeled as a first-order system between the attitude (roll/pitch) and the references $\phi_{\text{ref}} \in \mathbb{R}$ and $\theta_{\text{ref}} \in \mathbb{R}$, with gains $K_\phi, K_\theta \in \mathbb{R}$ and time constants $\tau_\phi, \tau_\theta \in \mathbb{R}$. The aforementioned terms model the closed-loop behavior of a low-level controller tracking ϕ_{ref} and θ_{ref}, which also implies that the UAV is equipped with a lower level attitude controller that takes thrust, roll, and pitch commands and provides motor commands for the UAV, such as in [39].

B. Cost Function

Let the state vector be denoted by $x = [p, v, \phi, \theta]^\top$ and the control action as $u = [T, \phi_{\text{ref}}, \theta_{\text{ref}}]^\top$. The system dynamics of the UAV are discretized with a sampling time T_s using the forward Euler method to obtain

$$x_{k+1} = \zeta(x_k, u_k). \tag{2}$$

This discrete model is used as the prediction model of the NMPC. This prediction is done with receding horizon, e.g., the prediction considers a set number of steps into the future. We denote this as the prediction horizon, $N \in \mathbb{N}$, of the NMPC. In some applications, the control horizon is distinct from N, but in this article, we will only consider the case where they are the same, meaning that both control inputs and predicted states are computed in the same horizon without loss of generality. By associating a cost to a configuration of states and inputs, at the current time and in the prediction, a nonlinear optimizer can be tasked with finding an optimal set of control actions, defined by the cost minimum of this cost function.

Let $x_{k+j|k}$ denote the predicted state at time step $k+j$, produced at the time step k. Also, denote the control action as $u_{k+j|k}$. Let the full vectors of predicted states and inputs along N be denoted as $\boldsymbol{x}_k = (x_{k+j|k})_j$ and $\boldsymbol{u}_k = (u_{k+j|k})_j$. The controller aims to make the states reach the prescribed set points while delivering smooth control inputs. To that end, we formulate the following cost function as:

$$\begin{aligned} J\big(\boldsymbol{x}_k, \boldsymbol{u}_k; u_{k-1|k}\big) &= \sum_{j=0}^{N} \underbrace{\|x_{\text{ref}} - x_{k+j|k}\|^2_{Q_x}}_{\text{State cost}} \\ &\quad + \underbrace{\|u_{\text{ref}} - u_{k+j|k}\|^2_{Q_u}}_{\text{Input cost}} + \underbrace{\|u_{k+j|k} - u_{k+j-1|k}\|^2_{Q_{\Delta u}}}_{\text{Input change cost}} \end{aligned} \tag{3}$$

where $Q_x \in \mathbb{R}^{8\times 8}$, Q_u, and $Q_{\Delta u} \in \mathbb{R}^{3\times 3}$ are symmetric positive definite weight matrices for the states, inputs, and input rates, respectively. In (3), the first term denotes the state cost, which penalizes deviating from a certain state reference x_{ref}. The second term denotes the input cost that penalizes a deviation from the steady-state input $u_{\text{ref}} = [g, 0, 0]$, i.e., the inputs that describe hovering in place. Finally, to enforce smooth control actions, a third term is added, which

penalizes changes in successive inputs, the input change cost. It should be noted that for the first time step in the prediction, this cost depends on the previous control action $u_{k-1|k} = u_{k-1}$. The UAV platform is susceptible to overly aggressive or sudden control actions in the roll and pitch, which can cause unnecessary wobbling or oscillations while translating, especially if it is carrying onboard sensing equipment, whose performance relies on stable and smooth flight behavior. The downside is that the system will be slower, depending on $Q_{\Delta u}$, in reacting to any new information or changing direction.

C. Obstacle Definition and Constraints

Following the constraint formulation structure for OpEn used in [27] and [29], while also keeping the constraints fully parametric so that their positions and size are part of the input fed to the NMPC scheme, we use the function $[h]_+ = \max\{0, h\}$ as described by (4). This allows us to formulate the constraints as equality expressions such that $[h]_+ = 0$ implies that the constraint is satisfied

$$[h]_+ = \begin{cases} 0, & \text{if } h \leq 0 \\ h, & \text{otherwise.} \end{cases} \tag{4}$$

Equation (4) is used for expressing a constrained area by choosing h as an expression that is positive while violating the constraint and negative when the constraint holds. In addition, by utilizing this constraint representation, it allows for the ability of defining more complex geometries by taking the product of multiple such terms, as they are zero where any of the terms are negative.

For the case of UAV navigation, there are multiple different types of obstacles that can be encountered in the surrounding environment. Being limited to 2-D information of the environment, two different obstacle types are included: circles (cylinders) and rectangles (general wall-like obstacles), and their parametric presentation as corresponding constraints will be discussed in the sequel. Moreover, to guarantee bounds on changes in the control actions, a constraint on the consecutive changes in control input will also be established. Finally, it should be highlighted that all the underlying constraints are considered in the full control horizon N to account for the obstacle constraints at all the predicted future time steps.

1) Circular Obstacle: A circular constraint can be used for any blocked area or general obstacle, where the radius of the circle envelops the area that is undesirable or blocked, forming an infinite cylinder in the 3-D space. The circular constraint is defined in (5) with a specified radius and x–y position as

$$h_{\text{circle}}\left(p, \xi^c\right) := \left[r_c^2 - \left(p_x - p_x^c\right)^2 - \left(p_y - p_y^c\right)^2\right]_+ = 0 \tag{5}$$

where $\xi^c = [p_x^c, p_y^c, r_c]$ define the x–and y-coordinates of the center and the radius of the obstacle. A visual representation of the obstacle is shown in Fig. 2, in the form of a cost map. This is a 2-D slice of the 3-D space where the new third axis represents the cost related to violating the obstacle constraint.

2) Rectangular Obstacle: The rectangular constraint represents wall-like obstacles with a limited width and length, described by intersecting lines (or hyperplanes) that form a rectangular area. As the NMPC structure requires that the obstacle structure is predefined, we require that the same obstacle types can be utilized for many situations. Thus, a wall-like obstacle can be defined by the intersection of four lines with safety distance d_s from an original line segment, provided from the environment data (Fig. 3). The constraint for a single line takes the form of (6)

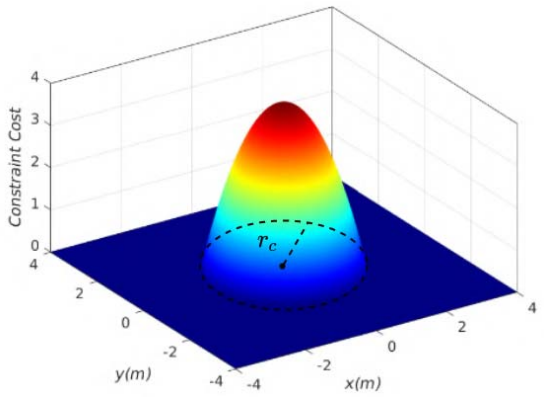

Fig. 2. Cost map of cylinder with radius 2 m.

$$h_{\text{line}}\left(p; \xi^l\right) := \left[m p_x - p_y + b\right]_+ = 0 \tag{6}$$

where m and b are standard line constants. To describe the rectangular constraint, we can take the product of four such terms as

$$\begin{aligned} h_{\text{rec}}(p, \xi^{\text{rec}}) : = & \left[m_{\text{par}} p_x - p_y + b_{\text{par},1}\right]_+ \\ & \times \left[-\left(m_{\text{par}} p_x - p_y + b_{\text{par},2}\right)\right]_+ \\ & \times \left[m_{\text{perp}} p_x - p_y + b_{\text{perp},1}\right]_+ \\ & \times \left[-\left(m_{\text{perp}} p_x - p_y + b_{\text{perp},2}\right)\right]_+ = 0 \end{aligned} \tag{7}$$

where line constants can easily be computed via simple algebraic operations to form the rectangle in Fig. 3, which produces a constraint as in Fig. 4, where $\xi^{\text{rec}} \in \mathbb{R}^6$ includes line constants for all four lines. It should be clear that to form such a constraint, we will have two lines parallel (with constants m_{par}, $b_{\text{par},1}$, and $b_{\text{par},2}$ and opposing signs) to the original line segment and two perpendicular (with constants m_{perp}, $b_{\text{perp},1}$, and $b_{\text{perp},2}$). To reduce large variations in the cost gradient of lines of varying slopes (especially close-to-1-D lines), we precondition this constraint using the computed slopes of the lines with the term $1/(\|m_{\text{par}}\| + \|m_{\text{perp}}\|)$ for an improved solver consistency.

3) Control Input Rate: We impose a constraint on the successive differences of control actions ϕ_{ref} and θ_{ref}, so as to directly prevent an overly aggressive behavior of the controller in-flight, i.e.,

$$|\phi_{\text{ref},k+j-1|k} - \phi_{\text{ref},k+j|k}| \leq \Delta\phi_{\max} \tag{8a}$$

$$|\theta_{\text{ref},k+j-1|k} - \theta_{\text{ref},k+j|k}| \leq \Delta\theta_{\max}. \tag{8b}$$

The above inequality constraints can be rewritten as equality constraints as it follows:

$$\left[\phi_{\text{ref},k+j-1|k} - \phi_{\text{ref},k+j|k} - \Delta\phi_{\max}\right]_+ = 0 \tag{9a}$$

$$\left[\phi_{\text{ref},k+j|k} - \phi_{\text{ref},k+j-1|k} - \Delta\phi_{\max}\right]_+ = 0 \tag{9b}$$

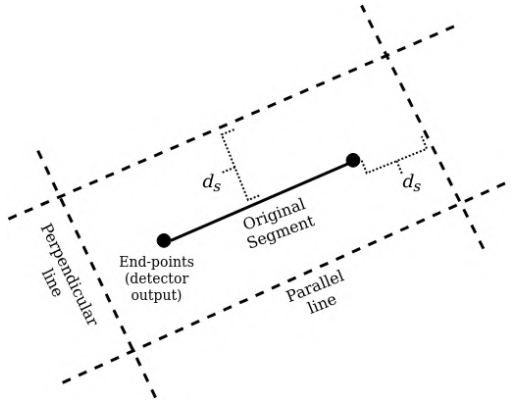

Fig. 3. Four lines envelop a line segment, with safety distance d_s.

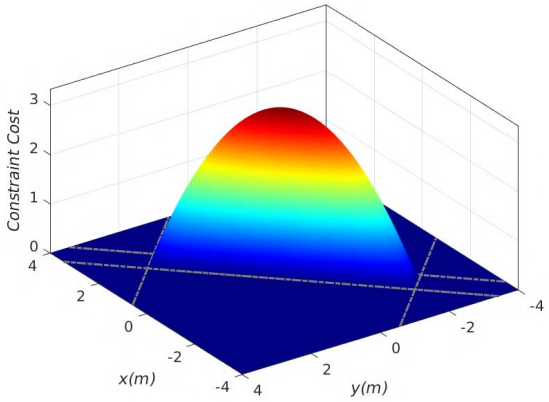

Fig. 4. Cost map of a rectangular obstacle as the intersection area of four lines.

e.g., setting a lower and upper bound with the maximum allowed change in the consecutive control action as $\Delta\phi_{\max}$. The exact same constraint is also formed for θ_{ref}, with the maximum change as $\Delta\theta_{\max}$.

4) Input Constraints: Finally, we also directly apply constraints on the control inputs. Since the NMPC is used with a real UAV, hard bounds on reference angles ϕ_{ref} and θ_{ref} must be considered, as a low-level controller will only be able to stabilize the attitude within a certain range. Since the thrust of a UAV is limited, such hard bounds must also be applied to the thrust input, T_{ref}. Thus, we can define bounds on inputs as

$$u_{\min} \leq u_{k+j|k} \leq u_{\max}. \tag{10}$$

D. Embedded Optimization

The NMPC problem is solved by the open-source OpEn [34], [35] and its associated algorithm PANOC [27], [29] that solves nonlinear nonconvex optimization problems. The OpEn generates embedded-ready source code from a specified cost function and set of constraints, while it can solve general parametric optimization problems on the form

$$\min_{z\in Z} \; f(z,\rho) \tag{11a}$$
$$\text{s.t. } G(z,\rho) = 0 \tag{11b}$$

where f is a continuously differentiable function with Lipschitz continuous gradient function and G is a vector-valued mapping so that $\|G(z,\rho)\|^2$ is a continuously differentiable function with Lipschitz continuous gradient. The decision variable and parameter are denoted by z and $\rho \in \mathbb{R}^{n_p}$, respectively. ρ is an input parameter to the NMPC module, including the initial measured state of the UAV $\hat{x}_k = x_{k|k}$, references u_{ref} and x_{ref}, the previous control action u_{k-1} (for the first term in the input change cost), and importantly the parametric obstacle data [as described by (5)–(7)], where n_p is the total number of such input parameters.

Based on the cost function and constraints, outlined in Sections II-B and II-C, we can formulate the NMPC problem, while in the presence of $N_c \in \mathbb{N}$ circular obstacles and $N_r \in \mathbb{N}$ rectangular obstacles as

$$\min_{\boldsymbol{u}_k, \boldsymbol{x}_k} \; J\left(\boldsymbol{x}_k, \boldsymbol{u}_k, u_{k-1|k}\right) \tag{12a}$$
$$\text{s.t. } x_{k+j+1|k} = \zeta\left(x_{k+j|k}, u_{k+j|k}\right)$$
$$j = 0, \ldots, N-1 \tag{12b}$$
$$u_{\min} \leq u_{k+j|k} \leq u_{\max}, \quad j = 0, \ldots, N \tag{12c}$$
$$h_{\text{circle}}\left(p_{k+j|k}, \xi_i^c\right) = 0, \quad j = 0, \ldots, N \tag{12d}$$
$$i = 1, \ldots, N_c \tag{12e}$$
$$h_{\text{rec}}\left(p_{k+j|k}, \xi_s^{\text{rec}}\right) = 0, \quad j = 0, \ldots, N \tag{12f}$$
$$s = 1, \ldots, N_r \tag{12g}$$
$$\text{Constraints (9)}, \quad j = 0, \ldots, N \tag{12h}$$
$$x_{k|k} = \hat{x}_k. \tag{12i}$$

This can be fit into the OpEn framework by performing single shooting of the cost function via the decision variable $z = \boldsymbol{u}_k$ and define Z by the input constraints (10). We also define G to cast the equality constraints presented in Section II-C. For the consideration of the constraints, a quadratic penalty method [31], [32] is applied, as it allows for the types of equality constraints proposed in Section II-C, especially the rectangles formed by products of $[h]_+$ terms, requiring only that the mapping of the constraint expression is continuously differentiable with Lipschitz continuous gradient. By formulating the problem as

$$\min_{z\in Z} f(z,\rho) + q\|G(z,\rho)\|^2 \tag{13}$$

where $q \in \mathbb{R}_+$ is a positive penalty parameter, and the PANOC algorithm [27] can be applied to the problem. Using a penalty method, an optimization problem, where the constraints are mapped to the cost domain, is resolved multiple times with an increasing penalty parameter q associated with the constraints while using the previous solution as the initial guess. In very simplified terms, this method gradually moves the cost minima of (13) by increasing q until none of the constraints are violated or rather until a specified tolerance is met.

In Fig. 5, the penalty method concept is displayed, where the solution from one to five penalty method iteration is shown for obstacle avoidance around a circular obstacle where $q^i = 10^i$, $i = 1, \ldots, 5$. As the penalty parameter is increased, the optimized trajectory, $\boldsymbol{u}_k$ (here displayed in position coordinates via the prediction model), is moved out of the obstacle. The trajectory will lie completely outside the

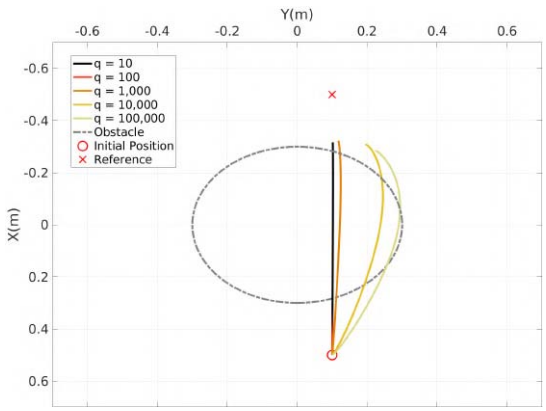

Fig. 5. NMPC optimized trajectories using one to five penalty method iterations for a circular obstacle with radius $r = 0.3$ m. The cost minima are gradually pushed out as the cost associated with violating the constraint is increased.

obstacle as q approaches infinity, and as such, small constraint violations should be expected and compensated by the safety distance d_s. In Fig. 5, it is depicted that the first iterations are not significantly different. In Section IV, we will be using penalties with $q^i = 10^3 \times (4^{i-1})$, $i = 1, \ldots, 4$, based on the results from initial testing and simulations.

E. Comparison Methods

For performing a comparison with the proposed NMPC approach, we will implement two APF formulations based on an approach very similar to the legacy APF by Warren [40], as APFs are one of the most common reactive avoidance formulations and fit very well into the 2-D LiDAR equipped UAV. The APF algorithms have the advantage of being able to work directly with the LiDAR scan 2-D point cloud, producing repulsive forces that shift the translational speed of the UAV based on the proximity and number of points within a certain radius of influence of the forces as $r_F \in \mathbb{R}$ of the APF. The APF should be seen as a pure reactive avoidance layer and requires a separate reference tracking controller, and to keep the comparison fair, we will use a similar NMPC, but without integrated obstacle avoidance. In Sections II-E1 and II-E2, we will define two formulations on the repulsive forces: one closely following the legacy approach, called the baseline APF, and one where we have added enhanced concepts that facilitate problems specific to the UAV platform in densely occupied environments. In regard to the selected optimization software and method for solving obstacle avoidance constraints, detailed comparisons with other optimizers/methods on the computational efficiency and ability to handle constraints of this type are offered in [32] and [34].

1) Baseline Approach: Let us define the LiDAR 2-D point cloud as an array of points {P}, where each point is described as the relative x and y position from the UAV/LiDAR (e.g., in the UAV body frame) as $\varrho = [\varrho_x, \varrho_y]$. Let us also denote the repulsive force as $F^r = [F_x^r, F_y^r]$ and the attractive force as $F^a = [F_x^a, F_y^a]$. As we are only interested in points inside the radius of influence, when considering the repulsive force, let us denote the list of such points $\boldsymbol{\varrho}_F \in \{P\}$, where $\|\varrho_F^i\| \leq r_F$ and $i = 1, 2, \ldots, N_{\varrho_F}$ (and as such, $N_{\varrho_F} \in \mathbb{N}$ is the number of points to be considered for the repulsive force). We can define the repulsive force as

$$F^r = \sum_{i=1}^{N_{\varrho_F}} L^r \left(1 - \frac{\|\varrho_F^i\|}{r_F}\right) \frac{-\varrho_F^i}{\|\varrho_F^i\|} + L^{\text{offset}} \frac{-\varrho_F^i}{\|\varrho_F^i\|} \tag{14}$$

where $L^r = [L_x^r, L_y^r] \in \mathbb{R}^2$ are the repulsive constants/gains representing the maximum repulsive force per point and $L^{\text{offset}} \in \mathbb{R}$ is an additional static potential. We can further define the attractive force to simply be $L^a(p_{\text{ref}} - \hat{p})$, where $L^a \in \mathbb{R}$ is the attractive gain, p_{ref} are the first two elements in x_{ref} related to the x and y position references, and $\hat{p} \in \mathbb{R}^2$ is the measured x and y position. The total force can then be obtained as $F = F^a + F^r$. To fit the APF to the reference tracking controller, the attractive force can be seen simply as the controller trying to reach the desired waypoint reference, while the repulsive force is the momentary shift in that waypoint as to avoid any obstacles.

2) Enhanced Approach: As we have discussed previously in Sections II-B and II-C, the UAV as a platform is susceptible to overly aggressive maneuvering, which in very tight spaces can lead to unwanted behavior. As such, we will impose a very similar repulsive force function but add saturation limits on the magnitude of forces, saturation limits on the rate of change of repulsive forces from one time step to the next, and a normalization of the attractive and summed total forces. In addition, instead of a static force per point inside r_F, we propose a larger static force for points inside a safety-critical radius $r_s \in \mathbb{R}$. We define the set of points, inside the safety-critical radius, as $\boldsymbol{\varrho}_s \in \{P\}$ where $\|\varrho_s^j\| \leq r_s$ and $j = 1, 2, \ldots, N_{\varrho_s}$ and $N_{\varrho_s} \in \mathbb{N}$ is the number of points inside the safety-critical radius. The advantages, compared to the baseline approach, are as follows: reduces excessive actuation by too rapid changes in the repulsive force, force normalization that leads to a much more consistent behavior of the reference tracking controller while making the tuning process much easier, and the addition of an extra safety bound if there are any points inside critical radius r_s. Thus, we can define the repulsive force as

$$F^r = \sum_{i=1}^{N_{\varrho_F}} L^r \left(1 - \frac{\|\varrho_F^i\|}{r_F}\right)^2 \frac{-\varrho_F^i}{\|\varrho_F^i\|} + \sum_{j=1}^{N_{\varrho_s}} L^s \frac{-\varrho_s^j}{\|\varrho_s^j\|} \tag{15}$$

with $L^s \in \mathbb{R}$ as the large repulsive gain to ensure that the UAV directly moves away from any point inside r_s. Imposing the suggested improvements (saturations and normalization) can most easily be explained in the depicted form in Algorithm 1, where $F_{\max} \in \mathbb{R}_+$ and $\Delta F_{\max} \in \mathbb{R}_+$ are saturation limits on the magnitude and rate of change on the repulsive force, F_k^r and F_{k-1}^r are the current and previous repulsive forces, respectively, and sgn() is the sign function.

3) Potential Field Tuning: The APF is extremely dependent on the tuning of repulsive and attractive gains L^a, L^r, as well as on the radius of influence r_F. Its performance is also very dependent on the tuning of the reference tracking controller that it is paired with, and performance can be reduced from moving aggressively as it leads to more abrupt changes in

Algorithm 1 Force Calculation

Inputs: F^a, F^r_k, F^r_{k-1}
if $|| F^r_k ||> F_{max}$ **then**
  $F^r_k \leftarrow sgn(F^r_k)F_{max}$
if $|| F_k - F_{k-1} ||> \Delta F_{max}$ **then**
  $F^r_k \leftarrow F^r_{k-1} + sgn(F_k - F_{k-1})\Delta F_{max}$
if $|| F^a ||> 1$ **then**
  $F^a \leftarrow \frac{F^a}{||F^a||}$
$F \leftarrow F^r_k + F^a$
if $|| F ||> 1$ **then**
  $F \leftarrow \frac{F}{||F||}$
Output: F

the repulsive forces. The APFs are evaluated in Section IV and in three experimental scenarios, where two require the UAV to pass in-between two obstacles. For a fair comparison, we will require that only one tuning of the APF is used for all three experiments to better evaluate the general performance. Based on the authors' experience, for the UAV case, the APF generally performs better when r_F can be chosen relatively large, while the UAV is moving relatively slowly as to make the reaction to changing forces less abrupt. However, to match the challenging scenarios that are used for evaluating the NMPC, we are forced to choose a small enough r_F to allow the UAV to move/fit through the densely occupied environment. As such, the force gains are chosen as $L^a = 1, L^r = [0.08, 0.16], L^{\text{offset}} = 0.04$, and $L^s = 1.5$, while $r_F = 0.75$ m and $r_s = 0.4$ m (slightly above the UAV size radius of 0.3 m), resulting in a comparatively aggressive tuning but with a smaller radius of influence, while we tune the reference tracking controller to be as fast as possible while maintaining no collisions. In addition, saturation limits for the enhanced approach are set to $F_{\max} = 6$ and $\Delta F_{\max} = 0.5$.

III. Experimental Setup

For the laboratory experiments, we use the robot operating system (ROS) [41] architecture for message handling between nodes. The general control structure can be seen in Fig. 6, where geometric environment data, registered from 2-D LiDAR scans, as well as state information (p, v, θ, and ϕ) from a Vicon motion capture system are fed to the NMPC module, together with references (x_{ref} and u_{ref} provided by the operator), to compute the control inputs ($T_{\text{ref}}, \theta_{\text{ref}}$, and ϕ_{ref}). The utilized low-level controller is ROSflight [39], which takes roll, pitch, thrust, and yaw-rate commands. The thrust command signal $u_t \in [0, 1]$ is assumed to have a quadratic relationship to the mass-less thrust (acceleration) input T_{ref} as

$$T_{\text{ref}} = \text{C}u_t^2 \tag{16}$$

where $C \in \mathbb{R}$ is the thrust constant that maps between T_{ref} and u_t.

A. Platform

The platform used for performing laboratory experiments is the Pixy [42] designed at the Luleå University of Technology, seen with its full sensor suite in Fig. 7, designed and used for underground constrained environments. The Pixy has been the testbed platform for multiple underground applications [2], [19], [43], designed to be a light and low-cost platform for aerial scouting purposes. In these experiments, we will use a simplified model, as state measurements are provided by the motion capture system, and thus, no onboard state estimation (except IMU) is required. As such, the important components of the platform are the RPLIDAR-A3 2-D LiDAR and the onboard computer, which is an Aaeon UP-Board with an Intel Atom x5-Z8350 processor and 4-GB RAM. As such, the NMPC optimization, as well as the 2-D LiDAR information processing, is done completely onboard with the limited computation power of the UP-board.

B. Obstacle Detection

Using the 2-D LiDAR measurements from the onboard LiDAR, we use the open-source ROS package obstacle detector [44] to detect and track obstacles. The package uses combined segmentation and merging of obstacles from 2-D LiDAR data and it is a perfect fit for the required needs, providing geometric approximations of the surrounding environment in the form of line segments and circles. Based on the onboard sensor, only the section of the environment within the line-of-sight of the UAV is visible. The obstacle detector matches the visible section of the obstacle to the best fitting circle segment or line segment. Assuming the obstacle to be circular, with only one side visible, it could lead to incorrect path generation, but as the detector and the NMPC are run at high frequencies, both the obstacle data and the generated paths are continually updated based on the new upcoming information (e.g., seeing more of the obstacle as the UAV moves past it), which mitigates such problems. Circular obstacles, c, are, just like in the constraint formulation, defined by a radius and the x- and y-coordinates of the center of the circle as

$$c \triangleq \left\{r_c, p^c\right\} = \left\{r_c, \left(p^c_x, p^c_y\right)\right\} \tag{17}$$

where r_c, p^c_x, and p^c_y define the radius and x and y position of the center of the circle. The obstacle detector also supports an additional safety distance, d_s, such that $r_c = r_{\text{real}} + d_s$. This safety distance is required since the constraints for obstacle avoidance assumes the position of the UAV to be expressed by a point. As such, d_s represents the size of the UAV and also in practice an extra increase to compensate for inaccuracies in measurements, solver tolerances, and limited penalty method iterations. Line segments, l, are defined by extreme points of the line segment as

$$l \triangleq \left\{p^{l,1}, p^{l,2}\right\} = \left\{\left(p^{l,1}_x, p^{l,1}_y\right), \left(p^{l,2}_x, p^{l,2}_y\right)\right\} \tag{18}$$

where $(p^{l,1}_x, p^{l,1}_y)$ defines the x and y position of the start point of the line segment and $(p^{l,2}_x, p^{l,2}_y)$ is the endpoint. From each such set of points, we can parameterize a rectangular constraint as defined in Section II-C2. This is done by computing the line equations, as shown in Figs. 3 and 4, to envelop the original line segment with a rectangle using the same extra safety distance d_s. As the detector is limited to outputting circles and line segments, obstacles are forced to be classified into either

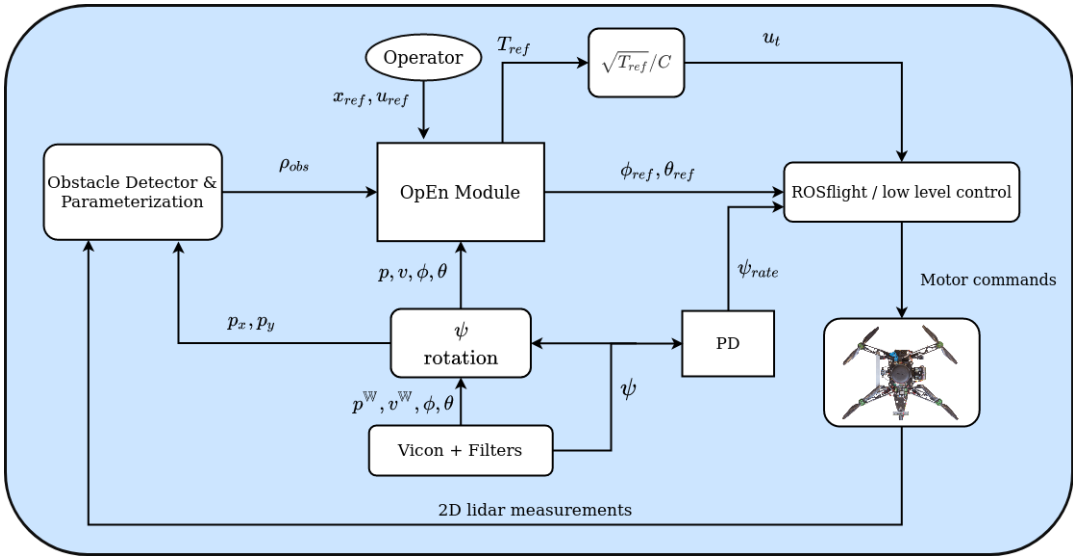

Fig. 6. Overall proposed control architecture overview. State estimation is done via a Vicon motion capture system and a median filter to estimate velocities. The 2-D LiDAR measurements are provided by an onboard RPLIDAR-A3. The OpEn module takes references (x_{ref} and u_{ref}), state measurements (p, v, θ, and ϕ), and obstacle parameters ρ_{obs} to compute the control inputs ($T_{\text{ref}}, \theta_{\text{ref}}$, and ϕ_{ref}). These are provided together with a yaw rate command ψ_{rate} (from a decoupled proportional-derivative controller) to the ROSflight low-level controller that calculates motor commands for the UAV.

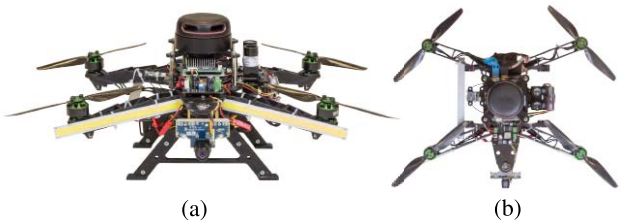

Fig. 7. UAV utilized for laboratory experiments. The subterranean scout, named the Pixy. (a) Front view. (b) Top view.

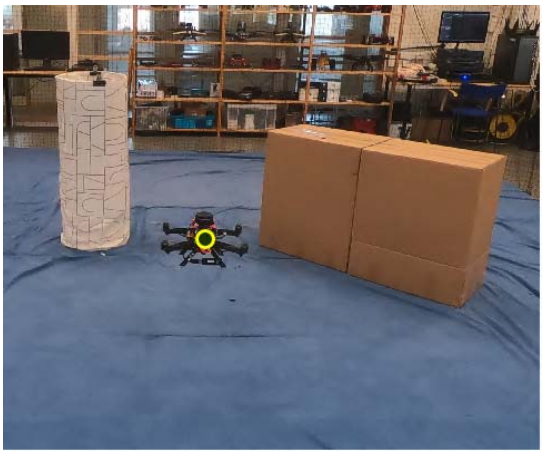

Fig. 8. Example scenario of UAV and obstacles in the laboratory environment.

one or divided into multiple obstacles, as it would be the case for a multiple-sided obstacle. The proposed architecture does not further merge obstacles, while the overlap of the obstacles poses no problems for the optimization method, except by unnecessarily using the limited number of constraints. An example of how the obstacle detector approximates the environment and how the constraints are formed can be seen in Figs. 8–10. Although being limited to only considering lines and circles, this method of obstacle detection performs very well and runs in real time and without a loss of generality, which is required for these types of applications. It should also be noted that, for the purpose of obstacle avoidance, only obstacles at a specified distance from the UAV, based on the horizon N, need to be considered in the constraint formulation. This greatly reduces the total number of obstacles and as such reduces the overall computational load.

C. Model Identification

In accordance with the model description presented in Section II-A, there are a set of unknown model parameters. Since the point of an NMPC scheme is to predict future states based on the optimized trajectory, the better model fit, the better the performance of the NMPC. Thus, the most impactful parameters are the first-order constants, describing the behavior of the UAV when a control input ($\theta_{\text{ref}}, \phi_{\text{ref}}$) is applied. These constants are gains $K_\phi, K_\theta \in \mathbb{R}$ and time constants $\tau_\phi, \tau_\theta \in \mathbb{R}$. These terms are dependent on both the platform and the tuning of the low-level controller and motor mixer, which in this case is the ROSflight.

We evaluate these constants by applying a step input in θ_{ref} and ϕ_{ref} to the attitude controller (ROSflight) on the UAV during flight and analyzing the corresponding response, as shown in Fig. 11. From the obtained results, it is evident that while not matching perfectly to that of a first-order system, we can approximate the time constants τ_ϕ and τ_θ to 0.23 and 0.25, respectively, while the gains K_ϕ and K_θ are (close to) unity. Similarly, we perform a height control test to identify the thrust constant C, as described in (16), and thus, this has been identified to be $(\sqrt{g}/0.48)$ for a fully charged battery. In-flight, we add a weak integrator based on the height measurement to

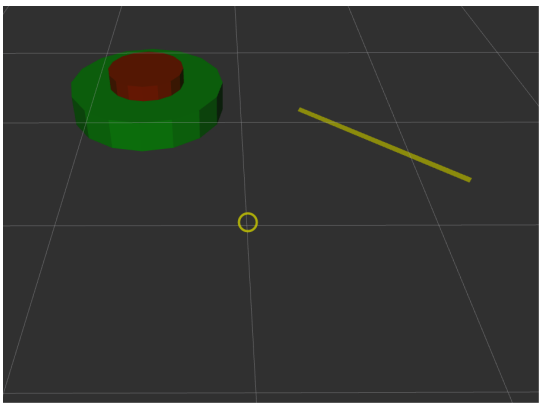

Fig. 9. Output from the obstacle detector in rviz from the scenario shown in Fig. 8, showing the radius and safety radius of the circular obstacle and the line segment from the wall-type obstacle.

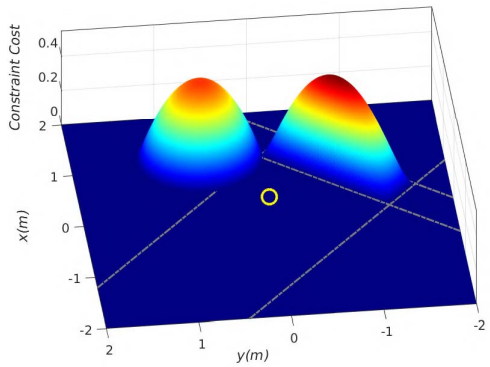

Fig. 10. Corresponding costmap from constraints formed with obstacle data from Fig. 9.

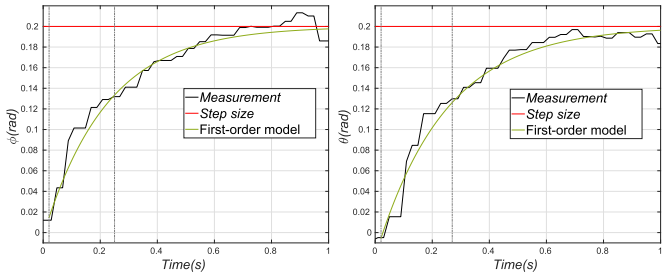

Fig. 11. System identification for the first-order time constants of LTU-Pixy. (a) ϕ step response. (b) θ step.

this constant to compensate for slight variations from battery drainage.

D. NMPC Tuning Parameters

For the experimental validation of the method, in addition to the model parameters tested for in Section III-C, we set linear damping terms A_x, A_y, and A_z to 0.1, 0.1, and 0.2 s^{-1}, respectively. The penalty method parameters are set as described in Section II-D, while we use a discretization of the cost function with a sampling time of 50 ms and a prediction horizon N of 40, implying a prediction of 2 s. Also, as such, we will be running the main control loop at 20 Hz as well, which, with the onboard IMU and ROSflight running at 100 Hz, is appropriate per common inner/outer control loop dynamics. To avoid run-time issues, we also impose a hard bound on the solver time to 40 ms, which means that if the solver does not converge in that time, we will use the nonconverged solution. The symmetric weight matrices in (3) are selected as: $Q_x = \text{diag}(2, 2, 40, 5, 5, 5, 8, 8)$, $Q_u = \text{diag}(5, 10, 10)$, and $Q_{\Delta u} = \text{diag}(10, 20, 20)$. The constraints on control inputs are selected (in SI units) as: $u_{\min} = [5, -0.2, -0.2]^T$ and $u_{\max} = [13.5, 0.2, 0.2]^T$

The constraints on the change of the input, described in (9), are selected as $\Delta\phi_{\max} = 0.08$ and $\Delta\theta_{\max} = 0.08$, and d_s is set to 0.4 m (UAV radius with propellers included approximately 0.3 m and an additional 0.1 m for safety). This tuning is extremely conservative with high weights on the inputs and low weights on the position states, which is what we will use for the set-point tracking. In addition, the input constraints only allow for a small magnitude in ϕ_{ref} and θ_{ref}.

This conservative tuning is selected for two reasons: 1) low x and y position weights lead to a smaller cost increase from deviating from the state reference to perform obstacle avoidance maneuvers and 2) the high input weights in addition to input constraints keep the UAV closer to the steady-state input of $[9.81, 0, 0]$ in-flight and as such keeps the 2-D LiDAR more stable, which was key to increase the performance of the obstacle detector. The tradeoff is a decrease in the distance covered inside the prediction horizon (as the UAV will be predicted to move slower). Such a conservative tuning might also be closer to that of a field application, as the onboard state estimation often suffers from too quick or sudden maneuvering.

Finally, we consider obstacles within a 3 m radius of the UAV in the optimization problem, since by the conservative tuning, this is slightly above the maximum distance covered within the prediction horizon. As stated in the optimization problem, the total number of obstacles has to be predefined, and we set this number to $N_c = 5$ and $N_r = 10$, which are also sorted by distance to the UAV. The average computation time naturally increases with more defined obstacle constraints and these numbers were chosen as a compromise between computation time and ensuring that all nearby detected obstacles can be included in the NMPC problem. We did not notice any case where this limitation caused any collisions due to not considering a certain obstacle.

IV. Result

The method for the evaluation of the proposed method will be for the UAV to take off to a set-point reference and will then be given a set point to go through various scenarios with obstacles or obstacle courses. We kindly recommend the reader to watch the corresponding video of the experiments, as it demonstrates the setup and the overall performance clearly (video link: https://youtu.be/xl_YQuDjs1M).

Figs. 12–14 show the scenario setup for the experimental validation, as well as the path of the UAV through the constrained environment, while Figs. 23–25 show snapshots from the above-mentioned video that also includes visualization of obstacle representations and the predicted optimized trajectory.

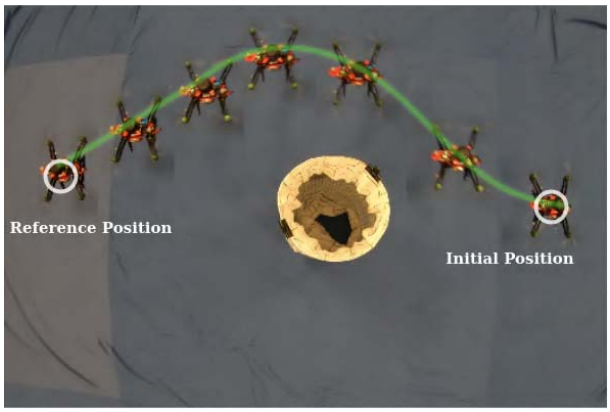

Fig. 12. Avoidance of a circular obstacle.

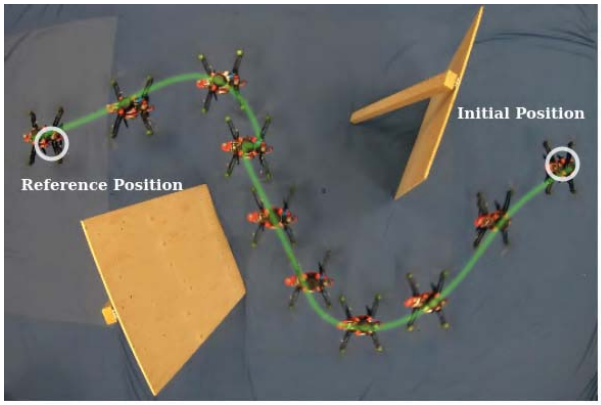

Fig. 13. Avoidance of two wall-like obstacles.

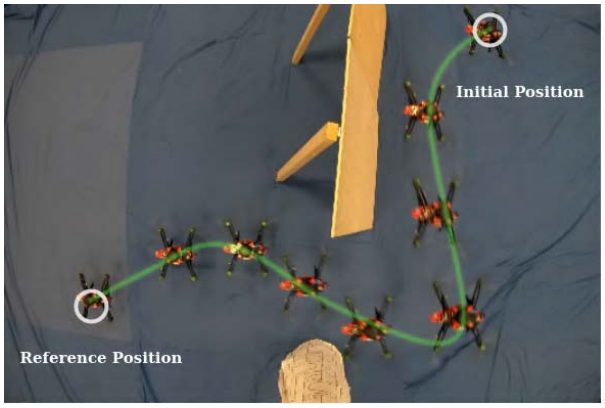

Fig. 14. Obstacle avoidance through a small opening.

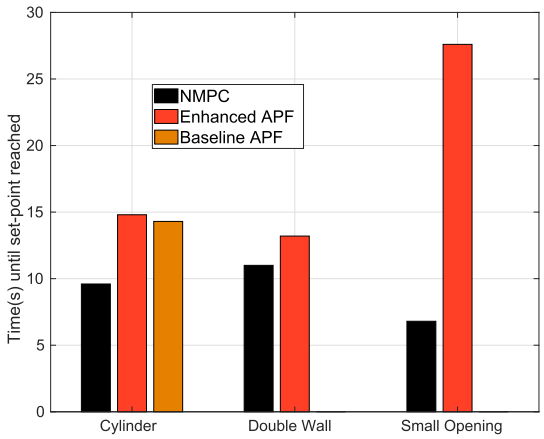

Fig. 15. Time duration from initiated movement until reaching the reference for NMPC and APF comparison.

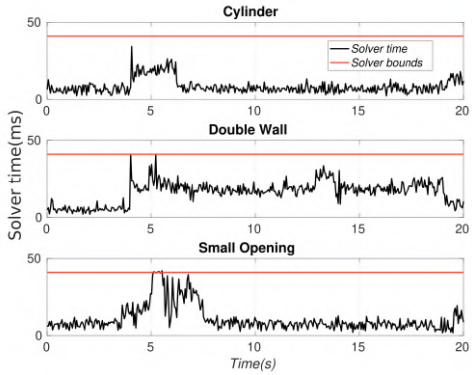

Fig. 16. NMPC-module solver time for the three experiments.

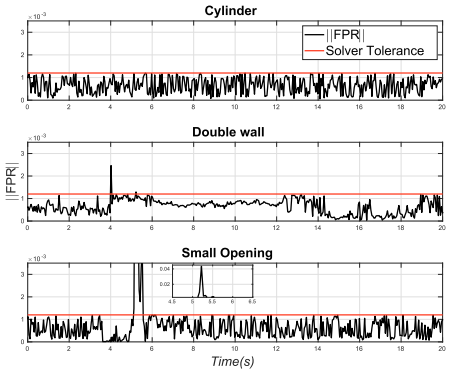

Fig. 17. NMPC-module norm of the fixed-point residual for the last inner problem for the three experiments.

Generally, we would like to evaluate three things: 1) various obstacle types; 2) the capability of dealing with multiple obstacles; and 3) the efficacy of the method at passing through tight openings or passages, which are tested in the three scenarios.

As for the comparison with the APF, we will also evaluate three things: 1) the time until the reference set point is reached; 2) the efficiency of the avoidance maneuvers; and 3) the ability to maintain the desired safety distance of 0.4 m. The time until mission completion is shown in Fig. 15, while a video of the APF performance is shown in https://youtu.be/dLnoLcNxPzs. Figs. 26 and 27 show the paths through the environment for each APF. From the obtained results, it is clear that the NMPC, by imposing hard bounds and considering a 2-s prediction on how to move around the obstacles, greatly outperforms the APF. While the enhanced APF is capable of completing all three scenarios (the baseline approach was simply incapable of moving in-between obstacles without excessive and dangerous movements), the NMPC objectively outperforms it on the time

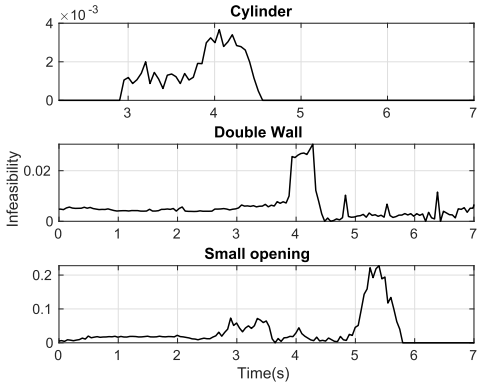

Fig. 18. NMPC-module constraint infeasibility during critical time instant during avoidance maneuvers.

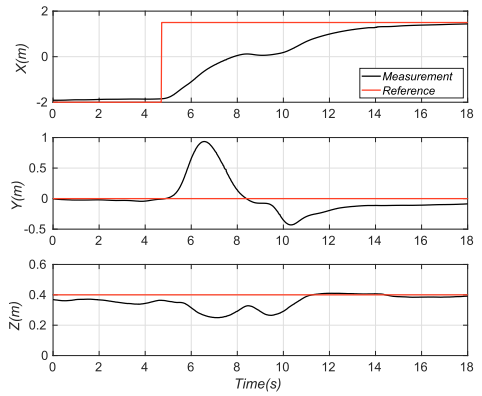

Fig. 19. Reference tracking for the obstacle avoidance scenario with two walls.

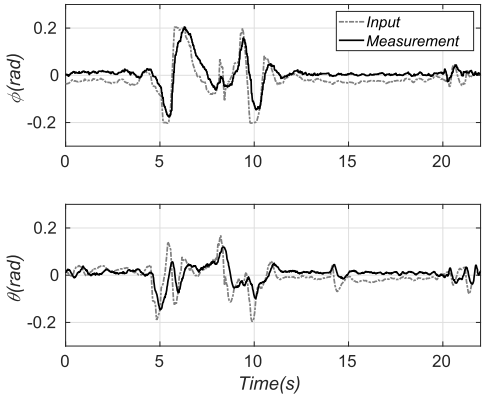

Fig. 20. Computed control signals for the scenario with two walls and corresponding angle states.

to reach the reference and, looking at the video results or Figs. 26 and 27 compared to Figs. 12–14, also very greatly outperforms it in the ability to execute efficient avoidance maneuvers. This can greatly be attributed to the proactive component of NMPC-based obstacle avoidance, where the UAV can start its avoidance maneuver, as soon as the obstacle is within the prediction horizon, without having to impose a huge safety radius around the obstacle. The main point of analysis for a reactive local planner is on how the safety distance is

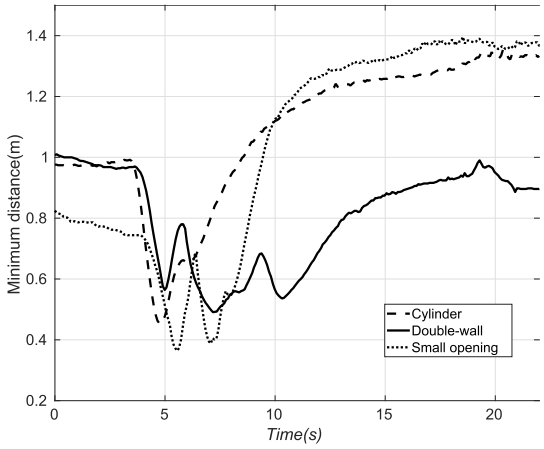

Fig. 21. Minimum distance from any obstacle by the closest LiDAR range measurement during the three experiment scenarios for the NMPC-based approach.

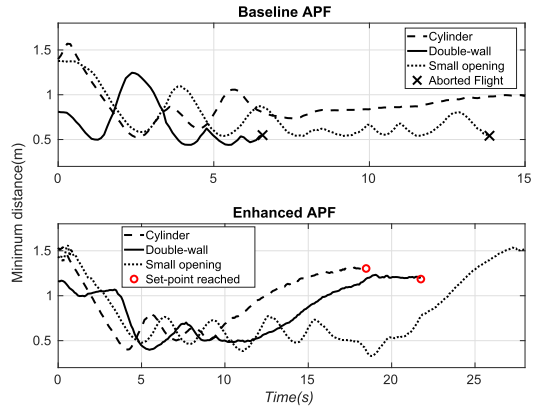

Fig. 22. Minimum distance from any obstacle by the closest LiDAR range measurement during the three experiment scenarios for the two APF approaches.

maintained from the constrained environment throughout the experiments, which is shown in Fig. 21 for the NMPC and in Fig. 22 for the APFs. These figures represent the closest range measurement from the onboard 2-D LiDAR throughout the flights, and as seen, the minimum safety distance is maintained completely throughout the experiments for the NMPC, except a small 0.03-m violation for the third experiment, while the APF has a slightly larger 0.07 m violation also during the third experiment while moving through the small opening. The main difference here is the ability for the NMPC to generate control signals that navigate the UAV as to precisely satisfy the desired safety distance, due to the method of constraining the available position space. It should also be noted that the safety distance for the NMPC is defined by the obstacle geometries, not the LiDAR measurements themselves, so there might not be a perfect overlap. For example, in the case of the first experiment of avoiding the cylinder, the minimum distance is 0.43 m, while the constraint is satisfied precisely.

Figs. 16 and 17 show the solver data for the three flights, namely, the solver time of the NMPC-module and the norm of the fixed-point residual of the last inner problem, describing the suboptimality of the solution (which here can be seen as

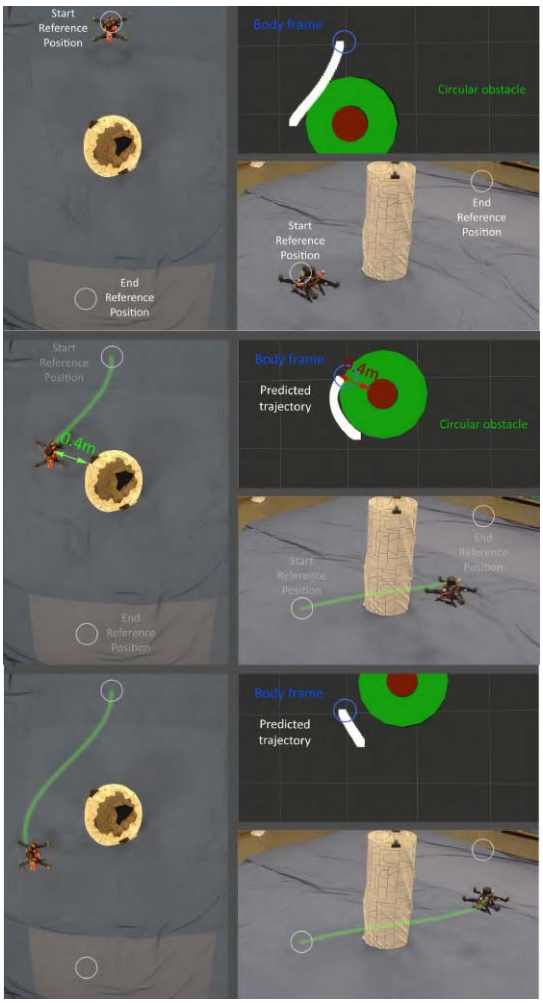

Fig. 23. Snapshot images of the first experiment. The UAV is tasked to avoid a cylindrical obstacle while performing set-point tracking. The maximum size radius of the UAV is 0.3 m; as such, we set the safety distance to 0.4 m.

Fig. 24. Snapshot images of the second experiment. The UAV is tasked to move through the multiobstacle constrained environment and avoid any obstacles.

a measurement of solver convergence or as the quality of the solution). Fig. 18 also shows the constraint infeasibility as the Euclidean norm of $G(z, \rho)$, during the critical part of the obstacle avoidance maneuver. Measuring the average solver time during the avoidance maneuver (e.g., between 4 and 7 s in the cylinder experiment), we get an average of 17.8, 19.6, and 22.3 ms for the three scenarios, while we momentarily hit the bounds for the solver time in the latter two scenarios. The result of hitting this bound can be clearly observed in Figs. 17 and 18 where the norm of the fixed-point residual peaks above the solver tolerance for these time instants, and we see a similar increase in the infeasibility. Generally, there is only one time instant where the optimizer is not close to converging, which is at 5.2 s into the third experiment, a case presented in Fig. 28. Despite this, since the trajectory is recalculated at 20-Hz intervals, a feasible solution is found at some moments later and the UAV still avoids the obstacle.

In addition, in Fig. 19, the set-point reference tracking from the double-wall experiment is displayed. From the obtained results, it is obvious that the UAV cleanly deviates from the reference to perform the obstacle avoidance maneuver. Fig. 20 shows the inputs and Euler angle states from the same experiment. These reflect the high input costs and the application of input and input rate constraints, where there is a relatively smooth behavior of the control inputs in ϕ_{ref} and θ_{ref} despite the multiobstacle constrained environment.

V. Limitations and Future Works

Working with a solution to the obstacle avoidance and local path planning problem in terms of nonlinear MPC with perception-based parametric constraints, we have discovered some limitations to this framework that provides directions for future works. The biggest limitation is the reliance on geometric data of the environment that has to run in real time and onboard. Small shifts in these geometric approximations can lead to large shifts in trajectory planning, where paths through the environment might close or open up due to these shifts or the obstacle classification suddenly changes (either by improper approximation or from seeing the obstacle from a different angle) resulting in a different position space being constrained by the NMPC, for example, as middle figure of Fig. 24 where the combined stand plus wall are merged into a circle as the UAV passes the obstacle. As such, proper

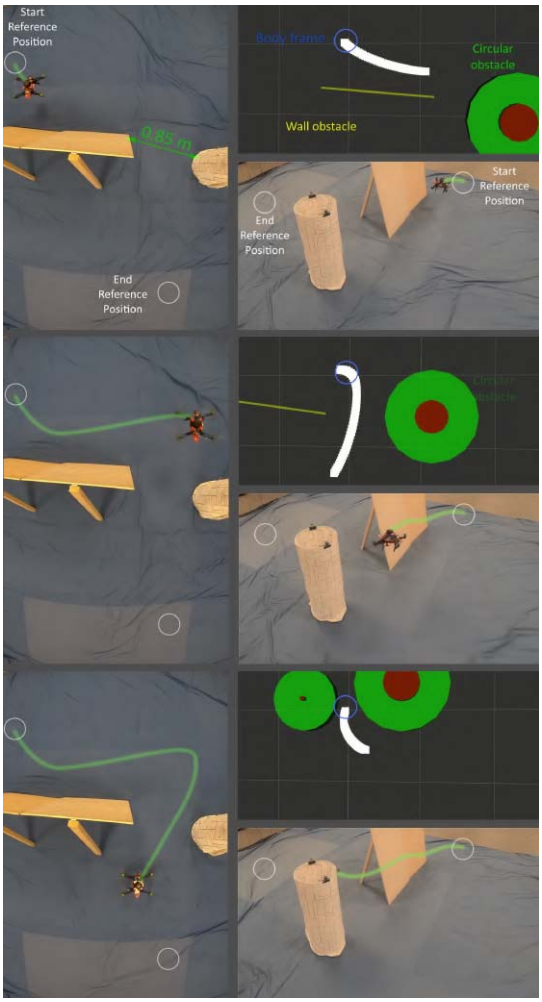

Fig. 25. Snapshot images from the third experiment. The UAV is tasked to move to the set-point reference, where the free path is a small constrained opening of 0.85 m between two obstacles. Using a safety radius of 0.4 m, the UAV can precisely pass through with an aggressive turn (within the dynamic constraints).

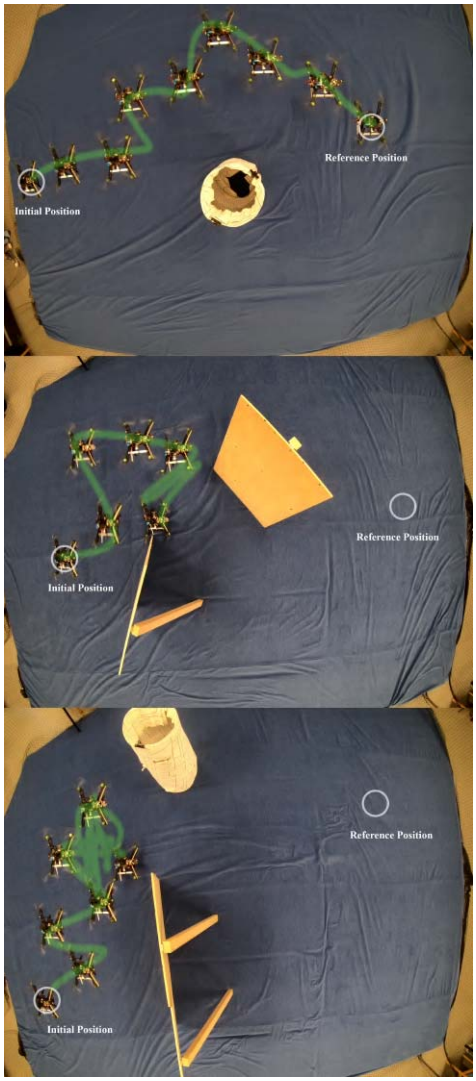

Fig. 26. Paths produced by the baseline APF for the three experiment scenarios. The scheme is incapable of completing the last two obstacle courses due to overly aggressive maneuvering.

filtering and tracking of the identified obstacles is a necessity for applying this method to avoid issues relating to obstacles suddenly changing positions, disappearing, or the filtering process replicating/duplicating certain obstacles due to fast enough changes in measurements when maneuvering so that the filter does not keep up (see results video). It should be noted that most methods (visual, distance to collision via convoluted neural networks [45], occupancy-based, and so on) have similar restrictions and requirements, with some exceptions like the APFs that can work directly with LiDAR data. In addition, as this work is limited to using two obstacles types (circles and line segments), either developing algorithms that can classify more types of obstacles or by using more general obstacles (by, e.g., using segments of high-degree polynomials to define obstacles boundaries) is a clear limitation and direction of future work.

Also, as discussed in Section IV, the solver might hit the bounds for computation time, which leads to nonconverged (or not optimal) solutions of the NMPC problem being applied to the UAV system (which does not guarantee obstacle-free trajectories). Adding conditions to restabilize midflight and recalculate and perhaps relax some input constraints when the solver does not converge would be a suitable addition to the method to guarantee that the trajectory $\boldsymbol{u_k}$ is obstacle-free before applying $u_{k|k}$ to the system. Further analysis and work on conditioning the constraints and investigating other methods for constraint-based NMPC navigation, such as the augmented Lagrangian method [46] [which is implemented in OpEn but would require a different approach to defining complex obstacles as in (7)] or the popular barrier functions [47] (which have previously been used in similar applications [48]), could be utilized instead of the penalty method used in this article. In addition, further developing the penalty method implementation to allow for different penalty update factors, initial penalties, and constraint tolerances for different constraints would supplement the very different constraints posed in Section II-C as, for example, the optimal parameters for the input rate constraints differ from the constraints that arise from obstacle avoidance.

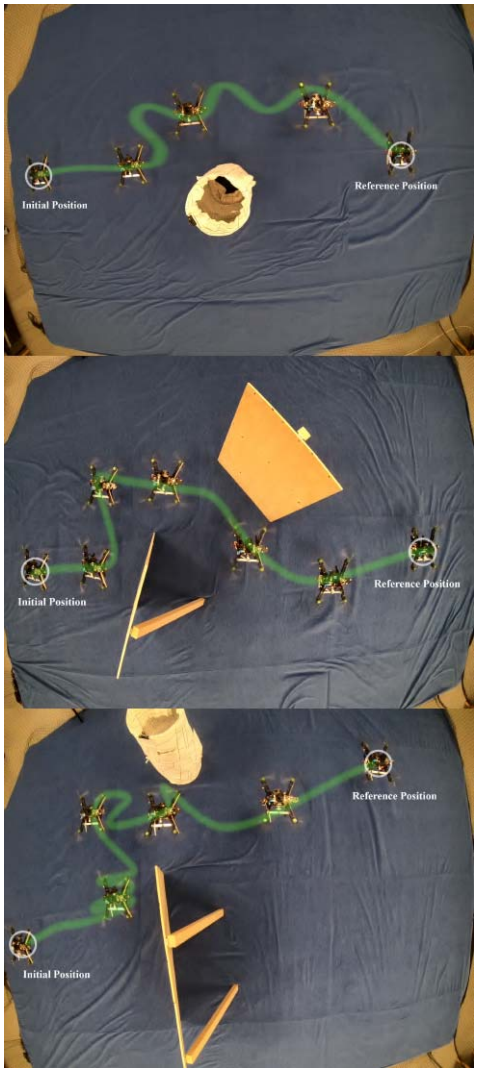

Fig. 27. Paths produced by the enhanced APF for the three experiment scenarios. The scheme completes all three scenarios but struggles to move through the small opening.

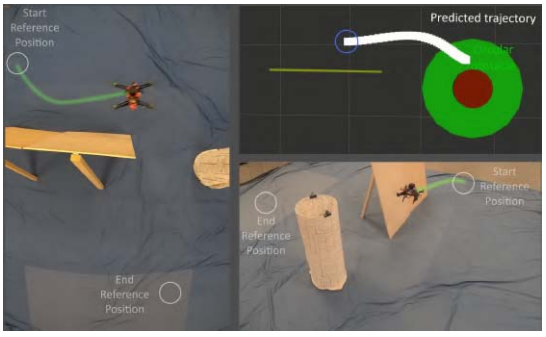

Fig. 28. Image from the third experiment, showing the nonconverged solution produced at 5.2 s into the small-opening experiment in Fig. 17.

The problems of solver time could also be prevented by running the method on more powerful onboard computers (which would also allow more computationally heavy mapping/environment awareness algorithms), such as the now-popular Intel NUC. In addition, purely for future works, an important consideration is to extend the method to 3-D obstacles, where obstacle data might come from depth cameras or 3-D LiDARs. Again, this would require reliable and fast algorithms for obstacle parameterization/approximation based on 3-D camera or LiDAR data and probably more powerful onboard computation. Another method for future investigation could be segmenting occupied space from an occupancy map representation [49] into obstacle geometries.

VI. Conclusion

In this article, we have presented a novel obstacle avoidance method based on nonlinear model predictive control using the OpEn used for a UAV in constrained environments. The method performs control, local path planning, and reactive obstacle avoidance all integrated into the control layer, and as such, the optimized trajectories are within the dynamic constraints of the system while being described by the dynamic system model. The efficacy of the control scheme has been demonstrated in multiple experimental scenarios, where the UAV avoids all obstacles while completing the mission of set-point tracking. To the best of our knowledge, this is the first time such a scheme has been applied on a real UAV for a fully autonomous reactive navigation where the parametric NMPC constraints are derived online from the surrounding environment using onboard computation and sensors.

The NMPC outperformed our APF algorithms on the ability to execute the mission of set-point tracking as quickly as possible, as well as performing considerably more efficient avoidance maneuvers. While APFs are great at maintaining distance from obstacles as an additional reactive safety layer, the NMPCs ability to precisely satisfy safety distances and proactively initiate its avoidance, without the need to impose very large areas of influence of the avoidance, greatly enhances its ability to move in-between multiple close-by obstacles or pass through tight openings, which becomes very significant in tightly constrained environments, such as in subterranean or indoors urban environments.

As demonstrated, the NMPC successfully solves the optimization problem online with few exceptions during the challenging obstacle-avoidance scenarios while maintaining input, input rate, and obstacle constraints. The proposed method is shown to handle multiple obstacle and it is also capable of passing through tight constrained spaces. Since the proposed complete architecture is totally novel, there are multiple directions of future works to improve the efficacy of the method, such as different methods for handling nonlinear nonconvex constraints and extending the method to include 3-D obstacles by using 3-D LiDARs or depth cameras for obstacle detection and parameterization.

References

[1] S. S. Mansouri, C. Kanellakis, E. Fresk, D. Kominiak, and G. Nikolakopoulos, "Cooperative coverage path planning for visual inspection," *Control Eng. Pract.*, vol. 74, pp. 118–131, May 2018.

[2] S. S. Mansouri, P. Karvelis, C. Kanellakis, D. Kominiak, and G. Nikolakopoulos, "Vision-based MAV navigation in underground mine using convolutional neural network," in *Proc. 45th Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, vol. 1, Oct. 2019, pp. 750–755.

[3] T. Tomic *et al.*, "Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue," *IEEE Robot. Autom. Mag.*, vol. 19, no. 3, pp. 46–56, Sep. 2012.

[4] J. K. Zègre-Hemsey, B. Bogle, C. J. Cunningham, K. Snyder, and W. Rosamond, "Delivery of automated external defibrillators (AED) by drones: Implications for emergency cardiac care," *Current Cardiovascular Risk Rep.*, vol. 12, no. 11, p. 25, Nov. 2018.

[5] *DARPA Subterranean (SubT) Challenge*. Accessed: May 20, 2021. [Online]. Available: https://www.subtchallenge.com/

[6] *Late Nights, Cool Hacks, and More Stories From the DARPA SubT Urban Circuit*. Accessed: May 20, 2021. [Online]. Available: http://disq.us/t/3mld3js

[7] *COSTAR*. Accessed: May 20, 2021. [Online]. Available: https://costar.jpl.nasa.gov/

[8] A. Agha *et al.*, "NeBula: Quest for robotic autonomy in challenging environments; TEAM CoSTAR at the DARPA subterranean challenge," 2021, *arXiv:2103.11470*.

[9] S.-K. Kim *et al.*, "PLGRIM: Hierarchical value learning for large-scale exploration in unknown environments," in *Proc. 31st Int. Conf. Automated Planning Scheduling*, vol. 31, 2021, pp. 652–662.

[10] M. Palieri *et al.*, "LOCUS: A multi-sensor lidar-centric solution for high-precision odometry and 3D mapping in real-time," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 421–428, Apr. 2021.

[11] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[12] C. Goerzen, Z. Kong, and B. Mettler, "A survey of motion planning algorithms from the perspective of autonomous UAV guidance," *J. Intell. Robot. Syst.*, vol. 57, no. 1, p. 65, Nov. 2009.

[13] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Proc. Millennium Conf., IEEE Int. Conf. Robot. Automat., Symp.*, vol. 2, Apr. 2000, pp. 995–1001.

[14] F. Duchoň *et al.*, "Path planning with modified a star algorithm for a mobile robot," *Proc. Eng.*, vol. 96, pp. 59–69, Jan. 2014.

[15] M. Otte and E. Frazzoli, "RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *Int. J. Robot. Res.*, vol. 35, no. 7, pp. 797–822, Jun. 2016.

[16] P. Pharpatara, B. Hérissé, and Y. Bestaoui, "3-D trajectory planning of aerial vehicles using RRT," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 3, pp. 1116–1123, May 2017.

[17] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robot. Automat.*, vol. 8, no. 5, pp. 501–518, 1992.

[18] D. Droeschel, M. Nieuwenhuisen, M. Beul, D. Holz, J. Stückler, and S. Behnke, "Multilayered mapping and navigation for autonomous micro aerial vehicles," *J. Field Robot.*, vol. 33, no. 4, pp. 451–475, Jun. 2016.

[19] C. Kanellakis, S. S. Mansouri, G. Georgoulas, and G. Nikolakopoulos, "Towards autonomous surveying of underground mine using MAVs," in *Proc. Int. Conf. Robot. Alpe-Adria Danube Region*. Cham, Switzerland: Springer, 2018, pp. 173–180.

[20] S. Hrabar, "Reactive obstacle avoidance for rotorcraft UAVs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 4967–4974.

[21] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online UAV replanning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 5332–5339.

[22] A. Schaub, D. Baumgartner, and D. Burschka, "Reactive obstacle avoidance for highly maneuverable vehicles based on a two-stage optical flow clustering," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 8, pp. 2137–2152, Aug. 2017.

[23] B. Ruf, S. Monka, M. Kollmann, and M. Grinberg, "Real-time on-board obstacle avoidance for UAVs based on embedded stereo vision," 2018, *arXiv:1807.06271*.

[24] K. Alexis, G. Nikolakopoulos, and A. Tzes, "Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances," *Control Eng. Pract.*, vol. 19, no. 10, pp. 1195–1207, Oct. 2011.

[25] U. Rosolia, S. de Bruyne, and A. G. Alleyne, "Autonomous vehicle control: A nonconvex approach for obstacle avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 2, pp. 469–484, Mar. 2017.

[26] R. Soloperto, J. Kohler, F. Allguwer, and M. A. Müller, "Collision avoidance for uncertain nonlinear systems with moving obstacles using robust model predictive control," in *Proc. 18th Eur. Control Conf. (ECC)*, Jun. 2019, pp. 811–817.

[27] A. Sathya, P. Sopasakis, R. Van Parys, A. Themelis, G. Pipeleers, and P. Patrinos, "Embedded nonlinear model predictive control for obstacle avoidance using PANOC," in *Proc. Eur. Control Conf. (ECC)*, Jun. 2018, pp. 1523–1528.

[28] L. Stella, A. Themelis, P. Sopasakis, and P. Patrinos, "A simple and efficient algorithm for nonlinear model predictive control," in *Proc. IEEE 56th Annu. Conf. Decis. Control (CDC)*, Dec. 2017, pp. 1939–1944.

[29] E. Small, P. Sopasakis, E. Fresk, P. Patrinos, and G. Nikolakopoulos, "Aerial navigation in obstructed environments with embedded nonlinear model predictive control," in *Proc. 18th Eur. Control Conf. (ECC)*, Jun. 2019, pp. 3556–3563.

[30] B. Lindqvist, S. S. Mansouri, and G. Nikolakopoulos, "Non-linear MPC based navigation for micro aerial vehicles in constrained environments," in *Proc. Eur. Control Conf. (ECC)*, May 2020, pp. 837–842.

[31] B. Hermans, P. Patrinos, and G. Pipeleers, "A penalty method based approach for autonomous navigation using nonlinear model predictive control," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 234–240, 2018.

[32] B. Hermans, G. Pipeleers, and P. Patrinos, "A penalty method for nonlinear programs with set exclusion constraints," *Automatica*, vol. 127, May 2021, Art. no. 109500.

[33] S. S. Mansouri *et al.*, "Subterranean MAV navigation based on nonlinear MPC with collision avoidance constraints," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 9650–9657, 2020.

[34] P. Sopasakis, E. Fresk, and P. Patrinos. (2019). *Optimization Engine*. [Online]. Available: http://doc.optimization-engine.xyz/

[35] P. Sopasakis, E. Fresk, and P. Patrinos, "OpEn: Code generation for embedded nonconvex optimization," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 6548–6554, 2020.

[36] S. S. Mansouri *et al.*, "A unified NMPC scheme for MAVs navigation with 3D collision avoidance under position uncertainty," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5740–5747, Oct. 2020.

[37] M. Kamel, J. Alonso-Mora, R. Siegwart, and J. Nieto, "Nonlinear model predictive control for multi-micro aerial vehicle robust collision avoidance," 2017, *arXiv:1703.01164*.

[38] B. Lindqvist, S. S. Mansouri, A.-A. Agha-mohammadi, and G. Nikolakopoulos, "Nonlinear MPC for collision avoidance and control of UAVs with dynamic obstacles," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 6001–6008, Oct. 2020.

[39] J. Jackson, G. Ellingson, and T. McLain, "ROSflight: A lightweight, inexpensive MAV research and development tool," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2016, pp. 758–762.

[40] C. W. Warren, "Global path planning using artificial potential fields," in *Proc. IEEE Int. Conf. Robot. Automat.*, Washington, DC, USA, May 1989, pp. 316–317.

[41] M. Quigley *et al.*, "ROS: An open-source robot operating system," in *Proc. ICRA Workshop Open Source Softw.*, Kobe, Japan, vol. 3, 2009, p. 5.

[42] D. Kominiak, S. S. Mansouri, C. Kanellakis, and G. Nikolakopoulos, "MAV development towards navigation in unknown and dark mining tunnels," 2020, *arXiv:2005.14433*.

[43] S. S. Mansouri, C. Kanellakis, D. Kominiak, and G. Nikolakopoulos, "Deploying MAVs for autonomous navigation in dark underground mine environments," *Robot. Auton. Syst.*, vol. 126, Apr. 2020, Art. no. 103472.

[44] M. Przybyla, "Detection and tracking of 2D geometric obstacles from LRF data," in *Proc. 11st Int. Workshop Robot Motion Control (RoMoCo)*, Jul. 2017, pp. 135–141.

[45] A. Kouris and C.-S. Bouganis, "Learning to fly by MySelf: A self-supervised CNN-based approach for autonomous navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1–9.

[46] E. G. Birgin and J. M. Martínez, *Practical Augmented Lagrangian Methods for Constrained Optimization*, vol. 10. Philadelphia, PA, USA: SIAM, 2014.

[47] A. G. Wills and W. P. Heath, "Barrier function based model predictive control," *Automatica*, vol. 40, no. 8, pp. 1415–1422, Aug. 2004.

[48] Y. Chen, H. Peng, and J. Grizzle, "Obstacle avoidance for low-speed autonomous vehicles with barrier function," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 1, pp. 194–206, Jan. 2018.

[49] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Robots*, vol. 34, no. 3, pp. 189–206, 2013.

Björn Lindqvist received the master's degree in space engineering with a specialization in aerospace engineering from the Luleå University of Technology, Luleå, Sweden, in 2019, where he is currently pursuing the Ph.D. degree in aerial robotics with the Department of Computer Science, Electrical and Space Engineering, Robotics and AI Team.

His research has so far been focused on collision avoidance and path planning for single-agent and multiagent unmanned aerial vehicle systems, as well as field applications of such technologies. He has worked as part of the JPL-NASA led Team CoSTAR in the DARPA Sub-T Challenge on subterranean UAV exploration applications, specifically in the search-and-rescue context.

Sina Sharif Mansouri (Member, IEEE) received the Bachelor of Science degree from the Technical University of Dortmund, Dortmund, Germany, in 2014, the Master of Science degree from the University of Tehran, Tehran, Iran, in 2012, and the Ph.D. degree from the Control Engineering Group, Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, Luleå, Sweden, in 2020.

He holds a post-doctoral position at the Robotics and AI Team, Luleå University of Technology. He currently works in the field of robotics, focusing on control, navigation, and exploration with multiple agents. His published scientific work includes more than 60 published international journals and conferences in the fields of his interest.

Dr. Mansouri received the Vattenfall's Award for the Best Doctoral Thesis in 2021.

Jakub Haluška received the master's degree in mechanical engineering from the Technical University of Liberec, Liberec, Czech Republic, in 2019.

He is currently a Research Engineer with the Control Engineering Group, Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology (LTU), Luleå, Sweden. His focus is on the design and construction of robust and field-applicable robots as well as 3-D printing technologies.

George Nikolakopoulos (Member, IEEE) was also affiliated with the NASA Jet Propulsion Laboratory, Pasadena, CA, USA, for conducting collaborative research on aerial planetary exploration and participated in the DARPA Grand challenge on Sub-T exploration with the CoSTAR team of JPL-NASA. He currently works as the Chair Professor in robotics and artificial intelligence at the Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology (LTU), Luleå, Sweden. His main research interests are in the areas of field robotics, space autonomy, unmanned aerial vehicles (UAVs), automatic control applications, networked embedded controlled systems, wireless sensor and actuator networks, cyber-physical systems, and adaptive control.

Prof. Nikolakopoulos is the Director of euRobotics and a member of the Scientific Council of ARTEMIS and the IFAC TC on Robotics. He established the Digital Innovation Hub on Applied AI at LTU and also represents Sweden in the Technical Expert Group on Robotics and AI in the project MIRAI 2.0 promoting collaboration between Sweden and Japan.