Towards Safe Human-Quadrotor Interaction: Mixed-Initiative Control Algorithm Using Real-Time NMPC

Bárbara Barros Carlos[1], Antonio Franchi[2] and Giuseppe Oriolo[1]

***Abstract*—This paper presents a novel algorithm for blending human inputs and motion generator commands, guaranteeing safety in mixed-initiative interactions between humans and quadrotors. The algorithm is based on nonlinear model predictive control (NMPC) and involves using the state solution to assess whether safety- and/or task-related rules are met to mix control authority. The mixing is attained through the convex combination of human and actual robot costs, driven by a continuous function that measures the violation of the rules. To achieve real-time feasibility, we rely on an efficient real-time iteration (RTI) variant of a sequential quadratic programming (SQP) scheme to cast the mixed-initiative controller. We demonstrate the effectiveness of our algorithm through numerical simulations, where a second autonomous algorithm is used to emulate the behavior of pilots with different competence levels. Simulations show that our scheme provides suitable assistance to pilots, especially novices, in a workspace with obstacles while underpinning computational efficiency.**

I. INTRODUCTION

Quadrotors have been widely used across various applications, resulting in a significant market expected to grow exponentially. Due to this increasing demand, one can envisage that, in the future, learning to fly a quadrotor will become a taken-for-granted skill, just as learning to drive a car nowadays. A first step could be a mixed-initiative control approach that guarantees the aerial system's "working conditions" while the novice operator learns to fly it. In other words, as long as some safety- and/or task-related rules are met, the operator commands are obeyed. Once novices start working in the context of complex tasks, it is essential to provide them with the assistance that gradually decreases as their competence increases – here, competence refers to the knowledge and ability that allow the use of the robot to accomplish a task. Conversely, one could think of an experienced operator who already has the competence level that enables him/her to fly a quadrotor. But as his/her cognitive load is primarily focused on the short-term aspect of the task, he/she cannot consider other underlying factors in the long run (e.g., safety). It is then essential to have a control approach that supervises the task in the long term to prevent the operator from being overwhelmed by (too) high engagement.

To fill these gaps, in this paper, we propose an efficient mixed-initiative controller based on nonlinear model predictive control (NMPC) to enforce safety in human-quadrotor interactions. The technical difficulties associated with the proposed controller are twofold: first, satisfy both the existing constraints and the human intentions; second, solve the underlying optimization problems that include a high-dimensional quadrotor system under the available computation time. We demonstrate how to handle these difficulties using concepts of zone MPC and online weight adaptation within the real-time iteration (RTI) scheme, an efficient Newton-type algorithm for real-time NMPC applications. To further speed-up solution times, we exploit high-performance numerical optimization algorithms, structure-exploiting convex solver, and linear algebra library in our implementation. The mixed-initiative controller is validated in simulation, where an autonomous algorithm emulates the behavior of pilots with different competence levels. To the best of our knowledge, this is the first mixed-initiative controller that overcomes all aforenamed dilemmas. For this reason, this work makes a significant contribution to human-robot interaction literature.

A. Related works

One way robots and humans can interact is to put the human in the loop with some blending scheme [1]. In a nutshell, the human and the automatic controller have control over the robot, and both systems must adapt to ensure task completion. For example, in bimanual robot manipulation, the robot can arbitrate between the user's command inputs and its underlying motion policies and understanding of bimanual tasks [2], [3]. In exoskeleton systems, the interplay between robot and human may be characterized as an interaction between teacher and student in a learning process. The teacher (robot) tries to minimize the student (human) error, applying a minimal effort [4], [5]. In the context of autonomous cars, the driver interacts with the automatic controller that aims at reducing the driver's workload and, at the same time, taking prompt actions in case of human failure [6], [7]. For human-swarm systems, the interaction paradigm is less obvious. As the size of the swarm increases, control should become more focused on the swarm as a whole rather than on the individuals, given the human's limited capacity to multitask. In this case, the interaction is more concerned with the human-swarm ability

[1]B. B. Carlos and G. Oriolo are with the Department of Computer, Control & Management Engineering (DIAG), Sapienza Università di Roma, Italy. {barros, oriolo}@diag.uniroma1.it

[2]A. Franchi is with the University of Twente, Faculty of Electrical Engineering, Mathematics & Computer Science, Robotics and Mechatronics Laboratory, The Netherlands and also with LAAS-CNRS, CNRS, Université de Toulouse, France a.franchi@utwente.nl

This work is partially funded by the European Commission project H2020 AERIAL-CORE (EC 871479).

to accomplish a particular task than with the spatial positioning of the swarm [8], [9].

Other interactive approaches rely on methods designed from the human perspective, where, in general, the presence of haptic cues increases situational awareness. These methods lean heavily in favor of perceiving the human as the source of action and the robot as the passive collaborator. For instance, virtual fixtures have been used to inform the operator of the highest comfort position, distance from the target position, or proximity to unsafe kinematic configurations [10], [11]. Obstacle avoidance is one of the most critical requirements to be met in many robotics scenarios. In this matter, haptic feedback can help inform the operator of instantaneous collisions [12], [13]. The work in [14] compares several human-collaborative schemes, highlighting their main aspects where haptic feedback is one of those.

B. Contributions and paper structure

The main contributions of this work are twofold:

- Novel mixed-initiative control algorithm for human-quadrotor systems based on real-time NMPC with safety guarantees.
- A numerical simulation that shows the computational efficiency and effectiveness of the proposed algorithm.

This paper is structured as follows. Section II states the problem we want to solve. Section III provides the details of the proposed algorithm. The numerical simulations are described, and their results are discussed in Section IV. Finally, the conclusion is made in Section V.

C. Notation

Operations such as $x \leq y$ is understood to be element-wise, whereas $\max(x)$ is vector-wise. Vectors are column vectors and we use the notation $(x; y; z)$ to denote concatenation. It is denoted by $\mathbf{I_n}$ the set of $n \times n$ identity matrices. An all-ones vector of dimension n is denoted by $\mathbf{1_n}$, whereas $\mathbf{0_n}$ denotes a n-vector of zeros. When dealing with norms of vectors $x \in \mathbb{R}^n$, we denote by $\|x\|$ the Euclidean norm. The set of positive real-valued vectors of dimension n is denoted by $\mathbb{R}^n_{>0}$.

II. PROBLEM STATEMENT

The fundamental problem in mixed-initiative control is how to blend human inputs and automatic controller commands to realize the former as much as possible while always enforcing safety. In this context, the formulation of our problem hinges upon the following components:

- A *robot* represented by a particular nonlinear, time-varying system
$$\dot{\xi} = f(\xi, u), \tag{1}$$
subject to a set of constraints, where states, control inputs and dynamics are denoted by $\xi \in \mathbb{R}^{n_\xi}$, $u \in \mathbb{R}^{n_u}$ and $f : \mathbb{R}^{n_\xi} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_\xi}$, respectively.
- Task variables denoted by $\eta = z(\xi)$ and described in $\mathbb{R}^{n_\eta}$. Typical examples are:
  - a subset of the state variables (which may even coincide with the state itself), e.g., cartesian points.
  - some nonlinear vector, e.g., kinematic map for some point of the robot.
- A motion generator that provides the collision-free *reference trajectory* $\eta^r \in \mathbb{R}^{n_\eta}$ for the task variables. It could be either a global planner or an autonomous controller.
- The human who specifies desired values to the control interface variables denoted by $\nu^r \in \mathbb{R}^{n_\nu}$. These values are henceforth called *human inputs*. As the notation suggests, they are reference signals representing human intentions.
- A set of *working conditions* defining the minimum requirements that the robot must meet to operate healthily. They can contain both safety rules (e.g., maximum kinetic energy, maximum dissipation, etc.) and/or task-related rules (e.g., the accuracy of end-effector positioning and orientation, etc.).

  Assumption 1. *Working conditions are defined such that the resulting set is convex.*

For this setting, we want to devise a *mixed-initiative controller* that will attempt to execute human inputs as much as possible without compromising the working conditions and the set of constraints inherent to the robot's physical limitations. As we will see, in addition to the mixed-initiative controller itself, the control algorithm also includes a blending mechanism that predicts the violation of working conditions and lends most of the control authority to the most capable agent at any time.

Let us now delve into the case of safe human-quadrotor interaction. The human operates the quadrotor providing it with four inputs: the desired roll, pitch, z angular rate, and total thrust. They are sent to the robot through a control interface, which is usually a joystick. Here, the working conditions concern safety that comes in the form of collision-free motions. Thus, for the human pilot to safely fly a quadrotor, he/she needs to perceive the surrounding cues and decide how to control the quadrotor by modifying the inputs above to avoid potential collisions. This job is always demanding for a novice, and in most cases, the quadrotor ends up crashing in the workspace [15]. That is why a mixed-initiative controller capable of guaranteeing safety during the learning process is of particular importance. Another key aspect of this process is repetition. Through repetition, the pilot can gradually reach a motor pattern that enables him/her to execute the task and increase his/her competence successfully. To that end, we assume that the human-quadrotor system must track a position reference repeatedly. This scenario will help us to illustrate the performance of our algorithm.

III. MIXED-INITIATIVE CONTROL

The mixed-initiative controller perceives the human operator and the motion generator as two different agents having the same control rights. The blending mechanism can assess the working conditions' violation and mixes the commands so as the most capable agent has the most control authority at any given moment. An example is when the working conditions include safety rules in the form of collision-free motion, as in this paper. Suppose human inputs compromise the quadrotor's safety. In that case, a violation will be predicted, and the

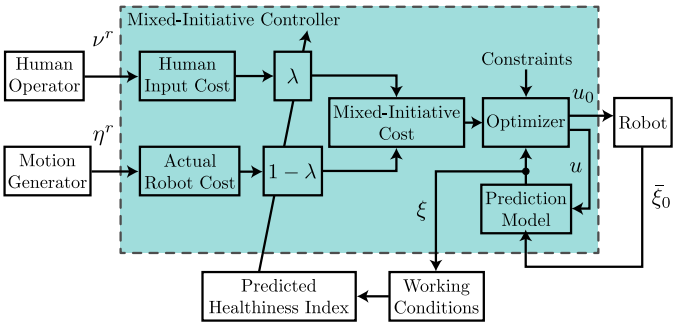


Fig. 1. Block diagram summarizing the proposed mixed-initiative control algorithm present in Section III.

blending will be carried out so that the control authority is mainly given to the motion generator.

Overall, this controller can be seen as an assistive scheme for novice and experienced pilots. Whereas for the former, it allows the pilot to learn to fly a quadrotor without crashing it, for the latter, it introduces some level of disengagement that enables the pilot to fly a quadrotor manually without focusing on avoiding obstacles.

A. Algorithm overview

Mixed control is accomplished through a *mixed-initiative NMPC controller* whose cost function – the *mixed-initiative cost* – is a convex combination between the *actual robot cost* and the *human input cost*. The latter is a cost term that penalizes the deviation from the human inputs. The proposed blending mechanism is a continuous function between 0 and 1, named *predicted healthiness index*, that drives the convex combination. It is computed in this way: when the index tends to 0, the working conditions' violation is too close and, hence, the mixed-initiative cost tends to the actual robot cost. When the index tends to 1, the violation is far enough and, therefore, the mixed-initiative cost tends to the human input cost.

Based on the state solution of the NMPC controller, at each time instant, the predicted healthiness index computes how far the robot is from violating the working conditions within a *virtual horizon* whose length is defined by the designer. It then selects a value between 0 and 1 and blends the commands. The idea is that the closer the index to 0, the more troublesome it will be to keep the system within the working conditions in the next virtual horizon, even if full control is given to the motion generator (and human inputs are heavily ignored from that moment on). Vice-versa, the closer the index to 1, the easier the motion generator's job will be in case it takes full control of the system. Fig. 1 shows a block diagram of the proposed algorithm.

B. Mixed-initiative controller

When using NMPC to control a system, at each sampling instant, a nonlinear nonconvex program is solved using the current state as the initial value. However, as the programs' solution times can be rather long, the employment of NMPC has only recently been extended to applications where shorter sampling times are required [16]. Typically, a continuous-time, infinite-dimensional optimal control problem (OCP) is tailored according to the problem at hand, discretized using some numerical strategy, and then solved. In doing so, the tailored OCP is transformed into a discrete-time, finite-dimensional nonlinear program (NLP) for which the optimally conditions are set up and solved at each sampling instant. In our approach, we cast the mixed-initiative (MI) controller as a constrained NLP formulated as follows:

Problem 1 (Mixed-Initiative Controller)**.**

$$\min_{\substack{\xi_0,\dots,\xi_N,\\ u_0,\dots,u_{N-1}}} \quad \sum_{i=0}^{N-1} L(\eta_i,\nu_i,\xi_i,u_i) + M(\eta_N,\nu_N,\xi_N) \tag{2a}$$

$$\text{s.t.} \quad \xi_0 - \bar{\xi}_0 = 0, \tag{2b}$$

$$\xi_{i+1} - F(\xi_i,u_i) = 0, \quad i=0,\dots,N-1, \tag{2c}$$

$$\xi_i \in \mathcal{X}, \quad i=0,\dots,N-1, \tag{2d}$$

$$u_i \in \mathcal{U}, \quad i=0,\dots,N-1, \tag{2e}$$

where

$$L(\eta_i,\nu_i,\xi_i,u_i) := \frac{1}{2}(\Delta\eta_i^T(1-\lambda)Q_\eta\Delta\eta_i + \Delta\nu_i^T\lambda Q_\nu\Delta\nu_i + \Delta e_i^T(1-\lambda)Q_e\Delta e_i + u_i^T(1-\lambda)Ru_i)$$

$$M(\eta_N,\nu_N,\xi_N) := \frac{1}{2}(\Delta\eta_N^T(1-\lambda)Q_{\eta_N}\Delta\eta_N + \Delta\nu_N^T\lambda Q_{\nu_N}\Delta\nu_N + \Delta e_N^T(1-\lambda)Q_{e_N}\Delta e_N).$$

Therein, stage and terminal cost terms are represented by L and M, respectively. We denote the task variables tracking error as $\Delta\eta_i = \eta_i - \eta_i^r$, $\Delta\eta_N = \eta_N - \eta_N^r$, and the human inputs tracking error as $\Delta\nu_i = \nu_i - \nu_i^r$, $\Delta\nu_N = \nu_N - \nu_N^r$. The cost function also includes other penalty terms that might by relevant to the specific application. Their tracking errors are denoted as $\Delta e_i = h(\xi_i) - e_i^r$, and $\Delta e_N = h(\xi_N) - e_N^r$, where the output functions $h(\xi)$, $h(\xi_N) \in \mathbb{R}^{n_h}$ and their respective references $e_i^r, e_N^r \in \mathbb{R}^{n_h}$ are defined by the relative complement $\mathbb{R}^{n_\xi} \setminus (\mathbb{R}^{n_\eta} \cup \mathbb{R}^{n_\nu})$. Note that $\xi := (\xi_0,\dots,\xi_N)$ and $u := (u_0,\dots,u_{N-1})$ represent the state and input trajectories of the discrete-time system whose dynamics are described by $F : \mathbb{R}^{n_\xi} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_\xi}$.

Moreover, $\mathcal{X}$ and $\mathcal{U}$ implement the state and input constraints associated with the physical limitations of the robot. Denoted by N is the horizon length and by $\bar{\xi}_0$ the current state estimate. The stage and terminal cost terms are weighted by the positive-definite weighting matrices $Q_\eta, Q_{\eta_N} \in \mathbb{R}^{n_\eta \times n_\eta}$, $Q_\nu, Q_{\nu_N} \in \mathbb{R}^{n_\nu \times n_\nu}$, $Q_e, Q_{e_N} \in \mathbb{R}^{n_h \times n_h}$, and $R \in \mathbb{R}^{n_u \times n_u}$. The scaling factor $\lambda \in (0,1)$ is the output of the blending mechanism and is used to determine how control authority should be mixed. Looking at the cost function, one can observe that increasing the value of λ will give more weighting to keeping the human inputs closer than the motion generator commands. Note that we consider an open interval because we are interested in convex problems. Finally, once the solution is computed, the first element of the input trajectory u_0 is applied to the system before shifting the horizon forward in time (see Fig. 1 for an illustration with the notations).

C. Working conditions

While the particular requirements that a robot has to obey may differ from one system to another, a generic approach is to express these requirements through suitable working conditions. In this paper, we are inspired by drone racing, where the conditions must be designed to help pilots safely fly a quadrotor along a trajectory in an arena with obstacles. To this end, we impose a maximum deviation $\alpha \in \mathbb{R}^{n_\eta}_{>0}$ from the reference trajectory to enforce safety. In terms of Euclidean norm, this condition reads as follows:

$$d(\eta) = \|\eta - \eta^r\| \leq \|\alpha\|. \tag{3}$$

Any point η satisfying (3) describes a convex free region $\mathcal{B}$, which we defined as

$$\mathcal{B} = \{\eta \in \mathbb{R}^{n_\eta} : \|\eta - \eta^r\| \leq \|\alpha\|\}. \tag{4}$$

We observe that $\mathcal{B}$ is a norm ball fully described by its center point η^r. This observation implies that the robot is softly constrained to move inside of a ball, where its actual trajectory is just one of the possible trajectories that avoid collisions, and at best, it is the reference trajectory itself.

Unlike the set of constraints intrinsic to the mixed-initiative NMPC, the working conditions are extrinsic constraints that facilitate the decoupled design of the safety- and/or task-related rules through the task variables. Ideally, the controller steers these variables to a steady value contained into the convex free region $\mathcal{B}$, while satisfying the constraints of Problem 1 throughout its evolution [17].

D. Predicted healthiness index

As previously established, the robot must remain inside $\mathcal{B}$ to enforce safety. Based on that, at each subproblem solve, we use the state solution of Problem 1 to compute the predicted healthiness (PH) index $\lambda : (0, \|\alpha\|) \mapsto (0, 1)$, our blending mechanism. More precisely, we measure how far the robot is from violating the boundary of the working conditions within a virtual horizon whose length is defined as $N_b \leq N$. Then, we select an index λ between 0 and 1 that is used to determine the actual weighting in Problem 1. Note that the weighting matrices have a clear and direct relationship with the MI controller behavior: the closer the index is to 0, the more the weighting to follow the motion generator commands; conversely, the closer the index is to 1, the more the weighting to follow the human inputs.

Moreover, the fact that N_b may differ from N introduces much flexibility in the controller's design. It allows one to determine the level of insight that the blending mechanism should have into the interaction, as we will see in simulation. In what follows, we formally define the PH index.

Definition 1 (PH index)**.** Given a virtual horizon with N_b intervals and the predicted η_i from the state solution of Problem 1, let us first select the largest excursion within $\mathcal{B}$

$$d_{\max} = \max_{i=0,\dots,N_b} \|\eta_i - \eta_i^r\|. \tag{5}$$

Then, to ensure that the PH index is a strictly positive function in the range of interest, let us consider a saturation function

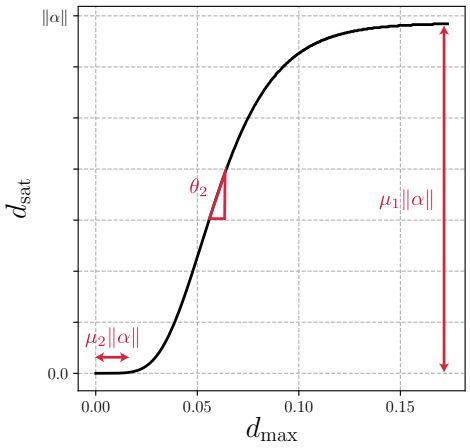


Fig. 2. Pictorial description of the saturation function (6). Note that when $d_{\max} = \|\alpha\|$ then $d_{\text{sat}} = \mu_1\|\alpha\|$.

$d_{\text{sat}} : [0, \|\alpha\|] \mapsto (0, \|\alpha\|)$ defined by the following Richards growth curve

$$d_{\text{sat}}(d_{\max}, \Lambda) := \frac{\mu_1\|\alpha\|}{(1 + \zeta \cdot \exp\{-\theta_2(d_{\max} - \mu_2\|\alpha\|)\})^{1/\zeta}} \in (0, \|\alpha\|), \tag{6}$$

where $\Lambda = (\mu_1, \mu_2, \theta_2, \zeta)$ is a quadruple that parameterizes the Richards' curve (see Fig. 2). In particular, μ_1 is a constant related to the the upper asymptote and whose value is close to 1 but strictly less than 1, μ_2 is a positive real number related to the lag phase, θ_2 is the growth rate, and ζ is a positive real number known as the shape parameter[1].

Finally, the PH index is defined as

$$\lambda(d_{\text{sat}}) := \frac{\sqrt{\|\alpha\|^2 - d_{\text{sat}}^2}}{\|\alpha\|} \in (0, 1). \tag{7}$$

An illustration of function (7) is provided in Fig. 3. Its profile indicates the aggressiveness with which the MI controller decreases human control authority. In particular, when d_{sat} is extremely high, the controller should be more belligerent and heavily ignore human inputs.

IV. NUMERICAL RESULTS AND DISCUSSION

This section is dedicated to describing the framework testbed and studies that demonstrate the effectiveness and efficiency of our control algorithm.

A. Human-quadrotor benchmark

1) Robot: Let us denote with $\{\mathcal{I}\}$ the inertial frame, and with $\{\mathcal{B}\}$ the body frame located at the center of mass (CoM) of the aerial vehicle. Consider a quadrotor with position $p = (x, y, z) \in \mathbb{R}^3$ expressed in $\{\mathcal{I}\}$, orientation $\gamma = (\phi, \theta, \psi) \in \mathbb{R}^3$, linear velocity $v_b = (v_x, v_y, v_z) \in \mathbb{R}^3$ expressed in $\{\mathcal{B}\}$, angular rate $\omega = (\omega_x, \omega_y, \omega_z) \in \mathbb{R}^3$ expressed in $\{\mathcal{B}\}$ and rotational speed of the propellers $\Omega = (\Omega_1, \Omega_2, \Omega_3, \Omega_4) \in \mathbb{R}^4$,

[1]The shape of the curve d_{sat} is due to parameter ζ. If $\zeta = 1$, one has the *logistic function*. If ζ tends to zero, the curve converges to the *Gompertz function*.

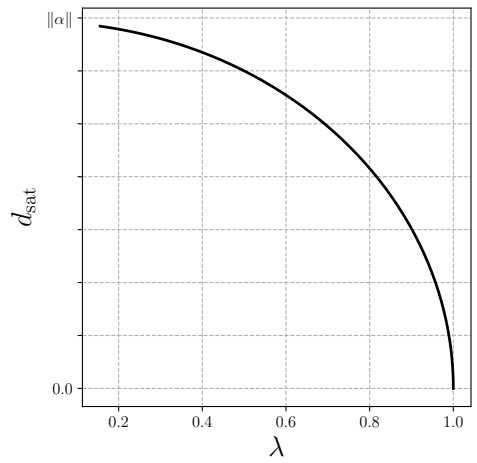


Fig. 3. An illustration of the function defined in (7) showing a profile relatively quadratic for penalizing excursions inside $\mathcal{B}$.

bounded by $\mathcal{X} = \{\Omega \in \mathbb{R}^4 : \underline{\Omega} \leq \Omega \leq \bar{\Omega}\}$. The system control inputs are the rotor torques, defined by $u := (\tau_1, \tau_2, \tau_3, \tau_4) \in \mathbb{R}^4$ and constrained in magnitude $\mathcal{U} = \{u \in \mathbb{R}^4 : \underline{u} \leq u \leq \bar{u}\}$. Its nonlinear dynamics are then given by the first-order ordinary differential equations:

$$\dot{\xi} = f(\xi, u) = \begin{pmatrix} S^T v_b \\ E\omega \\ \frac{1}{m} F_b - RG - \omega \times v_b \\ J^{-1}(M_b - \omega \times J\omega) \\ \frac{1}{J_m}(u - C_D \Omega \odot \Omega - d\Omega) \end{pmatrix} \tag{8}$$

with state $\xi := (p, \gamma, v_b, \omega, \Omega) \in \mathbb{R}^{16}$. The quadrotor's mass is denoted by m, $G = (0, 0, g) \in \mathbb{R}^3$ with g being the gravitational acceleration, d represents the drag coefficient of the rotor whose inertia is J_m. The element-wise product is denoted with $\odot$. The positive-definite matrix $J \in \mathbb{R}^{3\times3}$ denotes the vehicle inertia matrix. The rotation matrix from $\{\mathcal{I}\}$ to $\{\mathcal{B}\}$ is represented by $S \in SO(3)$. Matrix $E : \mathbb{R}^3 \to \mathbb{R}^{3\times3}$ expresses the relation between the instantaneous rates of change of γ and the instantaneous components of ω. The total external forces and moments applied to the CoM of quadrotor and expressed in $\{\mathcal{B}\}$ are defined, respectively, as

$$F_b := (0, 0, F_z), \quad M_b := (M_x, M_y, M_z) \tag{9}$$

with

$$F_z = C_T(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2), \tag{10a}$$
$$M_x = C_T \cdot l(\Omega_2^2 - \Omega_4^2), \tag{10b}$$
$$M_y = C_T \cdot l(\Omega_3^2 - \Omega_1^2), \tag{10c}$$
$$M_z = C_D(\Omega_1^2 + \Omega_3^2 - \Omega_2^2 - \Omega_4^2), \tag{10d}$$

where C_T is the thrust coefficient, C_D is the drag coefficient, and l is the distance between the quadrotor's CoM and the rotor's center. As an example, we report in Table I the values of the dynamic parameters appearing in (8) corresponding to the MikroKopter[2] quadrotor platform.

[2]https://www.mikrokopter.de/en/home

2) Reference trajectory: The global planner provides a cartesian helical trajectory η^r that avoids collisions with existing obstacles. This means that task variables are defined as $\eta := p \in \mathbb{R}^3$. Note that the reference trajectory implicitly enforces safety.

3) Pilot inputs: Usually, pilots provide the quadrotor with the desired roll, pitch, z angular rate, and total thrust, i.e., $g = (\phi_h, \theta_h, \omega_z, F_{z,h}) \in \mathbb{R}^4$. For simplicity, we assume that the desired z angular rate is zero throughout the task, and the total thrust is mapped into the speed of the propellers, i.e., $\nu^r : g \mapsto \mathbb{R}^6$. To simulate the pilot inputs, we use a second NMPC controller that reads as follows:

$$\begin{aligned} \min_{\substack{\xi_0,\dots,\xi_N,\\ u_0,\dots,u_{N-1}}} \; & \frac{1}{2}\sum_{i=0}^{N-1} \Delta\eta_i^T Q_1 \Delta\eta_i + \gamma_i^T Q_2 \gamma_i + v_{b,i}^T Q_3 v_{b,i} \\ & + \omega_i^T Q_4 \omega_i + \Delta\Omega_i^T Q_5 \Delta\Omega_i + u_i^T R_1 u_i \\ & + \frac{1}{2}(\Delta\eta_N^T Q_{1N} \Delta\eta_N + \gamma_N^T Q_{2N} \gamma_N + v_{b,N}^T Q_{3N} v_{b,N} \\ & + \omega_N^T Q_{4N} \omega_N + \Delta\Omega_N^T Q_{5N} \Delta\Omega_N) \\ \text{s.t.} \quad & \text{(2b)} - \text{(2e)}. \end{aligned} \tag{11}$$

Here, $\Delta\Omega_i = (\Omega_i - \Omega_{\text{hovering}})$ for $i = 0, \dots, N$ represents the propeller speed errors, where $\Omega_{\text{hovering}} = \sqrt{(mg/4C_T)}$. The positive-definite weighting matrices are denoted by Q_i, Q_{iN} for $i = 1, \dots, 5$ and R_1. In particular, constraint (2d) represents the control interface's real limitation.

Once the solution of (11) is computed, we can use the predicted state at stage $i = 1$ as pilot inputs, namely

$$\phi_h = \phi^\star, \quad \theta_h = \theta^\star, \quad \Omega_h = \Omega^\star, \tag{12}$$

where the superscript $\star$ indicates the optimal solution.

Remark 1. *In practice, the total thrust coming from the joystick can be easily mapped into the propellers' desired speed and passed to the MI controller if one uses Eq.* (10a) *assuming hovering condition, i.e.,* $\Omega_1 = \Omega_2 = \Omega_3 = \Omega_4 = \Omega_{ss}$*, which yields*

$$\Omega_{ss} = \sqrt{\frac{F_{z,h}}{4C_T}}, \quad \Omega_h = \Omega_{ss} \cdot \mathbf{1_4}.$$

The distinction between pilots with a different skill level is made through the weighting matrices of the controller. Flying a quadrotor is less automated for novice pilots and for this reason requires more of their attention span than experienced ones. Due to their limited self-regulatory ability, their inputs tend to be oscillatory (lower values in the weighting matrices). In contrast, the inputs of experienced pilots tend to be more precise (higher values in the weighting matrices), presumably

TABLE I
MIKROKOPTER PHYSICAL PARAMETERS

m	1.04 kg
l	0.23 m
C_D	10 Nm/kHz^2
C_T	595 N/kHz^2
d	0.5 kN·m·s
J_m	0.08 g·m^2
J	diag$(0.01, 0.01, 0.07)$ kg·m^2

reflecting their ability to use perceptual cues to support their actions.

Moreover, as we need to predict human inputs to solve Problem 1, we adopt a zero-order hold (ZOH) method. This prediction method assumes that the future human inputs will all be the same as the current ones, an assumption that proved to be surprisingly effective in experimentation (see [18]).

4) Additional penalty terms: As previously hypothesized, pilots would typically perform continuous maneuvers by changing their inputs all the time. To preserve smoothness in the generated trajectory, we consider additional penalty terms in the MI controller cost function. These terms rely on the following output functions:

$$h(\xi_i) = h(\xi_N) = (\psi; v_b; \omega) \in \mathbb{R}^7 \tag{13}$$

and their corresponding references

$$e_i = e_N = \mathbf{0_7}. \tag{14}$$

The vector of zeros implies two underlying assumptions: first, we are not dealing with aggressive maneuvers; second, we assume it is an educated guess to initialize and, thereby, speed-up the subsequent optimization algorithm.

5) NLPs parameterization: Among several approaches available to discretize continuous-time OCPs, direct multiple shooting [19] will be used in this work due to its convergence and initialization properties. Additionally, we approximate the Hessian of the Lagrangian using the Generalized Gauss-Newton method so that structure-exploiting convex solvers can profit from the particular block-structure of the resulting quadratic program (QP).

This endeavor's time horizon is T, which we divide into N intervals so that we have the discrete-time step $\Delta t = T/N$. Assuming an equidistant grid and piecewise constant control parameterization, the following initial value problem defines the state trajectory $\xi(\pi)$ at each shooting interval:

$$\dot{\xi}(\pi) = f(\pi, \xi(\pi), u_i), \quad \xi(t_i) = \xi_i, \quad \forall \pi \in [t_i, t_{i+1}]. \tag{15}$$

Function (15) is evaluated numerically, $\xi(t_{i+1}) \approx F(\xi_i, u_i)$, using one step of explicit Runge Kutta 4th order per Δt.

B. Implementation details

1) NMPC via real-time iteration scheme: In real-time NMPC applications, Problem 1 needs to be solved at each sampling instant under the available computation time. To that end, we use a numerical strategy based on sequential quadratic programming (SQP) that relies on the solution of a limited number of QP subproblems, the so-called real-time iteration (RTI) scheme [20]. More precisely, only one linearization and QP solve are carried out per sampling instant, leading to an approximate feedback control policy. An essential element in the RTI scheme is to keep the initial state ξ_0 as a constrained decision variable, often referred to as *initial value embedding*. This trait allows one to divide computations into a preparation and feedback phase, where the former is typically more expensive. In this work, we will be using the RTI method's implementation through the high-performance software package `acados` [21]. Through `acados` Python template-based interface, we generate the library that implements Problem 1, which is then wrapped by our framework written in Python.

2) Structure-exploiting QP solver and condensing approach: The QP subproblems arising in the SQP algorithm in `acados` are addressed using the high-performance `HPIPM` [22] solver, which implements a primal-dual interior-point method. It is built on top of the linear algebra package `BLASFEO` [23], finely tuned for multiple CPU architectures. This Riccati-based solver implements an efficient method for the solution of linear-quadratic (LQ) control problems, a special instance of equality constrained QP. Furthermore, `BLASFEO` is hardware-optimized for the moderately sized matrices present in our mixed-initiative NMPC. We use its `X64_INTEL_HASWELL` implementation, which exploits a set of vectorized instructions for the target CPU. Solution times are further reduced by reformulating the QPs resulting from our NMPC using the efficient partial condensing algorithm implemented in `HPIPM`. In particular, the algorithm finds the optimal level of sparsity for the solver, trading off horizon length for input vector size.

C. Performance analysis

The proposed algorithm, that is, the blending mechanism and the mixed-initiative NMPC, use the parameters reported in Table II in all test cases. In particular, we adopt a short virtual horizon N_b to compensate for the fact that with a ZOH method, pilot inputs would have a natural tendency to violate the norm ball's boundary in the long run. The chosen value is a compromise between the base level of "trust" in the human inputs and the switching frequency of control authority. Another assumption made concerns the initialization of the blending mechanism. We assume that during the first iterations $\lambda = 0.5$, notably, none of the agents has control authority. Furthermore, the tests consider pilots with different competence levels. More specifically, the autonomous algorithm emulates a novice and an experienced pilot. This will enable a thorough follow-on comparative analysis of the control algorithm.

Fig. 4 displays the trajectories generated by our mixed-initiative NMPC for both pilots. The results show that our controller was able to maintain the quadrotor inside $\mathcal{B}$, successfully avoiding all obstacles in all cases (see Fig. 6). In Fig. 5, we observe that the mixed-initiative NMPC commands closely follow the pilot inputs when the safety constraint is unlikely to be violated. This outcome is more evident for the experienced pilot, where the lines overlap almost completely, implying that the inputs issued kept the quadrotor closer to the reference trajectory. From Fig. 7, it is obvious that the blending mechanism begins to drop off rapidly the level of control authority for the novice. At the same time, it maintains a relatively high level for the experienced pilot. Overall, note that the control algorithm assists the novice through what is often a very challenging part of their learning journey, the damping in pitching-rolling motions. The algorithm can keep up considerably with the signals' pattern (preserving the pilot's primary intentions), but attenuates their magnitude to cope with the safety constraint. Also, as the physical constraints herein regarded are linear, they can be enforced very efficiently by the optimization algorithm, as shown in Fig. 8.

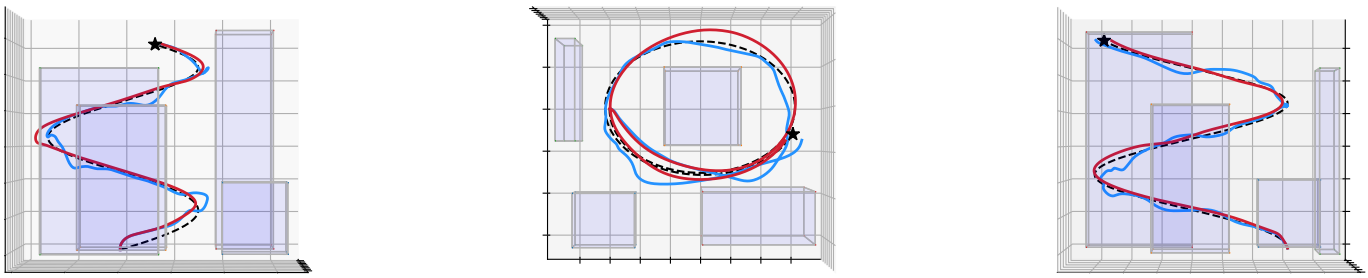
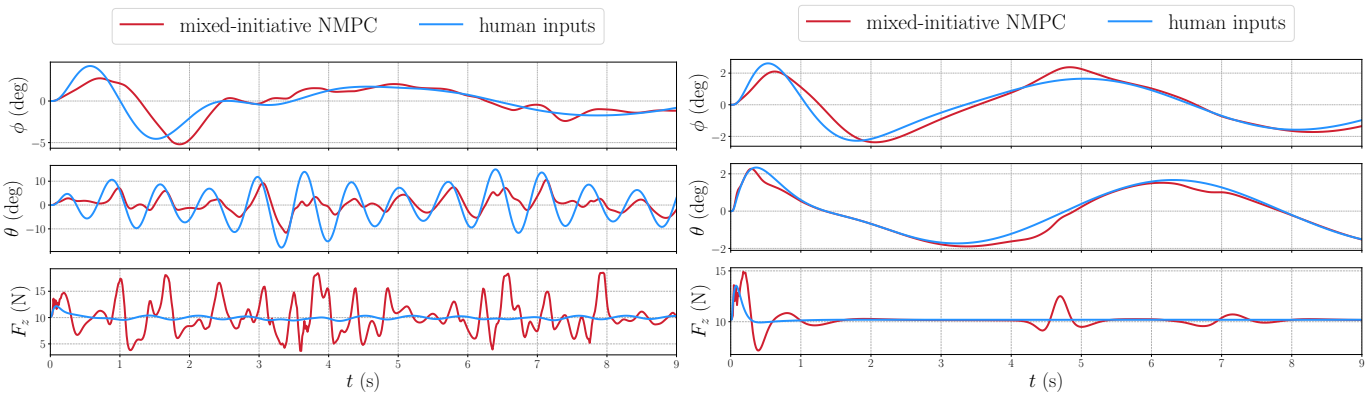


Fig. 4. Generated trajectories among obstacles (purple) using our mixed-initiative NMPC: novice pilot (blue); experienced pilot (red); reference trajectory is dashed; the $\star$ marks the final goal.



(a) Novice. (b) Experienced.

Fig. 5. Pilot inputs generated by an autonomous algorithm and mixed-initiative NMPC commands as a result of the blending mechanism.

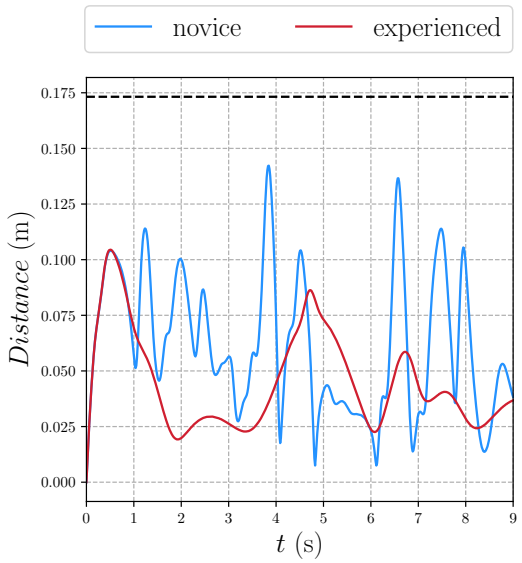


Fig. 6. Distance between generated trajectories and reference; $\|\alpha\|$ is dashed.

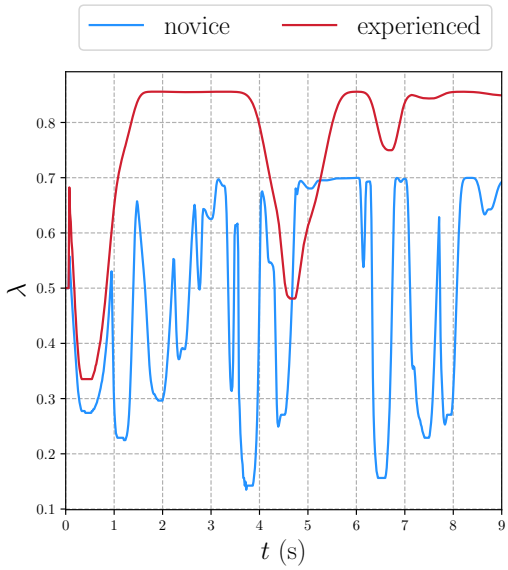
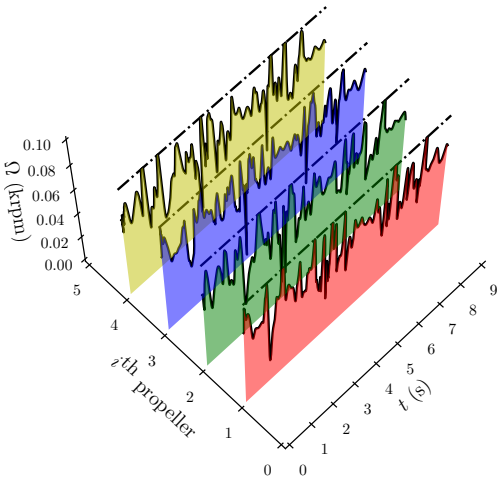


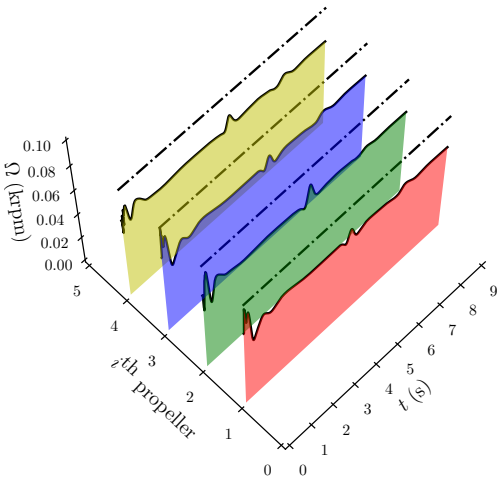
Fig. 7. Predicted healthiness index profiles indicating the level of human control authority.

Including maximum deviations provides versatility in the quadrotor motion generation. First, it gives a certain level of spatial freedom so that the quadrotor can leverage the motion by exploiting its dynamics. Second, it similarly adds new degrees of freedom to the controller. From a theoretic point of view, the control objective of the mixed-initiative NMPC can be seen as a target set (in the space of the task variables) instead of a target point, since inside $\mathcal{B}$ there are no preferences between one point and another [17].

The findings here indicate that our algorithm provides pilots with more control authority without sacrificing safety or even exceeding the robot's physical limits. Therefore, it has the potential to become a standard assistive scheme that can lead to a future improvement in pilot decision-making performance.



(a) Novice.



(b) Experienced.

Fig. 8. Generated propeller speeds; the upper bound $\bar{\Omega}$ is dashed.

TABLE II
PARAMETERS USED BY THE BLENDING MECHANISM AND THE MIXED-INITIATIVE NMPC

$\underline{\Omega}$	0	$\bar{\Omega}$	0.09 kHz
$\underline{u}$	-0.1285 Nm	$\bar{u}$	0.1285 Nm
T	0.75 s	N	50
α	$0.1{\cdot}\mathbf{1_3}$ m	N_b	10
μ_1	0.99	μ_2	0.3
θ_2	50	ζ	$1{\cdot}10^{-13}$

$Q_\eta = Q_{\eta_N} = \text{blkdiag}(250{\cdot}\mathbf{I_2}, 300)$, $Q_\nu = Q_{\nu_N} = 4{\cdot}10^3{\cdot}\mathbf{I_6}$,
$Q_e = Q_{e_N} = \text{blkdiag}(10, 3{\cdot}\mathbf{I_5}, 10)$, $R = 70{\cdot}\mathbf{I_4}$

D. Computational burden

Solution times reported are from an Intel Core i5-4288U @ 2.6 GHz running macOS Catalina. Simulations showed that `HPIPM` is significantly faster in solving the partially condensed QP arising in the NMPC formulation than the corresponding dense QP. The average times per SQP-iteration in `acados` were 4.27 ms for the novice and 3.87 ms for the experienced pilot. It is interesting to note that one can modify N_b and potentially improve the blending mechanism performance without affecting solution times significantly. Fig. 9 shows that all quantities appear to be related to N rather than N_b.

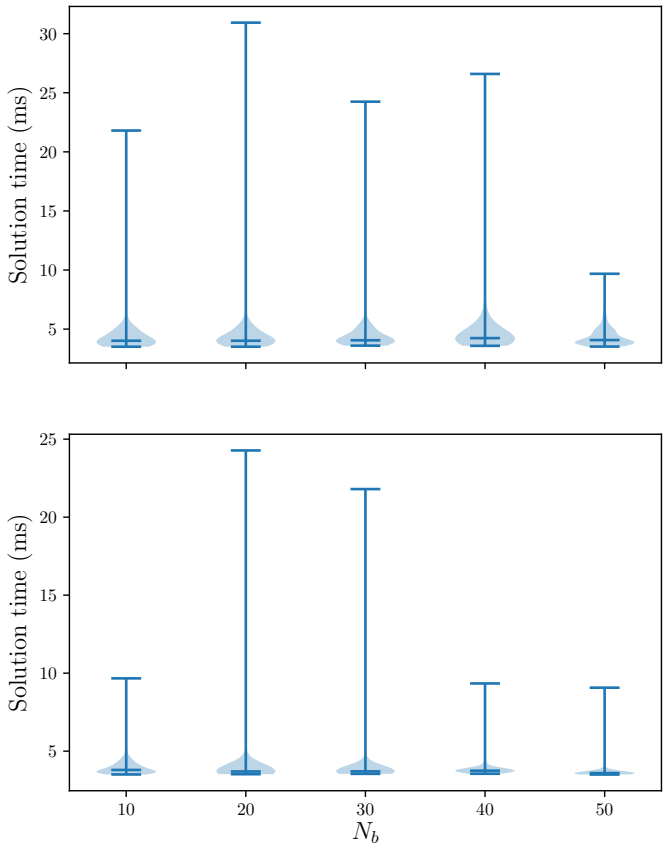


Fig. 9. Violin plot representations of the solution times associated with the mixed-initiative controller in simulations with a novice (top) and an experienced pilot (bottom). Central bars represent the median values for each virtual horizon N_b.

As previously mentioned, partial condensing is an approach in-between the sparse and the condensed one. Here, in particular, the horizon length can be reduced at the expense of a larger input vector. This degree of freedom is used to find the best subproblem size for `HPIPM`. Thus, given the dimensions of the resulting QP subproblems in our controller, computational efficiency is granted through partial condensing as it allows us to exploit hardware throughput better.

V. CONCLUSIONS

In this paper, we have presented a new mixed-initiative NMPC controller for enforcing safety in human-quadrotor interactions. We adopt a predict-then-blend framework to mix human inputs and motion generator commands during operation. Through the quadrotor's predicted positions, we assess whether safety rules are met to perform then the blending, a mechanism designed to continuously measure the violation of the rules. The presented controller uses an efficient RTI-based scheme that iteratively refines the combined commands to fulfill all intrinsic constraints within the available computation

time. In simulation, we evaluate our controller's performance against an autonomous algorithm that emulates the behavior of pilots with different competence levels. The results show that the control scheme allows safety enforcement with specific benefits for useful assistance to pilots, especially novices, and low computational effort. Thanks to high-performance software implementations, we can speed-up our controller's solution times, demonstrating efficient computational performance. Future work will focus on evaluating the pilot's cognitive load through real-world experiments.

REFERENCES

[1] N. Amirshirzad, A. Kumru, and E. Oztop, "Human adaptation to human–robot shared control," *IEEE Transactions on Human-Machine Systems*, vol. 49, no. 2, pp. 126–136, 2019.

[2] D. Rakita, B. Mutlu, M. Gleicher, and L. M. Hiatt, "Shared control–based bimanual robot manipulation," *Science Robotics*, vol. 4, no. 30, 2019.

[3] B. Xi, S. Wang, X. Ye, Y. Cai, T. Lu, and R. Wang, "A robotic shared control teleoperation method based on learning from demonstrations," *International Journal of Advanced Robotic Systems*, vol. 16, no. 4, pp. 1–13, 2019.

[4] X. Bao, V. Molazadeh, A. Dodson, B. E. Dicianno, and N. Sharma, "Using person-specific muscle fatigue characteristics to optimally allocate control in a hybrid exoskeleton–preliminary results," *IEEE Transactions on Medical Robotics and Bionics*, vol. 2, no. 2, pp. 226–235, 2020.

[5] W. M. dos Santos and A. A. Siqueira, "Optimal impedance via model predictive control for robot-aided rehabilitation," *Control Engineering Practice*, vol. 93, pp. 1–8, 2019.

[6] C. Huang, F. Naghdy, H. Du, and H. Huang, "Shared control of highly automated vehicles using steer-by-wire systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 2, pp. 410–423, 2019.

[7] Y. Lu, L. Bi, and H. Li, "Model predictive-based shared control for brain-controlled driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 2, pp. 630–640, 2019.

[8] C. C. Ashcraft, M. A. Goodrich, and J. W. Crandall, "Moderating operator influence in human-swarm systems," in *Proc. 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2019, pp. 4275–4282.

[9] C. Nam, P. Walker, H. Li, M. Lewis, and K. Sycara, "Models of trust in human control of swarms with varied levels of autonomy," *IEEE Transactions on Human-Machine Systems*, vol. 50, no. 3, pp. 194–204, 2019.

[10] R. Rahal, G. Matarese, M. Gabiccini, A. Artoni, D. Prattichizzo, P. R. Giordano, and C. Pacchierotti, "Caring about the human operator: haptic shared control for enhanced user comfort in robotic telemanipulation," *IEEE Transactions on Haptics*, vol. 13, no. 1, pp. 197–203, 2020.

[11] F. Abi-Farraj, C. Pacchierotti, O. Arenz, G. Neumann, and P. R. Giordano, "A haptic shared-control architecture for guided multi-target robotic grasping," *IEEE Transactions on Haptics*, vol. 13, no. 2, pp. 270–285, 2020.

[12] S. Parsa, D. Kamale, S. Mghames, K. Nazari, T. Pardi, A. R. Srinivasan, G. Neumann, M. Hanhaide, and A. Ghalamzan E., "Haptic-guided shared control grasping: collision-free manipulation," in *Proc. 2020 IEEE International Conference on Automation Science and Engineering (CASE)*, 2020.

[13] C. Masone, M. Mohammadi, P. Robuffo Giordano, and A. Franchi, "Shared planning and control for mobile robots with integral haptic feedback," *The International Journal of Robotics Research*, vol. 37, no. 11, pp. 1395–1420, 2018.

[14] A. Franchi, "Human-collaborative schemes in the motion control of single and multiple mobile robots," in *Trends in Control and Decision-Making for Human–Robot Collaboration Systems*. Springer, 2017, pp. 301–324.

[15] B. Xu and K. Sreenath, "Safe teleoperation of dynamic UAVs through control barrier functions," in *Proc. 2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7848–7855.

[16] B. Barros Carlos, T. Sartor, A. Zanelli, G. Frison, W. Burgard, M. Diehl, and G. Oriolo, "An efficient real-time NMPC for quadrotor position control under communication time-delay," in *Proc. 16th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2020, pp. 982–989.

[17] A. Ferramosca, D. Limon, A. H. González, D. Odloak, and E. Camacho, "MPC for tracking zone regions," *Journal of Process Control*, vol. 20, no. 4, pp. 506–516, 2010.

[18] R. Chipalkatty, G. Droge, and M. B. Egerstedt, "Less is more: Mixed-initiative model-predictive control with human inputs," *IEEE Transactions on Robotics*, vol. 29, no. 3, pp. 695–703, 2013.

[19] H. G. Bock and K. J. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," *9th IFAC World Congress*, vol. 17, no. 2, pp. 1603–1608, 1984.

[20] M. Diehl, H. G. Bock, and J. P. Schlöder, "A real-time iteration scheme for nonlinear optimization in optimal feedback control," *SIAM Journal on Control and Optimization*, vol. 43, no. 5, pp. 1714–1736, 2005.

[21] R. Verschueren, G. Frison, D. Kouzoupis, N. van Duijkeren, A. Zanelli, B. Novoselnik, J. Frey, T. Albin, R. Quirynen, and M. Diehl, "acados: a modular open-source framework for fast embedded optimal control," 2019. [Online]. Available: https://arxiv.org/abs/1910.13753

[22] G. Frison and M. Diehl, "HPIPM: a high-performance quadratic programming framework for model predictive control," 2020. [Online]. Available: https://arxiv.org/abs/2003.02547

[23] G. Frison, D. Kouzoupis, T. Sartor, A. Zanelli, and M. Diehl, "BLASFEO: Basic linear algebra subroutines for embedded optimization," *ACM Transactions on Mathematical Software (TOMS)*, vol. 44, no. 4, pp. 1–30, 2018.