applied sciences

MDPI

Article

Optimization of PID Controller to Stabilize Quadcopter Movements Using Meta-Heuristic Search Algorithms

Alaa Sheta [1,*], Malik Braik [2], Dheeraj Reddy Maddi [3], Ahmed Mahdy [3], Sultan Aljahdali [4] and Hamza Turabieh [5]

[1] Computer Science Department, Southern Connecticut State University, New Haven, CT 06515, USA
[2] Department of Computer Science, Al-Balqa Applied University, Salt 19117, Jordan; mbraik@bau.edu.jo
[3] Department of Computing Sciences, Texas A&M University, Corpus Christi, TX 78412, USA; dmaddi@islander.tamucc.edu (D.R.M.); ahmed.mahdy@tamucc.edu (A.M.)
[4] Department of Computer Science, College of Computers and Information Technology, Taif University, P.O. Box 11099, Taif 21944, Saudi Arabia; aljahdali@tu.edu.sa
[5] Department of Information Technology, College of Computers and Information Technology, Taif University, P.O. Box 11099, Taif 21944, Saudi Arabia; h.turabieh@tu.edu.sa
* Correspondence: shetaa1@southernct.edu

Abstract: Quadrotor UAVs are one of the most preferred types of small unmanned aerial vehicles, due to their modest mechanical structure and propulsion precept. However, the complex non-linear dynamic behavior of the Proportional Integral Derivative (PID) controller in these vehicles requires advanced stabilizing control of their movement. Additionally, locating the appropriate gain for a model-based controller is relatively complex and demands a significant amount of time, as it relies on external perturbations and the dynamic modeling of plants. Therefore, developing a method for the tuning of quadcopter PID parameters may save effort and time, and better control performance can be realized. Traditional methods, such as Ziegler–Nichols (ZN), for tuning quadcopter PID do not provide optimal control and might leave the system with potential instability and cause significant damage. One possible approach that alleviates the tough task of nonlinear control design is the use of meta-heuristics that permit appropriate control actions. This study presents PID controller tuning using meta-heuristic algorithms, such as Genetic Algorithms (GAs), the Crow Search Algorithm (CSA) and Particle Swarm Optimization (PSO) to stabilize quadcopter movements. These meta-heuristics were used to control the position and orientation of a PID controller based on a fitness function proposed to reduce overshooting by predicting future paths. The obtained results confirmed the efficacy of the proposed controller in felicitously and reliably controlling the flight of a quadcopter based on GA, CSA and PSO. Finally, the simulation results related to quadcopter movement control using PSO presented impressive control results, compared to GA and CSA.

Keywords: quadcopter; PID controller; Genetic Algorithms; Crow Search Algorithm; Particle Swarm Optimization

Citation: Sheta, A.; Braik, M.; Maddi, D.; Mahdy, A.; Aljahdali, S.; Turabieh, H. Optimization of PID Controller to Stabilize Quadcopter Movements Using Meta-Heuristic Search Algorithms. *Appl. Sci.* **2021**, *11*, 6492. https://doi.org/10.3390/app11146492

Academic Editor: Silvio Cocuzza

Received: 16 February 2021
Accepted: 3 July 2021
Published: 14 July 2021

1. Introduction

Recently, there has been an upward trend in the popularity of Unmanned Aerial Vehicles (UAVs) in a wide range of practical applications. While large outdoor UAVs are already in application for civil, commercial or military activities, the indoor flight of small UAVs remains a challenging field of application from a scientific point of view. Inland flight requires the right type of vehicle and appropriate navigation, control and collision-avoidance algorithms. In terms of vehicle type, helicopter-like vehicles are among the most encouraging candidates in terms of their weight, low cost, size, high maneuverability, slow-flying ability and even stable hovering flight in constrained conditions. One special helicopter-like vehicle with the added advantage of plain construction and rotor mechanics is the quadrotor.

Quadrotors have received much interest in the research community, where a large number of applications are being presented and experimented. In particular, there has been a growing interest in quadcopter applications, such as object detection and landing to drone stations [1], rescue and disasters [2], agriculture [3], and even other tasks, such as those reported in [4]. It is critically essential for the success of the missions performed that the UAVs precisely and rapidly follow the required path in civilian purposes, such as logistics, mapping, search and rescue, surveillance, exploration and many more military missions, such as attack, defense, supervision and surveillance [5,6].

This current expansion is mostly attributed to the lower cost of on-board sensors, actuators, small-scale embedded computing platforms and simplified rotor mechanics [7]. In spite of the significant advances, flight control is still an open topic of research. By varying the speed of the individual motors, the lift force can be varied, and lateral and/or vertical motion can be generated. However, despite the four actuators, the quadrotor is a non-linear, aerodynamically unstable system that must be stabilized by an appropriate control system. On the one hand, flight control essentially alludes to the ability to carry out highly time-sensitive sensory data acquirement, processing, and the computation of forces for application to aircraft actuators. On the other hand, to name a few, it is advisable that UAV flight controllers are able to withstand faults, adapt to changes in payload and/or the environment, and improve the flight path. UAV flight control systems are mostly implemented using Proportional-Integral-Derivative (PID) control systems. PID has shown exceptional performance in many conditions, including in the context of drone racing, where accuracy and lightness are key factors. In contrast, it has many handicaps: (1) at times, it provides a high control signal due to wind, thus overshooting and continuing to increase as the accumulated error is unwound; and (2) the differentiator leads to the amplification of noise. The PID controller system is the preferred control method [8]. The PID controller has only three parameters, giving it an easy adjustment space to obtain a better control point [9]. It is evident that fast response speed and high controllability can be achieved by implementing proper PID parameters [10]. Therefore, the method of tuning PID parameters has become one of the hot research topics in the engineering field. In stable environments, a PID controller displays close to the best possible performance. When bared to unknown dynamics, such as variable payloads, wind and voltage drops, however, a PID controller can be far from optimal [11]. On the other hand, because quadcopters are dynamically unstable, under-actuated and nonlinear systems, it is difficult to design a quadcopter's PID controller to track a moving target, particularly under the variable speed of the moving target and under the influence of the environment. This is because any error that occurs in any of them can cause serious damage to the quadcopter.

The PID controller successfully tracks a reference altitude, despite undergoing a large overshoot on takeoff. This may be related to the ground impact, which produces an unexpectedly strong lift before the quadcopter surveys the ground. Understanding how to balance the different aspects of flight to make a quadcopter respond optimally to a particular flight style is not a simple matter that can happen overnight. Practical knowledge of PID tuning helps to achieve this easily, and the more familiar the operator is with the PID settings, the easier it is to set the codecs to fly in any way.

Conventional methods, such as Ziegler–Nichols (ZN) [12], for the PID controller does not guarantee optimal control for a quadcopter. This method might cause system instability as well as major losses and damages to the system. Therefore, a more efficient control method is needed for the quadcopter that can make the system very stable with minimal losses and without damaging the system. A secondary goal is to achieve reliable and proper movements and directions for the quadcopter. For next-generation flight control systems to be intelligent, there is a need to devise a way to integrate adaptability to the changeable environment and dynamics. Accordingly, the evolution of intelligent flight control systems remains an active research domain [13], concretely through the use of artificial neural networks, which are an appealing choice that have been agreed on as global approximators, and noise-resistance [14]. Online learning methods [15] have the merit of learning the

dynamics of an aircraft in real time. The key restriction with these methods is that the flight control system is only aware of its past experiences. It follows that its performance level is constrained when exposed to an unprecedented event. Offline training models using supervised learning are problematic, as data acquisition is prohibitively expensive and derived from imprecise representations of rudimentary aircraft dynamics, which can result in suboptimal control policies [16].

At present, many research trends in adaptive control are gradually being turned toward the use of meta-heuristic algorithms to accommodate and solve professional engineering problems [17,18]; these algorithms have enticed the attention of many researchers [19]. Overall, a variety of studies have been conducted based on meta-heuristic algorithms to obtain appropriate PID parameters, such as Genetic Algorithms (GA) [20], Particle Swarm Optimization (PSO) [21], the Artificial Immune System (AIS) [22] and the Kidney-Inspired Algorithm (IKA) [23]. A PID controller designed by other meta-heuristic optimization methods was also applied in servo systems. Pršić et al. [24] used the Firefly algorithm to adjust the three parameters of the PID controller in parallel robot platforms with six degrees of freedom gain. The Beetle Antennae Search (BAS) algorithm was adapted for use in UAV avoidance [25], trajectory organizing and control systems [26,27].

Most of the previous studies show reasonable performance and results. However, they either use the Global Positioning System (GPS) to discover the location of the target, which provides inaccurate identification in indoor environments, or they use sensor fusion. Other researchers have focused more on the part of computer vision, ignoring the influence of external disturbances, such as the impact of the environment.

Objectives of the Work

The current work focuses on the PID controller of a quadcopter whose parameters are to be tuned by meta-heuristic algorithms, including Genetic Algorithms (GA) [28], the Crow Search Algorithm (CSA) [29,30] and Particle Swarm Optimization (PSO) [31,32]. These meta-heuristics have long been accepted by the artificial intelligence community for their adequacy in solving a wide range of real applications by arriving at optimal or accurate solutions with respect to a compact set of parameters, such as time or cost [33–35]. The optimization methods for the quadcopter PID controller, in this work, were mathematically formulated to control the movements of the quadcopter in the x, y and z directions over the quadcopter coordinates. In the assessment of the performance of the optimization algorithms used in the work, such parameters as overshooting, rising time and settling time were integrated in a multi-objective fitness function in order to achieve an optimal gain for the PID controller.

The key contributions of the proposed work lie in the following: (1) overcoming the nonlinearity and instability of a quadcopter system through the use of three meta-heuristics that control the movements and directions of a quadcopter with the aim of obtaining high gain for the quadcopter's PID controller; (2) it is expected that this will allow verification of the system performance and make a comparison between these meta-heuristics, using a set of related measures, as the second contribution; and (3) the third contribution is the comparison of the results of these meta-heuristics with the results of a traditional method.

The mathematical model used to simulate the movements of a quadcopter is given in Section 2. Section 3 then presents the PID controller for the quadcopter's motors. The following Section 4 presents a brief description of GA, CSA and PSO. The evaluation methods and experimental results are presented in Section 6. Section 7 then presents a statistical analysis test for the controlling methods. Section 8 then discusses the results with closing comments, and further outlooks are given in Section 9.

2. Mathematical Model of Quadcopter

A mathematical model is needed to simulate and/or monitor the movement of a quadcopter in a three-dimensional space space with the configuration shown in Figure 1. This model is essential to establish a relationship to link the inputs with the outputs of

the quadcopter system. The model can be used to predict the position and attitude of a quadcopter, using the four angular velocities of the propellers, where a quadcopter can operate under various operating conditions with respect to different input values. The stability and control of a quadcopter can be achieved with a suitable model [36].

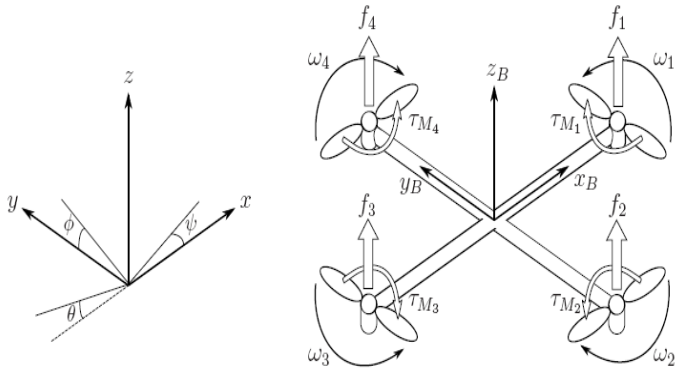

Figure 1. Quadcopter configuration.

The quadcopter is an UAV with four rotors connected to two horizontal bands. Rotors 1 and 3 rotate clockwise, while rotors 2 and 4 rotate counterclockwise to balance the torque. Quadcopters can move in six different ways by combining translational and rotational motion. The four inputs of the quadcopter can be described as follows:

1. **Thrust (z):** This is the vertical force that moves the quadcopter through the air in any direction, which is caused by increasing or decreasing the speed of all four rotors by the same amount, greater or lesser than the gravitational force.
2. **Roll angle (ϕ):** This identifies the rotation of the quadcopter around the x-axis, where the roll movement of the quadcopter is obtained by decreasing the speed of the second rotor and increasing the speed of the fourth rotor. This moves the quadcopter to the sides, engendering it to roll. However, it does not cause the quadcopter to alter its altitude position. These "rolls" make the quadcopter move left and right on its horizontal axis.
3. **Pitch (θ):** This identifies the rotation of the quadcopter around the y-axis, where the pitch movement is obtained by decreasing the speed of the first rotor and increasing the speed of the third rotor. The pitch means that the quadcopter tilts downwards or upwards on the basis of its orientation and the location of its nose. An upwards tilt will move the quadcopter in a backwards motion, while a downwards tilt will move it in a forwards motion.
4. **Yaw (ψ):** This identifies the rotation of the quadcopter around the z-axis, whereby the yaw motion is obtained by increasing the angular velocities of two opposite rotors and decreasing the velocities of the other two. Yaw refers to the direction that the front of the quadcopter faces when rotating either clockwise or counterclockwise on its vertical axis.

2.1. Quadcopter Coordinates

Referring to the dynamic model of a quadcopter presented in Figure 2, the following two reference frames can be used to describe the dynamic behavior movement of a quadcopter [37]:

- The linear and angular positions of the Earth's inertial frame (E) can be defined as follows:

$$\vec{\zeta} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \vec{\eta} = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \quad (1)$$

- The linear and angular velocities of the body-fixed frame (B) can be defined as follows:

$$\vec{Y} = \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \vec{\Omega} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2}$$

The following assumptions were taken into account when developing a dynamic model of a quadcopter:

- The body of the quadcopter is rigid and symmetrical.
- The center of gravity and center of body mass coincide.
- The aerodynamic effects are not significant in the case proposed in this work.

Equations (21) and (22) can be used to derive the formulas for $\dot{\zeta}$ and $\dot{\eta}$ as defined below:

$$\dot{\zeta} = \mathbf{R}Y \tag{3}$$
$$\dot{\eta} = \mathbf{J}\Omega \tag{4}$$

The rotation matrices along x, y and z axes are defined below:

- Rotation along the x-axis is as follows:

$$\mathbf{R}_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C_\phi & -S_\phi \\ 0 & S_\phi & C_\phi \end{bmatrix}$$

where C_ϕ and S_ϕ indicate $\cos\phi$ and $\sin\phi$, respectively.

- Rotation along the y-axis is as follows:

$$\mathbf{R}_\theta = \begin{bmatrix} C_\theta & 0 & S_\theta \\ 0 & 1 & 0 \\ -S_\theta & 0 & C_\theta \end{bmatrix}$$

where C_θ and S_θ indicate $\cos\theta$ and $\sin\theta$, respectively.

- Rotation along the z-axis is as follows:

$$\mathbf{R}_\psi = \begin{bmatrix} C_\psi & -S_\psi & 0 \\ S_\psi & C_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where C_ψ and S_ψ indicate $\cos\psi$ and $\sin\psi$, respectively.

- Equation (6) can be used to convert all the angles from the B frame to E frame:

$$\mathbf{R} = \mathbf{R}_\phi \times \mathbf{R}_\theta \times \mathbf{R}_\psi \tag{5}$$

$$\mathbf{R} = \begin{bmatrix} C_\psi C_\theta & C_\psi S_\theta S_\phi - S_\psi S_\phi & C_\psi S_\theta C_\phi + S_\psi C_\phi \\ S_\psi C_\theta & S_\psi S_\theta S_\phi + C_\psi C_\phi & S_\psi S_\theta C_\phi - C_\psi S_\phi \\ -S_\theta & C_\theta S_\phi & C_\theta C_\phi \end{bmatrix} \tag{6}$$

- The angular velocity translation matrix ($\mathbf{J}$) is defined in Equation (7).

$$\mathbf{J} = \begin{bmatrix} 1 & 0 & -S_\theta \\ 0 & C_\phi & S_\phi C_\theta \\ 0 & -S_\phi & C_\phi C_\theta \end{bmatrix} \tag{7}$$

Finally, the kinematic model for a quadcopter is given in Equation (8):

$$\begin{aligned}
\dot{x} &= w[S_\phi S_\psi + C_\phi C_\psi S_\theta] - v[C_\phi S_\psi - C_\psi S_\phi S_\theta] + u[C_\psi C_\theta] \\
\dot{y} &= v[C_\phi C_\psi + S_\phi S_\psi S_\theta] - w[C_\psi S_\phi - C_\phi S_\psi S_\theta] + u[C_\theta S_\psi] \\
\dot{z} &= w[C_\phi C_\theta] - u[S_\theta] + v[C_\theta S_\phi] \\
\dot{\phi} &= p + r[C_\phi T_\theta] + q[S_\phi T_\theta] \\
\dot{\theta} &= q[C_\phi] - r[S_\phi] \\
\dot{\psi} &= r \times C_\phi / C_\theta + q \times S_\phi / C_\theta
\end{aligned} \tag{8}$$

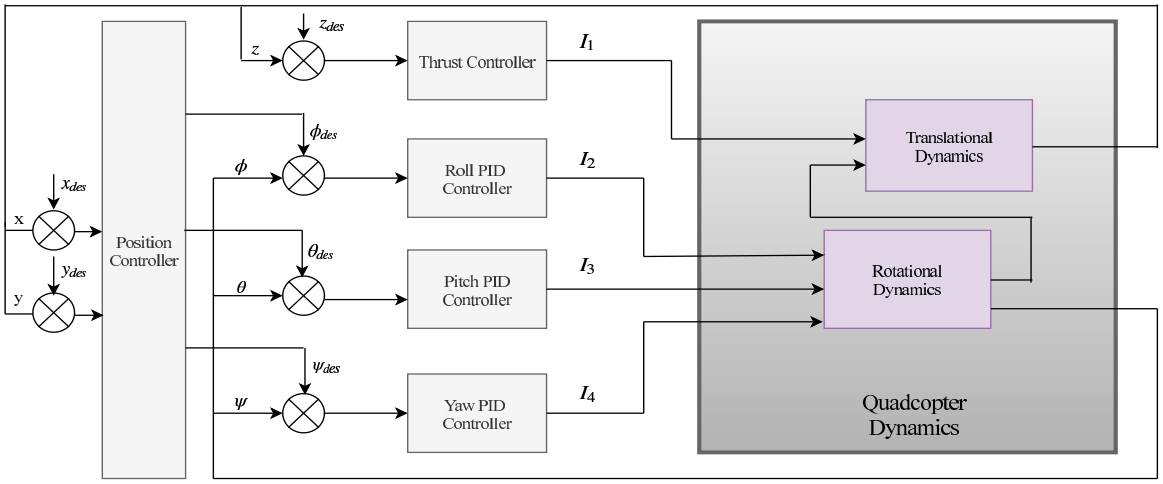

Figure 2. Block diagram of a quadcopter model.

2.2. Forces and Moment

The following terms are used in the control optimization process presented in this work:

- **The moment of inertia or angular mass** is the rotating body's resistance to angular acceleration or angular deceleration. It is best described as the product of the mass and the square of its perpendicular distance from the axis of rotation.
- **Torque** is a measure of how much force is applied on an object, causing the object to rotate. It can be defined mathematically as $\tau = r \times F$, where F is the force and r is the moment arm.

After getting the relationship between the body frame and the inertial frame, we can calculate the forces and torque acting on the quadcopter so that we can control the movements of the quadcopter. The thrust force acting upwards is given as follows:

$$T_B = \sum_{i=1}^{4} T_i = k \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} \omega_i^2 \end{bmatrix} \tag{9}$$

The torque force is the rotational force operating on the body of the quadcopter, making the body rotate in a specific direction.

$$\tau_B = \begin{bmatrix} L \cdot k \cdot (-\omega_1^2 + \omega_3^2) \\ L \cdot k \cdot (-\omega_2^2 + \omega_4^2) \\ b \cdot (-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2) \end{bmatrix} \tag{10}$$

In the E frame, the linear acceleration of a quadcopter is a combination of gravitational force and thrust force.

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + T_B/m \cdot \begin{bmatrix} \cos\psi \sin\theta \cos\phi + \sin\phi \sin\theta \\ \sin\phi \sin\theta \cos\phi - \cos\psi \sin\phi \\ \cos\theta \cos\phi \end{bmatrix} \tag{11}$$

where x is a positional vector, $\ddot{x}$ is the linear acceleration, g is the gravity force, and T_B is the thrust vector. The motion of angular acceleration can be defined as follows:

$$\begin{aligned} \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} &= \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \cdot \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} \\ &+ \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \cdot \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \cdot \begin{bmatrix} p \\ q \\ r \end{bmatrix} \end{aligned} \tag{12}$$

$$\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} I_{xx} \cdot \dot{p} \\ I_{yy} \cdot \dot{q} \\ I_{zz} \cdot \dot{r} \end{bmatrix} \begin{bmatrix} -I_{yy} \cdot qr + I_{zz} \cdot qr \\ I_{xx} \cdot pr - I_{zz} \cdot pr \\ -I_{xx} \cdot pq + I_{yy} \cdot pq \end{bmatrix} \tag{13}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{M_x + (I_{yy} - I_{zz}) \cdot qr}{I_{xx}} \\ \frac{M_y + (I_{zz} - I_{xx}) \cdot qr}{I_{yy}} \\ \frac{M_z + (I_{xx} - I_{yy}) \cdot pq}{I_{zz}} \end{bmatrix} \tag{14}$$

where ω is the angular velocity, τ is the torque and I is the moment of inertia. The translation and rotation movements can be defined as follows:

$$\ddot{x} = \frac{(\cos\psi \sin\theta \cos\phi + \sin\phi \sin\theta) \cdot T_B}{m} \tag{15}$$

$$\ddot{y} = \frac{(\sin\phi \sin\phi \cos\phi - \cos\psi \sin\phi) \cdot T_B}{m} \tag{16}$$

$$\ddot{z} = \frac{(\cos\phi \cos\theta) \cdot T_B}{m} \tag{17}$$

$$\dot{p} = \frac{M_x + (I_{yy} - I_{zz}) \cdot qr}{I_{xx}} \tag{18}$$

$$\dot{q} = \frac{M_y + (I_{zz} - I_{xx}) \cdot qr}{I_{yy}} \tag{19}$$

$$\dot{r} = \frac{M_z + (I_{xx} - I_{yy}) \cdot pq}{I_{zz}} \tag{20}$$

3. PID Controller

Due to the increasing number of applications and complexity of the quadcopter on a daily basis, approved control algorithms also need to progress so that they can deliver better performance and boost flexibility [38]. In the past, modest linear control methods were accepted, due to their lower computational requirements and stable behavior. However, with advanced modeling methods and faster on-board computational power, a wide-range of nonlinear methods have been used to run in real time. The proposed nonlinear control methodologies have grown rapidly, and they provide high performance, making the quadcopter more powerful [39,40].

In this paper we explore one of the well-recognized controllers in industry, called the PID controller [34,41]. The PID controller has many merits, such as reducing the downtime, rise time, and peak error, and smoothing the movement of the output in addition to the overshoot of the system. Meanwhile, this type of controller has many difficulties in adjusting its parameters. For example, its low ability to tune P, I and D parameters may result in poor performance, slow control and, in some cases, system instability. Consequently, many types of tuning methods for the PID controller demand considerable attention from the operator to identify the optimal set of values for those parameters to reach a reasonable gain. One of the conventional methods widely used to fine-tune the parameters of the PID controller is the Zeigler–Nichols method [12]. However, in a few past cases, this method did not produce sensible results in reducing the time response of the signal as reported in the literature.

Based on the developed mathematical model for the quadcopter, the four inputs of the PID controller of the quadcopter's motors were tuned to control the altitude, attitude, heading and position of the quadrotor in space. In this work, GAs, PSO and CSA are presented to optimize the gain of the PID controller in order to improve the system's dynamic response.

$$P = K_p \times e(t) \tag{21}$$

$$I = K_i \times \int_0^t e(t) \tag{22}$$

$$D = K_d \times \frac{de(t)}{dt} \tag{23}$$

The control function can be expressed mathematically as given in Equation (24).

$$u(t) = K_p e(t) + K_i \int_0^t e(t) + K_d \frac{de(t)}{dt} \tag{24}$$

The transfer function of the PID controller is defined as presented in Equation (25).

$$U(S) = K_p + K_i \times \frac{1}{s} + K_d \times s \tag{25}$$

The transfer function is capable of capturing the dynamics of the quadcopter's movements [42]. Furthermore, the transfer function is utilized to demonstrate the effectiveness of the presented method. The transfer functions for the inputs of the PID controller, ϕ, θ, ψ and z are defined in Equations (26)–(29), respectively [42].

$$G(s) = \frac{65 \times s + 4560}{s^3 + 109 \times s^2 + 1023 \times s + 2935} \tag{26}$$

$$G(s) = \frac{56.95 \times s + 4391}{s^3 + 105 \times s^2 + 870 \times s + 4430} \tag{27}$$

$$G(s) = \frac{105}{s^2 + 413 \times s} \tag{28}$$

$$G(s) = \frac{1.63}{s^2 + 5 \times s} \tag{29}$$

The PID controller was designed using closed-loop feedback control in order to stabilize the closed-loop system based on pre-defined equality characteristics. The PID controller can be described by a discrete transfer function as that given in Equation (30).

$$U(s) = \frac{K_d \times s^2 + K_p \times s + K_i}{s} \tag{30}$$

At each generation, the error is recomputed for the closed-loop system, where the resulting error is corrected by the PID controller and passed as inputs to the motors of the quadcopter.

In short, flight control systems for UAVs are mostly carried out using PID control systems. A PID controller is typically used to stabilize the rotational and translational motion of an UAV quadrotor system and enforce it to follow a specific path with minimum energy and error. It is evident that a fast response speed and high control capacity can be obtained by implementing adequate PID parameters. Thus, the method of tuning the PID parameter has become one of the hotspots of research in the engineering field. The tuning process, by which optimal values for the controller parameters are gained, is a crucial challenge. Many studies have been performed to locate the best way to fine-tune the PID parameters in order to obtain appropriate performance, such as rapid response, zero steady-state error, and minimal undershoot/overshoot. Although there are only three parameters, tuning PID parameters is a difficult process because it must meet complex criteria within the limitations of system actuators. In addition, the conventional PID controller only works for lower-order systems and lacks robustness against large uncertainties in system parameters. This is due to the inadequate number of parameters for handling the independent specifications of the time–domain response, such as the overshooting and settling time. Therefore, an effective PID controller control method is needed to have a better control point. The evolution of a method for tuning quadcopter PID parameters may save effort and time, and preferable control performance can be reached. For this intent, we propose, in this paper, a mathematics-based framework using several meta-heuristic optimization algorithms to design a robust PID controller for an UAV system. The produced control system should improve time-response dynamic properties, guarantee durability against large parameter uncertainties, and provide an effective tuning tool with fewer tuning parameters. After the determination of the parameters of the PID controller, the closed-loop system will achieve the required performance criteria.

4. Bio-Inspired Meta-Heuristic Algorithms

Bio-inspired optimizer algorithms are of great value for addressing many types of complex real-world problems, making it feasible to arrive at reasonable global solutions in a plausible time frame with better accuracy and reliability than classical search methods [43]. In fact, there is a great need, in practice, for large reliability and adequacy as well as low computational burden when the problem under investigation is so challenging and the search area is quite large. Moreover, in some cases, there is a need to compare the appropriateness of particular and global solutions.

The fundamental motivation for using meta-heuristic algorithms in this controlling and optimization problem is reliable optimization through the use of a local search strategy and the randomness in their evolutionary processes to find optimal or near-optimal solutions. However, this does not always result in meta-heuristics finding compelling or optimal solutions, as was shown in several cases [33,44,45]. In such a context, exploration and exploitation constitute the fundamental and important processes of meta-heuristic algorithms. While the meta-heuristic algorithm endeavors to assay various areas in the search domain during the exploration process, it seeks to converge the search into all sub-optimal solutions found in the exploitation process [33,43].

Broadly speaking, nature-inspired algorithms have intelligence behavior based on the collective behavior of a group of social individuals with multiple collaborating agents for the sake of optimizing a given problem. Although these algorithms are made up of relatively modest individuals, they offer harmonious social behavior to guide the individuals to their intended goals. These algorithms outset by randomly creating a predetermined number of possible solutions. The candidate solutions are then updated at each iteration loop, using a particular mechanism on the basis of the meta-heuristic algorithm used. Each algorithm evaluates each solution within each iteration loop, using an objective function. Evolutionary Algorithms (EAs), such as Genetic Algorithms, and Swarm Intelligence (SI), such as Particle Swarm Optimization and the Crow Search Algorithm, are two categories of bio-inspired meta-heuristic algorithms.

5. Proposed Multi-Objective Fitness Function

Fitness function plays a key role in the optimization problems that assists the optimization method in measuring the quality of the solution of the problem under study and how effective the solution of the problem is. The fitness function used by the meta-heuristic algorithms presented in this work, namely GA, CSA and PSO, to optimize the parameters of the PID controller for the inputs of the quadcopter is the integration of the weighted absolute error time and other three parameters identified as the overshoot, rise time and settling time of the quadcopter. Therefore, this fitness function is a multi-objective function as given in Equation (31):

$$Fitness = \beta_1 \times K + \beta_2 \times r + \beta_3 \times s + \beta_4 \times o \tag{31}$$

where $\beta_1, \beta_2, \beta_3$ and β_4 represent the weighting parameters that boost the efficacy of PSO; r, s and o are the rise time, settling time and overshoot, respectively; and K is the integral of the weighted absolute error time defined in Equation (32) [46]:

$$K = \int_0^n t \times |e| dt \tag{32}$$

where n stands for the number of samples, t represents the sample time and e stands for the error difference between unity and the response time of the closed-loop system.

In Equation (31), the values of $\beta_1, \beta_2, \beta_3$ and β_4 are arbitrarily defined as 0.7, 0.1, 0.1 and 0.1, respectively. So, the proposed fitness function is intended to reduce all the parameter values, K, r, s and o, concurrently based on the priority that is set as the weight parameters, namely $\beta_1, \beta_2, \beta_3$ and β_4.

Notably, the fitness function proposed for GA, CSA and PSO to implement the problem under study is a multi-objective function that integrates several parameters, including K, o, r and s. This combination of these parameters is targeted to increase the gain of the PID controller in the optimization process as well as to contribute to measuring the impact of the closed system performance based on the step response. As a matter of fact, K in Equation (32) is utilized in Equation (31) to arrive at an optimal PID control system design as confirmed in [30]. This parameter is integrated into the multi-objective fitness function to produce a system with a much faster settling time.

6. Experimental Results

This section presents the evaluation results of the conventional ZN method along with the results of GA, CSA and PSO in adjusting the PID parameters while controlling the movements of the quadcopter and provides an overall analysis of the varied computational time elapsed by these algorithms.

6.1. Parameter Setting

The parameter settings used by GA, PSO and CSA to optimize the gain of the PID controller for the developed experiments are displayed in Table 1. The parameters' values were selected based on the design of the experiments to conform each algorithm to the fundamental nature of the controlling problem. In fact, the values of the parameters were changed several times until a reasonable solution was found. However, only good settings are often obtained, not the *best* settings.

Table 1. Parameter setting of GA, PSO and CSA used in this study.

Algorithm	Parameter	Value
GA	Number of genes	100
	Mutation fraction	0.1
	Crossover fraction	0.8
	Lower bound	0
	Upper bound	50
	Number of iterations	100
PSO	Number of particles	100
	Inertia weight (ω)	1
	Cognitive parameter (α)	1.5
	Social parameter (γ)	2.0
	Lower bound	0
	Upper bound	50
	Number of iterations	100
CSA	Number of crows	100
	Awareness probability (AP)	0.1
	Flight length (fl)	2
	Lower bound	0
	Upper bound	50
	Number of iterations	100

6.2. Time Results

The overshoot, rise time and settling time values obtained by the ZN method, GAs, PSO and CSA during controlling the parameter inputs of the PID controller, namely, roll (ϕ), pitch (θ), yaw (ψ) and thrust (z), while controlling the movement of the quadcopter are presented in Table 2.

Table 2. Overshoot, rise time and settling time obtained by ZN, GAs, PSO and CSA.

Method		**ZN**			**GAs**	
Evaluation	**Overshoot**	**Rise time**	**Settling time**	**Overshoot**	**Rise time**	**Settling time**
Roll	6.334	0.26	0.781	1.9857	0.117	0.164
Pitch	4.791	0.168	0.79	0.0451	0.224	0.364
Yaw	10.266	0.0023	0.019	1.351	0.160	0.263
Thrust	10.2668	0.189	1.582	1.999	0.038	0.517
Method		**PSO**			**CSA**	
Evaluation	**Overshoot**	**Rise time**	**Settling time**	**Overshoot**	**Rise time**	**Settling time**
Roll	0	0.119	0.181	0.390	0.141	0.2182
Pitch	0.0386	0.192	0.28	0.0193	0.2236	0.362
Yaw	0.0425	0.166	0.296	5.84	0.141	1.507
Thrust	0.0159	0.044	0.564	0	0.099	0.9603

The overshoot, rise and settling times in Table 2 obtained by the ZN method and optimization process conducted by GAs, PSO and CSA for the PID controller parameters for a quadcopter illustrate the efficiency of these methods in controlling the movement of the quadcopter. It is clearly noted from Table 2 that the overshoot, rise time and settling time reported by the ZN method for the roll, pitch, yaw and thrust inputs of the quadcopter are relatively large. The overshoot for input yaw of the ZN method is analogous to the overshoot for the thrust input. The rise and settling times reported for the input yaw by the ZN method are very small, compared to the rise and settling times reported for the input thrust. Moreover, the settling time reported for the input roll by the ZN method is comparable to that of the input roll reported for the input pitch. The overshoot, rise time and settling time obtained by the GA-based approach are relatively small. Further,

the overshoot values obtained with the GA-based approach are small, compared to the corresponding overshoot values obtained by the ZN method. However, the rise and settling times obtained by the ZN method for inputs yaw and thrust are smaller than those inputs obtained with the GA-based approach. The overshoot values for inputs roll, pitch and thrust obtained by CSA are smaller than those obtained by GA. However, the overshoot value for the input yaw-obtained CSA-based approach is smaller than that obtained by GA. By comparing the control responses from GA- and CSA-based methods for the position and orientation of the PID controller, the CSA-based method has smaller overshoots for inputs roll and yaw of the quadcopter. However, the CSA-based method has longer settling times for these inputs.

As Table 2 shows, the overshoot, rise time and settling time values obtained based on PSO provide better large performance rates, compared to the corresponding time values obtained by the ZN, GAs and CSA methods. The respective obtained values based on PSO and those corresponding values obtained based on CSA are not significant. In short, the results of the performance values of the quadcopter inputs obtained based on GA, PSO and CSA are valuable, promising and denote that these meta-heuristic optimization algorithms have acquired desirable control for the movement of a quadcopter. These results confirm the eligibility of the proposed controlling optimization approach to optimize any nonlinear dynamic system, using the meta-heuristic algorithms utilized in this work.

6.3. PID Controller Tuning Parameters

The PID controller values, K_p, K_i and K_d, obtained by GA, PSO and CSA for the inputs of the quadcopter, denoted as roll (ϕ), pitch (θ), yaw (ψ) and thrust (z), along with the fitness values, are presented in Table 3.

Table 3. The fitness and PID controller values for the quadcopter inputs as obtained by GAs, PSO and CSA.

Method			**GAs**	
Parameter	K_p	K_i	K_d	**Fitness cost**
Roll	4.2936	9.999	0.2758	0.04247
Pitch	1.7989	10	0.20463	0.00993
Yaw	49.999	9.4626	0	0.1323
Thrust	50	50	43.0091	0.10289
Method			**PSO**	
Parameter	K_p	K_i	K_d	**Fitness cost**
Roll	4.3088	10	0.3254	0.0257
Pitch	2.0232	9.9998	0.1736	0.0405
Yaw	50	0.7083	0	0.0354
Thrust	50	32.798	37.814	0.1798
Method			**CSA**	
Parameter	K_p	K_i	K_d	**Fitness cost**
Roll	3.476	9.9357	0.2952	0.03737
Pitch	1.8020	9.9996	0.2037	0.00994
Yaw	49.796	49.941	0.0275	0.22835
Thrust	37.072	0.0011	20.8235	0.0450

The fitness cost values obtained by the GA-based approach for the quadcopter inputs in Table 3 are encouraging for all inputs of the quadcopter, which confirm reasonable control for the movement of the quadcopter. Table 3 confirms that PSO reported low fitness costs for all inputs of the quadcopter. Obviously, the fitness cost reported for the input ϕ is lower than the fitness cost reported for the other inputs of the quadcopter. However, the difference is relatively small compared to the inputs, θ and ψ, which are insignificant. Furthermore, PSO reported a fitness cost for the input ϕ that is significantly less than the

fitness cost reported for the input z. PSO also reported a fitness cost for the input θ that is comparable to the fitness cost reported for the input ψ. In a nutshell, the low fitness values obtained by PSO in Table 3 underline the suitability of PSO in controlling the movement of the quadcopter through optimizing the PID controller parameters. The fitness values obtained by CSA are promising, where the values obtained for inputs roll and thrust are smaller than those obtained by GA. However, the fitness values reported by CSA for inputs pitch and yaw are larger than those reported by GA.

6.4. Convergence Results

The convergence curves of GA, PSO and CSA used to accomplish the optimization process presented in this work are presented, using different population sizes to demonstrate the ability of these algorithms to provide accurate control for the quadcopter movements through tuning the PID parameters of the quadcopter. Figures 3–5 show the convergence characteristic curves of the roll control input of the quadcopter obtained by GA, PSO and CSA, respectively, over a different number of population sizes.

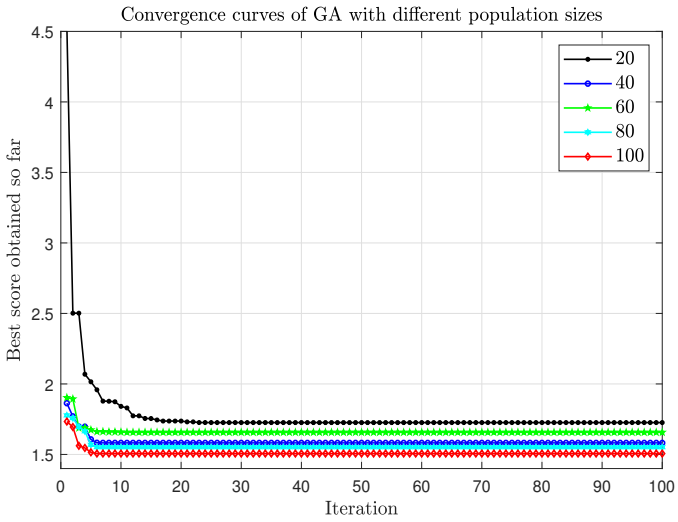

Figure 3. Convergence curves of GA with various population sizes.

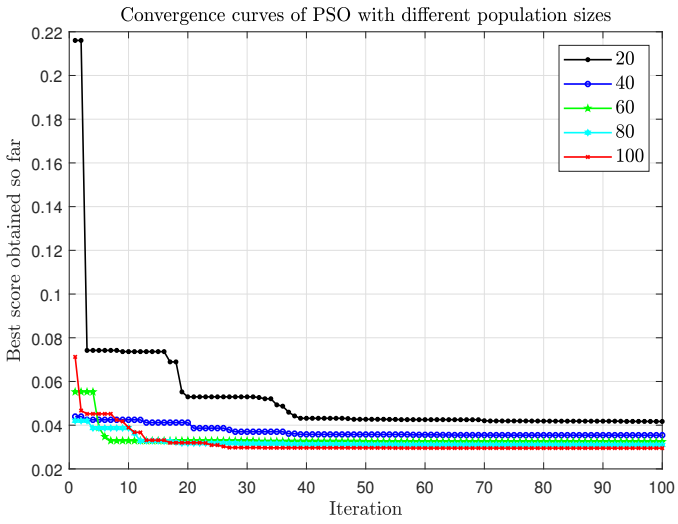

Figure 4. Convergence curves of PSO using various numbers of population sizes.

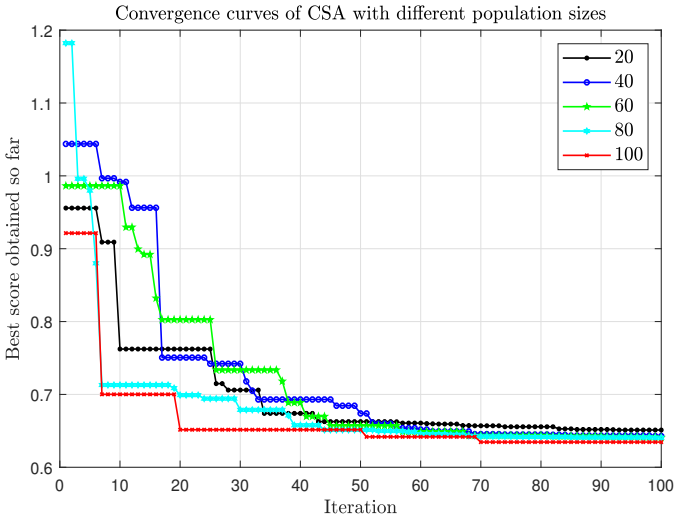

Figure 5. Convergence curves of CSA using various numbers of population sizes.

It is observed from Figure 3 that there is significant convergence in the evolutionary process of the GA-based approach in optimizing the PID controller of the quadcopter. These convergence curves are promising and affirm the effectiveness of GA in controlling the quadcopter's movements. The convergence curves of controlling the quadcopter movement using the PSO-based approach are rendered for up to 100 iterations. These convergence curves show that PSO lays out fast convergence with very low error. It can be seen from Figure 4 that the convergence curve for the population of size 60 converges more rapidly to the optimal cost than that for the other population sizes, although PSO with population sizes of 40, 60, 80 and 100 roughly reach near fitness cost values. The fitness cost value obtained by PSO for the population of size 20 is not optimal and is significantly higher than the fitness cost values reported by PSO for larger population sizes. Thus, any further increase in the population sizes is likely to reach a very small fitness cost value. It is possible that a very large increase in the size of populations will not significantly reduce the fitness cost value further. The fitness cost value for some population sizes continues to decline slightly with further population sizes. The fitness cost value may reduce further with more than 100 particles in the population of PSO. There is a big difference between the curves of population size 20 and the other curves, which indicates that the controlling process implemented by PSO with 100 particles and 80 particles achieves better control than the controlling process carried out by PSO with 20 particles. The performance of CSA in optimizing the roll input for the quadcopter is shown for up to 100 iterations in Figure 5. These curves represent the fitness cost values obtained using the fitness criterion in Equation (31). The convergence curves in Figure 5 demonstrate that the CSA-based approach provides stable control of the quadcopter with fast convergence and very low error.

Figure 6 compares the convergence characteristic curves of the roll control input of the quadcopter obtained by GA, PSO and CSA for a population size of 100.

The convergence curves in Figure 6 confirm that the control method presented based on GA, PSO and CSA reaches a sensible degree of performance in controlling the roll input of the quadcopter for all of these meta-heuristic methods. It is clearly observed that PSO converges to a very small error value, compared to GA and CSA.

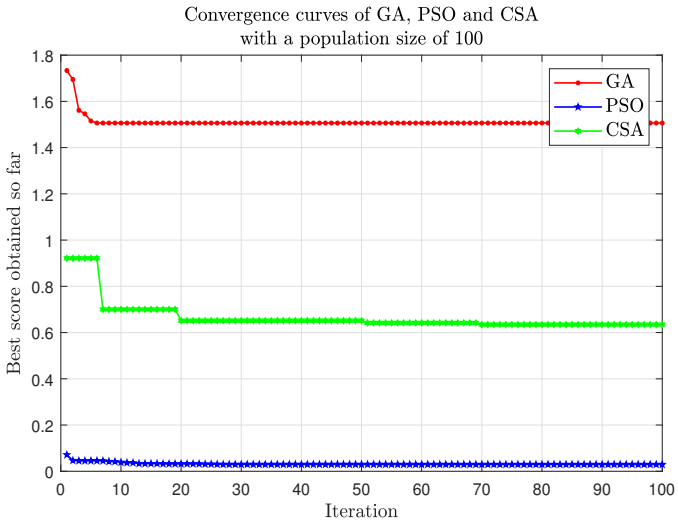

Figure 6. Convergence curves of GA, PSO and CSA with population size 100.

6.5. Simulation Results

The simulated step signal representations of the quadcopter's inputs, roll (ϕ), pitch (θ), yaw (ψ) and thrust (z), using a PID controller along with simulations of the amplitudes of square signals of the inputs' responses obtained by the GA and ZN methods, are illustrated in Figures 7, 8, 9 and 10, respectively.

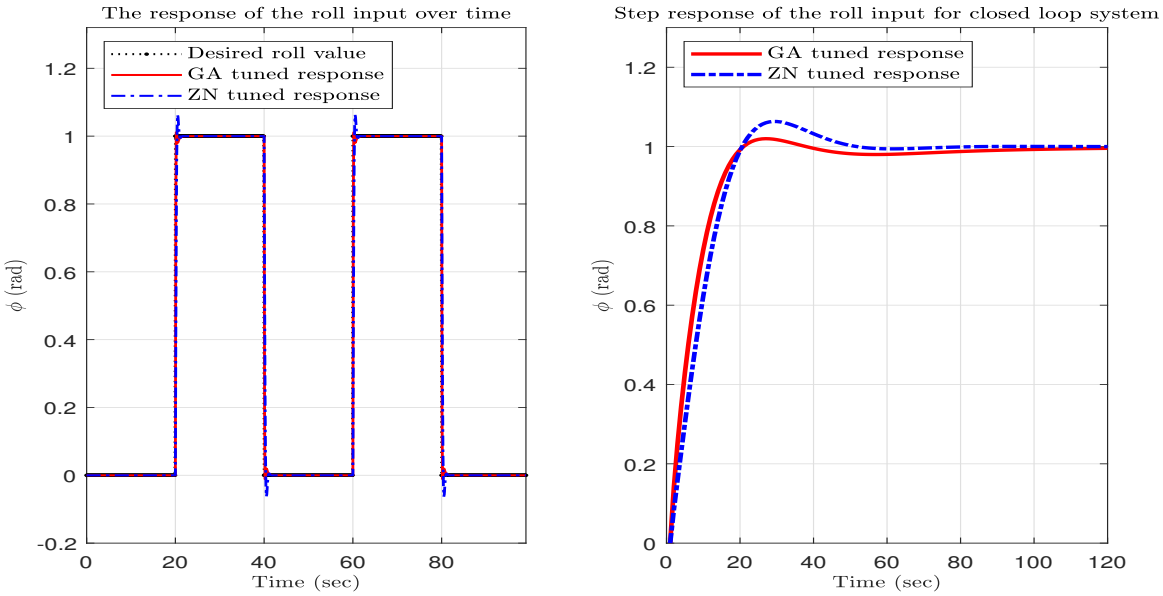

Figure 7. Simulated output responses of the roll input of a quadcopter using GA and ZN methods.

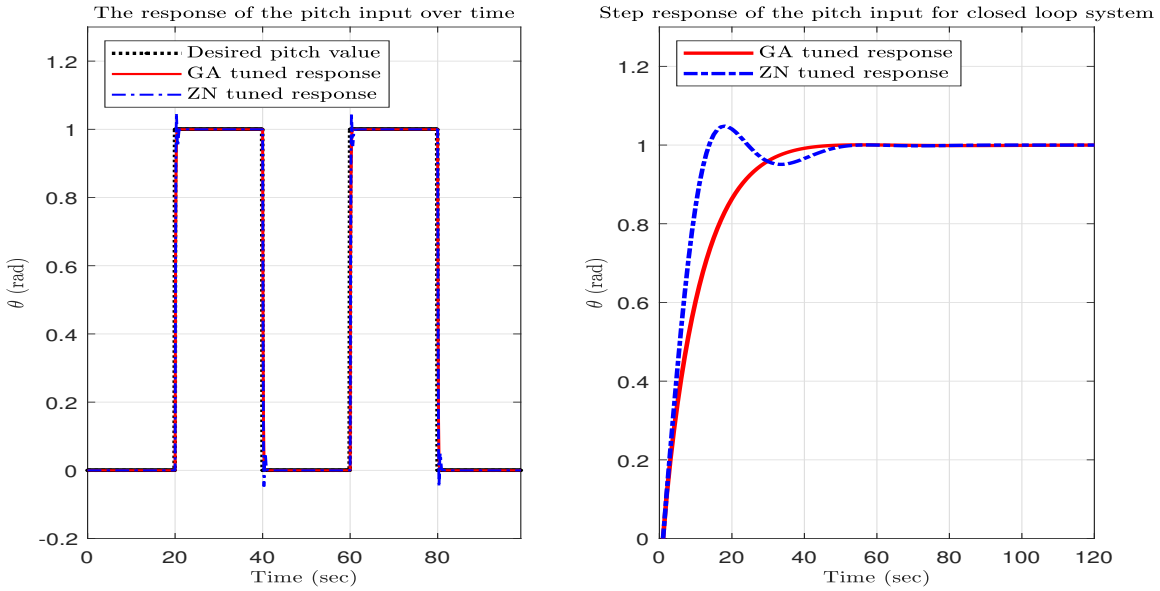

Figure 8. Simulated output response of the pitch input of a quadcopter using GA and ZN methods.

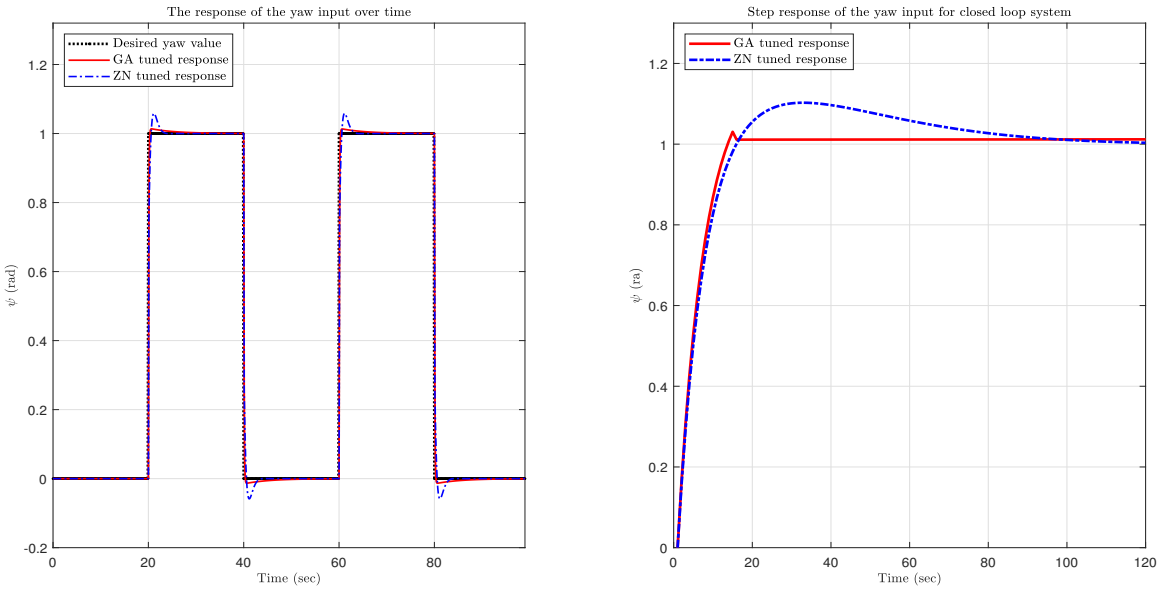

Figure 9. Simulated output responses of the yaw input of the quadcopter using GA and ZN methods.

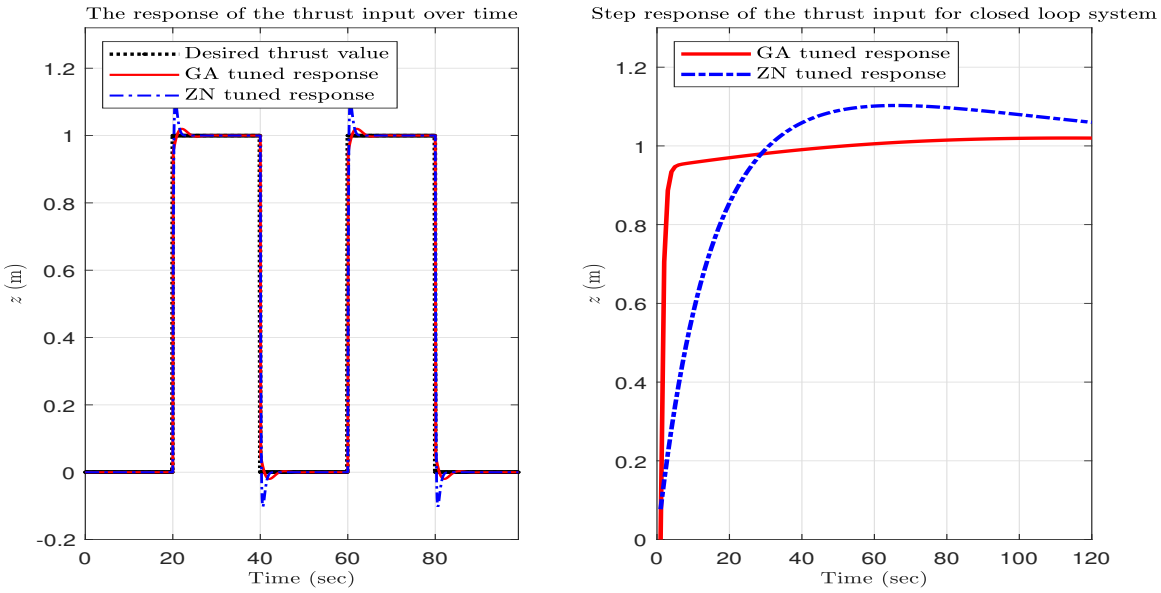

Figure 10. Simulated output responses of the thrust input of a quadcopter using GA and ZN methods.

It is clearly evident from Figures 7–10 that there are significant differences between GA and ZN step response curves, indicating that GA achieves a slight overshoot, a small rise time and a small settling time, compared to those arrived at by the ZN method. By comparing the simulated step responses from the GA- and ZN-based methods, the GA-based approach is faster and has a smaller overshoot for input yaw of the quadcopter. However, the GA-based approach has a longer settling time. Remarkably, the degree of separation between the GA and ZN curves is relatively large for all quadcopter inputs.

The simulated step signals of the inputs—roll, pitch, yaw and thrust—using the PID controller of the quadcopter as well as the simulations of amplitudes of the input responses obtained by the PSO and ZN methods are presented in Figures 11, 12, 13 and 14, respectively.

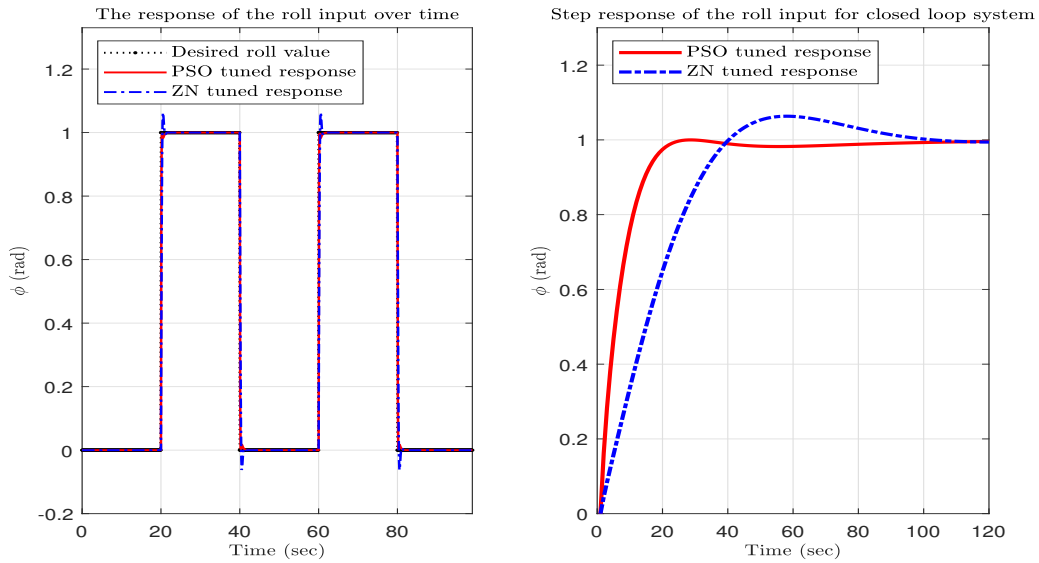

Figure 11. Simulated output responses of the roll input of a quadcopter using PSO and ZN methods.

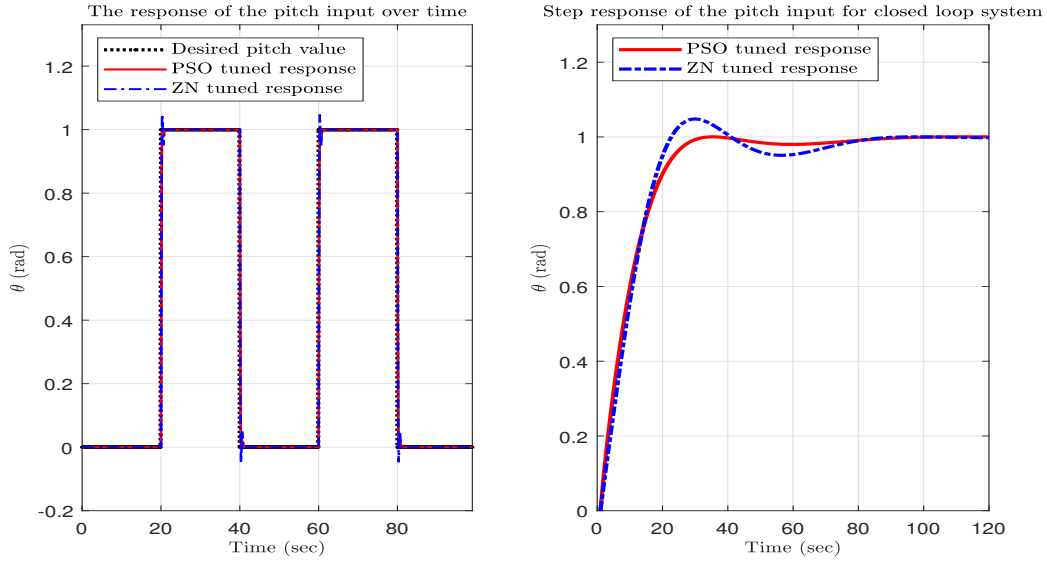

Figure 12. Simulated output responses of the pitch input of a quadcopter using PSO and ZN methods.

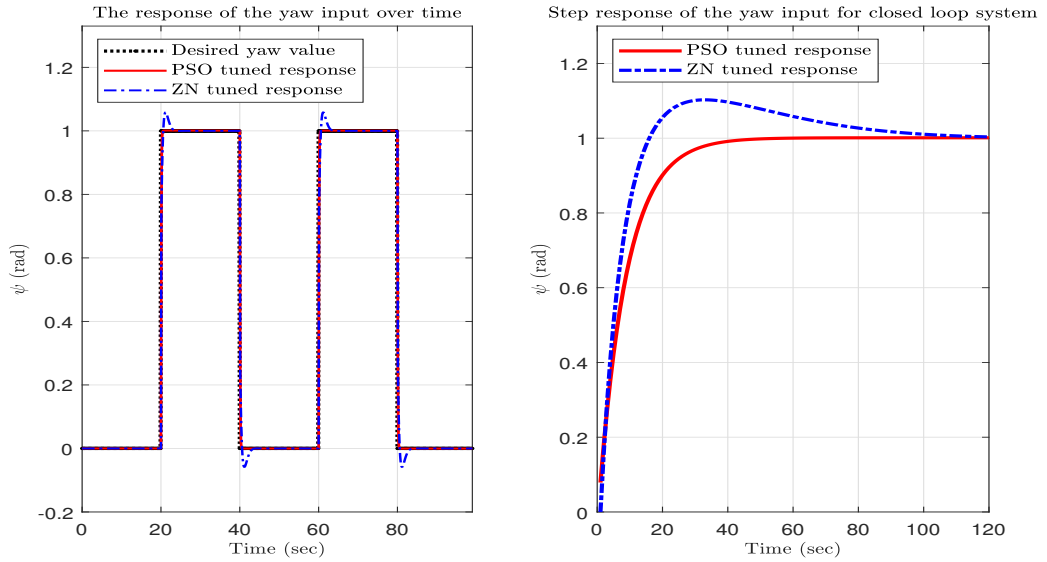

Figure 13. Simulated output responses of the yaw input of a quadcopter using PSO and ZN methods.

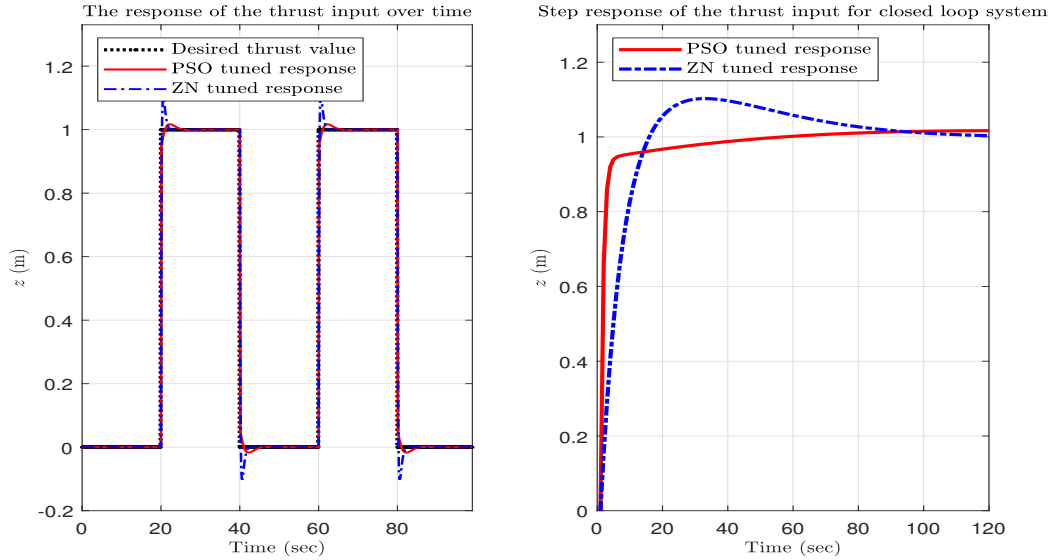

Figure 14. Simulated output responses of the thrust input of a quadcopter using PSO and ZN methods.

Obviously, Figures 11–14 confirm that PSO is substantially more effective than the ZN method in controlling the movement of the quadcopter. This is due to the fact that the signal representations obtained by PSO report small overshoot, modest rise and settling times, compared to those reported by the ZN method.

The simulated step signal representations using the PID controller of the quadcopter inputs—roll, pitch, yaw and thrust—along with the simulations of amplitudes of the input responses obtained by the CSA and ZN methods are displayed in Figures 15, 16, 17 and 18, respectively.

It is clearly observed from Figures 15–18 that there are appreciable differences between CSA and ZN step signal representations, indicating that CSA arrives at slight overshoot, small rise and small settling times compared to those reached by the ZN method. Significantly, there is a relatively large degree of separation between the CSA and ZN curves for all of the quadcopter inputs.

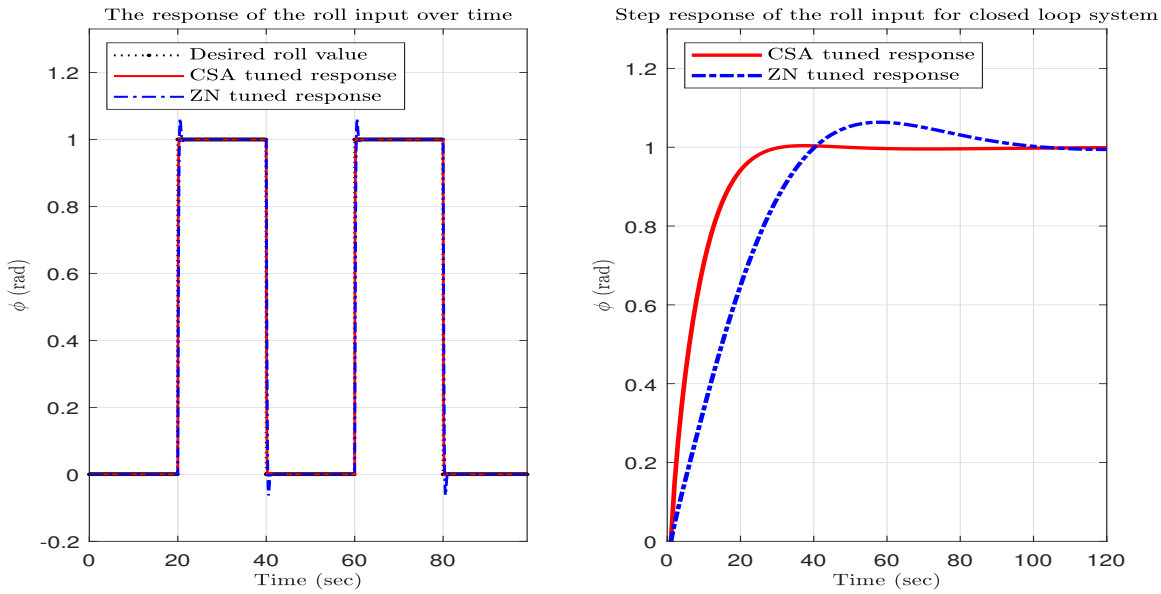

Figure 15. Simulated output responses of the roll input of a quadcopter using CSA and ZN methods.

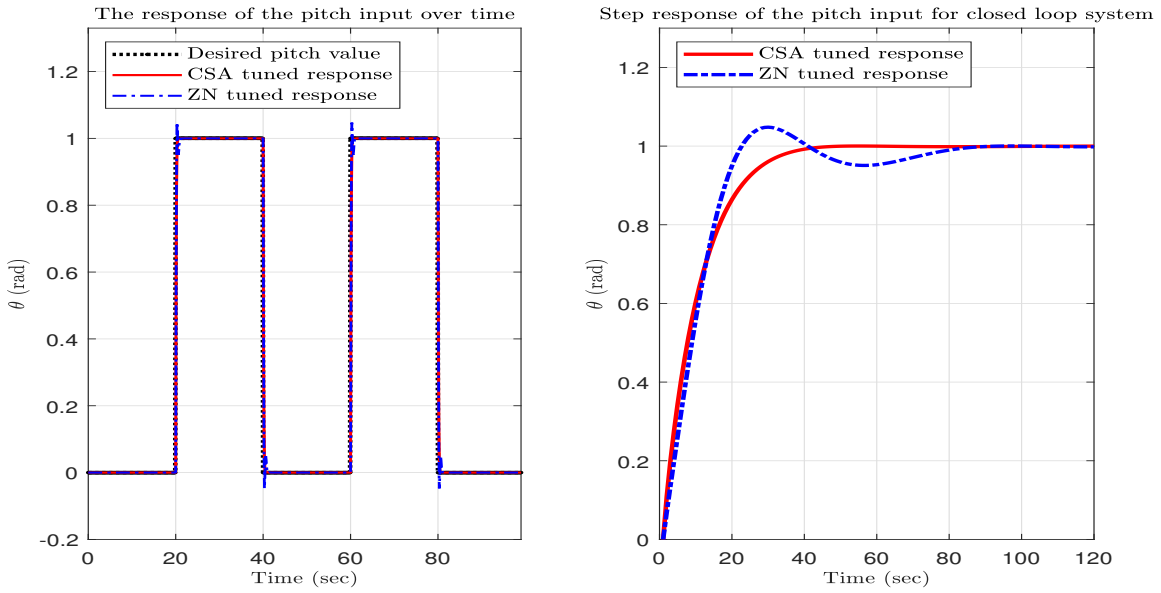

Figure 16. Simulated output responses of the pitch input of a quadcopter using CSA and ZN methods.

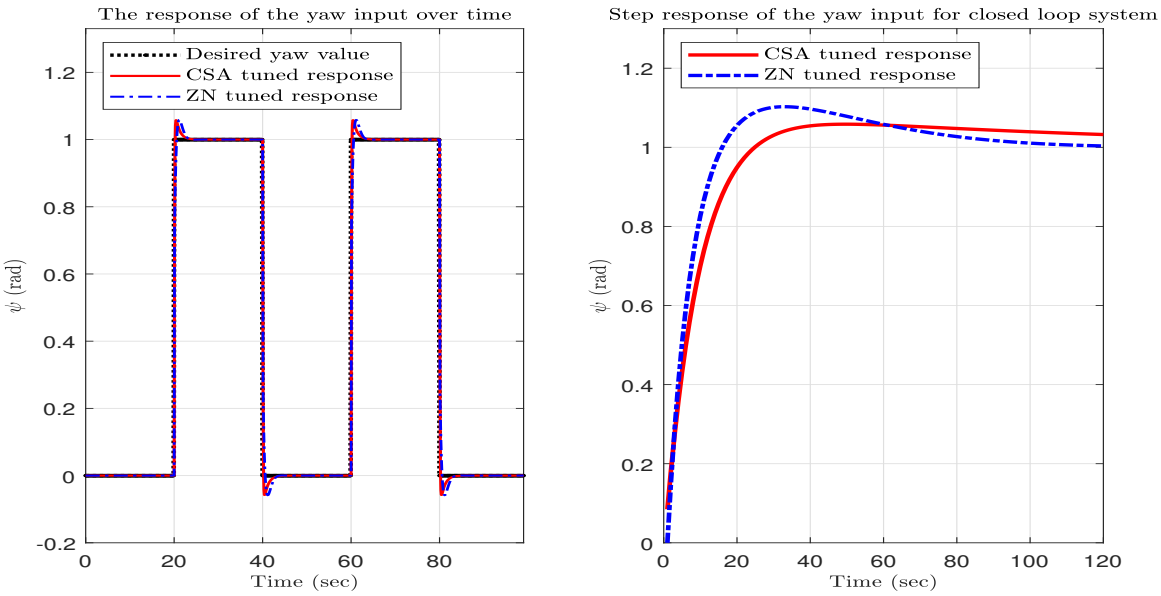

Figure 17. Simulated output responses of the yaw input of a quadcopter using CSA and ZN methods.

The simulated step signals for the roll, pitch, yaw and thrust inputs of the quadcopter shown in Figure 1 based on the use of GA, PSO, CSA and ZN methods by optimizing the PID controller of the quadcopter to stabilize its movements are presented in Figure 19.

The results shown in Figure 19a–d demonstrate the effectiveness of GA, PSO and CSA in reliably controlling the roll, pitch, yaw and thrust inputs of a quadcopter, respectively. As it is clear, the simulated signals for the roll, pitch, yaw and thrust inputs in Figure 19a–d, respectively, obtained based on GA, PSO and CSA are better than those corresponding responses obtained by the ZN method. Interestingly, the responses obtained based on GA, PSO and CSA for roll, pitch, yaw and thrust inputs of the quadcopter are almost similar to each other, with not much difference between them. However, PSO provides the best results, compared to the others.

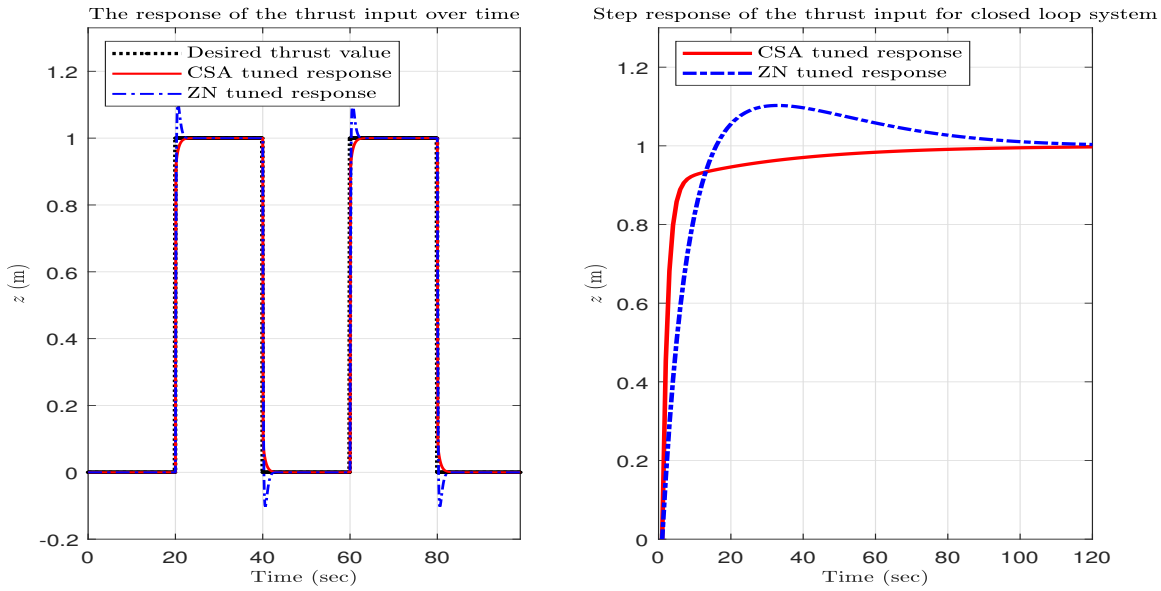

Figure 18. Simulated output responses of the thrust input of a quadcopter using CSA and ZN methods.

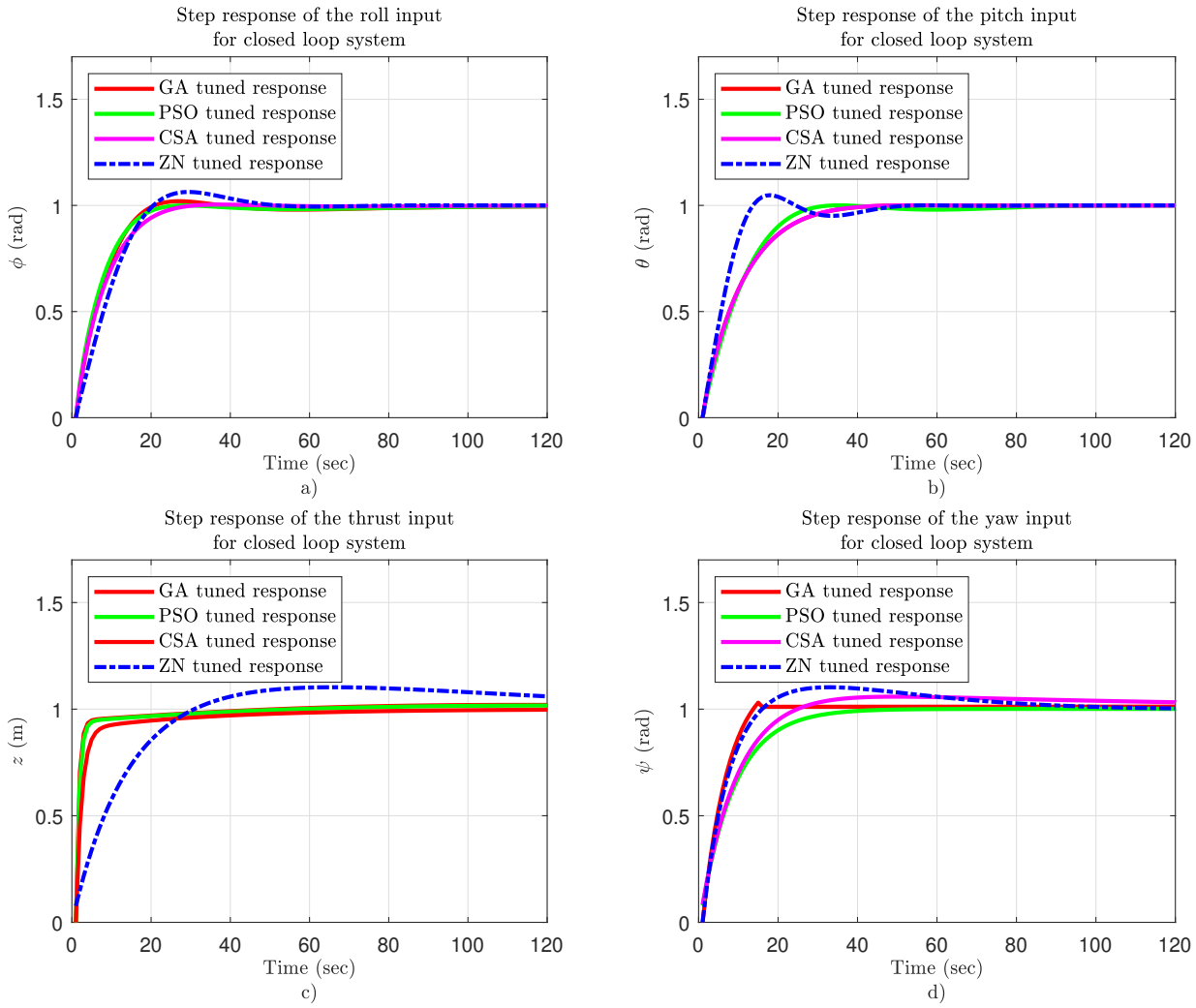

Figure 19. Simulated output responses of the roll, pitch, yaw and thrust inputs of a quadcopter using (**a**) GA, (**b**) PSO, (**c**) CSA and (**d**) ZN methods.

7. Statistical Test Analysis

To locate the importance of the differences between the optimization outcomes arrived at by each method, we utilize the Friedman statistical test. This test is conducted to find out if there are perceivable differences between the overshoot results of the methods presented in this work [47]. If there are statistically notable differences between the optimization results, the Holm method [47] will be conducted as a posterior statistical procedure that is usually used to compare the best method of performance (i.e., the control method) with the rest of the methods. The level of confidence, pointed out as α, is 5% in all evaluated cases.

In more detail, Friedman's test requires the calculation of the average ranked value. Then, a comparison is used to assess the critical values obtained for the level of significance tested ($\alpha = 0.05$) with Friedman's test to check whether or not the null hypothesis is rejected. In such a context, if the level of significance (i.e., the p-value) divulged by Friedman's test is less than or equal to 0.05, the null hypothesis is rejected, denoting that there is a big difference between the performance scores of the algorithms. After that, further steps are needed to determine which algorithm's performance is crucially different from the best algorithm and which algorithms are similar. This is to check whether the outcomes arrived at by the best algorithm differ from the outcomes of the other methods in a statistically significant manner. For this intent, this study looks at a post-hoc statistical analysis method of Holm's test [48] to see which methods are worse or better than the best algorithm and at what level of significance. Holm's test is a widely applicable test method based upon the sequential rejection method. It orders all optimization algorithms on the basis of their p-values and compares them to $\alpha/k - i$, where α is the significance level, k represents the freedom degree and i denotes the algorithm number. The method starts with the most significant p-value and consecutively rejects the null hypothesis as long as $p_i < \alpha/k - i$. Once the method is not able to reject the hypothesis, it stops and stops all the remaining hypotheses passable. In applying Friedman's test, the best-performing algorithm obtains the lowest ranking, while the worst performing algorithm obtains the highest rank. The lowest ranked method is used as a control algorithm for post-hoc analysis.

The average order of the optimization results arrived at by Friedman's method as per the overshoot metric is tabulated in Table 4. The controlling results based the presented methods in this work are ranked as PSO in the first order, followed, in order, by CSA, GA and ZN.

Table 4. Average order of the optimization methods based on Friedman's test using the overshoot metric measure.

Algorithm	Ranking
ZN	4.0
GA	2.75
CSA	1.75
PSO	1.5

The p-value, computed using Friedman's statistical test for the overshoot measure in Table 2, is 0.025557; the null hypothesis of the homologous performance level is unacceptable to assert if there are statistically significant differences between the performance of all evaluated methods.

The Holm's method is then appealed as a post-test procedure to determine if there are statistically substantial differences between the best performing algorithm (that is, PSO) and the remaining algorithms (ZN, GA and CSA). Holm's method dismisses the hypotheses with p-values of less than 0.025. The results obtained based on the Holm's procedure are presented in Table 5.

Table 5. Results of Holm's method based on the overshoot measure.

i	Method	z	p-Value	$\alpha \div i$	Hypothesis
1	CSA	0.27386	0.78419	0.0500	Not rejected
2	GA	1.36930	0.17090	0.0250	Not rejected
3	ZN	2.73861	0.00616	0.0166	Rejected

The findings of Holm's method reveal that the optimization results obtained based on PSO in controlling the movement of the quadcopter are statistically better than the optimization results obtained based on the ZN-, GA- and CSA-based approaches. In the case of the optimization results obtained using the CSA-based approach, there is no significant difference between these results and those obtained using the PSO-based approach, in accordance with the results given in Tables 4 and 5. In summary, these end results imply that controlling the movement of the quadcopter using a CSA-based approach is a feasible alternative approach to the PSO-based approach, which needs further investigation.

8. Discussion of the Results

It is clearly observed from Table 2 that PSO reports overshoot values for the quadcopter inputs that are significantly lesser than the overshoot values reported by the ZN method, shown in Table 2, for the same corresponding quadcopter inputs. Further, GA and CSA reported overshoot values as shown in Table 2, which are lower than those reported by the ZN method for the corresponding inputs. These findings are attributed to the powerful features of GA, CSA and PSO as meta-heuristic optimization methods and the robustness of the presented multi-objective fitness function given in Equation (31). However, the overshoot values reported by PSO for the inputs of the quadcopter in Table 2 are lower than those reported by GA and CSA as shown in Table 2.

The rise and settling times reported by GA, CSA and PSO for all quadcopter inputs are remarkably lower than those reported by the ZN method for the same corresponding inputs. This clarifies and may prove that GA, CSA and PSO are reliable and more effective than the classical ZN method in controlling the movements of the quadcopter, using PID control parameters. Additionally, the differences in the rise and settling times between the control process developed based on PSO, GA and CSA for the four inputs of the quadcopter are respectively less than 0.0045, 0.0209, 0.00925, and 0.431625, on average, which are small and not statistically significant.

Overshoot of the motor signal of the quadcopter is a critical factor that should be slashed. If the temperature of the insulation of the motor of a quadcopter is elevated further away than its rated limits, then the motor will be ruined rapidly [49]. Table 2 states that the overshoot values of all movements of the quadcopter reported by PSO are close to zero in all cases considered; the overshoot values for all movements of the quadcopter reported by GA and CSA are very small in all cases considered. Outstandingly, this outcome is highly desirable for reaching an extremely low temperature for the motors of the quadcopter to protect them from destruction.

The control approaches developed based on GA, CSA and PSO report very low fitness values for all quadcopter inputs, which simply means that the controlling processes developed using these meta-heuristic algorithms for the PDI controller of the quadcopter reached an optimal control for the movements of the quadcopter. Moreover, there is no significant difference between the cost values of these algorithms, whereas the differences in fitness cost values between the optimization developed based on PSO, GA and CSA for the four inputs of the quadcopter are, respectively, less than 0.0015475 and, 0.0209, 0.00925, and 0.009815, on average, which are not statistically significant.

Simply put, the step signals representations of the quadcopter inputs plotted using GA, CSA and PSO demonstrate that these meta-heuristic algorithms are robust tuning algorithms for the quadcopter's PID controller and are better than the conventional tuning

methods, such as the evaluated ZN method. In short, this expected upshot asserts the reliability level of GA, CSA and PSO in optimizing the PID parameters for the quadcopter.

All in all, the following observations can be made from the experimental results of GA, CSA and PSO when used to optimize the PID parameters to control the movements of the quadcopter.

- As per the overshoot evaluation method, GA, CSA and PSO outperform the ZN method through reporting significant reductions in the overshoot values as presented in Table 2.
- In accordance with the rise and settling time values, GA, CSA and PSO report rise and settling time values that are considerably lower than the rise and settling time values reported by the ZN method for all inputs of the quadcopter as presented in Table 2.
- As per the fitness function values for the inputs of the quadcopter, GA, CSA and PSO reach extremely low fitness costs for all inputs, indicating that GA, CSA and PSO are robust and effective in optimizing the PID parameters for the quadcopter flight as presented in Table 3.
- The convergence curves of the evolutionary processes of GA, CSA and PSO confirm the appropriateness of these methods and the robustness of the proposed multi-objective fitness function in controlling the flight of a quadcopter.

For all evaluation criteria, the meta-heuristic algorithms presented in this paper, including GA, CSA and PSO, are able to control the PID controller of the quadcopter to control its movement in all orientations. The superior performance of these algorithms in such an optimization and control problem is attributed to the astonishing features of meta-heuristics and the fitness function, which suggested carrying out such work. The meta-heuristic optimization algorithms is presented in Appendix A.

9. Conclusions and Future Work

This work presented the use of Genetic Algorithms (GAs), Particle Swarm Optimization, and the Crow Search Algorithm (CSA) to adjust the parameters of the Proportional Integral-Derivative (PID) controller to control the movements of a quadcopter. The goal was a reliable scheme to control the movements and direction of a quadcopter. We adopted a multi-objective fitness function for GA, PSO and CSA to optimize the parameters of the PID controller by controlling the quadcopter's flight. The performance of these algorithms was studied in detail with respect to time and accuracy. Convergence curves were presented to demonstrate the ability of this algorithm to provide an accurate controlling process. A fitness function was used to evaluate the performance of the PID controller to control the movements of a quadcopter. Superb results and credible flight orientation of the quadcopter were obtained through the use of GA, PSO and the traditional Ziegler–Nichols (ZN) method. The gain of the developed PID controller was tested through a closed-loop step response system. Further study is needed to assess the suitability of this method to control a quadcopter in a range of directions. It would be beneficial to perform a comparison of alternative meta-heuristic algorithms to control the PID of a quadcopter. Further work is needed to generalize the proposed controlling method based on meta-heuristics to tune the parameters of the PID controller for different types of aerial vehicles.

Author Contributions: Conceptualization, A.S., D.R.M., M.B. and S.A.; Methodology, A.S., M.B., A.M. and D.R.M.; Data curation, A.S., D.R.M., S.A. and H.T.; implementation and experimental work, A.S., M.B., D.R.M. and H.T.; Validation, A.S., M.B., H.T. and D.R.M.; Writing original draft preparation, A.S., M.B., D.R.M., A.M. and H.T.; Writing review and editing, A.S., M.B., A.M. and S.A.; Proofreading, A.S. and A.M.; Supervision, A.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: Sultan Aljahdali acknowledge that this work was supported by Taif University Researchers Supporting Project Number (TURSP-2020/73), Taif University, Taif, Saudi Arabia.

Conflicts of Interest: The authors declare that there is no conflict of interest regarding the publication of this paper.

Appendix A. Meta-Heuristic Optimization Algorithms

Appendix A.1. Genetic Algorithms

Genetic Algorithms are one of the most well-known EAs inspired from Darwin's theory of evolution by John Holland in 1975 [50] and subsequently extended and explored in depth by [51] and Goldberg [52]. The key inspiration features of GAs were inspired based on the evolutionary concepts of natural selection and genetic operators, referred to as crossover and mutation operators. GA does not demand any gradient information and is more likely to converge to the search area for optimum global solutions. The population of GA is initialized by a number of candidate solutions. The iterative procedure of GAs is prolonged through the application of the difference vector based on GAs reproduction operators, in order, including mutation and crossover operators followed by the selection mechanism. Then, each solution within each loop in the evolutionary computational process is evaluated using a pre-determined objective criterion presented to fit to the optimization problem under consideration. In such a context, an objective value is allocated to each individual of the GAs to determine the solution quality of the individuals at each iteration loop of the evolutionary process. In general terms, the problem under study is usually represented by a number of individuals (i.e., chromosomes) that are commensurate with the nature of the problem. Each chromosome acts for a candidate potential solution. The fitness function in GA is always used to judge the quality of the solution of each chromosome in the population. Chromosomes undergo mutation and crossover operators throughout the evolutionary process, eventually leading to the following generation. The crossover operation integrates picked pairs of chromosomes to evolve the following generation that comes up with a new solution. Mutation is always used to prohibit coming down into local solutions. GA can be adapted to a wide range of problems and domains. Hence, it is widely known for its ability to solve complex optimization problems. GA has few mathematical requirements, and these implement an important characteristic of GA when used to address optimization issues. The main steps of GAs can be briefly outlined by a pseudo-code as illustrated in Algorithm A1.

Algorithm A1: Pseudo-code of GAs.

$N \leftarrow$ Population size.
$L \leftarrow$ Number of iterations.
$l \leftarrow$ Current iteration.
$P_c \leftarrow$ Crossover rate.
$P_m \leftarrow$ Mutation rate.
Initialize the population of GA with a size of N.
Evaluate the fitness function of each chromosome *using Equation* (31) given below.
while $l \leq L$ **do**
    Selection: Select Parents from the current population.
Crossover: Perform crossover operation for offsprings with a probability of P_c.
if $(rand(0.0, 1.0) < P_m)$ **then**
        Mutation: Mutate the generated offsprings from the population.
    end
    Evaluation: Evaluate every chromosome using the fitness function.
Updating: Update the population by removing certain limit of lease fit solutions.
    end
    Results: Return the global best solution.

In the proposed work, the evolutionary GA process comprises a number of basic steps to reach the optimal movement and orientation of a quadcopter as given below:

- **Representation**: This step uses a proper chromosome representation so that it can optimize the gain of the PID controller. In this case, the candidate solutions (i.e., individuals) are stored in a three-dimensional parameter vector.
- **Initialization**: This step defines a set of chromosomes in the population and creates random values of genes for those chromosomes.
- **Evaluation**: In this step, the fitness function value is recomputed for each chromosome generated in the initialization phase.
- **Selection**: In this step, the roulette wheel selection mechanism is used to select the suitable chromosomes from the population for mating and the creation of the off-springs produced.
- **Crossover**: A single point crossover mechanism is used to identify one point for each parent and to generate a new child, using a combination of all the data before the allocated crossover point from the first parent with all data after the allocated crossover point of the second parent.
- **Mutation**: This genetic step implements a random tweak in the chromosomes to boost the diversity concept in the population. In other words, it is implemented by replacing the gene at a random position with a new value.

Appendix A.2. Particle Swarm Optimization

Particle swarm optimization is a well-known and familiar type of SI algorithm proposed by Kennedy and Eberhart in 1995 [31,53]. The main inspiration of PSO is to mimic the collective interactive behavior of non-intelligent social swarms, such as birds and fishes, with collaborating agents in nature. It represents a set of solutions (i.e., particles) that move in groups in the search space. The particles of PSO explore possible solutions in the search area and accelerate to the global solutions based on a predetermined criterion. The term "particles" was adopted because particles experience positions, velocities and accelerations of the swarms in accordance with specific equations that update the particles' movement. The particles of the PSO algorithm are updated at each iteration based on the optimal solutions found so far by all particles, as the best solution can be explicitly identified by the best updated particle. The position of each particle is controlled based on the fitness score of each particle as well as the fittest particle of the swarms in the search area. Equation (A1) can be used to update the velocity, v, and the position, x, for each particle for a number of n particles in PSO.

$$\begin{aligned} v_{id}(t+1) &= \omega v_{id}(t) + \alpha r_1[p_{id}(t) - x_{id}(t)] \\ &+ \gamma r_2[p_{gd}(t) - x_{id}(t)] \\ x_{id}(t+1) &= x_{id}(t) + v_{id}(t+1) \end{aligned} \tag{A1}$$

where i identifies the index number of each particle, d indicates the dimension of the problem under consideration, t identifies the iteration number, v_{id} is the velocity of the ith particle in the dth dimension, p_{id} represents the best position of the ith particle visited so far in the dth dimension, p_{gd} represents the global best position of a particle in the swarm in the dth dimension, ω is a positive inertia weight, r_1 and r_2 are two randomly generated numbers between 0 and 1, α and β are two constants that represent the values of the degree of influence of p_{id} and p_{gd} on the particles velocity, respectively, and x_{id} is the position of the ith particle in the dth dimension.

Equation (A1) can be used to find a new solution in the search space and to update the global best solution. The key steps of PSO can be summarized as shown by the pseudo-code in Algorithm A2.

The PSO algorithm embraces exploration search properties, making it well suited for optimizing nonlinear functions and complex real-world problems. It is expected that through the use of PSO, there is a high probability of evading minimal local solutions over and over by encouraging more exploration of the search domain. This issue is concerned with the fact that survival is the ultimate goal for fish schooling and bird

flocking. Consequently, PSO exhibits a coordinated behavior that distinguishes the social behavior of bird and fish swarms in nature.

There are many applications presented in the literature based on PSO that have reported convincing results in a wide range of optimization issues [28,54]. Today, there is rapid development in the use of meta-heuristic algorithms, and especially PSO, to address real contemporary problems, such as the problem under consideration. The particle representation for the problem under study is represented in Table A1.

Table A1. PSO particle representation.

k_p	k_i	K_d

Algorithm A2: Pseudo-code of PSO.

```
N ← Population size.
L ← Number of iterations.
l ← Current iteration.
pbest ← Personal best solution.
gbest ← Global best solution.
p ← Best particle.
Initialize the positions of all N particles.
Evaluate the fitness function of each particle using Equation (31)
while While l ≤ L do
    foreach particle in p do
      Update pbest:
  if (f(p) < f(pbest)) then
        pbest = p.
    end
    Update gbest
if (f(pbest) < f(gbest))then
        gbest=pbest
    end
  end
  foreach particle in p do
     Update velocity and position: using Equation (A1)
  end
  end
  Results: return gbest.
```

Appendix A.3. Crow Search Algorithm

The Crow Search Algorithm is a meta-heuristic algorithm inspired from the collective behavior of crows in nature [29]. Specifically, abstract ideas of living social behavior and crows feeding aspects led to the evolution of this algorithm with a mathematical model, described in detail in [29]. Many studies have reported in the literature that crows exhibit intelligent behavior and capabilities of intelligently hiding food in a specific location and the capabilities of finding it again. In such a context, crows communicate in a complex way, as they have a good memory of where they store their food. As such, CSA is a meta-heuristic population-based search algorithm, and it is implemented in accordance with the following rules:

- Crows inhabit in nature in flocks.
- Crows can recall the places in which they store their food and fetch it once more.
- Crows can see other animals when they pillage their stored food.
- Crows have the capacity to manage themselves in ratios to safeguard their unattended food.

To solve a particular optimization problem using CSA, the solution is represented as the position of the crows at any instant of time as given in Equation (A2).

$$x^{j,g} = \left[x_1^{j,g}, x_2^{j,g}, \ldots, x_D^{j,g}\right] \tag{A2}$$

where $x^{j,g}$ stands for the position of the jth crow at generation g and D identifies the dimension of the problem under study.

Similar to any population-based algorithm, CSA begins by randomly generating a pool of possible solutions, representing a flock of crows. As mentioned earlier, crows make use of their memory to recover hidden food placements, which implicitly typifies the quality of the crows' positions. The quality of each solution is judged by the fitness value computed from a fitness evaluation criterion used to solve the given problem. The quality of each solution is collected in a predefined array as stated in Equation (A3).

$$Memory = \begin{bmatrix} m_1^1 & m_2^1 & \ldots & m_d^1 \\ m_1^2 & m_2^2 & \ldots & m_d^2 \\ \vdots & \vdots & \vdots & \vdots \\ m_1^N & m_2^N & \ldots & m_d^N \end{bmatrix} \tag{A3}$$

The crows in flocks update their positions as per the following procedure: each crow j realizes another crow k from the flock to pursue it with the aim to locate the hidden food by the last crow, and is referred to as m^k. This mechanism is illustrated in Equation (A4).

$$x^{j,k+1} = \begin{cases} x^{j,g} + r_j \times fl^{j,g} \times \left(m^{k,g} - x^{j,g}\right) & r_k \geq AP^{k,g} \\ a\ random\ position & Otherwise \end{cases} \tag{A4}$$

where r_k indicates a random number generated from a uniform distribution in the interval from 0 to 1, and $AP^{k,g}$ stands for the awareness probability of crow kth at generation g.

The solution quality is valued for each new crow position, using the fitness score obtained in each iteration. The crow's position is updated if its solution is better than the current crow's solution. However, the crow remains in place if the quality of its solution is lesser than that of the current crow. The crows' location memory can be updated, using Equation (A5).

$$m^{j,g+1} = \begin{cases} x^{j,g+1} & f\left(x^{j,g+1}\right)\ is\ better\ than\ f\left(m^{j,g}\right) \\ m^{j,g} & Otherwise \end{cases} \tag{A5}$$

where f denotes the score value of the fitness criterion.

Likewise, each crow updates its memory if its solution quality in the new position is higher than the memorized position. CSA is implemented in an iterative procedure: the new positions and memories of the crows are created, assessed, and updated at each iteration loop until convergence, in which a defined evaluation method fit to the problem under study is met.

In short, the iterative steps of CSA can be described briefly by an pseudo-code as outlined in Algorithm A3.

While the usefulness of CSA in optimization problems is incontrovertible, it has not been used to optimize a control problem. Therefore, in light of the stimulating results of the CSA in many fields, we explore, in this work, its significance in a control problem for optimizing the parameters of the PID controller of a quadcopter to control its movement.

Algorithm A3: Pseudo-code stating the basic steps of the Crow Search Algorithm.

$N \leftarrow$ Number of crows in the search space.
$AP \leftarrow$ Awareness probability.
$fl \leftarrow$ Flight length.
$k \leftarrow$ Iteration counter.
$K \leftarrow$ maximum number of iterations.
Set the initial values of N, AP, fl and K
Initialize all crows positions, x_i, randomly in the search space, where $i = 1, 2, \ldots, N$.
Evaluate the fitness value for each crow using the fitness function $f(x)$
Initialize the memory, m, of all crows
Set $k \leftarrow 1$ **while** $(k \leq K)$ **do**
    for $i = 1, 2, \ldots, N$ **do**
    Randomly pick one of crows, such as crow z, to pursue **if** $(r_z \geq AP^{z,l})$ **then**
        $x_i^{k+1} = x_i^k + r_i \times fl_i^k \times \left(m_z^k - x_i^k\right)$
    end
    $x_i^{k+1} =$ Random location in the search space
    end
Check the feasibility of each position for each crow (x_i^{k+1})
Evaluate each new position for each crow $f(x_i^{k+1})$
Update the memory for each crow in the population m_i^{k+1}
$k = k + 1$
    end
Return best solution of all crows x

References

1. Talha, M.; Asghar, F.; Rohan, A.; Rabah, M.; Kim, S.H. Fuzzy logic-based robust and autonomous safe landing for UAV quadcopter. *Arab. J. Sci. Eng.* **2019**, *44*, 2627–2639. [CrossRef]
2. Saha, H.; Basu, S.; Auddy, S.; Dey, R.; Nandy, A.; Pal, D.; Roy, N.; Jasu, S.; Saha, A.; Chattopadhyay, S.; et al. A low cost fully autonomous GPS (Global Positioning System) based quad copter for disaster management. In Proceedings of the 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 8–10 January 2018; pp. 654–660.
3. Navia, J.; Mondragon, I.; Patino, D.; Colorado, J. Multispectral mapping in agriculture: Terrain mosaic using an autonomous quadcopter UAV. In Proceedings of the 2016 International Conference on Unmanned Aircraft Systems (ICUAS), Arlington, VA, USA, 7–10 June 2016; pp. 1351–1358.
4. Hunt, G.; Mitzalis, F.; Alhinai, T.; Hooper, P.A.; Kovac, M. 3D printing with flying robots. In Proceedings of the 2014 IEEE international conference on robotics and automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 4493–4499.
5. Hu, C.; Zhang, Z.; Yang, N.; Shin, H.S.; Tsourdos, A. Fuzzy multiobjective cooperative surveillance of multiple UAVs based on distributed predictive control for unknown ground moving target in urban environment. *Aerosp. Sci. Technol.* **2019**, *84*, 329–338. [CrossRef]
6. Altan, A.; Hacıoğlu, R. Model predictive control of three-axis gimbal system mounted on UAV for real-time target tracking under external disturbances. *Mech. Syst. Signal Process.* **2020**, *138*, 106548. [CrossRef]
7. Paiva, E.; Soto, J.; Salinas, J.; Ipanaqué, W. Modeling, simulation and implementation of a modified PID controller for stabilizing a quadcopter. In Proceedings of the 2016 IEEE International Conference on Automatica (ICA-ACCA), Curico, Chile, 19–21 October 2016; pp. 1–6.
8. Fan, Y.; Shao, J.; Sun, G.; Shao, X. Improved beetle antennae search algorithm-based Lévy flight for tuning of PID controller in force control system. *Math. Probl. Eng.* **2020**, *2020*, 4287315. [CrossRef]
9. de Jesus Rubio, J.; Cruz, P.; Paramo, L.A.; Meda, J.A.; Mujica, D.; Ortigoza, R.S. PID anti-vibration control of a robotic arm. *IEEE Lat. Am. Trans.* **2016**, *14*, 3144–3150. [CrossRef]
10. Kumar, J.; Kumar, V.; Rana, K. Design of robust fractional order fuzzy sliding mode PID controller for two link robotic manipulator system. *J. Intell. Fuzzy Syst.* **2018**, *35*, 5301–5315. [CrossRef]
11. Maleki, K.N.; Ashenayi, K.; Hook, L.R.; Fuller, J.G.; Hutchins, N. A reliable system design for nondeterministic adaptive controllers in small UAV autopilots. In Proceedings of the 2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC), Sacramento, CA, USA, 25–29 September 2016; pp. 1–5.
12. Meshram, P.; Kanojiya, R.G. Tuning of PID controller using Ziegler-Nichols method for speed control of DC motor. In Proceedings of the IEEE-International Conference on Advances in Engineering, Science and Management (ICAESM-2012), Nagapattinam, India, 30–31 March 2012; pp. 117–122.

13. Santoso, F.; Garratt, M.A.; Anavatti, S.G. State-of-the-art intelligent flight control systems in unmanned aerial vehicles. *IEEE Trans. Autom. Sci. Eng.* **2017**, *15*, 613–627. [CrossRef]
14. Miglino, O.; Lund, H.H.; Nolfi, S. Evolving mobile robots in simulated and real environments. *Artif. Life* **1995**, *2*, 417–434. [CrossRef]
15. Dierks, T.; Jagannathan, S. Output feedback control of a quadrotor UAV using neural networks. *IEEE Trans. Neural Netw.* **2009**, *21*, 50–66. [CrossRef]
16. Shepherd, J.F., III; Tumer, K. Robust neuro-control for a micro quadrotor. In Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, Portland, OR, USA, 7–11 July 2010; pp. 1131–1138.
17. Braik, M. A Hybrid Multi-gene Genetic Programming with Capuchin Search Algorithm for Modeling a Nonlinear Challenge Problem: Modeling Industrial Winding Process, Case Study. *Neural Process. Lett.* **2021**, *36*, 1–44.
18. Braik, M.S. Chameleon Swarm Algorithm: A bio-inspired optimizer for solving engineering design problems. *Expert Syst. Appl.* **2021**, *174*, 114685. [CrossRef]
19. Braik, M.; Sheta, A.; Al-Hiary, H. A novel meta-heuristic search algorithm for solving optimization problems: Capuchin search algorithm. *Neural Comput. Appl.* **2021**, *33*, 2515–2547. [CrossRef]
20. Alkamachi, A.; Erçelebi, E. Modelling and genetic algorithm based-PID control of H-shaped racing quadcopter. *Arab. J. Sci. Eng.* **2017**, *42*, 2777–2786. [CrossRef]
21. Yeom, K. Intelligent controller modelling for steerable robotic bar using bio-inspired control synthesis. *Microsyst. Technol.* **2019**, *25*, 1493–1504. [CrossRef]
22. Wang, M.; Feng, S.; He, C.; Li, Z.; Xue, Y. An artificial immune system algorithm with social learning and its application in industrial PID controller design. *Math. Probl. Eng.* **2017**, *2017*, 3959474. [CrossRef]
23. Ekinci, S.; Hekimoğlu, B. Improved kidney-inspired algorithm approach for tuning of PID controller in AVR system. *IEEE Access* **2019**, *7*, 39935–39947. [CrossRef]
24. Pršić, D.; Nedić, N.; Stojanović, V. A nature inspired optimal control of pneumatic-driven parallel robot platform. *Proc. Inst. Mech. Eng. Part C J. Mech. Eng. Sci.* **2017**, *231*, 59–71. [CrossRef]
25. Wu, Q.; Shen, X.; Jin, Y.; Chen, Z.; Li, S.; Khan, A.H.; Chen, D. Intelligent beetle antennae search for UAV sensing and avoidance of obstacles. *Sensors* **2019**, *19*, 1758. [CrossRef]
26. Wu, Q.; Lin, H.; Jin, Y.; Chen, Z.; Li, S.; Chen, D. A new fallback beetle antennae search algorithm for path planning of mobile robots with collision-free capability. *Soft Comput.* **2020**, *24*, 2369–2380. [CrossRef]
27. Fan, Y.; Shao, J.; Sun, G. Optimized PID controller based on beetle antennae search algorithm for electro-hydraulic position servo control system. *Sensors* **2019**, *19*, 2727. [CrossRef]
28. Sheta, A.; Braik, M.S.; Aljahdali, S. Genetic algorithms: A tool for image segmentation. In Proceedings of the 2012 International Conference on Multimedia Computing and Systems, Tangiers, Morocco, 10–12 May 2012; pp. 84–90.
29. Askarzadeh, A. A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. *Comput. Struct.* **2016**, *169*, 1–12. [CrossRef]
30. Sheta, A.; Faris, H.; Braik, M.; Mirjalili, S. Nature-inspired metaheuristics search algorithms for solving the economic load dispatch problem of power system: A comparison study. In *Applied Nature-Inspired Computing: Algorithms and Case Studies*; Nilanjan, D.; Amira, S.; Ashour, S.B., Ed.; Springer: Singapore, 2020; pp. 199–230.
31. Kennedy, J.; Eberhart, R.C. Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
32. Braik, M.; Sheta, A. *Exploration of Genetic Algorithms and Particle Swarm Optimization in Improving the Quality of Medical Images*; Lambert Academic Publishing (LAP): Saarbrücken, Germany, 2011; pp. 329–360.
33. Sheta, A.; Braik, M.; Al-Hiary, H. Modeling the Tennessee Eastman chemical process reactor using bio-inspired feedforward neural network (BI-FF-NN). *Int. J. Adv. Manuf. Technol.* **2019**, *103*, 1359–1380. [CrossRef]
34. Maddi, D.; Sheta, A.; Davineni, D.; Al-Hiary, H. Optimization of PID Controller Gain Using Evolutionary Algorithm and Swarm Intelligence. In Proceedings of the 2019 10th International Conference on Information and Communication Systems (ICICS), Irbid, Jordan, 11–13 June 2019; pp. 199–204.
35. Braik, M.; Al-Zoubi, H.; Al-Hiary, H. Artificial neural networks training via bio-inspired optimisation algorithms: Modelling industrial winding process, case study. *Soft Comput.* **2021**, *25*, 4545–4569. [CrossRef]
36. Jiřinec, T. Stabilization and Control of Unmanned Quadcopter. Master's Thesis, LuleåUniversity of Technology, Luleå, Sweden, 30 May 2011. Validerat; 20111004 (anonymous).
37. Gibiansky, A. Quadcopter Dynamics and Simulation. 23 November 2012. Available online: https://andrew.gibiansky.com/blog/physics/quadcopter-dynamics/ (accesed on 1 July 2021).
38. Henriques, B.S.M. *Estimation and Control of a Quadrotor Attitude*; Master of Science; Instituto Superior Técnico, Lisbon University: Lisbon, Portugal, 2011.
39. Thu, K.M.; Gavrilov, A. Designing and Modeling of Quadcopter Control System Using L1 Adaptive Control. *Procedia Comput. Sci.* **2017**, *103*, 528–535. [CrossRef]
40. HadiAbbas, N.; Sami, A.R. Tuning of PID Controllers for Quadcopter System using Hybrid Memory based Gravitational Search Algorithm–Particle Swarm Optimization. *Int. J. Comput. Appl.* **2017**, *172*, 9–18.

41. Ang, K.H.; Chong, G.; Li, Y. PID control system analysis, design, and technology. *IEEE Trans. Control Syst. Technol.* **2005**, *13*, 559–576.
42. Li, J.; Li, Y. Dynamic analysis and PID control for a quadrotor. In Proceedings of the 2011 IEEE International Conference on Mechatronics and Automation, Beijing, China, 7–10 August 2011; pp. 573–578.
43. Yang, X.S. *Nature-Inspired Metaheuristic Algorithms*; Luniver Press: Frome, UK, 2010.
44. Sheta, A.; Faris, H.; Aljarah, I. Estimating ARMA Model Parameters of an Industrial Process Using Meta-Heuristic Search Algorithms. *Int. J. Eng. Technol.* **2018**, *7*, 187–194. [CrossRef]
45. Krell, E.; Sheta, A.; Balasubramanian, A.P.R.; King, S.A. Collision-Free Autonomous Robot Navigation in Unknown Environments Utilizing PSO for Path Planning. *J. Artif. Intell. Soft Comput. Res.* **2019**, *9*, 267–282. [CrossRef]
46. Patel, A.; Juneja, P.K.; Chaturvedi, M.; Rawat, S. Integral Error Based Controller Design & Performance Analysis for a TOPDT Process. In Proceedings of the 2016 International Conference on Advances in Computing, Communication, & Automation (ICACCA) (Spring), Dehradun, India 8–9 April 2016.
47. Ross, S. *Introduction to Probability and Statistics for Engineers and Scientists, Student Solutions Manual*, 4th ed.; Academic Press: Cambridge, MA, USA, 2009.
48. Holm, S. A simple sequentially rejective multiple test procedure. *Scand. J. Stat.* **1979**, *6*, 65–70.
49. Bloch, H.P.; Geitner, F.K. *Practical Machinery Management for Process Plants*; Gulf Publishing Company: Houston, TX, USA, 1988; Volume 3.
50. Holland, J. *Adaptation in Natural and Artificial Systems*; University of Michigan Press: Ann Arbor, MI, USA, 1975.
51. De Jong, K. Analysis of Behavior of a Class of Genetic Adaptive Systems. Ph.D. Thesis, University of Michigan, Ann Arbor, MI, USA, 1975.
52. Goldberg, D. *Genetic Algorithms in Search, Optimization, and Machine Learning*; Addison-Wesley: New York, NY, USA, 1989; p. 989.
53. Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization. *Swarm Intell.* **2007**, *1*, 33–57. [CrossRef]
54. Braik, M.; Sheta, A.; Ayesh, A. Particle swarm optimisation enhancement approach for improving image quality. *Int. J. Innov. Comput. Appl.* **2007**, *1*, 138. [CrossRef]