IEEE International Conference on Automation, Quality and Testing, Robotics - THETA 20th edition (AQTR)
Cluj-Napoca, Romania, 19-21 of May, 2016.

AR.Drone UAV control parameters tuning based on particle swarm optimization algorithm

Thi Thoa Mac[1,2], Cosmin Copot[1,3], Trung Tran Duc[2] and Robin De Keyser[1]
[1] Department of Dynamical Systems and Control (DySC), Ghent University, Belgium.
[2] School of Mechanical Engineering, Hanoi University of Science and Technology, Vietnam.
[3] Op3Mech, University of Antwerp, Belgium.
Email of corresponding author: Thoa.MacThi@Ugent.be

***Abstract*—In this paper, a proposed particle swarm optimization called multi-objective particle swarm optimization (MOPSO) with an accelerated update methodology is employed to tune Proportional−Integral−Derivative (PID) controller for an AR.Drone quadrotor. The proposed approach is to modify the velocity formula of the general PSO systems in order for improving the searching efficiency and actual execution time. Three PID control parameters, i.e., the proportional gain K_p, integral gain K_i and derivative gain K_d are required to form a parameter vector which is considered as a particle of PSO. To derive the optimal PID parameters for the Ar.Drone, the modified update method is employed to move the positions of all particles in the population. In the meanwhile, multi-objective functions defined for PID controller optimization problems are minimized. The results verify that the proposed MOPSO is able to perform appropriately in Ar.Drone control system.**

I. Introduction

In recent years, research interest in Unmanned Aerial Vehicles (UAVs) has been grown rapidly because of their potential use for a wide range of applications. Thanks to the ability to perform dangerous and repetitive tasks in remote and hazardous environments, UAV is very promising to play more important roles such as flight maneuverer [1], ball throwing and catching [2], formation control of UGVs using an UAV [3], illustration in the field of precision agriculture using UAV as smart flying sensor [4], collaborative construction tasks [5].

As an UAV is a complex system that electromechanical dynamics is involved [6], the robust controller is an essential requirement. As a consequence, conventional PID controller is widely applied in UAV for its practicability and robustness in the past decades. In [7], the dynamical characteristics of a quadrotor are analyzed to design a PID controller which aims to regulate the posture (position and orientation) of the quadrotor. In [8], a classical PID is implemented on a quadrotor for autonomous visual tracking and landing on a moving carrier. Based on the PID controller, the autonomous control problem of a quadrotor UAV in GPS-denied unknown environments is studied [9], [10]. Overall, the primary problem associated with UAV is how to optimal tuning the parameters of PID controller. In order to obtain reasonable dynamical performance, guarantee security and sustainable utilization of equipments and plants, PID controller parameters must be well tuned.

978-1-4673-8691-3/16/$31.00 ©2016 IEEE

PID controller contains three adjustable gain parameters, the proportional gain K_p, integral gain K_i and derivative gain K_d, respectively. Many approaches have been proposed for improving the setting performance of PID gains of dynamical systems using heuristic optimization methods such as Genetic Algorithm (GA), Ant Colony Optimization (ACO), Artificial bee colony algorithm (ABC). GA is used to obtain optimized parameters values for PID aircraft pitch controller [11] and for autonomous underwater vehicle (AUV) motion control [12]. An approach which proposes an ant colony optimization algorithm for tuning fuzzy PID controller is presented in [13]. It points out that the main drawback of fuzzy PID controllers is a huge amount of parameters to be tuned. This problem is solved by using ACO algorithm. This method is capable of generating the optimum or quasi-optimum parameters to the control system in a high dimensional space. In [14] a tuning method for determining the parameters of PID controller using ABC algorithm is proposed for ball and hoop system.

Particle swarm optimization (PSO) algorithm is a new evolutionary computation technique and also is applied to derive the optimal PID gains for various dynamical systems. In [15], a robust controller for a hose transportation system performed by three UAVs is implemented. PSO is used to tune PID controller so that the three UAVs are able to carry two sections of a hose. In [16], PSO algorithm is used to optimize the Fractional Order Proportional Integral Derivative (FOPID) controllers parameters to solve the problem of path tracking of an autonomous ground vehicle. The advantage of this approach is minimizing the path tracking error and the complement of the path following.

Many of those studies treat the PID parameters optimization as a single objective problem. Thus, finding the optimal parameters of PID controller considering multiple desired criteria (objectives) is a challenging problem. Consequently, this study proposes a particle swarm optimization algorithm called multi-objective particle swarm optimization (MOPSO) with an accelerated update methodology to obtain the optimal parameters of PID controllers for the quadrotor. MOPSO algorithm is deployed to tune PID controllers considering desired criteria which are overshoot, steady-state error, settling time and rise time. Thus, finding PID gains is a multi-objective optimization problem in which different objectives are in conflict, i.e., the improvement of one criterion may lead to

the deterioration of other criteria.

The paper is organized as follows: a brief description of the AR.Drone 2.0 UAV used in this study is presented in section II, the particle swarm optimization algorithm for control parameters optimization is described in detail in section III. The Ar.Drone control results are presented in section IV. In section V, the main outcomes of this work are summarized.

II. AR.Drone 2.0 Quadrotor Platform

A. Plant Description

An Ar. Drone 2.0, a commercial and low-cost micro UAV, is used in this study. It has four propeller blades arranged symmetrically around a central unit which includes the sensory equipment and the circuit board. There are four basic motions of this quadrotor: pitch, roll, throttle and yaw as shown in Fig.1.

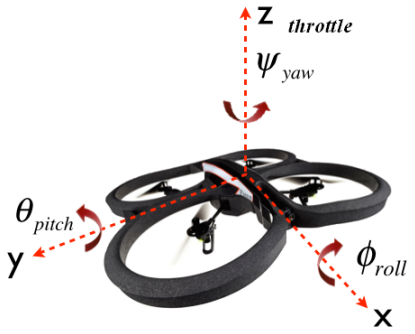

Fig. 1. The movements of an AR.Drone 2.0.

The sensor system consists of several motion sensors which together form the Inertial Measurement Unit (IMU). Communication between Ar.Drone and a command station is performed via Wi-Fi connection within a 50 meters range. AR.Drone 2.0 is equipped with two cameras, one in the bottom part and the other in frontal part. They have resolutions of 320 x 240 pixels at 30 frames per second (fps) and 640 x 360 pixels at 60 fps, respectively.

Several Software Development Kits (SDK) have been developed for Windows, Linux or iOS operating systems [17][18][19], thus enabling AR. Drone 2.0 to be manipulated from a computer, smart phone or tablet. In this work, the AR.Drone is controlled from a computer using Windows 7 with Visual Studio C++, OpenCV and AR.Drone libraries.

B. Analysis of Inputs and Outputs

The developed 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. The control parameters given to the internal controllers are floating point values between [-1, 1]. Those parameters are not directly the control parameters values, but a percentage of the maximum corresponding values of the mentioned controller.

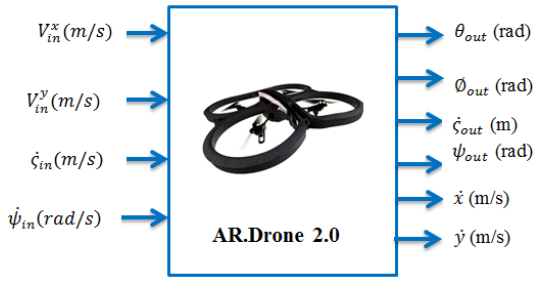

Fig. 2. Inputs and Outputs of an AR.Drone 2.0.

C. System identification

An Ar. Drone is a multi-variable and naturally unstable system. However, because of 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 SISO loops. Transfer functions are obtained via parametric identification using the prediction error method and Pseudo-Random Binary Signal (PRBS) input signals [20]. 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 performed on the previous studies [21]. The identified transfer functions are:

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

III. Particle Swarm Optimization Algorithm for Control Parameters Optimization

The particle swarm optimization (PSO) algorithm is an evolutionary computation technique, proposed by Eberhart and Kennedy [22] and has been applied in various application fields in recent years. PSO is inspired by social behavior of bird flocking or fish schooling. This algorithm is a population based stochastic optimization technique. In PSO, a swarm consists of a set of particles and each particle is a potential solution of the optimization problem. Basically, PSO is initialized with a set of random solutions and then updated each generation based on optimal schema. The computational efficiency of this optimal algorithm is an excellent feature and it is also easy to be implemented. In addition, unlike other heuristic optimization methods, it has a flexibility to enhance the exploration and exploitation abilities.

A. Particle swarm optimization algorithm

Considering the search space $\mathscr{D}$ that has N dimension ($\mathscr{D} \subset \mathscr{R}^N$), the position and the velocity of i^{th} particle in the swarm are denoted as $X_i = (X_{i1}, X_{i2}, ..., X_{iN}) \in \mathscr{D}$ and $V_i = (V_{i1}, V_{i2}, ..., V_{iN}) \in \mathscr{D}$. The particles will update their locations in the swarm towards the global optimum (or target position) based on two factors: 1) the personal best position (Pb_i) and 2) the global best position (Gb). The first factor is the best position found by the i^{th} particle itself which is termed local leader and represented as $Pb_i = (Pb_{i1}, Pb_{i2}, ..., Pb_{iN})$. The second factor is the best position of the whole particles in the swarm, which is termed global leader and represented as Gb. At the iteration $t+1$ of the search process, the velocity and the position will be updated according to following equations:

$$\begin{aligned} V_i(t+1) &= wV_i(t) + c_1 r_1 (Pb_i(t) - X_i(t)) \\ &+ c_2 r_2 (Gb(t) - X_i(t)) \end{aligned} \tag{2}$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{3}$$

where:
w is inertia weight;
c_1 and c_2 are two nonnegative constants, referred as cognitive and social factors, respectively;
r_1 and r_2 are uniform random numbers in [0, 1] that brings the stochastic state to the algorithm.
The pseudo code of this algorithm for minimizing a cost function f is provided in Algorithm 1.

Algorithm 1 PSO pseudo-code

```
Initialize population, parameters
While Termination criterion is unsatisfied
    For i=1 to Population Size
    Calculate particle velocity according to (2)
    Update particle position according to (3)
     If f(X_i) <f(Pb_i)
       Pb_i = X_i
       If f(Pb_i)<f(Gb)
         Gb = Pb_i
       End
     End
   End
End
```

B. PSO-based PID controller approach

To this day, over 95 percentage of industrial applications are predominantly controlled by (PID) controllers. However, it is time consuming to find a set of parameter which satisfies several requirements at the same time specially to whom do not have knowledge about this controller behaviors. This subsection introduces the PSO−based PID approach and a proposed multi-objective optimization tool to obtain the optimal PID control parameters set.
First, consider the transfer function of a PID controller:

$$G_{PID}(s) = K_p + \frac{K_i}{s} + K_d s \tag{4}$$

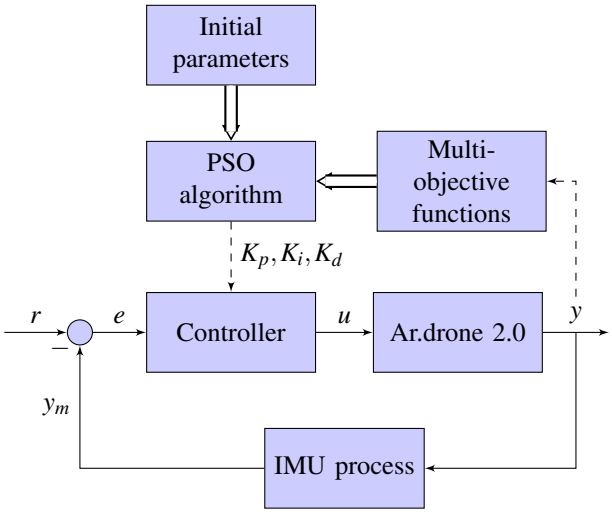

Fig. 3. PSO-based PID controller approach.

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 proposed approach based on PSO techniques is applied to find the optimal values for controller parameters that minimizes the three desired objective functions such as:

$$\begin{aligned} J_1(X) &= |SSE| \\ J_2(X) &= OS \\ J_3(X) &= T_s - T_r \end{aligned} \tag{5}$$

Where X is a set of parameters to be optimized,
$X = (K_p, K_i, K_d)$.
Fig. 3 shows the block diagram of PSO-based PID controller approach. In this procedure, the dimension of the particle is 3. 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.

C. Multi−objectives optimization approach

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.

$$J(X) = w_1 J_1(X) + w_2 J_2(X) + w_3 J_3(X) \tag{6}$$

where w_1, w_2 and w_3 are positive constants; $J_1(X)$, $J_2(X)$ and $J_3(X)$ are the objective functions defined as in section B. In this study, those values are set as $w_1 = 0.59$, $w_2 = 0.49$ and $w_3 = 0.88$.

D. Accelerated particle updates of MOPSO

In the conventional PSO, each particle updates its position based on both the current global best Gb and the personal

best Pb_i (or local best). More details can be found in [22]. 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, in order to accelerate the convergence of the algorithm, it is possible to use the global best only [23]. Based on that statement, the velocity vector and position vector are formulated as:

$$V_i(t+1) = V_i(t) + c_1 r + c_2(Gb(t) - X_i(t)) \quad (7)$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (8)$$

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:

$$c_1 = c_0 \gamma^t * (UB - LB) \quad (9)$$

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 $\gamma \in$ (0 1) is a control parameter.

E. The algorithm flow

In this subsection, the proposed MOPSO algorithm is presented according to the following steps.

Step 1(Initialization) 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]. In order to increase the optimal convergence, an initial swarm X is randomly generated in [LB UB]. The pseudo code for this step is:

- For i = 1 to N (N is number of particles)
- Initialize X_i in [LB_i UB_i]

Step 2 (Main loop) While (t<T_{max})
(T_{max}: predefined maximum iterations)
Step 2.1 (Update control parameter) $c_1 = c_0 * \gamma^t$ with γ is a control parameter, chosen in (0 1).
Step 2.2 (Update position) The velocity and position of particle is updated according to equations (7), (8).
Step 2.3 (Update the objectives) The composite objective value is calculated based on equation (6).
Step 2.4 (Update Gb) When the current position of the particle is better than the Gb contained in its memory, it will be updated using:
for i = 1: N
if $J(X_i) \leq Jbest$
$Gb = X_i$;
$Jbest = J(X_i)$
Step 2.5 (Update the iteration) $t = t$+1

Step 3 (Display results) Output optimal results.

IV. AR.DRONE 2.0 CONTROL RESULTS

In the following simulations, 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; γ = 0.97.

Having the dynamic model of the Ar.Drone 2.0 (section II), controllers parameters is achieved by applying PSO-based PID controller as presented in section III. Optimal parameters of PID controllers are obtained through simulations. In order to better understand the performance of the controllers, each one is analyzed separately. It can be noticed that the models obtained for X and Y position controllers are the same, hence their controllers have the same topology and parameters. Fig. 4 - Fig. 6 show the step response obtained in altitude, $X(Y)$ position and yaw control using the proposed MOPSO, Frtool [24] and PID tuner Matlab toolbox .The sets of optimal K_p, K_i, K_d parameters obtained by MOPSO method together with rise time, settling time, overshoot, undershoot, peak, peak time and the cost function of each Ar. Drone controller are presented in Table I. Four PID controllers based on MOPSO algorithm are successfully applied. 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 I, all designed controllers have no overshoot/ undershoot, 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 PSO-PID controller than for PID-Frtool and PID tuner.

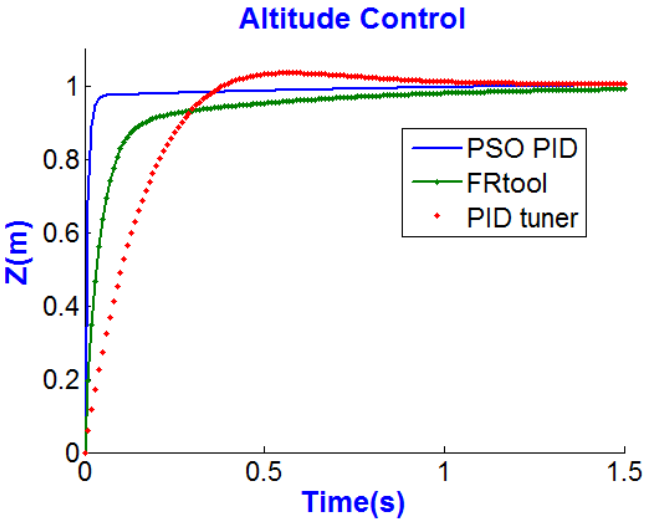

Fig. 4. Altitude step response with MPSO tuning, Frtool, PID tuner.

TABLE I
OPTIMAL CONTROL PARAMETER SELECTED BY PSO ALGORITHM FOR AR. DRONE 2.0 PID CONTROLLERS

PSO-PID PARAMETERS	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
Undershoot	0	0	0	0
Peak	0.9976	0.9999	0.9999	0.9925
Peak Time (s)	0.5338	0.0346	0.0346	0.0025
Cost function value	0.26	0.21	0.21	0.2950

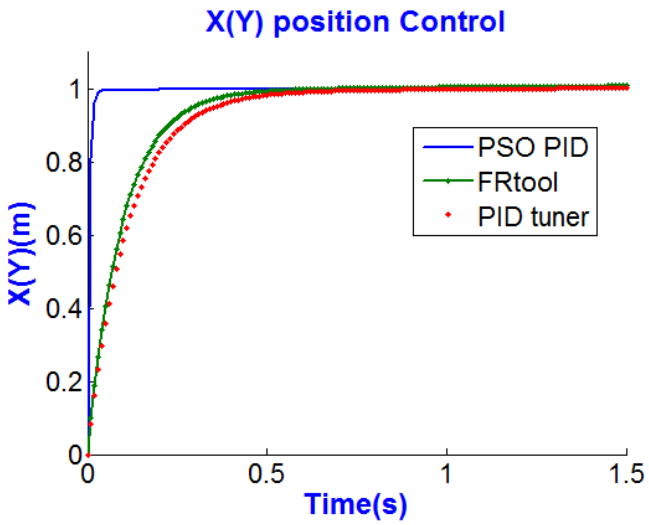

Fig. 5. $X(Y)$ position step response with MPSO tuning, Frtool, PID tuner.

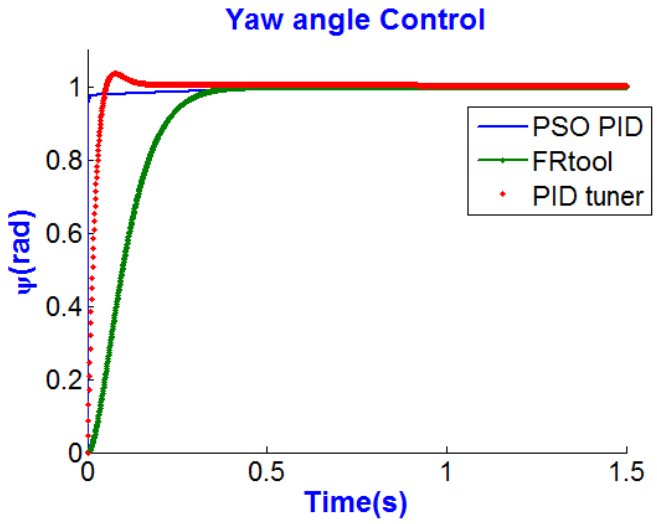

Fig. 6. Yaw angle step response with MPSO tuning, Frtool, PID tuner.

To investigate the robustness and sensitivity of the approach in changing of the weighted constants, we modify parameters w_1, w_2 , w_3 in the range of 20% those values. Therefore, the composite objective optimization is as following:

$$J(X) = (w_1 \pm \Delta w_1)J_1(X) + (w_2 \pm \Delta w_2)J_2(X) + (w_3 \pm \Delta w_3)J_3(X) \quad (10)$$

where:
w_1, w_2, w_3 are defined in III.*C*.
$\Delta w_1 \leq 0.2w_1$; $\Delta w_2 \leq 0.2w_2$; $\Delta w_3 \leq 0.2w_3$.

Fig.7 illustrates 6 different PSO-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 PSO-PID controllers react very fast and without overshoot and track the reference input very well. In addition, there are only slightly difference between 6 PSO-PID controllers' outputs. To conclude, the proposed approach is highly robust and not very sensitive in term of changing of the weighted constants.

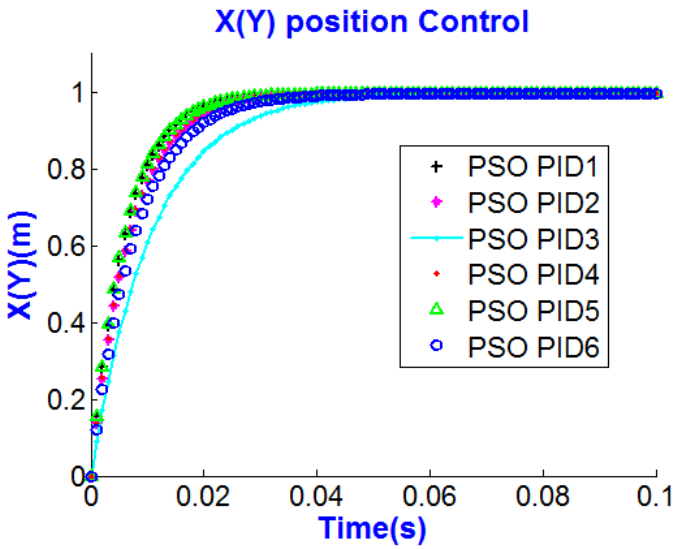

Fig. 7. Simulation results obtained from PSO-PID X(Y) control in the variance of w_1, w_2 , w_3.

Once the simulated results were satisfactory, the controllers were implemented on the real plant. The experiments were performed in an indoor environment without strong external disturbances as: wind or irregular ground. The reference for altitude is 2.4 meter. The experiment result is shown in Fig. 8. Since Ar. Done system is strongly dynamical system and includes internal noises, the experiment output is not as good

as simulation result. The error may also come from the identification process because the obtained model is never perfect. It can be observed that the error is accumulated over time. However, the designed controller tracks the reference appropriately. The results for $X(Y)$ positions and yaw angle control have similar behaviors with the one for altitude control hence they are skipped here.

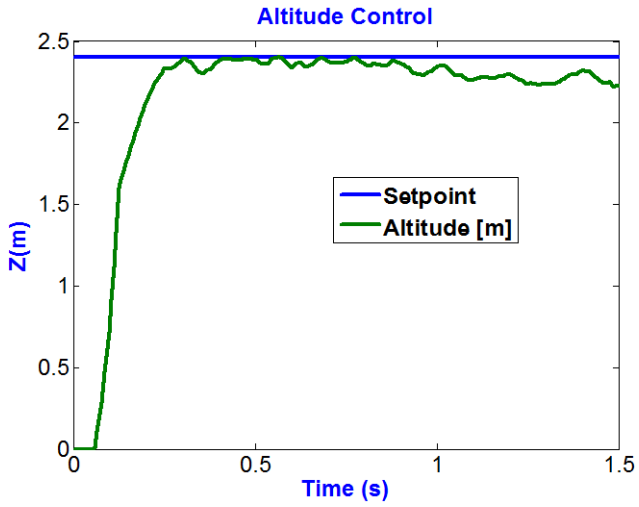

Fig. 8. Result obtained from PSO-PID altitude control on Ar.Drone 2.0.

V. Conclusion

In this paper, a proposed particle swarm optimization called multi-objective particle swarm optimization (MOPSO) with an accelerated update methodology is employed for UAV PID controller parameters tuning. Three PID control gains, i.e., the proportional gain K_p, integral gain K_i and derivative gain K_d are required to form a parameter vector which is considered as a particle of PSO. The PID controller parameters for altitude (Z), X and Y positions, yaw angle are optimized considering three essential criteria of PID controller as settling time/ rise time ($T_s - T_r$), overshoot (OS) and steady-state error (SSE). The outputs are compared to other tuning method to illustrate the better performances of the proposed approach. The results verify that the proposed MOPSO algorithm is an effective method for optimal tuning of PID parameters in term of no overshoot, zero steady state error and extremely short setting time and rise time. The proposed approach is highly robust and not very sensitive in term of changing of the weighted constants. This proposed tuning method may create some benefit in real-life and industrial applications since it saves time for none expert control engineering to find optimal controller parameters to enhance the performance quality. For future work, this method will be applied to tuning fractional-order (FO) PID controller.

References

[1] D. Mellinger and V. Kumar, Minimum snap trajectory generation and control for quadrotors, in Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA), 2011.

[2] R. Ritz, M. Mueller, and R. DAndrea, Cooperative quadrocopter ball throwing and catching, in Proc. IEEE Intl. Conf. on Intelligent Robots and Systems (IROS), 2012.

[3] A. Hernandez, C. Copot, J. Cerquera, H. Murcia, R. De Keyser, Formation Control of UGVs using an UAV as Remote Vision Sensor, Proceedings of the 19th IFAC World Congress, Cape Town, South Africa, , pp. 618 −623 2014.

[4] A. Hernandez, H. Murcia, C. Copot, R. De Keyser, Towards the Development of a Smart Flying Sensor: Illustration in the Field of Precision Agriculture, Sensors (Basel) 15(7), pp. 16688-16709, 2015.

[5] Q. Lindsey, D. Mellinger, and V. Kumar, Construction of cubic structures with quadrotor teams, in Proc. of Robotics: Science and Systems (RSS), 2011.

[6] T. T. Mac, C. Copot, A. Hernandez, R. De Keyser, Improved Potential Field Method for Unknown Obstacle Avoidance Using UAV in Indoor Environment, IEEE 14th International Symposium on Applied Machine Intelligence and Informatics (SAMI), Kosice, Slovakia,

[7] J. Li, Y. Li, Dynamic Analysis and PID Control for a Quadrotor, IEEE International Conference on Mechatronics and Automation, Beijing, China, pp. 573−578, 2011.

[8] Y. Bi, H. Duan, Implementation of autonomous visual tracking and landing for a low-cost quadrotor, Optik - International Journal for Light and Electron Optics, Vol.124, Issue 18, pp. 3296−3300, 2013.

[9] Y. Song, B. Xian, Y. Zhang, X. Jiang, X. Zhang, Towards autonomous control of quadrotor unmanned aerial vehicles in a GPS-denied urban area via laser ranger finder, Optik - International Journal for Light and Electron Optics, Vol.126, Issue 23, pp. 3877−3882,

[10] J. Engel, J. Sturm, D. Cremers, Scale-Aware Navigation of a Low-Cost quadrocopter with a Monocular Camera, Robotics and Autonomous Systems, Vol. 62 (11), pp. 1646−1656, 2014.

[11] V. Chugh, GA tuned LQR and PID controller for aircraft pitch control, IEEE 6th India International Conference on Power Electronics (IICPE), Kurukshetra , India, pp. 1−6, 2014.

[12] Q. Chen, T. Chen, Y. Zhang, Research of GA-based PID for AUV Motion Control, 2009 IEEE International Conference on Mechatronics and Automation, Changchun, China, pp. 4446−4451.

[13] H. Boubertakh, M. Tadjine, P. Y. Glorennec, S. Labiod, Tuning Fuzzy PID Controllers using Ant Colony Optimization, 17 th Mediterranean Conference on Control Automation, Thessaloniki, Greece, pp. 13−18, 2009.

[14] S. Pareek, M. Kishnani, R. Gupta, Application of Artificial Bee Colony Optimization For Optimal PID Tuning, IEEE International Conference on Advances in Engineering Technology Research (ICAETR), Unnao, India, pp. 1−5, 2014.

[15] J. Estevez, M. Grana, Robust Control Tuning by PSO of Aerial Robots Hose Transportation, IWINAC, Part II, LNCS 9108, pp. 291300, 2015, DOI: 10.1007/978-3-319-18833-1$_3$1

[16] A. A. Mayahi, W. Wang, P. Birch, Path Tracking of Autonomous Ground Vehicle Based on Fractional Order PID Controller Optimized by PSO, IEEE 13th International Symposium on Applied Machine Intelligence and Informatics (SAMI), Herlany, Slovakia, pp. 109−114, 2015.

[17] S. Piskorski, N. Brulez, E. Eline, and F. D'Haeyer, Ar. drone developer guide-sdk 2.0, 2012.

[18] T. Krajnik, V. Vonasek, D. Fiser, and J. Faigl, Ardrone as a platform for robotic research and education, Research and Education in Robotics - Eurobot, Volume 161, pp.172−186, 2011.

[19] https://github.com/puku0x, cv drone free software, December 2013.

[20] L. Ljung, System identication: theory for the user, Prentice-Hall, 2007.

[21] A. Hernandez, C. Copot, J. Cerquera, H. Murcia, R. De Keyser, Model Predictive Path-Following Control of an AR.Drone Quadrotor, XVI Latin American Control Conference The International Federation of Automatic Control, Cancun, Mexico, 2014.

[22] M. Clerc, J. Kennedy, The particle swarm−explosion, stability, and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation, Vol. 6, pp. 58−73, 2002.

[23] X. S. Yang, S. Deb, S. Fong, Accelerated Particle Swarm Optimization and Support Vector Machine for Business Optimization and Applications, Networked Digital Technologies, Communications in Computer and Information Science, Vol. 136, Springer, pp. 53−66, 2011.

[24] R. De Keyser, C.M. Ionescu, FRTool: a frequency response tool for CACSD in Matlab, in Proc. of the IEEE Conf. on Computer Aided Control Systems Design, Munich, Germany, pp. 2275-2280, 2006.