Mechatronics 49 (2018) 187–196

Contents lists available at ScienceDirect

Mechatronics

journal homepage: www.elsevier.com/locate/mechatronics

The development of an autonomous navigation system with optimal control of an UAV in partly unknown indoor environment☆

Thi Thoa Mac*[a,b], Cosmin Copot[c], Robin De Keyser[a], Clara M. Ionescu[a]

[a] DySC research group on Dynamical Systems and Control, Ghent University, Technologiepark 914, Zwijnaarde 9052, Belgium
[b] School of Mechanical Engineering, Hanoi University of Science and Technology, Dai Co Viet street 1, Hanoi, Vietnam
[c] Department of Electromechanics, Op3Mech, University of Antwerp, Groenenborgerlaan 171, Antwerp 2020, Belgium

ARTICLE INFO

Keywords:
Unmanned aerial vehicles (UAVs)
Sensor fusion
Autonomous navigation
Optimal control
Multi-objective particle swarm optimization

ABSTRACT

This paper presents an autonomous methodology for a low-cost commercial AR.Drone 2.0 in partly unknown indoor flight using only on-board visual and internal sensing. Novelty lies in: (*i*) the development of a position-estimation method using sensor fusion in a structured environment. This localization method presents how to get the UAV localization states (position and orientation), through a sensor fusion scheme, dealing with data provided by an optical sensor and an inertial measurement unit (IMU). Such a data fusion scheme takes also in to account the time delay present in the camera signal due to the communication protocols; (*ii*) improved potential field method which is capable of performing obstacle avoiding in an unknown environment and solving the non-reachable goal problem; and (*iii*) the design and implementation of an optimal proportional - integral - derivative (PID) controller based on a novel multi-objective particle swarm optimization with an accelerated update methodology tracking such reference trajectories, thus characterizing a cascade controller. Experimental results validate the effectiveness of the proposed approach.

1. Introduction

In the last few years, Unmanned Aerial Vehicles (UAVs) stir up both scholar and commercial interest within the robotics community as the real and potential applications are numerous [1]. To undertake the challenging task of autonomous navigation and maneuvering, a versatile flight control design is required.

A large number of studies have emerged in the literature on UAVs. Some examples of its application can be found in precision agriculture [2], formation control of Unmanned Ground Vehicles (UGVs) using an UAV [3], habitat mapping [4]. Modeling, identification and control of an UAV using on-board sensing are presented in [5]. Catching a falling object using a single UAV, has been accomplished in [6] and for a group of UAVs in cooperative formation in [7], where high-speed external cameras were applied to estimate the position of both the objects and UAVs. Simultaneous localization and mapping (SLAM) was implemented to navigate UAV in working space [8]. Current implementations in UAV still require collision avoidance, adaptive path-planing and optimal controller. There exists a need to design methodologies to cope with these requirements to increase the degree of intelligence and therefore autonomy of UAV.

An autonomous UAV consists of four essential requirements: (i) *perception*, the UAV uses its sensors to extract meaningful information; (ii) *localization*, the UAV determines its pose in the working space; (iii) *cognition and path planning*, the UAV decides how to steer to achieve its target; (iv) *motion control*, the UAV regulates its motion to accomplish the desired trajectory.

The path planning problem can be divided into classical methods and heuristic methods [9]. The most important classical methods consist of cell decomposition method (CD), potential field method (PFM), subgoal method (SG) and sampling-based methods. Heuristic methods include neural network (NN), fuzzy logic (FL), nature inspired methods (NIM) and hybrid algorithms. The potential field method (PFM) is particularly attractive since it has a simple structure, low computational complexity and easy to implement. In literature, there has been a significant amount of work based on this method applied to ground agents path planning [10–13]. An interesting work on implementing and flight testing of this approach on an UAV is studied in [14]. To operate in real-time, a layered approach is developed in uncharted terrain: plan globally and react locally. The global planner is based on an implementation of Laplace equation that generates a potential function with a unique minimum at the target. The local planner uses modification of

☆ This paper was recommended for publication by Associate Editor Dr. Lei Zuo.

* Corresponding author.
E-mail addresses: Thoa.MacThi@ugent.be (T.T. Mac), Cosmin.Copot@uantwerpen.be (C. Copot), Robain.Dekeyser@ugent.be (R.D. Keyser), ClaraMihaela.Ionescu@ugent.be (C.M. Ionescu).

https://doi.org/10.1016/j.mechatronics.2017.11.014
Received 18 August 2017; Received in revised form 8 November 2017; Accepted 29 November 2017

conventional potential field method in which not only the position of the UAV (as in the traditional PFM) but also the relative angles between the goal and obstacles are taken into account. However, this approach sometimes encounters problems when the repulsion from obstacles exceeds the physical constraints of the UAV. It is pointed out that the potential field method has several inherent limitations [15] in which the non-reachable target problem is the most serious one and is worth investigating since it causes an incomplete path in the navigation task.

As an UAV is a complex system in which electromechanical dynamics is involved, the robust controller is an essential requirement. In [16], the dynamical characteristics of a quadrotor are analyzed to design a controller which aims to regulate the posture (position and orientation) of the quadrotor. An autonomous control problem of a quadrotor UAV in GPS-denied unknown environments is studied [17,18]. In order to obtain reasonable dynamical performance, guarantee security and sustainable utilization of equipment and plants, controller performance has to be constantly optimal.

In the current study, a real-time implementation for an AR. Drone 2.0 UAV autonomous navigation in indoor environment is proposed to trigger its identification, able to estimate the UAV pose, detect obstacles, generate the suitable path and to perform the parametric optimization of its optimal proportional-integral-derivative (PID) controller. The main contributions are the development of: (*i*) a position-estimation method based on sensor fusion using only on-board visual and inertial sensing considering the time delay of the camera signal and reducing drift errors; (*ii*) a solution to solve the non-reachable target problem in conventional PFM; (*iii*) multi-objective optimization PID controller based on a proposed multi-objective particle swarm optimization (MOPSO) with an accelerated update methodology to execute navigation task. The motivation behind this research is to illustrate that autonomous navigation is feasible on low-cost UAV devices.

This paper is structured as follows: the next section gives a description of AR.Drone 2.0, identification, system setup and localization. Section 3 discusses UAV path planning based on improved potential field method. Multi-objective particle swarm optimization algorithm for control parameters optimization and simulation results are described in detail in Section 4. Next, the effectiveness of the proposed real-time collision-free path planning for an AR. Drone 2.0 UAV using only on-board visual and inertial sensing application in indoor environment is presented in Section 5. The final section summarizes the main outcome of this contribution and presents the next challenges.

2. System setup, identification and localization

A description of the Ar.Drone 2.0 main characteristics, system identification, sensory equipment, system setup and localization are presented in this section.

2.1. Ar.Drone 2.0 description and coordinates system

There are four basic motions of this UAV: pitch, roll, throttle, yaw and translational movements over *x, y* and *z*, as shown in Fig. 1 **(Left)**.

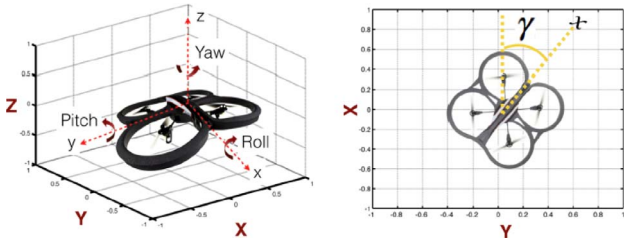


Fig. 1. The movements of an AR.Drone 2.0 in absolute and relative planes **(Left)** and UAV displacement on (*x; y*) plane respect to the absolute plane **(Right)**.

It is worth mentioning that the coordinate system described above (*x; y; z*), represents a relative coordinate system used by the internal controllers (low layer). Using such a coordinate system instead of absolute coordinates (*e.g., X; Y; Z*) in the high layer will yield errors. For example, notice that by rotating the quadrotor, the relative coordinates (*x; y*) will change with respect to the absolute coordinates, as depicted in Fig. 1 **(Right)**. In which, the rotation angular of *XY* coordinate system respect to the absolute *xy* coordinate system is γ. It is possible to state that the relation between the two-coordinate system depends directly of this angle.

The IMU provides the software with pitch, roll and yaw angle measurements. Communication between Ar.Drone and a command station is performed via Wi-Fi connection within a 50 m range. AR.Drone 2.0 is equipped with two cameras in the bottom and in frontal parts with the resolutions of 320 × 240 pixels at 30 frames per second (fps) and 640 × 360 pixels at 60 fps, respectively.

2.2. Analysis of inputs and outputs and system identification

The developed Software Development Kit (SDK) mode allows the quadrotor to transmit and receive the information roll angle (rad), pitch angle (rad), the altitude (m), yaw angle (rad) and the linear velocities on longitudinal/transversal axes (m/s). They are denoted by $\{\theta_{out}, \phi_{out}, \zeta_{out}, \psi_{out}, \dot{x}, \dot{y}\}$ respectively. The system is executed by four inputs $\{V_{in}^{x}, V_{in}^{y}, \dot{\zeta}_{in}, \dot{\psi}_{in}\}$ which are the linear velocities on longitudinal/ transversal axes, vertical speed and yaw angular speed references as depicted in Fig. 2.

An Ar. Drone is a multi-variable and naturally unstable system. However, due to the internal low layer control implemented in the embedded operative system, it is considered as a Linear Time Invariant (LTI) System, which is able to be decomposed into multiple single input single output (SISO) loops. Transfer functions are obtained via parametric identification using the prediction error method (PEM) and Pseudo-Random Binary Signal (PRBS) input signals [19]. A sampling time of 5 ms for yaw and 66 ms for other degrees of freedom are chosen based on the analysis of dynamics characteristic. The identified transfer functions are given in Eq. (1).

Validation of transfer function of pitch/roll, altitude and yaw are presented in Fig. 3. The validation of the transfer function is made against a different set of data to prove that quadrotor movements are approximated appropriately.

$$\begin{aligned} H_x(s) &= \frac{x(s)}{V_{in}^{x}(s)} = \frac{7.27}{s(1.05\,s+1)} \\ H_y(s) &= \frac{y(s)}{V_{in}^{y}(s)} = \frac{7.27}{s(1.0fs+1)} \\ H_{altitude}(s) &= \frac{\zeta_{out}(s)}{\dot{\zeta}_{in}(s)} = \frac{0.72}{s(0.23\,s+1)} \\ H_{yaw}(s) &= \frac{\psi_{out}(s)}{\dot{\psi}_{in}(s)} = \frac{2.94}{s(0.031\,s+1)} \end{aligned} \tag{1}$$

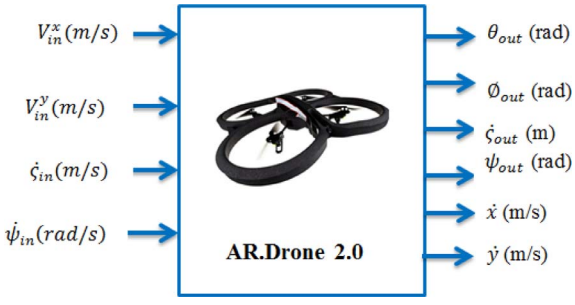
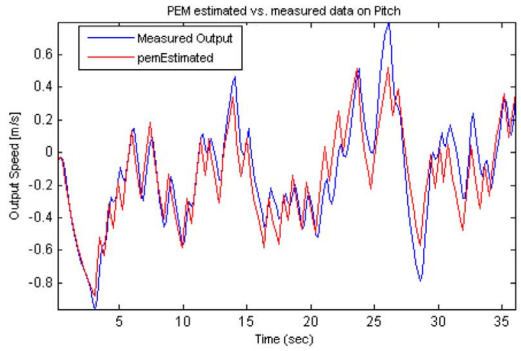
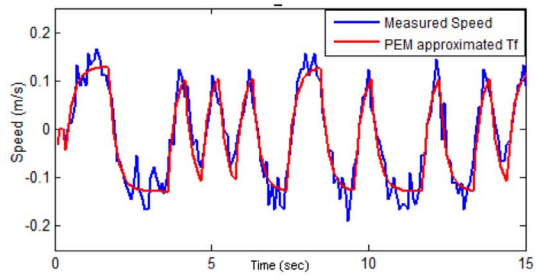


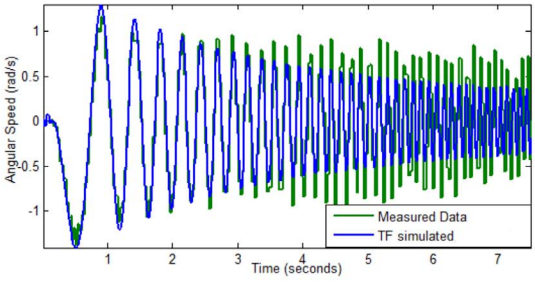
Fig. 2. Inputs and Outputs of an AR.Drone 2.0.



(a) The comparison of the PEM-estimated pitch/roll transfer function response and measured data (speed in m/s)



(b) The comparison of the PEM-estimated altitude transfer function response and measured data (speed in m/s)



(c) Response of PEM-estimated yaw angular speed transfer function compared to measured yaw angular speed for a chirp input signal

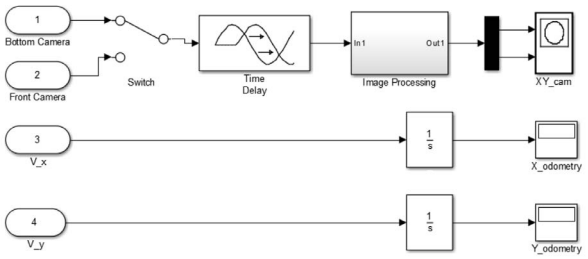
Fig. 3. Validation of pitch/roll (a), altitude (b), yaw (c) transfer function of an AR.Drone 2.0.

2.3. System setup and localization using sensor fusion

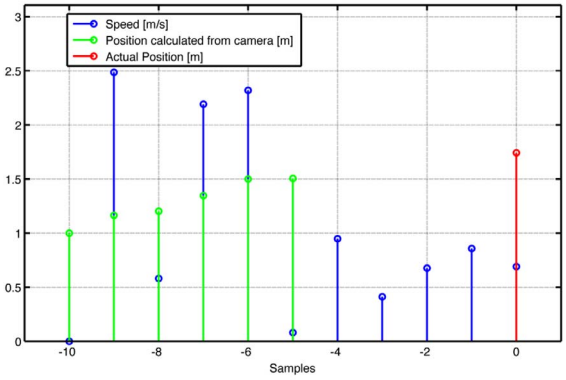
In our approach, the localization based on the sensor fusion approach which fuses data from ground patterns and IMU data is shown in Fig. 4a. A time delay is presented in the camera signal due to the communication protocols. Time delay depends directly on the resolution of the camera, i.e: 0.33 second approximately for a size of 320×240 pixels. Fig. 4b explains the information channels for cameras and IMU speed sensors. The sensor fusion localization allows Ar. Drone 2.0 to determine its location and orientation in a working space.

Fig. 5 presents all components of our navigation approach. The first component is sensor fusion and the second one is a cascade control, which guides the quadrotor to follow the designed trajectories.

First, the velocity data of the drone in the local coordinate system are transformed into the world coordinate system, then the position is retrieved by using Euler integration:



(a) Sensors for localization of the AR.Drone in the x, y plane.



(b) Position sensor combination to eliminate the time delay on video signals. green: position calculated from video signal with time delay of 5 samples (ycam), blue: quadrotor speed (y) and red: final calculated position (y)

Fig. 4. Optical sensor and IMU for localization of the Ar.Drone 2.0.

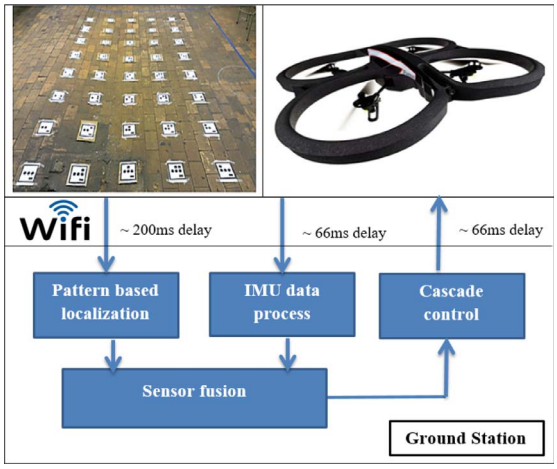


Fig. 5. Our navigation approach of an AR.Drone 2.0.

$$velocity_{world} = velocity_{local} * \begin{bmatrix} cos(yaw) & -sin(yaw) \\ sin(yaw) & cos(yaw) \end{bmatrix}$$
$$pos_{world} = prevpos_{world} + velocity_{world} \triangle time \quad (2)$$

The bottom camera uses a grid of ground patterns to estimate the pose of the drone. Each pattern inside this grid represents (x, y) coordinates which are calculated based on the information in the first and second rows. Each row includes three bits, the white and black ones are corresponding to '0' and '1'. Fig. 6 represents the position of the ground pattern with $(x = 1, y = 3)$ coordinates calculated as below:

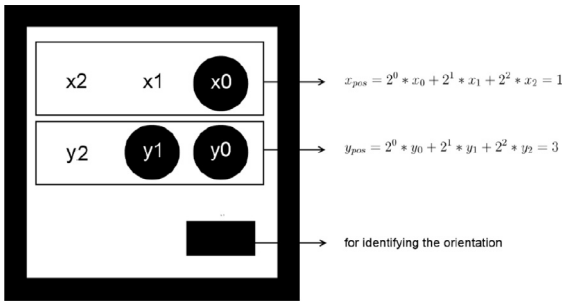


Fig. 6. A Ground pattern representing position (x = 1, y = 3).

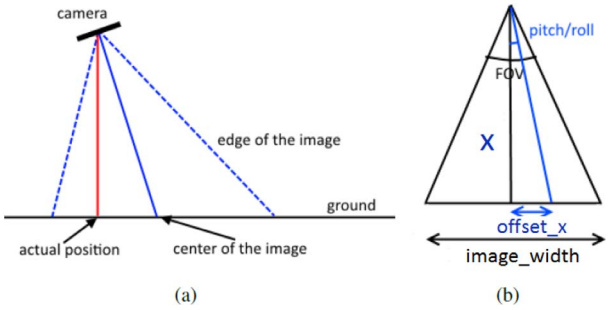


Fig. 7. (a) Center of the image versus the actual position of the quadrotor; (b) Calculation of the offset to the center.

$$x = 2^0x_0 + 2^1x_1 + 2^2x_2$$
$$y = 2^0y_0 + 2^1y_1 + 2^2y_2 \quad (3)$$

The rectangle at the right bottom of the pattern is used to approximate the orientation of the quadrotor. The distance between two patterns is 50 cm to ensure that there is always at least one pattern visible in one image.

Due to the quadrotor having different pitch angles during flight, the center of the image from the bottom camera is not always pointing perpendicular to the ground (Fig. 7(a)). Therefore, a correction to this center needs to be made using the information from the on board pitch sensors. This offset is dependent on the pitch/roll angles of the quadrotor and the field of view (FOV) of the camera which can be calculated using trigonometry rules as illustrated in Fig. 7(b). In order to correct the offset for x and y, the following relationships can be used:

$$\begin{aligned} offset_x &= \frac{tan(roll)}{x} \\ tan(FOV/2) &= \frac{image_{width}/2}{x} \\ offset_x &= \frac{2*tan(roll)*tan(FOV/2)}{image_{width}} \\ offset_y &= \frac{2*tan(pitch)*tan(FOV/2)}{image_{height}} \end{aligned} \quad (4)$$

The image processing algorithm is depicted in flowchart as shown in Fig. 8 in which the input image is converted into gray image, then applied a suitable threshold to find the contours. This threshold depends on the light condition and therefore has to be appropriately chosen.

In this work, the proposed solution to achieve a reliable position estimation consists of combining the information from the two sensors. In order to reduce the drift effect and noise, IMU is used to read the variations and the optical sensor is used to find an offset-free measurement. The simplest and functional combination consists of using the optical sensor only when the standard deviation of the last five samples obtained from odometry is bigger than a tolerance value. The first step is to synchronise the two signals (video and speeds) as is shown in the Fig. 4b.

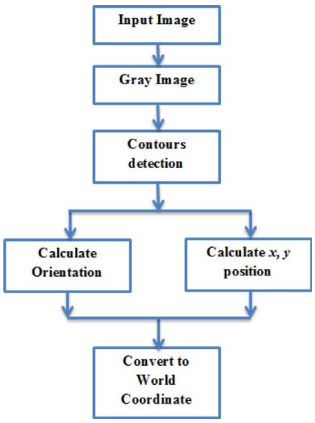


Fig. 8. Image process flowchart of pattern based localization.

The position obtained from the camera represents the offset-free position n samples before, where n represents the time delay in samples (i.e., $n = 5$ with $T_s = 66$ (ms)). Next, it is possible to integrate the speed values of the last five samples, in order to obtain the position estimation from odometry up to time $n - 1$. Eq. (5) presents the method to eliminate the time delay effect on the video signal using a combination with odometry, assuming the dead time is a multiple of the sample time for y axis.

$$y = T_s \sum_{i=-(N_d-1)}^{0} v_y(i) + y_{cam}(N_d) \quad (5)$$

where y is the final position in meters, $N_d = T_d/T_s$, and T_s is the sample time: 66 ms; i represents the samples; v_y is the speed on the y axis; and y_{cam} represents the position obtained from the camera with a constant time delay $T_d = 200$ ms. The position of the quadrotor in x axis is calculated in similar way.

Fig. 9 shows an estimation of the position with odometer (IMU), optical sensor (camera) and the combination which fuses two sensors. It is clear that it is not possible to only use the camera for position estimation as it starts to drift very quickly. The IMU has a robust position estimation but it is not very accurate. The sensor fusion combines the advantages of both signals. It is a robust estimation without noise.

2.4. Obstacle detection

The proposed method in this paper considers both unknown and known obstacles, where the known obstacles are predefined from the beginning. This information could be extracted from the provided map or from previous flights through that environment. Unknown obstacles become only visible upon detection, which may force the algorithm to

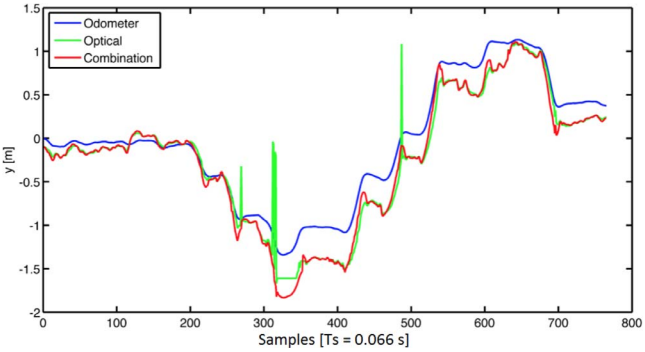


Fig. 9. Position values in an open loop obtained from the image processing-optical sensor (green), the IMU-odometry (blue) and the fused response (red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

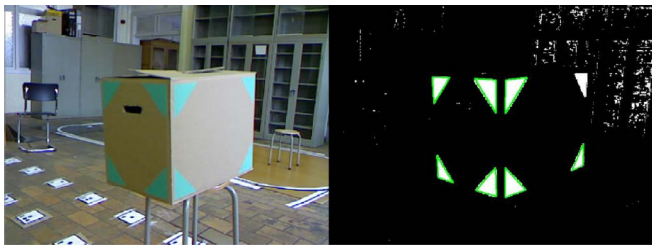


Fig. 10. Detected Obstacle by ArDrone 2.0.

adjust the previous trajectory. The corners on the obstacle are marked by colored (blue) patterns in order to distinct them from the rest of the room as it can be seen in Fig. 10. The orientations of the triangles also provide information the obstacle's geometry. The obstacle and its location can still be detected when is only partially visible to the frontal camera. In order to estimate the distance to the obstacle, the size of the triangles in the image can be used. The larger they appear, the closer they are and vice-versa.

3. Path planning based on improved potential field method

Path planning is defined as designing a collision-free path in a working environment with obstacles. A path is a set of configuration $\vec{q} = \{q_0, q_1, \dots, q_{goal}\} \in \mathscr{R}^n$ of the agent that connects the starting position q_0 and the final position q_{goal}.

Although the conventional PFM generates an effective path, it suffers from the non-reachable target problem. This problem occurs when the target is close to obstacles. In that case, when the agent approaches the target, it approaches the obstacles as well. As a consequence, the attractive force reduces while the repulsive force increases. Therefore, the agent is trapped in local minima and oscillations might occur.

Fig. 11 (**Left**) presents the case that there are several obstacles located near the target.The repulsive force is considerably larger than the attractive force, therefore the agent is repulsed away rather than reaching the target.

Fig. 12 (**Left**) illustrates another case in which the attractive field and the repulsive field are co-linear in opposite directions and the total force approximates zero thus the agent is trapped in local minima.

3.1. Proposed attractive and repulsive potential

The crucial cause of the non-reachable target problem is that the goal position is not a global minimum of the total potential U in Eq. (6). When the agent reaches the target, attractive potential U_{att} is equal to zero; however, the repulsive potential U_{rep} is none-zero if there is at least one obstacle which satisfies the condition $d(q, q_{obs}) < d_0$. To overcome that drawback, a proposed attractive and repulsive potential is proposed to ensure that the total potential field force has the unique global minimum at the target position.

Obviously, if the repulsive potential approaches zero as the agent reaches the target, the total potential attains the global minimum at the

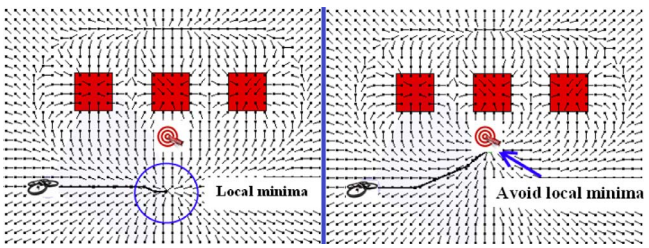


Fig. 11. The problem and the solution when the position of target is very close to obstacles. **Left:** the non-reachable target problem of conventional PFM, **Right:** Avoid local minima solving the target non-reachable problem of MPFM.

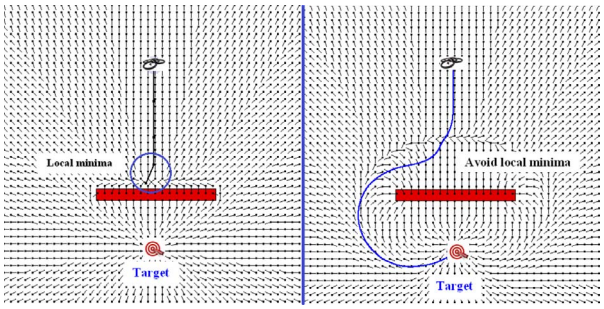


Fig. 12. The problem and the solution when the agent, obstacle and target are aligned in which the obstacle in the middle and the attractive force approximates the repulsive force. **Left:** the non-reachable target problem of conventional PFM, **Right:** Avoid local minima solving the target non-reachable problem of MPFM.

goal. As a consequence, it is necessary to introduce the relative distance between the agent and the target ($d(q, q_{goal})$) into the formula of repulsive potential. Furthermore, since the agent is unable to stop suddenly at the target position while it is moving at a high speed, the agent velocity term ($\dot{q}$) is taken into account in the proposed attractive potential formula. The total potential $U = U_{att} + U_{rep}$ obtains the global minimum (0) if and only if $q = q_{goal}$ and $\dot{q} = 0$. The MPFM are formulated as follows:

$$\begin{aligned} &U = U_{att} + U_{rep} \\ &U_{att}(q, \dot{q}) = \rho_p d^2(q, q_{goal}) + \rho_v \dot{q}^2 \\ &U_{rep} = \begin{cases} \frac{1}{2}\alpha\left(\frac{1}{d(q, q_{obs})} - \frac{1}{d_0}\right)^2 d^\beta(q, q_{goal}) & \text{if } d(q, q_{obs}) \le d_0 \\ 0 & \text{if } d(q, q_{obs}) > d_0 \end{cases} \end{aligned} \tag{6}$$

where ρ_p, ρ_v, α, β are positive coefficients; d_0 is the affected distance of obstacle; $d(q, q_{goal})$ is the distance between the agent and the target; $d(q, q_{obs})$ is the minimum distance between the agent and the obstacles.

The attractive force and repulsive force are the negative gradients of attractive potential and repulsive potential as follows:

$$\begin{aligned} F &= F_{att} + F_{rep} \\ F_{att} &= -2\rho_p d(q, q_{goal}) - 2\rho_v \dot{q} \\ F_{rep} &= \begin{cases} F_{rep1} + F_{rep2} & \text{if } d(q, q_{obs}) \le d_0 \\ 0 & \text{if } d(q, q_{obs}) > d_0 \end{cases} \\ F_{rep1} &= -\alpha\left(\frac{1}{d(q, q_{obs})} - \frac{1}{d_0}\right)\frac{d^\beta(q, q_{goal})}{d^2(q, q_{obs})} \\ F_{rep2} &= -\frac{\alpha\beta}{2}\left(\frac{1}{d(q, q_{obs})} - \frac{1}{d_0}\right)^2 d^{\beta-1}(q, q_{goal}) \end{aligned} \tag{7}$$

Applying conventional PFM, the agent is not successful in autonomous navigation tasks when the obstacles are located near the target as mentioned before. However, the proposed method can handle such problems properly because it reduces the repulsive force when the agent moves towards the target. Thus, the agent enables to reach the target successfully as shown in Fig. 11 (**Right**) and Fig. 12 (**Right**). It indicates that the proposed method effectively solves the non-reachable target problem when the obstacles are located near the target.

The proposed approach is developed to appropriately work in known and unknown complex environment. First, the global agent path planning is generated based on the proposed modified potential field method (MPFM), then this path is renovated when the agent senses a new obstacle until reaching the target (local path). The algorithm is presented in Fig. 13.

3.2. Simulation results in complex environment

To validate the proposed algorithm, simulations are executed under various complex environment conditions with known and unknown

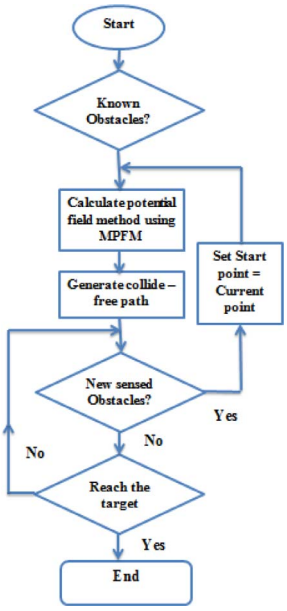


Fig. 13. The flowchart of the agent path planning in known and unknown environment based on MPFM.

obstacles. There are large obstacles with different dimensions and shapes located in the way of the agent. Fig. 14 illustrates two different cases of complex indoor environment with completely known obstacles in which the agent arrives at the target without collision with obstacles using MPFM.

Fig. 15 shows the result of the algorithm in complex scenarios with unknown obstacle. The known obstacle is presented in red while the unknown obstacle is presented in black. Since agent has no information about the black obstacle in advance, the trajectory is generated to avoid only red obstacles as displayed in Fig. 15 (**Left**). However, it updates the path immediately (local path) as soon as detecting a new obstacle (the obstacle changes its color to orange) as shown in Fig. 15 (**Right**). The results prove the feasibility and effectiveness of the algorithm in complex environment with known and unknown obstacles.

4. Control parameters optimization based on particle swarm optimization algorithm

Unlike the existing approaches, in our work the PID controllers of UAV are designed and implemented based on the proposed multi-objective particle swarm optimization (MOPSO) with an accelerated update methodology. This algorithm aims to facilitate convergence to optimal set of PID parameters. This section is structured as follows. In Section 4.1, a proposed accelerated particle updates of MOPSO is presented. MOPSO-based PID controller approach is described in Section 4.2. The final subsection presents Ar.Drone 2.0 control results.

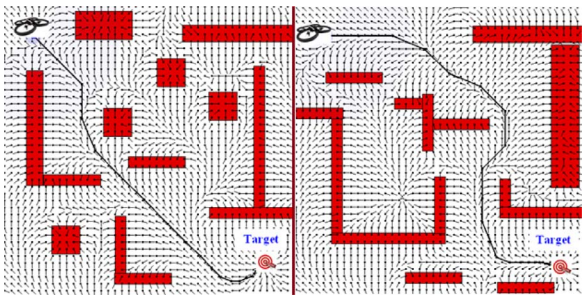


Fig. 14. Proposed MPFM under the complex environment.

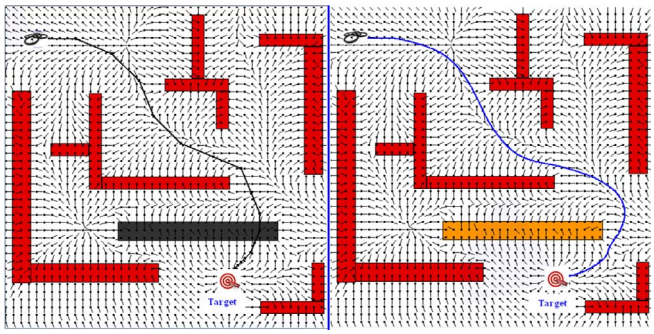


Fig. 15. Proposed MPFM under the complex partial known environment, **Left:** The agent collides black (unknown) obstacle. **Right:** The agent re-plans the path according to perceived environment.

4.1. Proposed accelerated particle updates of MOPSO

In the conventional PSO, the particle's position is updated based on both the current global best *Gb* and the personal best Pb_i (or local best) [20]. The purpose of using the local best is primarily to expand the diversity of the quality solutions, however, the diversity can be simply simulated by some randomness. Therefore, to accelerate the convergence of the algorithm, it is possible to use the global best only. Based on that statement, the velocity vector and position vector are formulated as:

$$V_i(t+1) = V_i(t) + c_1 r + c_2(Gb(t) - X_i(t)) \tag{8}$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{9}$$

where:

$c_1 \in [0.1\ 0.5]^*(UB - LB)$;

$c_2 \in [0.1\ 0.7]$;

r is a uniform random number in [0,1] that brings the stochastic state to the algorithm.

$Gb(t)$ is the global best in iteration t;

$V_i(t)$, $V_i(t+1)$ are velocities of particles i in iteration t, $t+1$;

$X_i(t)$, $X_i(t+1)$ are positions of particles i in iteration t, $t+1$;

LB, UB are lower bound and upper bound of X. In this study, the values of *LB* and *UB* are (0, 0, 0) and (50, 50, 50).

To reduce the randomness as iterations are updated, the value of c_1 can be designed as:

$$c_1 = c_0 \xi^{t*}(UB - LB) \tag{10}$$

where $c_0 \in [0.1\ 0.5]$ is the initial value of the randomness parameter while t is the number of the iterations and $\xi \in (0\ 1)$ is a control parameter. The pseudo code is presented in Algorithm 1.

4.2. A proposed MPSO-based PID controller approach

Starting from the transfer function of a PID controller:

$$G_{PID}(s) = K_p + \frac{K_i}{s} + K_d s \tag{11}$$

The controller parameters K_p, K_i, K_d are chosen to satisfy prescribed performance criteria regarding the settling times (T_s) and the rise time (T_r), the overshoot (*OS*) and the steady-state error (*SSE*). Since the PID is a very well-known controller, the definition of T_r, T_s, *OS* and *SSE* are not mentioned in this paper. The three desired objective functions are:

$$\begin{aligned} J_1(X) &= |SSE| \\ J_2(X) &= OS \\ J_3(X) &= T_s - T_r \end{aligned} \tag{12}$$

where X is a set of parameters to be optimized, $X = (K_p, K_i, K_d)$.

The block diagram of MPSO-based PID controller approach is presented in Fig. 16. In this procedure, the dimension of the particle is 3.

Initialize parameters: The time (iteration) counter is set to 0, initial values $r \in (0\ 1)$, $c_0 \in [0.1\ 0.5]$, $c_2 \in [0.1\ 0.7]$
Initialize particle: X is randomly generated in $[LB\ UB]$
while {maximum iterations or minimum error criteria is not attained} **do**
  Update control parameter: $c_1 = c_0 * \xi^t$ with ξ is a control parameter, chosen in (0 1).
  Update position
  Update the objectives
  Update Gb:
    for $i = 1: N$
    if $J(X_i) \leq Jbest$
    $Gb = X_i$;
    $Jbest = J(X_i)$
  Update the iteration: $t = t+1$
end while
(Display results) Output optimal results

Algorithm 1. The proposed MPSO pseudo-code.

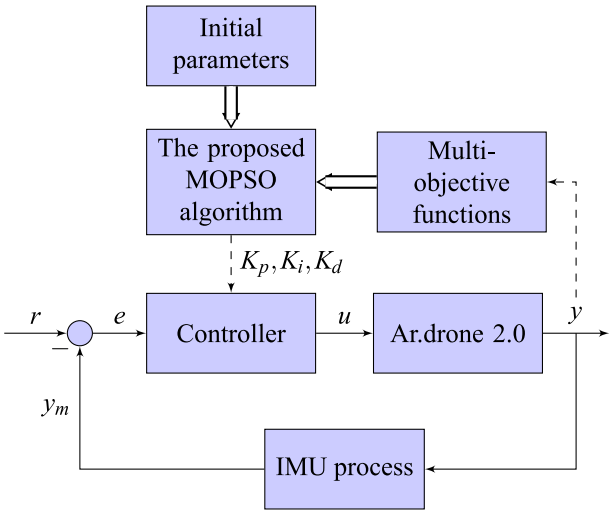


Fig. 16. The proposed MOPSO-based PID controller approach.

Initially, PSO algorithm assigns arbitrary values of K_p, K_i, K_d and computes the objectives function and continuously update the controller parameters until the objective functions are optimized. A composite objective optimization for PSO-based PID controller is obtained by summing values of three mentioned objective functions through the following weighted-sum method.

$$J(X) = \beta_1 J_1(X) + \beta_2 J_2(X) + \beta_3 J_3(X) \tag{13}$$

where β_1, β_2 and β_3 are positive constants; $J_1(X)$, $J_2(X)$ and $J_3(X)$ are the objective functions defined as in equation (12). In this study, those values are set as $\beta_1 = 0.59$, $\beta_2 = 0.49$ and $\beta_3 = 0.88$.

4.3. Ar.Drone 2.0 control results

The proposed algorithm MOPSO used a set of parameters as: the swarm size $N = 50$, the maximum number of iterations $T_{max} = 50$, $c_0 = 0.2$; $c_2 = 0.7$; $\xi = 0.97$ in the following simulations.

Having the dynamic model of the Ar.Drone 2.0 (Section 2.2), controllers parameters is obtained by applying proposed MOPSO-based PID controller as presented in Section 4.2. PID controllers' optimal parameters are obtained through simulations. It is worth noting that the models obtained for X and Y position controllers are the same, therefore their controllers have the same topology and parameters. Fig. 17 show the results of $X(Y)$ position control using Frtool [21], PID tuner Matlab toolbox and the proposed MOPSO. The optimal K_p, K_i, K_d parameter sets obtained by the proposed MOPSO method together with rise time, settling time, overshoot, peak and the cost function of each Ar. Drone 2.0 controller are shown in Table 1. The proposed MOPSO is used to minimize three cost functions in the term of settling times/ rise time $(T_s - T_r)$, overshoot (OS) and steady-state error (SSE) of each controller. As shown in Table 1, all designed controllers have no overshoot, zero steady state error, extremely short rise time and setting time.

The results of the proposed approach (blue solid curves, name PSO-PID) are compared with the PID using Frtool (green dash-dot curves) and PID tuner Matlab toolbox (red dot curves). Both PSO-PID and PID-Frtool have better performances than the third one with no overshoot. However, the setting time is clearly less for the proposed MPSO-PID controller than for PID-Frtool and PID tuner.

In order to investigate the robustness and sensitivity of the approach in changing of the weighted constants, the parameters β_1, β_2 , β_3 are modified in the range of 20% those values. Therefore, the composite objective optimization is as following:

$$J(X) = (\beta_1 \pm \Delta\beta_1)J_1(X) + (\beta_2 \pm \Delta\beta_2)J_2(X) + (\beta_3 \pm \Delta\beta_3)J_3(X) \tag{14}$$

where:

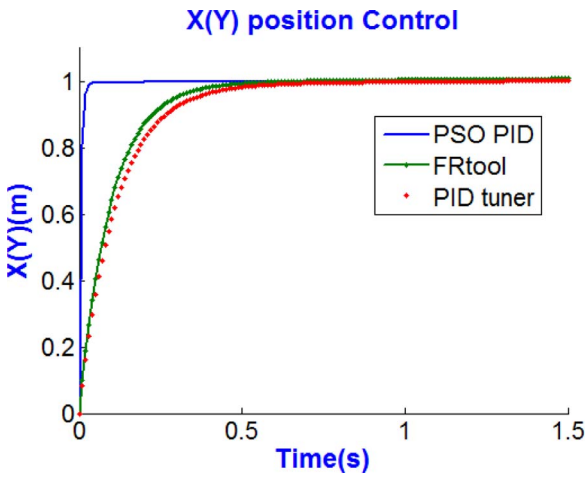


Fig. 17. $X(Y)$ position step response with MPSO tuning, Frtool, PID tuner.

Table 1
Optimal control parameter selected by pso algorithm for AR. Drone 2.0 PID controllers.

MOPSO-PID	Altitude controller	X controller	Y controller	YAW controller
K_p	21.9820	43.9582	43.9582	24.0011
K_i	38.1941	40.9473	40.9473	43.4267
K_d	42.8751	9. 2215	9.2215	29.6480
Rise time (s)	0.0363	0.0072	0.0072	5.3959e-04
Settling time (s)	0.2368	0.0129	0.0129	0.0010
Overshoot	0	0	0	0
Peak	0.9976	0.9999	0.9999	0.9925
Cost function value	0.26	0.21	0.21	0.2950

β_1, β_2, β_3 are defined in Section 4.2.

$\Delta\beta_1 \leq 0.2\beta_1$; $\Delta\beta_2 \leq 0.2\beta_2$; $\Delta\beta_3 \leq 0.2\beta_3$.

Fig. 18 illustrates different MOPSO-PID controllers for X(Y) position on Ar.Drone 2.0 obtained by randomly changing the weights of three objective functions. It was observed that all MOPSO-PID controllers react very fast and without overshoot and track the reference input very well. In addition, there are only slightly difference between MPSO-PID controllers' outputs. In conclusion, the proposed approach is highly robust and not very sensitive in term of changing of the weighted constants.

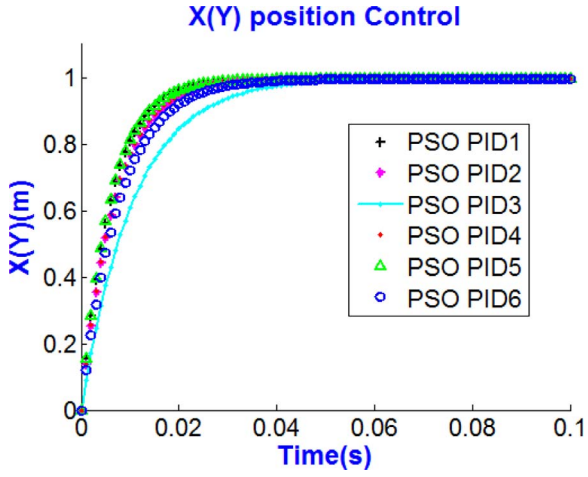


Fig. 18. Simulation results obtained from MOPSO-PID X(Y) control in the variance of β_1, β_2 , β_3.

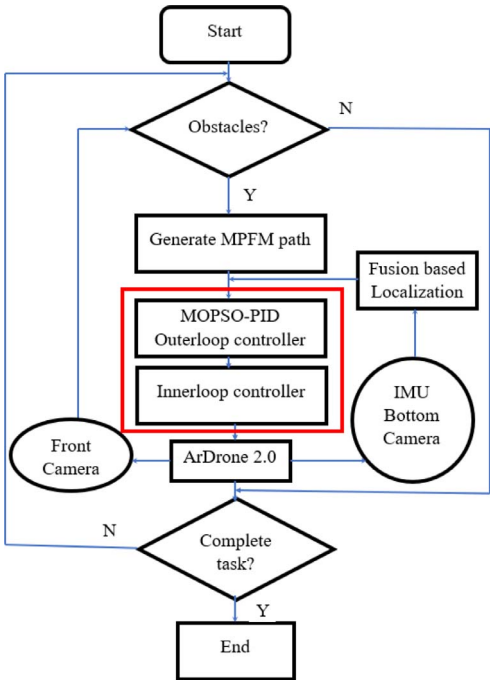


Fig. 19. The proposed solution structure with cascade control.

5. Experimental validation

The proposed solution structure of this study is depicted in Fig. 19. At the first stage, the quadrotor uses its sensor to extract obstacles information then generates MPFM path based on the proposed path planning algorithm. At this phase, the quadrotor should know its pose based on localization process and decides how to steer to achieve its goal. After that the drone regulates its motion to accomplish the desired trajectory. In real experiments, a cascade control is designed such that the Ar.Drone 2.0 accurately performs this task. There are two parts of the cascade controller: inner-loop controller and outer-loop controller. The inner-loop controller is performed inside the quadrotor as a black-box. The model identified in Section 2.2 is used to identify the relationships between the inputs and the outputs of this black-box. The outer-loop controller or MOPSO-PID is represented by the command station, which defines the references to the internal controllers located in the low layer.

In the outer-loop controller, the localization process provides the current Ar.Drone 2.0 pose in the world coordinate based on the ground patterns. The obtained free-collision trajectory using MPFM is sent to the controller by a list of way-points. However, due to missing communication, oscillating and uncertain noise, the Ar.Drone 2.0 may be unable follow the designed trajectory. Therefore, it is necessary to generate a compensation path, named, $\overrightarrow{Tar}$ that guides the quadrotor to return to the designed trajectory.

To perform the compensation, several definitions are introduced. Suppose the drone is currently in the position with the two nearest way-points, named, *Previous way-point* and *Next way-point*.

- *Path error*: the distance between the drone and the designed path at a time during flight.
- $\overrightarrow{Target_{WP}}$: the vector from *Previous way-point* to the *Next way-point*
- $\overrightarrow{Target_{pathline}}$ as the vector towards the perpendicular path

A schematic representation is depicted as shown in Fig. 20. The compensation $\overrightarrow{Tar}$ is only executed when the *Path error* (L) is larger than threshold value L_0. The formula of the target vector is:

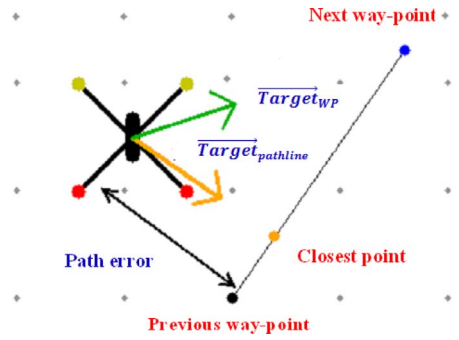


Fig. 20. Compensation strategy for the quadrotor.

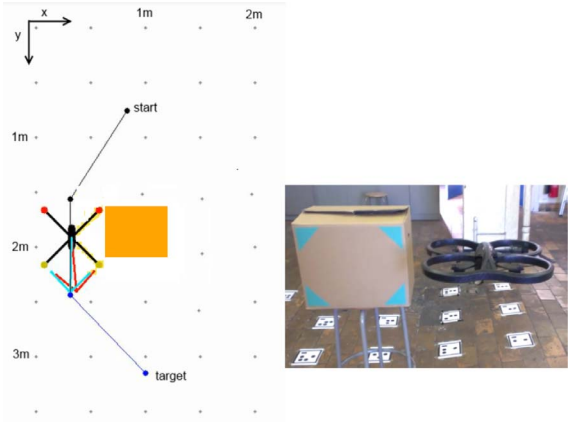


Fig. 21. Virtual vs. Actual environment in the real-time experiment.

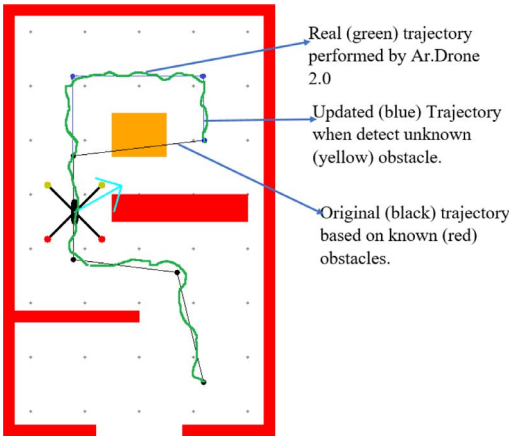


Fig. 22. Real-time obstacle avoidance of the AR. Drone 2.0 with pattern-based localization, MPFM and optimal PID controller.

$$\overrightarrow{Tar} = \lambda^* \overrightarrow{Target_{WP}} + (1-\lambda)^* \overrightarrow{Target_{pathline}} \tag{15}$$

where λ is defined parameter as following:

$$\lambda = \begin{cases} 1-(L/L_0)^n & \text{if } L < L_0 \\ 0 & \text{if } L > L_0 \end{cases}$$

It is important to notice that the $\overrightarrow{Tar}$ should not significantly change when the drone is around the trajectory with a small *Path error* and should quickly react with a large *Path error*. For our system, $n = 4$ and $L_0 = 0.5\ m$ has been chosen.

In this work, a virtual environment is developed in order to monitor on-line and facilitate off-line tests of the quadrotor navigation tasks as presented in Fig. 21. In this figure, the comparison between the virtual environment (**Lelf**) and actual environment (**Right**) is shown. When the drone detects the real obstacle in actual environment, that one is displayed in the virtual environment as black box. The MPFM is generates the path in order taking account the ArDrone 2.0 dimension. The start point, the target and the dimensions of the working space are also shown in the virtual environment.

The results obtained with pattern-based localization, MPFM and optimal PID control in the real system are presented in Fig. 22. The bounded rectangle presents the walls of indoor working environment. The red obstacles are known obstacles and the yellow one is unknown obstacle. In the beginning, MPFM generate the black trajectory which collides with unknown (yellow) obstacle since the drone only avoid red obstacles. However, it updates the path immediately (local path-blue path) as soon as its detecting a new (yellow) obstacle as shown in Fig. 22. The green path is the real path obtained by experiment. It is possible to observe that the quadrotor has few deviations to the desired trajectory, with an acceptable overshoot at the moment of performing sharp bends.

6. Conclusions

This paper proposed an autonomous navigation approach for a low-cost commercial AR. Drone 2.0 using only on-board visual and internal sensing. The main achievements for this work are: (*i*) sensor fusion for localization in a structured environment taking into account camera signals time delay; (*ii*) proposed potential field method for path planning; and (*iii*) the design and implementation of an optimal PID controller based on a proposed multi-objective particle swarm optimization (MOPSO) with an accelerated update methodology that allows accomplishing path-following task using a cascade control approach. The proposed modified potential filed method enables to solve the non-reachable target problem and successfully avoids unknown obstacles. The performed tests using the virtual environment and real-time experiment demonstrate the feasibility of the proposed strategy, which opens new possibilities of autonomous navigation for the mobile agent in complex known and unknown environment.

Regarding future work, the approach is currently implemented in a dynamic working space. An extension to multiple UAVs and a combination with ground vehicles is also under inspection.

Acknowledgment

All authors from Ghent University are with EEDT group, member of Flanders Make consortium.

References

[1] Woods AC, La HM. A novel potential field controller for use on aerial robots. IEEE Trans Syst Man Cybern Syst 2017;99:1–12.
[2] Hernandez A, Murcia H, Copot C, De Keyser R. Towards the development of a smart flying sensor: illustration in the field of precision agriculture. Sensors (Basel) 2015;15(7):16688–709.
[3] Hernandez A, Copot C, Cerquera J, Murcia H, De Keyser R. Formation control of UGVs using an UAV as remote vision sensor. Proceedings of the 19th IFAC World Congress, Cape Town, South Africa. 2014. p. 618–23.
[4] Dijkshoorn N. Simultaneous localization and mapping with the AR.drone. University of Amsterdam; 2012. Master's thesis.
[5] Dullerud G. Modeling, identification and control of a quad-rotor drone using low-resolution sensing. Champaign, IL, USA: University of Illinois at Urbana-Champaign; 2012. Master's thesis.
[6] Bouffard P. On-board model predictive control of a quadrotor helicopter: Design, implementation, and experiments. Berkeley, CA, USA: University of California; 2012. Master's thesis.
[7] Ritz R, Muller M, Hehn M, D'Andrea R. Cooperative quadrocopter ball throwing and catching. Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Portugal, Algarve. 2012. p. 4972–8.
[8] Alvarez H, Paz LM, Sturm J, Cremers D. Collision avoidance for quadrotors with a monocular camera. Exp Rob Springer Trans Adv Rob 2015;109:195–209.
[9] Mac TT, Copot C, Tran DT, De Keyser R. Heuristic approaches in robot path planning. A Surv Rob Auton Syst 2016;86:13–28.
[10] Malone N, Chiang H, Lesse K, Oishi M, Tapia L. Hybrid dynamic moving obstacle avoidance using a stochastic reachable set-based potential field. IEEE Trans Rob

2017;33(5):1124–38.
[11] Moreau J, Melchior P, Victor S, Anioun F, Guillemard F. Path planning with fractional potential fields for autonomous vehicles. IFAC-PapersOnLine 2017;50(1):14533–8.
[12] Rasekhipour Y, Khajepour A, Chen S, Litkouhi B. A potential field-based model predictive path-planning controller for autonomous road vehicles. IEEE Trans Intell Transp Syst 2017;18(5):1255–67.
[13] Valbuena L. Hybrid potential field based control of differential drive mobile robots h. g. tanner. J Intell Robot Syst 2012;68:307–22.
[14] Scherer S, Singh S, Chamberlain L, Elgersma M. Flying fast and low among obstacles: methodology and experiments. Int J Rob Res 2008;27(5):549–74.
[15] Golan Y, Edelman S, Shapiro A, Rimon E. On-line robot navigation using continuously updated artificial temperature gradients. IEEE Rob Autom Lett 2017;2(3):1280–7.
[16] Li J, Li Y. Dynamic analysis and PID control for a quadrotor. IEEE international conference on mechatronics and automation, Beijing, China. 2011. p. 573–8.
[17] Song Y, Xian B, Zhang Y, Jiang X, Zhang X. Towards autonomous control of quadrotor unmanned aerial vehicles in a GPS-denied urban area via laser ranger finder. Optik - Int J Light Electron Opt 2015;126:3877–82. Issue 23
[18] Engel J, Sturm J, Cremers D. Scale-aware navigation of a low-cost quadrocopter with a monocular camera. Rob Auton Syst 2014;62(11):1646–56.
[19] Ljung L. System identication: theory for the user. Prentice-Hall; 2007.
[20] Yang XS, Deb S, Fong S. Accelerated particle swarm optimization and support vector machine for business optimization and applications. Networked digital technologies, communications in computer and information science. 136. Springer; 2011. p. 53–66.
[21] De Keyser R, Ionescu CM. FRTool: a frequency response tool for CACSD in matlab. Proc. of the IEEE Conf. on Computer Aided Control Systems Design, Germany, Munich. 2006. p. 2275–80.

Thi Thoa Mac received the B.Eng. degree in mechatronics engineering from Hanoi University of Science and Technology, Vietnam, in 2006. She obtained her master degree in 2009 at National Taiwan University of Science and Technology in Faculty of Engineering, Taiwan. She is currently a Ph.D. Researcher with the Department of Electrical Energy, Systems, and Automation, Ghent University. She also is a lecturer at Hanoi university of science and technology. Her interests include fuzzy logic control, neural networks, path planning and optimization control for autonomous robot.

Cosmin Copot received his M.Sc. and M.E. degrees in systems engineering from Gheorghe Asachi Technical University of Iasi, Romania, in 2007, and 2008, respectively. In 2011 he received Ph.D. degree from the same university on control techniques for visual servoing systems. He is currently a postdoctoral research at Ghent University. His research interests include robotics, visual servoing systems, identification and control.

Robin De Keyser received the M.Sc. degree in electromechanical engineering and the Ph.D. degree in control engineering from GhentUniversity, Ghent, Belgium, in 1974 and 1980, respectively. He is currently a Full Professor of control engineering with the Faculty of Engineering, Ghent University. He is the author or co-author of over 300 publications in journals, books, and conference proceedings. He has acted as an external review expert in several European Commission Research Programs and is one of the pioneers who produced the original concepts of predictive control during the 1980s. The research is application-driven, with many pilot implementations in technical and nontechnical systems, amongst others in the chemical, steel, marine, mechatronic, semiconductor, power electronics, and biomedical spheres. His current research interests include model predictive control, auto-tuning and adaptive control, modeling and simulation, and system identification.

Clara-Mihaela Ionescu graduated with a M.Sc. degree in Industrial Informatics and Automation at Dunarea de Jos University of Galati, Romania, in 2003. She obtained the Ph.D. degree in November 2009 from Ghent University, on modelling of the human respiratory system with non-integer order models. She was holder of the prestigious Flander Research Foundation (FWO) post-doctoral grant at the same university. She is currently a Professor of control engineering with the Faculty of Engineering, Ghent University. She is involved in several international projects, with both industrial and biomedical applications. Her main research interests revolve around the principle of emerging tools from mathematics into control engineering practice, i.e. generalized order models and control laws.