Reactive and Predictive Control Scheme for Evasive Maneuvers in Aerial Robots

Jossué Cariño Escobar, Alberto Castillo, Pedro Castillo, Pedro García

Reactive and predictive control scheme for evasive maneuvers in aerial robots

J. Cariño
Université de technologie de Compiegne, France

A. Castillo
Universidad Politécnica de Valencia, Spain.

P. Castillo, Member, IEEE
CNRS, Heudiasyc laboratory, France

P. García
Universidad Politécnica de Valencia, Spain.

***Abstract*— A theoretical control solution to improve navigation performance of an aerial vehicle, focusing on a quadcopter vehicle, is presented in this paper. The main control purpose is to endow the aerial vehicle with predictive and reactive properties (evasive maneuvers) to avoid collisions with dynamic objects (or other aerial agents) that may be in direct collision with it. The proposed control architecture is formed by three modules: a predictor algorithm, an upper-level command generator and a lower-level drone controller. The trajectory predictor algorithm gives a horizon of possible future positions where an object could collide with the aerial vehicle. These predictions are sent to the second module which, based on an artificial particle field, generates velocity commands to avoid possible collisions. These commands are then sent to the drone controller, which was designed using the Lyapunov formalism to guarantee an agile and stable response to the aggressive nature of the problem. The overall architecture was validated and tested in real time with different scenarios and using a cyber-physical twin framework, showing good capabilities to anticipate (predictive properties) and avoid (reactive properties) imminent collisions.**

***Index Terms*—Reactive control; collision avoidance; dynamic environment; predictor; unmanned aerial vehicles**

This work was supported by project FPU15/02008, *Ministerio de Educación y Ciencia*, Spain, the *Ministére de l'Education Nationale, de l'Enseignement Supérieur et de la Recherche* and the National Network of Robotics Platforms (ROBOTEX), from France, and the *Consejo Nacional de Ciencia y Tecnología* from México. *Corresponding author: P. Castillo*. Authors have contributed equally in this work.

A. Castillo and P. García are with the Department of Ingeniería de Sistemas y Automática, Universidad Politécnica de Valencia, Spain. (e-mails: alcasfra@gmail.com, pggil@isa.upv.es). J. Carino and P. Castillo are with the Université de technologie de Compiegne, CNRS, Heudiasyc (Heuristics and Diagnosis of Complex Systems), CS 60319 - 60203, Compiegne, France (e-mail: josuecarinoe@gmail.com, castillo@hds.utc.fr).

I. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) have shown a great potential for many applications, such as: agriculture, rescue, transport, inspection of infrastructures, smart response to disasters or video recording, among others [1], [2]. Since its early beginnings, their construction, equipment and control have been significantly improved [3], [4] making possible to develop novel applications that move from its classical individual passive tasks into new active cooperative tasks, like grasping or manipulation of objects [5], [6], [7], [8]. According to the European Robotics Strategic Research Agenda (eSRA) [9], UAVs are intended to be employed in the near future as workers and co-workers in both private and public activities [8].

A fundamental issue to guarantee a safe integration of UAVs into civil activities is its capability to avoid collisions with other environmental objects [10], [11]. In fact, UAVs will be expected to carry out autonomous tasks in aerial/terrestrial spaces that will be shared with other non-cooperative agents; such as, other autonomous UAVs, Remotely Piloted Aircraft Systems (RPAS), human-driven vehicles, workers doing different tasks or, in general, moving obstacles of any kind. A large-scale implementation of workers UAVs, although having promising features, will not be possible unless effective methods to avoid collisions in open and dynamic environments are developed.

The problem of avoiding collisions between robots, or between a robot and the environmental objects, has been extensively studied in the control literature. The main collision avoidance algorithms can be categorized into two groups: optimization-based methods and reactive controllers [10], [12], [13]. The first ones are characterized by performing an optimization procedure in order to compute a finite-length free-of-collisions trajectory to reach a desired destination. Conversely, the last ones directly generate a control output in response to some received measurements that indicate a possible collision [12].

The most popular optimization-based solutions go from the classics Rapidly-exploring Random Trees (RRT) [14], Probabilistic Roadmaps (PRM) [15], [16] or Curvature-Velocity methods [17]; to some of its recent extensions [18], [19], [20], [21], [22] and other similar solutions that are based on optimal or statistical control [23], [24], [25], [26]. Optimizing feasible trajectories is found to be advantageous for moving in known, highly-complex and closed spaces –such as the interior of buildings, where a wide number of obstacles and infeasible trajectories may exist. Also, it is appropriate for robots with complex configurations or strong moving constraints –such as robotic arms or non-holonomic vehicles, where the generation of a feasible collision-free trajectory could be considerably complex.

However, the optimization-based solutions are probably not the most effective algorithms to avoid collisions

between drones operating in open aerial spaces. In order to compute an optimized collision-free trajectory, the algorithm needs to know the future trajectories that the surrounding UAVs are going to follow [20], [22].

Reactive controllers seem to be a more suitable solution to collision avoidance, as they directly take control actions based on current measurements of the environmental state. A wide number of reactive controllers have been proposed for different kind of robots, sensors configurations and scenarios, e.g. [13], [12], [27], [28], [29], [30], [31], [32]. The most popular approaches are based on Artificial Potential Fields (APF), Force Field (FF) and Velocity Field (VF) methods [33], [34], [35], [36], [37]. The main implementation behind these strategies is quite straightforward: an upper-level controller generates a local repulsive APF/FF/VF in the vicinity of each UAV. Thus, if two or more UAVs –which can be now assumed to move "freely" and independently– get too close to each other, they will be automatically repelled by the artificial fields; maintaining always a safety distance between them.

These repulsive field-based methods, although being quite effective in many cases, contain some drawbacks [38]. The most important one for the problem being considered here –which was also highlighted in [12]– is that they are just static solutions that only consider the current position of each agent. In a highly dynamic environment, where probably some UAVs are moving and accelerating; a single implementation of static repulsive fields could not avoid a collision. Indeed, if a collision is going to happen in less time than the response-time of the UAV inner controller; then –independently of how the repulsive field has been configured– the UAV will not have enough time to modify its velocity; ending in an inevitable crash, see Figure 1.

As can be seen from the references and the state-of-the-art of predictive and reactive control, autonomous collision avoidance in dynamic and open environments is a field of study that requires a roadmap to achieve. We propose, as a first step towards this goal, a novel reactive and predictive controller that allows the drone to avoid being hit by a dynamic object on a direct collision course. It has been assumed that information about the object's and the drone's state is known in order to corroborate the viability of the proposed control strategy.

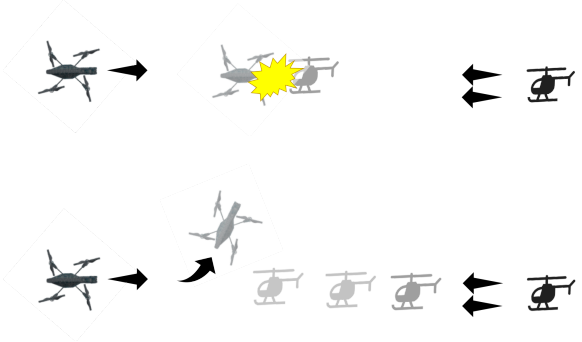

Fig. 1. Two drones are flying with same direction. The right drone is moving two times faster the left drone. Up). Using (robust) controllers without trajectory prediction, the drones are not capable to avoid the collision even if some before instants the static VF generates repulsion commands. Down). The left drone includes the trajectory prediction and then it has estimated the collision horizon of the right drone. As a consequence, it has already started the evasive maneuver by this time and, now, it has enough time to avoid the collision.

The solution is developed based on a novel predictor-algorithm that estimates the future trajectories that other agents[1] –with uncertain or unmodelized motion– may follow in the near future. The horizons of these predicted trajectories are then employed to generate predictive repulsive Velocity Fields (VF) that are sent as desired commands to the drone controller in order to move itself, anticipating and avoiding (imminent) collisions, see Figure 1.

In addition, once the collision horizon is predicted the controller reacts, by making reactive and aggressive movements, to produce evasive maneuvers to be transported far enough of the collision object. This action is only limited by the physical characteristics of the vehicle used.

The idea of using a predictor in this kind of scenario has already been explored in the literature. The three most prominent predictors used are based on a constant-velocity extrapolation [20], [12], on a relatively complex model-based statistical extrapolation [39], [22], or on learning movement patterns [40]. The presented prediction strategy is a viable alternative that is constructed based on ideas of some recent results on disturbed systems and high-order disturbance observers [41], [42]. Our trajectory prediction algorithm is straightforward to implement and is able to make short-time predictions of bodies with unmodelized motion. Additionally, it permits to include model information –if available– in order to enhance the predictions' accuracy.

The rest of the paper is structured as follows: in Section II, the overall reactive and predictive control architecture is developed. It is further subdivided to develop in detail each one of the components of the proposed solution. The predictor algorithm is described in Section A, followed by the velocity-command generator in Section B. Sections C and D illustrate the quadcopter model and the control strategy used to enable aggressive maneuvers. The main experimental results together with link to videos of the tests are presented in Section III. Finally, the conclusions are discussed in Section IV.

II. REACTIVE AND PREDICTIVE CONTROL SCHEME

Our proposed control architecture is summarized in Fig. 2. It can be observed from this figure that this control scheme is composed by a quadcopter system (named *drone* layer), a controller (named *drone controller*), a predictor (named *Predictor*) and the desired references for the control algorithm (named *velocity command generator*).

[1] we use the term agents, including static or dynamics objects and others kind of vehicles

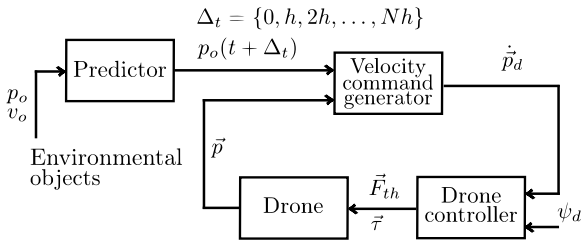

Fig. 2. Block-diagram of the reactive and predictive control architecture. Here, p_o, v_o denote the current position/velocity of the object, $\hat{p}_o(t+\Delta_t)$ represents the predicted obstacle position, Δ_t defines the prediction-horizon, $\dot{\vec{p}}_d$ means the desired velocity of the vehicle, ψ_d is the heading of the drone, $\vec{F}_{th}, \vec{\tau}$ denote the drone's control inputs and $\vec{p}$ defines the drone's position.

A. Trajectory-predictor algorithm

The environment under consideration is characterized by being dynamic and uncertain. This implies that the future position of the moving objects –i.e. the environmental evolution– can only be predicted with certain accuracy for short-time intervals. In this subsection, we propose a short-time trajectory predictor in order to monitor the environmental evolution. Our solution assumes that the current position/velocity of the objects, $p_o(t)$, $v_o(t)$, are available. How this information is obtained is outside the scope of this work, but some practical solutions include embedded sensors, such as cameras, lidar, GPS, etc [43].

Based on this information, a predictor algorithm is constructed from a deterministic perspective in order to make predictions of the future trajectories, i.e. $p_o(t+\Delta t)$. These predictions contain information about the moving intentions and they can be effectively used to anticipate collisions. To further clarify the development of the algorithm, we will develop this algorithm in 2D, however it can be extended to 3D withour any loss of generality.

Denote by $p_o(t) \triangleq p_o \triangleq [p_{o,x}, p_{o,y}] \in \mathbb{R}^2$ (m) the obstacle position, by $v_o(t) \triangleq v_o \in \mathbb{R}^2$ (m/s) its velocity and by $m_o \in \mathbb{R}_+$ (kg) its mass (considering any dynamic rigid-body). Hence, its movement can be determined by the Newton's second law as follows

$$\begin{aligned} \dot{p}_o(t) &= v_o(t), \\ \dot{v}_o(t) &= \frac{1}{m_o} F_o(t) \triangleq \omega_o(t), \end{aligned} \tag{1}$$

being $F_o(t) \triangleq F_o \in \mathbb{R}^2$ the external (unknown) forces acting on it. Notice from (1) that if the object accelerates, it is because there exist some forces (probably unknown/unmeasurable) that are acting on it. If a more accurate model for the obstacle movement is available, it could be used instead of (1). However, Eq. (1) is quite general and accurate enough for short-time predictions.

From (1), the future obstacle position, $p_o(t+\Delta_t)$, is

$$p_o(t+\Delta_t) = p_o(t) + \Delta_t v_o(t) + \int_t^{t+\Delta_t} (t+\Delta_t - s)\omega_o(s) ds \tag{2}$$

Note that Eq. (2) cannot be directly computed in practice because $\omega_o(s)$ is not known for $s \in [t, t+\Delta_t]$. However, consider that $\omega_o(t)$ and its first i-derivatives are *observed* with the available position/velocity measurements. Then, using a Taylor series decomposition, the function $\omega_o(s)$, $s \in [t, t+\Delta_t]$, could be reconstructed as:

$$\omega_o(s) \approx \hat{\omega}_o(s) \triangleq \hat{\omega}_o(t) + (s-t)\hat{\dot{\omega}}_o(t) + \ldots + \frac{(s-t)^n}{n!}\hat{\omega}_o^{(n)}(t) \tag{3}$$

up to a value $n \in \mathbb{N}$ and a sufficiently small Δ_t; being $\hat{\omega}_o^{(n)}(t) \triangleq \hat{\omega}_o^{(n)} \in \mathbb{R}^2$ an observation, or estimation, of $\omega_o^{(n)}(t)$.

Therefore, using (3) in (2), the future obstacle-trajectory could be predicted by:

$$\hat{p}_o(t+\Delta_t) = p_o(t) + \Delta_t v_o(t) + \int_t^{t+\Delta_t} (t+\Delta_t - s)\hat{\omega}_o(s) ds. \tag{4}$$

From current results on disturbed systems and disturbance observers, a high-order disturbance observer can be constructed for system (1) in order to estimate $\boldsymbol{\omega_o}(t) \triangleq [\omega_o^T(t), \dot{\omega}_o^T(t), \ldots, \omega_o^{(n)\,T}(t)]^T$. To this purpose, note that system (1) can be alternatively expressed in the following extended-state form [42], [44]:

$$\begin{aligned} \dot{\eta}(t) &= \bar{A}\,\eta(t) + \bar{B}\,\omega_o^{(n+1)}(t), \\ y_o(t) &\triangleq \bar{C}\eta, \end{aligned} \tag{5}$$

being $\eta(t) \triangleq [p_o^T(t),\, v_o^T(t),\, \boldsymbol{\omega}_o{}^T(t)]^T$ and

$$\bar{A} \triangleq \begin{bmatrix} 0 & I_3 & 0 \\ 0 & 0 & \Pi_1 \\ 0 & 0 & \Phi \end{bmatrix}, \quad \bar{B} \triangleq \begin{bmatrix} 0 \\ 0 \\ \Pi_2 \end{bmatrix}, \quad \bar{C} = [I_3,\, I_3,\, 0]$$

with $\Phi \triangleq \begin{bmatrix} 0_{3n\times 3} & I_{3n} \,;\; 0_{3\times 3} & 0_{3\times 3n} \end{bmatrix}$, $\Pi_1 \triangleq [I_3,\, 0_{3\times 3n}]$ and $\Pi_2 \triangleq [0_{3\times 3n}, I_3]^T$.

Eq. (5) is observable, allowing to construct the next observer that provides estimates of $\boldsymbol{\omega_o}(t)$:

$$\dot{\hat{\eta}}(t) = \bar{A}\,\hat{\eta}(t) + L\left(y_o(t) - \bar{C}\hat{\eta}(t)\right). \tag{6}$$

Provided that $\omega_o^{(n+1)}(t)$ is bounded, the observer (6) estimates $\boldsymbol{\omega_o}(t)$ with bounded error for any L such that $(\bar{A} - L\bar{C})$ is Hurwitz stable [42]. The matrix L can be designed by using the standard pole-placement method so that the eigenvalues of $(\bar{A} - L\bar{C})$ contain a sufficiently high bandwidth.

REMARK 1. *The embedded system into the drone computes the equations (3), (4) and (6) in order to predict the future trajectory that each obstacle may follow. To this purpose, it computes (3)-(4) for $\Delta_t = \{0, h, 2h, \ldots, Nh\}$; being $h \in \mathbb{R}$ a fixed step-size and $N \in \mathbb{N}$ an integer defining the prediction-horizon, as illustrated in Fig. 3. The prediction horizon, Nh, can be regarded as a tuning parameter. For our application, it can be set equal to the response-time of the drone controller; so that it anticipates possible collisions that will not be avoided by the static VF.*

REMARK 2. *The observer order is also a tuning parameter. The bigger it is, the more derivatives of $\omega_0(t)$ are observed. Equations (5)-(6) provide a useful clue to select*

this parameter as they show that the error $\eta(t) - \hat{\eta}(t)$ is driven by $\omega_0^{(n+1)}(t)$. It is interesting to chose n so that $\omega_0^{(n+1)}(t)$ is kept small. Thus, for obstacles that are moving in high-chaotic form –i.e. with high-frequency accelerations– $\omega^{(n+1)}(t)$ grows as n is increased. In these cases, it is better to reduce n. However, for obstacles that move with relatively-low accelerations; increasing n contributes to reduce the observation error.

B. Velocity-command generator

This subsection proposes a velocity-command generator to set escape-velocities if a possible collision could happen. The challenge is to provide a safe autonomous navigation in dynamic environments. It is assumed that most of the moving objects can be classified as either static or slowly-moving, or quick agents that could potentially collide with the vehicle.

The algorithm implements a local static and a predictive repulsive velocities fields –VFs. The static VF is just generated with the current position of the obstacle; whereas the predictive VF depends on the predicted trajectory of the obstacle. The solution is constructed so that, if no immediate impact is detected by the predictor subsystem, the overall architecture behaves as a conventional static VF; which tries to keep a safety distance with all the objects. However, if an immediate collision is detected –i.e. a collision that may happen in less time than the response time of the drone controller– then, the predictive VF generates an anticipated reaction in order to increase the amount of time that the UAV has to react.

Our solution follows the classical approaches that employ security zones around the robot defined by a security radius, as depicted in Figure 3. Two cylinders around the drone position, $p(t)$, are defined: a big outer cylinder of radius r_{max} and a small inner cylinder of radius r_{min}. However, other geometrical forms like spheres, cubes or boxes can also be considered to define the different security zones used for the strategy. In practice, we found that cylinders work best as spheres tended to generate a reference velocity that could send the drone too close to the ground or too far up for practical and illustration purposes. The cubes and boxes have too much volume around the drone and would prematurely trigger the predictive action.

On the one hand, the outer security zone generates (static) repulsive velocities, $\overrightarrow{v_s}(t) \in \mathbb{R}^2$, if the obstacle position, $p_0(t)$, falls within it. This repulsive velocity is expressed as:

$$\overrightarrow{v_s}(t) \triangleq \begin{cases} -\,k_{11}\left[r_{max} - \|d(t)\|\right]^2 \overrightarrow{u_s}(t), & \|d(t)\| \leq r_{max}, \\ 0, & \|d(t)\| > r_{max}; \end{cases} \tag{7}$$

where $k_{11} \in \mathbb{R}_+$ is a positive constant gain, $d(t) \triangleq p_o(t) - p(t)$ and $\overrightarrow{u_s}(t) \triangleq \frac{p_0(t)-p(t)}{\|p_0(t)-p(t)\|}$, being $p_o(t)$ and $p(t)$ the obstacle and drone coordinates, respectively.

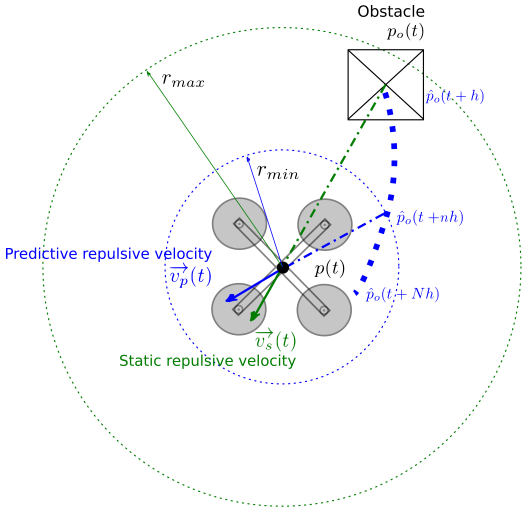

Fig. 3. Illustrative scheme of the reactive and predictive control architecture.

On the other hand, the inner security zone produces (predictive) repulsive velocities, $\overrightarrow{v_p}(t) \in \mathbb{R}^2$, if the predicted object trajectory, given by Eqs. (3)-(4), (6), with $\Delta_t = \{0, h, 2h, \ldots, Nh\}$, intersects to the inner cylinder at some point, $\hat{p}_0(t + n\,h)$ with $n \leq N$; as illustrated in Figure 3. This predictive velocity is proposed as:

$$\overrightarrow{v_p}(t) \triangleq \begin{cases} -\left[k_{21}(N-n)^2 + k_{22}\right] \overrightarrow{u_p}(t), \\ 0, \quad \nexists\, n \leq N \,:\, \hat{p}_o(t + n\,h) \leq r_{min} \end{cases} \tag{8}$$

being $\overrightarrow{u_p}(t) \triangleq \frac{\hat{p}_0(t+n\,h)-p(t)}{\|\hat{p}_0(t+n\,h)-p(t)\|}$ and k_{21}, k_{22} positive constant parameters.

Observe that (8) depends quadratically on $(N-n)$, in this way, it quadratically increases as the time to collision, nh, is reduced. This represents a natural behavior as far as higher repulsive velocities are desired for smaller time-to-collision.

These repulsive velocities (7) and (8) are sent to the drone control system as follows:

$$\dot{\vec{p}}_d := \dot{\vec{p}}_d\,(t) = \overrightarrow{v_s}(t) + \overrightarrow{v_p}(t), \tag{9}$$

where $\dot{\vec{p}}_d$ denotes the drone desired velocity.

If the drone is tracking position commands –instead of velocity commands– then the desired position is modified by adding the following term:

$$\vec{p}_d = \int_{t_0}^{t} \dot{\vec{p}}_d\,(\tau)\; d\tau. \tag{10}$$

Note that these desired references may induce fast and high-angle maneuvers that the drone needs to apply. These unforeseen movements should appropriately carried out by the drone control system. So it is needed to include a drone controller for position/velocity tracking that has the capability of performing agile, fast and high-angle robust maneuvers. The next section proposes a quaternion-based controller to this purpose.

C. Drone system

The quadcopter vehicle is a multi-rotor with parallel motors, its underactuated dynamics can be expressed using the quaternion formulation as

$$\begin{aligned} m\,\ddot{\vec{p}}_e &= \boldsymbol{q}\otimes\vec{F}_{q_b}\otimes\boldsymbol{q}^* + m\,\vec{g} + \vec{\zeta}_p \\ J\,\dot{\vec{\Omega}}_e &= \vec{\tau} - \vec{\Omega}\times J\,\vec{\Omega} + \vec{\zeta}_{\Omega_e} \end{aligned} \tag{11}$$

where $\vec{p}_e \in \mathbb{R}^3$ denotes the error position vector of the system w.r.t. a local ENU inertial reference frame and is defined as $\vec{p}_e \coloneqq \vec{p} - \vec{p}_d$, where $\vec{p}_d$ represents the desired position. Note that the desired position was previously defined for 2D, but it can be extended to 3D without any loss of generality. $m \in \mathbb{R}_+$ and $J \in \mathbb{R}_+^{3\times3}$ are the system's mass and inertia tensor, respectively. $\vec{\Omega}$ represents the angular velocity w.r.t. the body frame. The effect of the gravity on the system is denoted by $\vec{g} \in \mathbb{R}^3$, $\vec{\Omega}_e = \vec{\Omega} - \vec{\Omega}_d \in \mathbb{R}^3$ defines the angular velocity error, $\vec{\Omega}_d \in \mathbb{R}^3$ is the desired angular velocity, $\vec{F}_{q_b} = [0, 0, \sum_i f_i]^T \in \mathbb{R}^3$ denotes the main thrust (or main control input). The thrusts $f_i \in \mathbb{R}_+$, produced by the motor $i \in \{1,2,3,4\}$ are assumed to depend on the motor's angular velocities and usually expressed in the z-body axis. $\vec{\tau} = [\tau_\phi, \tau_\theta, \tau_\psi]^T \in \mathbb{R}^3$ are the torques produced in the aerial vehicle. $\boldsymbol{q} \in \mathbb{H}$ is the system's attitude in unit quaternion form and $\boldsymbol{q}^*$ denotes the quaternion conjugate term. The vector $\vec{\zeta}_k \in \mathbb{R}^3; \forall k \in \left\{\vec{p}, \vec{\Omega}_e\right\}$ represents nonlinear uncertainties, unmodeled dynamics, and/or external and unknown perturbations, like drag effects and/or wind turbulence.

Notice that (11) is an underactuated system, attractive for the control and robotic community and a lot control algorithms have been proposed for stabilizing it. The design of the control law starts with the assumption that system (11) is fully actuated, by changing the effect on the thrust. Thus, even though the system is an underactuated platform, it can be analyzed and a control law can be designed for it from this "virtual full-actuated" transformation. This changes the form of (11) as :

$$m\,\ddot{\vec{p}}_e = \vec{F}_u + m\,\vec{g} + \vec{\zeta}_p \tag{12}$$

$$J\,\dot{\vec{\Omega}}_e = \vec{\tau} - \vec{\Omega}\times J\,\vec{\Omega} + \vec{\zeta}_{\Omega_e} \tag{13}$$

where $\vec{F}_u = [F_x, F_y, F_z]^T \in \mathbb{R}^3$ is an independent control law stabilizing (12).

Observe that (12)-(13) can be seen as two different actuated systems, nevertheless they are linked by a desired quaternion, $\boldsymbol{q}_d$, and related with a proposed control law $\vec{F}_u \in \mathbb{R}^3$ and with the constant thrust vector direction $\vec{F}_{q_b} \in \mathbb{R}^3$. Therefore, for a given controller $\vec{\tau}$ stabilizing the attitude of the vehicle and doing $\boldsymbol{q} \to \boldsymbol{q}_d$ will imply that $\boldsymbol{q}\otimes\vec{F}_{q_b}\otimes\boldsymbol{q}^* \to \vec{F}_u$.

Hence, the quaternion error is defined as $\boldsymbol{q}_e \coloneqq \boldsymbol{q}_d^*\otimes\boldsymbol{q}$, while the desired quaternion is proposed as

$$\boldsymbol{q}_d = \boldsymbol{q}_{f_d}\otimes\boldsymbol{q}_{\psi_d} \tag{14}$$

where the attitude quaternions $\boldsymbol{q}_{f_d} \in \mathbb{H}$ and $\boldsymbol{q}_{\psi_d} \in \mathbb{H}$ are defined in the following form using the quaternion natural logarithm and exponential functions [45, Chapter 5]

$$\boldsymbol{q}_{f_d} \coloneqq e^{\frac{\ln\left(\hat{F}_u\otimes\hat{F}_{q_b}^*\right)}{2}}; \qquad \boldsymbol{q}_{\psi_d} \coloneqq e^{\frac{\psi_d\hat{F}_{q_b}}{2}} \tag{15}$$

where $\hat{F}_j \coloneqq \vec{F}_j/\|\vec{F}_j\|; \quad \forall j \in \{u, q_b\}$ represents a normalized vector and ψ_d denotes the desired heading of the vehicle. In addition, $\boldsymbol{q}_{f_d}$ describes the shortest path between $\boldsymbol{q}\otimes\vec{F}_{q_b}\otimes\boldsymbol{q}^*$ and $\vec{F}_u$ and is related with the pitch and roll movement.

The desired angular velocity can be obtained from (14) and is denoted as

$$\vec{\Omega}_d = \boldsymbol{q}_{\psi_d}^*\otimes\vec{\Omega}_{f_d}\otimes\boldsymbol{q}_{\psi_d} + \vec{\Omega}_{\psi_d} \tag{16}$$

where $\vec{\Omega}_{f_d}$ denotes the angular velocity for pitch and roll angles and $\vec{\Omega}_{\psi_d}$ for the heading of the vehicle.

D. Drone controller

For designing the controllers, propose the following Lyapunov candidate functions as

$$V_{p_e} = \overline{x}_p^T\,P_p\,\overline{x}_p \tag{17}$$

$$V_{q_e} = \overline{x}_\Omega^T\,P_\Omega\,\overline{x}_\Omega \tag{18}$$

with $\overline{x}_p \coloneqq \begin{bmatrix} \vec{p}_e^T & \dot{\vec{p}}_e^T \end{bmatrix}^T$ and $\overline{x}_\Omega \coloneqq \begin{bmatrix} (2\ln\boldsymbol{q}_e)^T & \vec{\Omega}_e^T \end{bmatrix}^T$ and $P_p, P_\Omega \in \mathbb{R}^{6\times6}$ are positive real symmetric matrices. The velocity error can be defined as $\dot{\vec{p}}_e = \dot{\vec{p}} - \dot{\vec{p}}_d$, where $\dot{\vec{p}}$ is the vehicle's velocity w.r.t. the inertial frame.

Differentiating (17) w.r.t. time, and proposing the position control law $\vec{F}_u$ as

$$\vec{F}_u = -m\,K_{pt}\,\vec{p}_e - m\,K_{dt}\,\dot{\vec{p}}_e - m\vec{g} \tag{19}$$

where $K_{pt}, K_{dt} \in \mathbb{R}^{3\times3}$ are the gain matrices, it follows that

$$\dot{V}_{p_e} = -\bar{x}_p^T\,Q_p\,\bar{x}_p + \frac{\partial V_{p_e}}{\partial\dot{\vec{p}}_e}\frac{\vec{\zeta}_p}{m}$$

with $Q_p > 0$ and

$$\begin{bmatrix} \vec{0} & -K_{pt} \\ I_{3\times3} & -K_{dt} \end{bmatrix} P_p + P_p \begin{bmatrix} \vec{0} & I_{3\times3} \\ -K_{pt} & -K_{dt} \end{bmatrix} = -Q_p \tag{20}$$

Therefore

$$\dot{V}_{p_e} \leq -\bar{x}_p^T\,Q_p\,\bar{x}_p + \|P_p\|\,\|\overline{x}_p\|\,\frac{\left\|\vec{\zeta}_p\right\|}{m} \tag{21}$$

Remark that, due mainly to its physical properties, aerial vehicles cannot generate infinity energy to compensate infinitely external perturbations. This signifies that the external perturbations affecting the aerial robot are assumed to be bounded. Remember that for our study, the perturbations are considered to be uniform Lipschitz continuous, implying that

$$\left\|\vec{\zeta}_p\right\| \leq L_p\left\|\begin{bmatrix} \vec{p}_e \\ \dot{\vec{p}}_e \end{bmatrix}\right\| \leq L_p\,\|\overline{x}_p\| \tag{22}$$

for some $L_p \in \mathbb{R}_+$. From (21), the above implies that

$$\dot{V}_{p_e} \leq -\left[\|Q_p\| - \|P_p\|\,\frac{L_p}{m}\right]\|\overline{x}_p\|^2 \tag{23}$$

which implies that $\dot{V}_{p_e} \leq 0$ if

$$\|Q_p\| > \|P_p\| \frac{L_p}{m} \tag{24}$$

A similar analysis can be done for the attitude dynamics with (18) and using the following algorithm

$$\vec{\tau} = J\left[-K_{p\vartheta}\left(2\ln \boldsymbol{q}_e\right) - K_{d\vartheta}\,\vec{\Omega}_e\right] + \vec{\Omega} \times J\vec{\Omega}, \tag{25}$$

From the quaternion algebra the following property for $\vec{\Omega}_d$ can be obtained

$$\left\|\vec{\Omega}_d\right\| \leq L_{\vec{\Omega}_d} \|2\ln \boldsymbol{q}_e\| \tag{26}$$

where $L_{\vec{\Omega}_d} \in \mathbb{R}_+$ is a positive real number. Therefore, similarly to (22), it follows that

$$\left\|\vec{\zeta}_{\Omega_e}\right\| \leq L_{\vec{\zeta}_{\Omega_e}} \left\|\begin{bmatrix} 2\ln \boldsymbol{q}_e \\ \Omega_e \end{bmatrix}\right\| \tag{27}$$

for some $L_{\vec{\zeta}_{\Omega_e}} \in \mathbb{R}_+$. Then the gains matrices, $K_{p\vartheta}, K_{d\vartheta} \in \mathbb{R}^{3\times 3}$ are chosen such that

$$\begin{bmatrix} \vec{0} & -K_{p\vartheta} \\ I_{3\times 3} & -K_{d\vartheta} \end{bmatrix} P_\Omega + P_\Omega \begin{bmatrix} \vec{0} & I_{3\times 3} \\ -K_{p\vartheta} & -K_{d\vartheta} \end{bmatrix} = -Q_\Omega \tag{28}$$

with $Q_\Omega > 0$. Therefore, this results in

$$\dot{V}_{q_e} \leq -\left[\|Q_\Omega\| - \|P_\Omega\| \left\|J^{-1}\right\| L_{\vec{\zeta}_{\Omega_e}}\right] \|\overline{x}_\Omega\|^2 \tag{29}$$

which implies that $\dot{V}_{q_e} \leq 0$ if

$$\|Q_\Omega\| > \|P_\Omega\| \left\|J^{-1}\right\| L_{\vec{\zeta}_{\Omega_e}}. \tag{30}$$

Observe that controllers (19) and (25) stabilize robustly the quadcopter dynamics (12)-(13) for any mission, even for the trajectory tracking or navigation purposes (using $\vec{p}_d$ and $\dot{\vec{p}}_d$). Note also that the repulsive velocities (7) and (8) included in $\dot{\vec{p}}_d$, see equation (9), may induce strong "undesirable" and "uncertain" dynamics into the system, that the controller must tolerate.

The "uncertain" dynamics come from the sudden response needed from the drone's controller in reaction to the predictive system. The prediction, due to its very nature, is extremely sensitive and can produce sudden velocity commands, which translates into an almost instantaneous reference change that the controller must follow. If the controller's response time or robustness is not adequate, it could induce undesirable dynamics, like oscillations, into the system.

III. EXPERIMENTAL TESTS: REACTIVE DRONE

This section illustrates the main advantages that our predictive control strategy has over other conventional collision avoidance controllers that just employ static VFs –such as (7). As previously mentioned, one of the main disadvantages of the static solutions is that they may not avoid collisions with fast moving objects due to the delay introduced by the inner system dynamics.

An experimental scheme was tested in order to have an objective comparison between the proposed control strategy against the static VFs. A person is tasked to attempt to damage the vehicle using a stick while the drone is ordered to remain in a hover position and only move to prevent being hit by the stick. See Figures 4 and 5 for an illustration of the configuration. The top of the stick simulates the moving object with uncertain motion that needs to be avoided, and the fact that a person is moving it makes the movements far more unpredictable. In addition, we chose in this scenario, to use a human hitting the aerial vehicle, to compare the reactive and bio-inspired performance of the drone with respect to a fly when a human try to catch it (the fly reacts very quickly). Nevertheless, other kind of scenarios can be proposed.

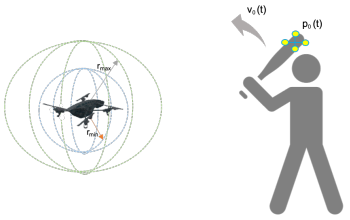

Fig. 4. Scenario for validating the proposed reactive and predictive control.

A. Experimental platform

The overall control scheme is implemented in the drone embedded system. The quadrotor used for the experiments is an AR Drone with an embedded ARM Cortex A8 processor at 1 GHz, running Linux 2.6.32, it also has a 1GB DDR2 RAM at 200 MHz, the sampling frequency is 200 Hz. The navigation sensors were an Inertial Measurement Unit, as well as an OptiTrack Motion Capture system for position feedback. The motion capture reference frame uses a local NED frame, but it is transformed into a local ENU frame in order to be compatible with the proposed control strategy. The hardware of the quadrotor has been conserved from its factory design, nevertheless the original software has been completely replaced by our framework FL-Air (Framework libre AIR) C++ libraries, such that custom programs can be run,

Fig. 5. Overall scenario for evaluating the performance of the proposed architecture. A user tries to hit and dammage the aerial vehicle.

accessing all states and variables that usually can not be modified with a brand new Parrot.

In this particular case, the tests are performed indoors and most of the state information for the vehicle and the obstacle is acquired with the aid of an external motion capture system. This does not undermine the results, as any noise in the vehicle's state can be mitigated using a wide array of filtering techniques, such as a Kalman filter. The Optitrack system measures the stick position, $p_o(t)$, and its velocity, $v_o(t)$, and sends it to the drone[2]. The drone predicts the future stick trajectory according to Remark 1 with $h = 0.01$ (sec) and $N = 50$. The security zones are defined with $r_{min} = 1.5$ m and $r_{max} = 2.5$ m. The velocity-command generator gains ((7) and (8)) are set to $k_{11} = 10$, $k_{21} = 0.002$ and $k_{22} = 2.5$. They were tuned experimentally and can be used in other scenarios if physical drone parameters or sensors are not changed. The observer is designed with $n = 1$ and L =[1.5, 1.5, 1.5, -0.2, -0.2, -0.2, -13.6, -13.6, -13.6, 130.4, 130.4, 130.4; 0.1, 0.1, 0.1, 4.7, 4.7, 4.7, 107.3, 107.3, 107.3, 1115.9, 1115.9, 1115.9]^T.

The quaternion-based control laws, (19) and (25), have been experimentally tuned using a circle reference trajectory with constant speed. The gains obtained were $m\,K_{pt} = I_3\,0.4$, $m\,K_{dt} = I_3\,0.3$, $J\,K_{p\vartheta} = I_3\,1.5$ and $J\,K_{d\vartheta} = I_3\,0.1$. In order to have a fair comparison, the same gains and platform was used for all test scenarios.

REMARK 3. *Obviously, this strategy could be used in other type scenarios, such as: surveillance, building inspections, trajectory tracking, or even when the vehicle is moving. For a practical implementation in other environments, it is just needed to add the desired references* (9) *or* (10) *to the specific ones that are being considered.*

B. Experimental tests

In what follows, different experiments are reported in order to illustrate how the proposed control architecture behaves. Three scenarios are proposed for better illustrating the behavior of the components of the control architecture.

Scenario A: the mobile object is moving with a slow velocity in the direction of the aerial robot. The drone tries to keep a safety distance with respect to the object. The goal here is to illustrate the performance of the static VF ($\vec{v}_s(t)$ in equation (7)). This term is generated whenever the obstacle falls within the cylinder defined by r_{max} see Fig. 3. Observe in Fig. 6-up that the vehicle tries to maintain a safety distance of 2.5 m with respect to the object in order to keep safe. A video of this experiment can be seen at https://youtu.be/LovW4c2FGSA.

The bottom plot, in Fig. 6, depicts the lateral components of the reference and drone velocities, $v_{s,y}(t)$, $\dot{p}_y(t)$, respectively. It is seen that the drone response-time is about 0.5 s, i.e. there exist a time-delay of approximately 0.5 s between the reference signal and the drone velocity. This delay, which is a normal consequence due to the nature of the system's dynamics, makes the vehicle unable to avoid collisions with sufficiently fast-moving objects. In fact, if the same experiment is repeated in a more aggressive environment (where the object moves in the drone's direction with a velocity $\dot{p}_{o,y} > \frac{r_{max}}{0.5}$), it is found that the collision will not be avoided only with the static term, $\vec{v}_s(t)$.

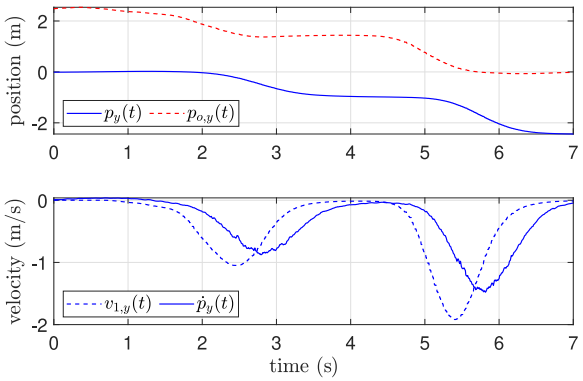

Fig. 6. Data extracted from the maneuver of Video 1.

Scenario B: in this scenario a human tries to hit quickly the aerial robot. A video of this experiment can be seen at https://youtu.be/eGGn5cgoINs. The bottom video represents the system response when only the term $\vec{v}_s(t)$ is considered; whereas top video represents the system response when only the predictive-term, $\vec{v}_p(t)$, is considered. In this scenario, observe that when only the term $\vec{v}_s(t)$ is used, the drone is not fast-enough to avoid a possible impact. In fact, the user needs to stop the hit in order to not damage the vehicle. However, when using the predictive term, $\vec{v}_p(t)$, the collision is avoided. This can be clearly seen in Figure 7, which contains the data extracted from this experiment. Note that the predictive-term starts generating repulsive velocities *before* the static-term. This predictive nature allows start the maneuver before so that the drone has time-enough to avoid the impact.

Observe also from Figure 7 that this fast maneuver implies big roll angles, $\phi \approx 80^o$ and $\phi \approx 50^o$. Both movements are done without losing the vehicle stability. This is possible as far as a quaternion-based controller is being used in the closed-loop system. A linear controller based on Euler angles could not be capable to hold the stability in these kind of maneuvers.

The experiment of scenario B clearly illustrates the main role of $v_p(t)$, which is: i) generating anticipated repulsive velocities if an immediate impact is detected; ii) doing nothing if no impact is detected. A third experiment was carried out using scenario B, whose main goal is to illustrate in a graphical form the behavior of the predictive term. A third video of this experiment can be seen at https://youtu.be/AjbeMfaNUt0. The computed

[2]The perception system only measures the position, but the velocity is computed onboard the embedded system using a Kalman filter.

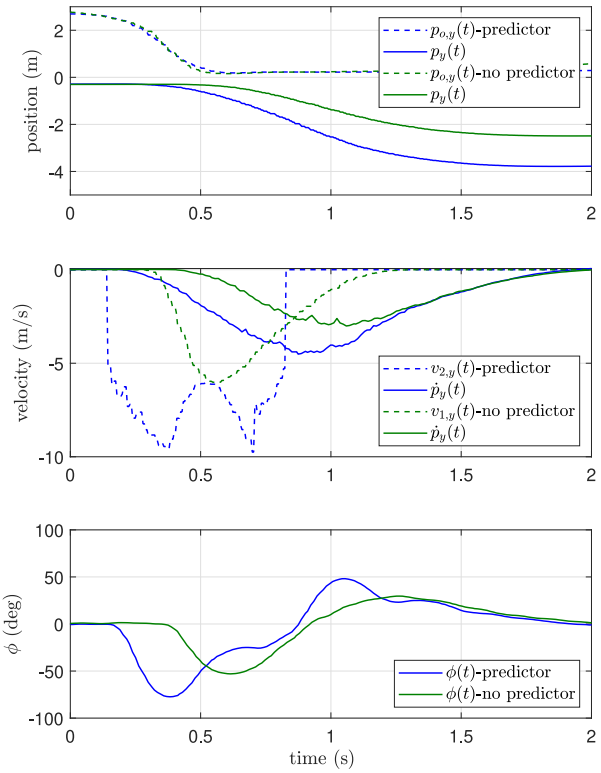

Fig. 7. Extracted from the maneuver of Video 2.

predictions, $p_o(t+\Delta_t)$, are overlapped into the video and the prediction $p_o(t+n\,h)$ intersecting to the inner cylinder, is highlighted with a red circle. Observe that, as soon as the obstacle starts moving in the drone direction with fast velocity, the computed predictions rapidly intersect to the inner cylinder generating anticipated repulsive velocities.

Figs. 8-9 contain the extracted data of this experiment. Fig. 8 represents, in the $x-z$ plane, the stick and the drone trajectories as well as the computed predictions at four time-instants. Observe in this figure how the proposed predictor extrapolates the current trajectory. This predictor allows to extrapolate curve trajectories of unmodelized motions in a very simple form that would not be possible with other approaches. On one hand, predictions based on statistical approaches tends to increase their variance the further steps into the future one takes. This translates into an increasingly bigger volume that would prematurely trigger the drone's reaction. On the other hand, a model with constant velocity restricts the movement of the trajectory to lines and omits the effect of second order dynamics on the system.

Fig. 9 represents the extracted data in the $x-y$ plane where, additionally, the security cylinders have been included. Note in the upper-right plot that, as soon as the obstacle movement has been initiated, the computed predictions are intersecting to the inner cylinder *before* they have reached the bigger cylinder. This generates an anticipated reaction that allows the drone controller to have some additional tenths of second to modify its velocity so that, now, the collision can be avoided.

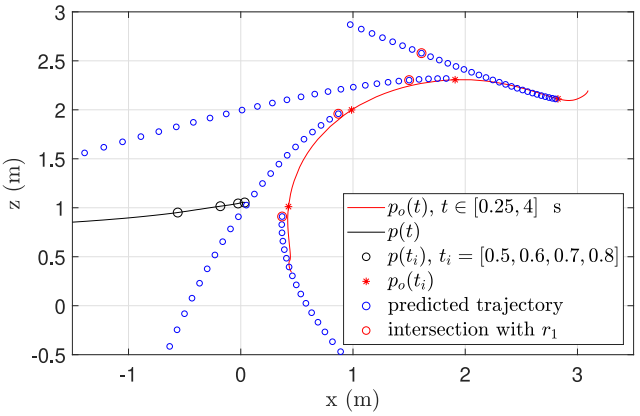

Fig. 8. Extracted from the maneuver of Video 3.

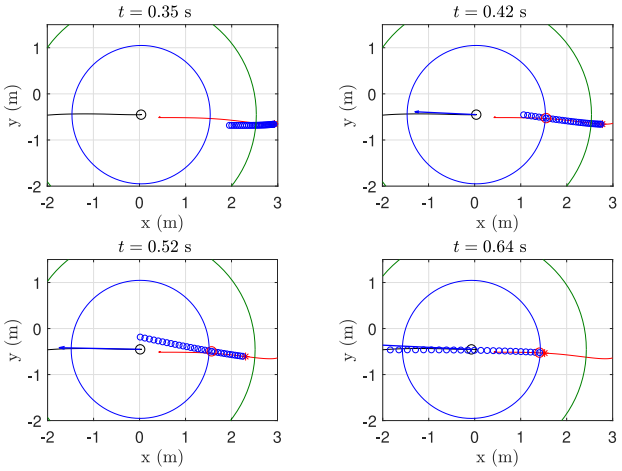

Fig. 9. Extracted from the maneuver of Video 3. Plot $x-y$

Finally, a last scenario C is proposed to evaluate the predictor behavior with respect to false intentions to hit the drone. The main purpose is to show that the aerial vehicle is, in fact, analyzing the future objects trajectories and, if none affects to its safety, then none reaction is commanded. This experiment can be seen at https://youtu.be/B2DbJ9c28I4. Observe in the video that the vehicle does not move if the obstacle goes slowly or in another direction. It only reacts whenever it detects an immediate impact. Similarly to Fig. 9, Fig. 10 contains the extracted data during the maneuver at the 25 seconds mark in the video; where a hit was directed in another direction that will not cause a collision. In this case, it can be seen how the computed predictions do not intersect to the inner cylinder and, thus, no anticipated escape velocity is generated as this object movement will not impact to the drone.

IV. Conclusions

A reactive and predictive control scheme for evasive maneuvers in aerial robots was proposed in this paper. This reactive scheme endows to the aerial robot the capabilities of performing predictive evasive maneuvers if an imminent collision may happen.

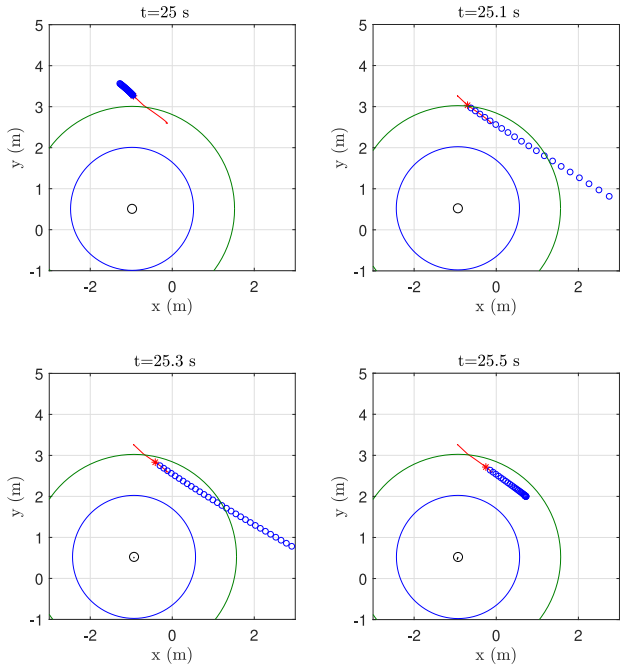

Fig. 10. Extracted from the maneuver of scenario B. Plot $x - y$

The solution is composed by three modules: a novel trajectory predictor algorithm, a velocity-command generator and a robust quaternion-based controller. The trajectory predictor was constructed from a simple model-free perspective based on some recent results on disturbed systems and disturbance observers and it is able to make short-time predictions of bodies with unmodeled/uncertain motion. The velocity-command generator was then constructed by combining a conventional static repulsive Velocity-Fields (VF) with a novel predictive VF in order to anticipate for imminent collisions. The predictive VF is found to be quite useful to compensate for the inherent delay caused by the system dynamics. The quaternion-based controller was designed by using the Lyapunov theory.

The overall architecture was experimentally tested in an aerial drone whose security was compromised. The experiments shown the main advantages that the insertion of the predictive term brings to the conventional solutions based on repulsive VF. Future works could be focused on implementing this strategy in aerial aerial robots navigating in environments where other non-cooperative agents are evolving.

Observe that our main goal for the presented work was indeed to develop a theoretical control solution to improve navigation behavior of an aerial vehicle, focusing on a quadcopter vehicle, when it navigates in the presence of non-cooperative agents. From a control point of view, the algorithm was conceived for having a good performance even in open and dynamic environments and in the presence of non-cooperative agents that are moving randomly. Note that, we proposed a practical proof of concept (cyber-physical twin) of the theoretical solution. This practical validation gave us a reasonable prediction of the reliability for analyzing future performances and allowing to adapt the algorithm to future behaviors or scenarios.

The presented work is a step in the direction of autonomous navigation in dynamic environments. Some of the possible research venues that could be explored include the addition of more objects. Because the presented velocity-command is based on VFs, it is possible to present local minima under certain conditions, but this can be avoided by improving the velocity-command algorithm.

Other scenarios can also be considered to test its viability, including outdoor tests. These would require the inclusion of more sensors like cameras, laser rangefinders, acoustics sensors, etc. for detecting obstacles.

REFERENCES

[1] T. Rakha, A. Gorodetsky, Review of unmanned aerial system (UAS) applications in the built environment: Towards automated building inspection procedures using drones, Automation in Construction 93 (2018) 252–264.

[2] M. Hassanalian, A. Abdelkefi, Classifications, applications, and design challenges of drones: A review, Progress in Aerospace Sciences 91 (2017) 99–131.

[3] T. P. Nascimento, M. Saska, Position and attitude control of multi-rotor aerial vehicles: A survey, Annual Reviews in Control 48 129–146.

[4] H. Shraim, A. Awada, R. Youness, A survey on quadrotors: Configurations, modeling and identification, control, collision avoidance, fault diagnosis and tolerant control, IEEE Aerospace and Electronic Systems Magazine 33 (7) (2018) 14–33.

[5] H. Wu, X. Tao, N. Zhang, X. Shen, Cooperative UAV cluster-assisted terrestrial cellular networks for ubiquitous coverage, IEEE Journal on Selected Areas in Communications 36 (9) (2018) 2045–2058.

[6] A. Khan, B. Rinner, A. Cavallaro, Cooperative robots to observe moving targets, IEEE transactions on cybernetics 48 (1) (2016) 187–198.

[7] A. Yamashita, T. Arai, J. Ota, H. Asama, Motion planning of multiple mobile robots for cooperative manipulation and transportation, Transactions on Robotics and Automation 19 (2) (2003) 223–237.

[8] F. Ruggiero, V. Lippiello, A. Ollero, Aerial manipulation: A literature review, IEEE Robotics and Automation Letters 3 (3) (2018) 1957–1964.

[9] E. Robotics, Strategic research agenda for robotics in europe 2014–2020, IEEE Robot. Autom. Mag 24 (2014) 171.

[10] A. Pandey, S. Pandey, D. Parhi, Mobile robot navigation and obstacle avoidance techniques: A review, International Robotics & Automation Journal 2 (3) (2017) 1–12.

[11] V. Tran, F. Santoso, M. Garratt, Adaptive trajectory tracking for quadrotor systems in unknown wind environments using particle swarm optimization-based strictly negative imaginary controllers, IEEE Transactions on Aerospace and Electronic Systems 57 (3) (2021) 1742 –1752.

[12] S. Choi, E. Kim, K. Lee, S. Oh, Real-time nonparametric reactive navigation of mobile robots in dynamic environments, Robotics and Autonomous Systems 91 (2017) 11–24.

[13] S. Choi, E. Kim, S. Oh, Real-time navigation in crowded dynamic environments using gaussian process motion control, in: International Conference on Robotics and Automation (ICRA), IEEE, 2014, pp. 3221–3226.

[14] S. LaValle, Rapidly-exploring random trees : a new tool for path planning, The annual research report.

[15] F. F. Arias, B. Ichter, A. Faust, N. M. Amato, Avoidance critical probabilistic roadmaps for motion planning in dynamic environ-

ments, in: 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021, pp. 10264–10270.
[16] G. Chen, N. Luo, D. Liu, Z. Zhao, C. Liang, Path planning for manipulators based on an improved probabilistic roadmap method, Robotics and Computer-Integrated Manufacturing 72 (2021) 102196.
[17] R. Simmons, The curvature-velocity method for local obstacle avoidance, in: Proceedings of IEEE international conference on robotics and automation, Vol. 4, IEEE, 1996, pp. 3375–3382.
[18] S. Karaman, E. Frazzoli, Sampling-based algorithms for optimal motion planning, The international journal of robotics research 30 (7) (2011) 846–894.
[19] S. Chakravorty, S. Kumar, Generalized sampling-based motion planners, IEEE Transactions on Systems, Man, and Cybernetics 41 (3) (2011) 855–866.
[20] Y. Lin, S. Saripalli, Sampling-based path planning for UAV collision avoidance, IEEE Transactions on Intelligent Transportation Systems 18 (11) (2017) 3179–3192.
[21] Y. Kuwata, J. Teo, S. Karaman, G. Fiore, E. Frazzoli, J. How, Motion planning in complex environments using closed-loop prediction, in: AIAA Guidance, Navigation and Control Conference and Exhibit, 2008, p. 7166.
[22] G. S. Aoude, B. D. Luders, J. M. Joseph, N. Roy, J. P. How, Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns, Autonomous Robots 35 (1) (2013) 51–76.
[23] T. B. Wolf, M. J. Kochenderfer, Aircraft collision avoidance using monte carlo real-time belief space search, Journal of Intelligent & Robotic Systems 64 (2) (2011) 277–298.
[24] D. H. Shim, S. Sastry, An evasive maneuvering algorithm for UAVs in see-and-avoid situations, in: American Control Conference, IEEE, 2007, pp. 3886–3891.
[25] J. Van Den Berg, D. Wilkie, S. J. Guy, M. Niethammer, D. Manocha, Lqg-obstacles: Feedback control with collision avoidance for mobile robots with motion and sensing uncertainty, in: 2012 IEEE International Conference on Robotics and Automation, IEEE, 2012, pp. 346–353.
[26] D. Bareiss, J. Van den Berg, Reciprocal collision avoidance for robots with linear dynamics using LQR-obstacles, in: International Conference on Robotics and Automation, IEEE, 2013, pp. 3847–3853.
[27] J. Alonso-Mora, P. Beardsley, R. Siegwart, Cooperative collision avoidance for nonholonomic robots, IEEE Transactions on Robotics 34 (2) (2018) 404–420.
[28] F. Belkhouche, Reactive path planning in a dynamic environment, Transactions on Robotics 25 (4) (2009) 902–911.
[29] K. Cole, A. M. Wickenheiser, Reactive trajectory generation for multiple vehicles in unknown environments with wind disturbances, Transactions on Robotics 34 (5) (2018) 1333–1348.
[30] V. Sezer, M. Gokasan, A novel obstacle avoidance algorithm: Follow the gap method, Robotics and Autonomous Systems 60 (9) (2012) 1123–1134.
[31] E. Baklouti, N. B. Amor, M. Jallouli, Reactive control architecture for mobile robot autonomous navigation, Robotics and Autonomous Systems 89 (2017) 9–14.
[32] J. Lin, H. Zhu, J. Alonso-Mora, Robust vision-based obstacle avoidance for micro aerial vehicles in dynamic environments, in: 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 2682–2688.
[33] N. Lu, W. Zhou, H. Yan, M. Fei, Y. Wang, A two-stage dynamic collision avoidance algorithm for unmanned surface vehicles based on field theory and colregs, Ocean Engineering 259 (2022) 111836.
[34] Y. Li, J. Zheng, Real-time collision avoidance planning for unmanned surface vessels based on field theory, ISA Transactions 106 (2020) 233–242.
[35] J. Wubben, C. T. Calafate, J.-C. Cano, P. Manzoni, Ffp: A force field protocol for the tactical management of uav conflicts, Ad Hoc Networks 140 (2023) 103078.
[36] C. Y. Kim, Y. H. Kim, W.-S. Ra, Modified 1d virtual force field approach to moving obstacle avoidance for autonomous ground vehicles, Journal of Electrical Engineering & Technology 14 (3) (2019) 1367–1374.
[37] H. J. Asl, T. Narikiyo, An assistive control strategy for rehabilitation robots using velocity field and force field, in: 2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR), 2019, pp. 790–795.
[38] Q. Yao, Z. Zheng, L. Qi, H. Yuan, X. Guo, M. Zhao, Z. Liu, T. Yang, Path planning method with improved artificial potential field—a reinforcement learning perspective, IEEE Access 8 (2020) 135513–135523.
[39] D. Fridovich-Keil, A. Bajcsy, J. F. Fisac, S. L. Herbert, S. Wang, A. D. Dragan, C. J. Tomlin, Confidence-aware motion prediction for real-time collision avoidance1, The International Journal of Robotics Research 39 (2-3) (2020) 250–265.
[40] D. Vasquez, T. Fraichard, O. Aycard, C. Laugier, Intentional motion on-line learning and prediction, Machine Vision and Applications 19 (5-6) (2008) 411–425.
[41] K.-S. Kim, K.-H. Rew, S. Kim, Disturbance observer for estimating higher order disturbances in time series expansion, IEEE Transactions on automatic control 55 (8) (2010) 1905–1911.
[42] A. Castillo, P. García, R. Sanz, P. Albertos, Enhanced extended state observer-based control for systems with mismatched uncertainties and disturbances, ISA transactions 73 (2018) 1–10.
[43] D. Falanga, K. Kleber, D. Scaramuzza, Dynamic obstacle avoidance for quadrotors with event cameras, Science Robotics 5 (40).
[44] A. Castillo, P. Garcia, Predicting the future state of disturbed LTI systems: A solution based on high-order observers, Automatica.
[45] J. P. Morais, S. Georgiev, W. Sprößig, Quaternions and Spatial Rotation, Springer Basel, Basel, 2014, pp. 35–51.

BIOGRAPHIES

Jossué Cariño Escobar received his Ph.D. in Automatic Control from the Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional - CINVESTAV, Mexico. He also has a MS in Autonomous Aerial and Underwater Navigation Systems from the same institution. He made a postdoctoral position at the Heudiasyc laboratory in France and currently he is at the ONERA in France.His research concentrates on cooperative control, autonomous vehicles and their mathematical models, and simultaneous localization and mapping (SLAM).

Alberto Castillo was born in Valencia, Spain, in 1992. He received his B.S. degree in Industrial Engineering from the School of Industrial Engineers, Polytechnic University of Valencia (UPV), Valencia, Spain, in 2013. He has been working in the Systems Engineering and Control Department, UPV, since 2014. In 2016, he received his M.S. in Industrial Engineering with a major in Process Control from the School of Industrial Engineers, UPV. He received his Ph.D. fellowship, FPU15/02008, in 2016; a project in which he is currently working towards the Ph.D. degree. He has been research collaborator at the Institute of Cyber-Systems and Control, Zhejiang University, China; at the Universidade Federal de Santa Catarina, Brazil; and at the Heudiasyc Laboratory, Universite de Technologie de Compiegne, France. His current research interests are focused on control of disturbed/uncertain systems, on quadrotor control algorithms and machine learning techiques.

Pedro Castillo (Member, IEEE), received the B.S. degree in electromechanics engineering from the Instituto Tecnologico de Zacatepec (Mexico) in 1997, the M. Sc. degree in electrical engineering from the Centro de Investigación y de Estudios Avanzados (CINVESTAV) (Mexico) in 2000, and the Ph.D. degree in automatic control from the University of Technology of Compiegne (France) in 2004. His research topics include real-time control applications, nonlinear dynamics and control, aerospace vehicles vision, and underactuated mechanical systems. He has obtained his HDR (Habilitation à Diriger des Recherches) degree from the University of Technology of Compiègne, France in 2014. He has held several visiting positions at the University of Sydney, Australia, at the Massachusetts Institute of Technology (MIT), at the Polytechnic University of Valencia, Spain, at Virginia Tech, USA. He has held a detachment position at the LAFMIA UMI CNRS 3175 CINVESTAV -IPN in Mexico, from December 2012 to November 2014. At the moment, he is a DR2 researcher at the French National Research Foundation (CNRS), in the Laboratory Heudiasyc, at the University of Technology of Compiègne, France. His research topics cover : non-linear control, delays systems and predictors, observers, autonomous vehicles, robust navigation, real-time control, robotics.

Pedro Garcia received the Ph.D. degree in Computer Science from the Universitat Politecnica de Valencia, Spain, in 2007, where he is currently an Associate Professor within the Department of Systems Engineering and Control. He has been a visiting researcher at the Lund Institute of Technology, Lund, Sweden, at Université de Technologie de Compiègne, France, at University of Florianopolis, Brazil, at the University of Sheffield, UK, and at the University of Zhejiang, Hangzhou, China. He has coauthored one book, and more than 80 refereed journal and conference papers. His current research interests include control of time-delay systems, disturbance observers and control in type 1 diabetes.