SATELLITE TRACKING CONTROL SYSTEM USING DIFFERENT OPTIMAL CONTROLLERS BASED ON EVOLUTIONARY OPTIMIZATION TECHNIQUES

1 Mohamed El-Sayed M. Sakr and 2Mohamed A. Moustafa Hassan

1,2Department of Electrical Power, Faculty of Engineering, Cairo University, Giza, Egypt

Email: Mohamed.201920007@eng-st.cu.edu.eg

ABSTRACT

Satellite tracking control system is a control unit which automatically steers the parabolic antenna to the desired satellite. It precisely traces the satellite as it spins in its orbit across the sky. In order to maintain continuous communication signal during multiple satellite tracking missions, the tracking process must be very fast and smooth with minimum deviations from the desired position. Over time, various controller models have been proposed to tackle the problem of antenna pointing in satellite as well as to track movable targets utilizing servomechanism. This paper aims to present and discuss several control strategies applied in this work. Also, introduces various control models that have been applied in closed loop control systems to position the satellite dishes [1]. Over time, various controller models have been proposed to tackle the problem of antenna pointing in satellite as well as to track moveable targets utilizing servomechanism system [3, 4, 5] and [6]. The aim of the proposed controller is to make the system meets the desired requirements regarding overshoot, rise time, settling time and steady-state error while maintaining the system's high stability and robustness, also, at the same time, providing the system with ability to reject any disturbance and noise [7]. Optimal controllers based on evolutionary techniques are proposed in order to design and perform a Two Degree of Freedom (2DOF) control system to stabilize the satellite tracking control system's azimuth and elevation angles.

1. Introduction

This section provides a background information on satellite control system and illustrates the problem of satellite tracking. Also, introduces various control models that have been developed over time to tackle this problem, depicts the proposed control strategies applied in this work.

1.1. Background Information

As a result of advances in satellite technology, satellites have many incredible uses in the modern world in the following areas: Meteorology, Weather Forecasting, Communications, Radio and TV Broadcast, Navigation, Military and Space Exploration [1]. Receiving and transmitting systems are installed on a fixed station or mobile station such as a ship, train, car or aircraft. For multiple missions satellite ground station, in order to maintain continuous communication signals, antenna system must be oriented in both the azimuth and elevation angles to track the satellite. An earth station's tracking system, who's presented in Fig. 1, is required to carry out some of the tasks include satellite acquisition, manual tracking and automatic tracking. For automatic tracking there are two sorts of satellite searching techniques, they are mechanical and electrical searching techniques. The mechanical technique controls both the elevation and azimuth angles by pressing the elevation up and down keys or the azimuth right and left keys to operate a motor system. The searching mechanism for the electrical technique is done automatically by steering the antenna according to the elevation and azimuth angle computed by the programmed software [2]. For many years DC servo motor-based controllers have been applied in closed loop control systems to position the satellite dishes [1]. Over time, various controller models have been proposed to tackle the problem of antenna pointing in satellite as well as to track moveable targets utilizing servomechanism system [3, 4, 5] and [6]. The aim of the proposed controller is to make the system meets the desired requirements regarding overshoot, rise time, settling time and steady-state error while maintaining the system's high stability and robustness, also, at the same time, providing the system with ability to reject any disturbance and noise [7]. Optimal controllers based on evolutionary techniques are proposed in order to design and perform a Two Degree of Freedom (2DOF) control system to stabilize the satellite tracking control system's azimuth and elevation angles.

1.2. Problem Statement

Earth stations, which are mainly utilized in satellite tracking applications, are susceptible to environmental disturbances [8]. In order to maintain continuous communication signals, antenna system must be oriented in both the azimuth and elevation angles to trace the desired satellite [2]. The most typical issue when aligning a dish is pointing at the right satellite or target for the broadcasts required. The issue may be solved by creating a specific control system that utilizes the received satellite signals to control DC servo motor system, which will direct the antenna to the appropriate location. In addition to tracking problem, DC servo motor system is sensitive to disturbances and errors produced by nonlinear fluctuations in load conditions, motor saturation, backlash, friction and wind pressures and gusts [3]. Several controllers models have been developed over time to solve the problem of antenna pointing in satellite as well as to track moveable targets utilizing servomechanism [2], [3, 4, 5, 6, 8, 9, 10] and [11]. These controllers include conventional controllers such as Proportional-Integral (PI), Proportional-Derivative (PD), Proportional-Integral-Derivative (PID), Fractional Order Proportional-Integral-Derivative (FOPID), Fuzzy PID (FPID), Variable Coefficient PID (V-PID), Fractional Order PID (FOPID), Variable Coefficient Fractional Order PID (V-FOPID), and Fractional Order PID (FOPID). These controllers are proposed for satellite tracking control system. The system and control strategies designed and simulated in MATLAB & SIMULINK. The response of the system is analyzed, and the results of different control strategies are measured and compared with others. The obtained results implies that the proposed controller is able to track the desired position precisely with the fastest settling time and free overshoot compared to other control strategies.
controller parameters. In various applications, several systematic approaches for optimal tuning of the proposed controller are established to enhance the transient state response of the system. The controllers, such as PI and PID, are sensitive to changes in motor characteristics and load. It's also difficult to tune PI or PID gains to prevent overshoot caused by load disturbance. However, the PID response is still regarded as poor due to significant overshoot and a long settling time value [10]. LQG controllers are not only optimum, but they can also estimate non-measurable states by utilizing observers to rebuild them and perform better in the presence of wind gust noise [9]. A state-feedback controller with pole placement assignment is used to achieve very minor overshoot, a small settling time value, and zero steady-state error in order to produce a good transient response specification [3]. If the system parameters perfectly known (certainties) LQR is effective to achieve the desired performance by choosing the control law. Also, a hybrid PID-LQR controller is proposed by combining PID with a Linear Quadratic Regulator, in order to optimize system response and increases the precision of system. The hybrid PID-LQR controller's performance outperforms other controllers significantly in terms of decreased settling time and overshoot [10]. However, the performances of these controllers depend on the accuracy of system models and parameters. Generally, an accurate nonlinear model of actual DC motor is difficult to find. As a result, many researchers are now interested in using intelligent control techniques such as Fuzzy Logic Controllers (FLC), Self-Tuning Fuzzy Logic Controller (STFLC), Fuzzy PID (FPID) controller, Neural Network and Adaptive Neuro-Fuzzy System (ANFIS) controller. Self-tuning controllers, such as PI and PID, are sensitive to changes in motor characteristics and load. It’s also difficult to tune PI or PID gains to prevent overshoot caused by load disturbance. However, the PID response is still regarded as poor due to significant overshoot and a long settling time value [10]. LQG controllers are not only optimum, but they can also estimate non-measurable states by utilizing observers to rebuild them and perform better in the presence of wind gust noise [9]. A state-feedback controller with pole placement assignment is used to achieve very minor overshoot, a small settling time value, and zero steady-state error in order to produce a good transient response specification [3]. If the system parameters perfectly known (certainties) LQR is effective to achieve the desired performance by choosing the control law. Also, a hybrid PID-LQR controller is proposed by combining PID with a Linear Quadratic Regulator, in order to optimize system response and increases the precision of system. The hybrid PID-LQR controller's performance outperforms other controllers significantly in terms of decreased settling time and overshoot [10]. However, the performances of these controllers depend on the accuracy of system models and parameters. Generally, an accurate nonlinear model of actual DC motor is difficult to find. As a result, many researchers are now interested in using intelligent control techniques such as Fuzzy Logic Controllers (FLC), Self-Tuning Fuzzy Logic Controller (STFLC), Fuzzy PID (FPID) controller, Neural Network and Adaptive Neuro-Fuzzy System (ANFIS) controller. The Adaptive Neuro-Fuzzy System (ANFIS) controller is more efficient because it combines the advantages of the FLC and NN approaches to establish a nonlinear self-tuning controller. When using the PID, FLC, and ANFIS techniques, it is observed that ANFIS provides the best performance in terms of stabilizing the system response in a much shorter rise time with the least amount of rise time and overshoot [11]. On the other hand, artificial intelligence techniques have high cost and complexity.

Fractional order Proportional-Integral-Derivative (FOPID) controller approach provides a small settling time, in addition to increase the system's robustness and makes it more stable [12]. PID and FOPID controllers have constant coefficient which are proportional gain \( K_p \), integral gain \( K_i \), and derivative gain \( K_d \), these parameters have the same value during whole operation in spite of the system error. As a consequence, it is impossible to interfere with the transient and steady-state responses separately. Variable Coefficient PID (V-PID) controller and Variable Coefficient Fractional Order PID (V-FOPID) controller are established to enhance the transient state while maintaining the steady-state response unaffected. Unfortunately, whenever the system's performance enhances in terms of settling and rising times and overshoot, the number of tuned parameters increases. As a result, the system's optimization problem becomes more challenging [13]. In order to achieve a certain performance, it is desired to develop a systematic approach for optimal tuning of the proposed controller parameters. In various applications, several evolutionary techniques are used for tuning PID controller parameters.

In this research, Particle Swarm Optimization (PSO) [14], Adaptive Weighted Particle swarm optimization (AWPSO), Adaptive Acceleration Coefficients Particle Swarm Optimization (AACPSO), Modified Adaptive Accelerated Coefficients Particle Swarm Optimization (MAACPSO) [15], Crazy Particle Swarm Optimization (C-PSO) [16], Phasor Particle Swarm Optimization (PPSO) [17], Gravitational Search Algorithm with Particle Swarm Optimization (GSA-PSO) [18], and Eagle Strategy with Particle Swarm Optimization (ES-PSO) [19] are proposed for optimal tuning of (PID, FLP, V-PID, FOPID, V-FOPID) controllers in Satellite Control System. Performance Index measured using Integral Based Objective Functions and Dynamic Performance Indices Based Objective Functions [20]. Also, Self-Tuning Fuzzy PID (STF-PID) controller [2], and Self Tuning Fuzzy FOPID (STF-FOPID) are proposed for satellite tracking control system.

https://drive.google.com/drive/folders/111RfoFG1LV6KJ81B1ipNSjNMghxK-J3?usp=sharing , this link contains simulation files and used code.

2. Satellite Tracking Control System

For multiple missions satellite ground station, the same earth station can track and communication with LEO, MEO, HEO and GEO satellites by adjusting the azimuth and elevation angles and used the suitable sending/receiving frequency bandwidth. It's important to notice that the antenna movements are only applicable to cases of LEO, MEO and HEO satellites and for GEO satellites, the control algorithms are commonly utilized for a one-time for placement of an antenna dish automatically rather than carrying out the process manually. The issue of antenna azimuth/elevation position control has become one among the various aspects that have piqued the interest of researchers within the control of antenna placement [21]. These interests stem from the significant roles and improved performance that would be obtained from an antenna's precise azimuth/elevation placement. Several approaches have been presented to obtain optimal control of the azimuth/elevation position, but such placement remains a difficult control problem. Yet crucial accuracy is necessary to maintain accurate signal levels, necessitating the use of a high-performance controller; hence, the goal for this research is to provide a more robust solution to the aforementioned problems. The suggested controller enhances the response of the antenna position system by eliminating overshoot and steady-state error and reducing rising time, setting time, and peak time. A ground station's tracking control system presented in Fig. 1, used for tracking the desired satellite in both azimuth and elevation angles. The summer contains two inputs; the first input is the desired point at which azimuth or elevation motor is anticipated to run to and stop, the second input is the current position of the azimuth or elevation motor as measured by the feedback sensor. The difference between these two inputs is referred to as the position error signal, and it is provided to the controller. The controller receives the error signal and generates the corresponding output signal, known as controller output. Therefore, the controller output is sent to the motor driver, which generates a proportionate output to rotate the appropriate motor in either direction, depending on the sign of the error signal (positive or negative). As the desired position is achieved, the error signal decreases to zero, and the motors stop.
3. Tracking System Motors

Electric motors are often categorized based on their functions, such as servomotors, gear motors, also, categorized based on their electrical configurations, such as (DC) and (AC) motors. In terms of operating principles they classified into single phase and poly-phase for AC motors, and static magnet and shunt DC motors for DC motors [22]. An armature-controlled DC stepper motor is frequently used to align the antenna to the required location. Also, two-phase hybrid stepper is used for tracking system. In this research work, an armature-controlled DC servo motor is employed to track satellite movement across the sky.

3.1. Preference of Servo Motors over Stepper Motors

In this study, the DC servo motors were preferred to the stepper motors because of the following reasons:

1) Servo motors can produce high torque across a wide speed range on demand also are offered in a wider torque range and higher voltages.
2) Servo motors consume power only to rotate to the desired position and doesn’t consume power when at rest. Stepper motors require more current (power) to lock into an and maintain the desired position.
3) At certain speeds or load dynamics, stepper motors suffer from vibration and resonance issues, resulting in missed steps, stopping, excessive vibration and noise.
4) Stepper motors are commonly utilized in open-loop position control, with the number of steps of movement previously set, a controller is required to deliver the position of the stepper motor at power up. On the other hand, servomotor will instantaneously move to whatever angle the controller directs it to, regardless of initial position at starting.
5) Stepper motors lack feedback, therefore limiting their performance to drive loads that are within their capability, resulting in positioning inaccuracies due to miss steps under increasing loads.
6) Stepper motors produce more continuous torque at lower speeds than servo motors. Otherwise, servo motors deliver intermittent peak torques in the same low-speed range as well as peak and continuous torques across a considerably wider-higher speed range.

7) Servo motors are better choice for variable load systems than stepper motors which cannot react to changes is load.

3.2. DC Servo Motor System

The DC servo motor is essentially a torque transducer that transforms electrical energy into mechanical energy. Servo motors are an automated device, that are used in closed-loop control systems to regulate position or speed. Today, DC servo motors are utilized in disc drives of computers, numeric control machines, industrial equipment, actuator for automation control process, satellite tracking antennas, weapon industry and speed control of alternators. So, we consider DC servo motors for position tracking because they have some limitations in terms of precise measurement of position rather than speed. The rotor inertia and time constant of a DC servomotor are exceedingly tiny. As a result, the torque-to-inertia ratios of the motors are quite high for commercially applications. Field resistance control, armature voltage control, and armature resistance control are the three most popular speed control methods of DC servo motors. Because servo motors are less sensitive to changes in field current, we will focus on armature voltage control approach. Fig. 2 shows a typical model of a servomotor system with gear system [21, 23] and [24]. A servomotor is made up of two major parts: the first is the electrical part, which is made up of armature resistance ($R_a$), armature inductance ($L_a$), armature voltage ($V_a$), besides the back electromotive force ($E_b$). The mechanical component of the servomotor is the second component through which we obtain the useful mechanical rotational movement at the shaft. The mechanical component consists of the motor’s shaft, inertia of the motor ($J_m$) and load inertia ($J_L$) and viscous friction coefficient ($B_L$). The angle ($\theta$) refers to the angular position of the motor shaft which can be used to find the angular speed of the shaft ($\omega$). $(N_1,N_2)$ refers to gear teeth. ($\theta_{\alpha}$) denotes to the angular position of the load shaft which can be used to find the angular speed of the load shaft ($\omega$). Also, ($T_m$) is the load torque disturbance which consist of the sum of torque due to load ($T_L$) and torque due to friction($T_F$).
3.2.1. Modeling of DC Servo Motor System

For an armature voltage controlled separately excited DC motor shown in Fig. 2, the voltage applied to the motor’s armature circuit is changed while the voltage applied to the field circuit remains constant. Applying Kirchhoff’s Voltage Law to armature loop:

\[ V_a(t) = R_a i_a(t) + L_a \frac{d i_a(t)}{dt} + E_b(t) \]  

\[ E_b = K_b \omega_m \]  

(1)

\[ E_b \] is the back electromotive force due to the movement of armature conductor through the field flux established by the field current:

\[ E_b = K_b \frac{d \theta}{dt} = K_b \omega(t) \]  

(2)

The motor torque equation given by Eq. (3):

\[ T_m = K_f L_p(t) \]  

(3)

Based on the fact that at any instant of time, the developed torque must be equal and opposite torques due to friction, inertia and load:

\[ T_m(t) = T_p(t) + B_e q \omega(t) + T_L(t) + J_\text{eq} \frac{d \omega(t)}{dt} \]  

(4)

Where:

\[ J_\text{eq} = J_m + \left( \frac{N_c^2}{N_a^2} \right) J_L \]  

(5)

\[ B_e q = B_m + \left( \frac{N_c^2}{N_a^2} \right) B_L \]  

(6)

Let \( T_a(t) \) as in Eq. (7), to get Eq (8):

\[ T_a(t) = T_p(t) + T_L(t) \]  

(7)

\[ T_m(t) = T_a(t) + B_e q \omega(t) + J_\text{eq} \frac{d \omega(t)}{dt} \]  

(8)

Now, the four Eqs. (1), (2), (3) and (8) constitute a basic set of equations that model the DC Motor and from which the transfer function or block diagram may be obtained. Applying Laplace transform on both sides of basic set of equations leads to:

\[ V_a(s) - E_b(s) = (L_a S + R_a) I_a(s) \]  

(9)

\[ E_b = K_b \omega_m(s) \]  

(10)

\[ T_m = K_f L_p(s) \]  

(11)

\[ T_m - T_d(s) = (J_\text{eq} S + B_e q) \omega(s) \]  

(12)

\[ T_m - T_a(s) = (J_\text{eq} S^2 + B_e q S) \theta(s) \]  

(13)

\[ K_f I_a(s) - T_a(s) = (J_\text{eq} S^2 + B_e q S) \theta(s) \]  

(14)

\[ K_f \left( \frac{V_a(s) - E_b}{L_a S + R_a} \right) - T_d(s) = (J_\text{eq} S^2 + B_e q S) \theta(s) \]  

(15)

3.2.2. DC Servo System Block Diagram Representation

The block diagram model of the open loop DC servo motor system with the load is shown in Fig. 3.

3.2.3. DC Servo System Transfer Function

Mechanical motion transferred from motor shaft to the load through a gear box. The transfer function of armature voltage-controlled DC servo motor when loaded without gear system given by Eq. (16) and when loaded with gear system given by Eq. (17). Let \( T_d(s) = 0 \) in the block diagram model shown in Fig. 3 leads to:

\[ G_\text{angle}(s) = \frac{\theta(s)}{V_a(s)} = \frac{1}{K_f} \]  

(16)

\[ G_0 \text{angle}(s) = \frac{\theta_0(s)}{V_a(s)} = \frac{K_t}{K_f} \frac{1}{L_a J_\text{eq} S^3 + \left( R_a J_\text{eq} + B_e q L_a \right) S^2 + \left( R_a B_e q + K_t K_\theta \right) S} \]  

(17)

3.3. Mathematical Modeling of Satellite Tracking System

Fig. 4 depicts a block schematic of a system for regulating the azimuth (or elevation) position of a satellite tracking antenna. The overall system includes three subsystems: Controller; power amplifier, motor and load, and feedback position sensor. Every subsystem has its own transfer function. To model and analyze this system, the following assumptions were made [25]:

i. The project data supplied adequately describes the system. This implies that the values used to simulate the motor, gears, load, Power Amplifier, Position Sensor and Voltage References are precise.

ii. The transfer functions provided for the power amplifier are precise, and saturation is never achieved.

iii. There are no disturbances or interference in the signals transferred between control system components. This implies that the position sensor and power amplifier dynamics are ignored.

3.3.1. Selected Earth Station Antenna System

Antesky 9.0-meter C/Ku-band Earth Station Antenna weights approximately 450 kg and employs a shaped paraboloid reflector to provide high gain and excellent pattern features. Antesky earth station antenna systems are offered for a variety of applications worldwide, including wireless telecommunications service providers, Internet service providers, system operators, and broadcasters. This earth station based on motorized drive. This motorized system can
steer the antenna in azimuth angle from 0 to 360 degrees, also can steer antenna in elevation angle from 0 to 90 degrees [26].

Moment of inertia for antenna system is calculated using Eq. (18):

\[ J_L = \frac{wR^2}{2g} \cong 827 \text{ kg} \cdot \text{m}^2 \] (18)

W: Weight (N), R: Radius (m) and g: Gravitational constant (N/kg).

3.3.2. Selected Servomotor for Antenna System

Kollmorgen’s AKM82T Servomotor is selected, which could produce continuous torque up to 75 N.m. Kollmorgen’s AKM™ brushless servomotors offer exceptional choice and flexibility from a diverse collection of standard products, allowing you to choose the most effective servomotor for your application. Choosing the right motion control devices has never been easier, thanks to the combination of AKM servomotors with family of plug-and-play AKDTM servo drives it becomes easier. It’s possible to choose from hundreds of servomotor/servo drive combinations mentioned in the selection guide or check their website to get the most straightforward solution for your application [27].

3.4. Parameters of Tracking Satellite System

In order to model the system, several parameters and variables which represent the DC servomotor, significant inputs, outputs and signals were defined in table format. Table 1 shows parameters of tracking system.

| Parameters | Description | Value |
|------------|-------------|-------|
| \( V_a \) | Motor armature input voltage | 0 up to 640 VDC |
| \( R_a \) | Motor armature resistance | 0.092 ohm |
| \( L_a \) | Motor armature inductance | 2.73 mH |
| \( I_a \) | Motor armature current | 0.108 V/s/rad |
| \( K_b \) | Back EMF constant of motor | 1.08 V/s/rad |
| \( E_b \) | Back electromotive force | V |
| \( T \) | Torque constant of motor | 1.6 Nm/A |
| \( J_m \) | Moment of inertia of motor | 1.72 kg.m² |
| \( J \) | Moment of inertia of load | 827 kg.m² |
| \( B_m \) | Viscous friction coefficient of motor | 0.35 Nms. /rad |
| \( B_L \) | Viscous friction coefficient of load | 1.5 Nms. /rad |
| \( J_{eq} \) | Equivalent moment of inertia | 1.726577 Nms. /rad |
| \( B_{eq} \) | Equivalent viscous coefficient of load | 0.350011 Nms. /rad |
| \( T_L \) | Torque due to load | 2.3 Nm |
| \( T_F \) | Torque due to friction | 1.7 Nm |
| \( K_1 \) | Power Amplifier gain | 100 |
| \( K_p \) | Power Amplifier pole. | 100 |
| \( K_g \) | Gearbox ratio | 0.002777 |
| \( N_1 \) | Gear Teeth | 5 |
| \( N_2 \) | Gear Teeth | 1800 |
| \( K_{ps} \) | Position sensor gain | 1 |
| \( \alpha_m \) | Motor/load shaft angular velocity | (18) |
| \( \theta \) | Motor shaft angle |

4. Control Strategies

This section introduces various control strategies: PID, FPID, V-PID, FOPID and V-FOPID involved in satellite position control problems. Also, displays their analyses for controlling the satellite’s standalone position.

4.1. Conventional PID Controller

PID controllers were extensively utilized in various industrial control systems owing to its simple construction, reliable performance and inexpensive cost [28]. However, these controllers provide better performance only at particular operating range and that they got to be retuned if the operating range is modified. However, these controllers give superior performance only within a specific operating range and must be retuned if the operating range is modified. Furthermore, conventional controllers performance doesn’t meet predicted for nonlinear systems [29]. Fig. 5 describes the architecture of a PID controller, which comprises of Proportional (\( K_P \)), Integral (\( K_I \)) and Derivative (\( K_D \)) gains. These gains are placed in parallel along the feed forward path of a closed loop system. The aim of proportional control is to minimize the rising and settling times whereas the function of integral control is to eliminate the system’s steady state error. The derivative control is often used to enhance the closed-loop system’s transient response. The calculated error between the reference value and actual measurement is supplied into the PID controller. Finally, the formulas for a continuous-time parallel-form PID controller are stated below:

\[ U(t) = K_P e(t) + K_I \int_0^t e(t) + K_D \left( \frac{d}{dt} e(t) + k \right) \] (19)

Also, the transfer function of the PID controller is also given by Eq. (20):

\[ G_{PID}(S) = K_P + K_I \frac{1}{S} + K_D \left( \frac{N}{S+N} \right) \] (20)

4.1.1. Optimal Design of the (PID) Controller

The (PID) controller tuning parameters are \( K_P \), \( K_I \) and \( K_D \). The searching space of each parameter as follows:

\[ K_P \in [0 \ 1000], \quad K_I \in [0 \ 1000], \quad K_D \in [0 \ 1000], \quad N \in [0 \ 1000] \]

4.2. The FOPID Controller
Igor Podlubny introduced a fractional order PID controller symbolized by \( P^{1+D^\mu} \), in 1999 [30]. Is one among the most recent advanced controllers. The FOPID controller originates from the field of fractional order calculus. Theoretically, fractional calculus is an extension of classical calculus. Fractional calculus has been thought a generalization of classical calculus in which \( d^n y/dt^n \), \( n \) is accepted as a non-integer form. In general, both of fractional order derivatives and integrals are expressed by \( D \) letter, which implies exactly the sort of fractional calculus. The mathematical form of fractional order integrator and derivative is donated by \( aD_t^p \) which defined by Eq. (21):

\[
\begin{align*}
\frac{d^a}{dt^a} a > 0 \\
1 \quad a = 0 \\
\int_a t a < 0
\end{align*}
\]

(21)

Where \( a \) denotes the fractional order, \( a \) and \( t \) indicate the lower and upper limits of the integral operator. Negative terms of \( a \) refers to fractional integrals, whereas positive values indicated fractional derivatives. \( a \) can also be a complex number that allows us to define systems in a more powerful and effective manner. Many systems are frequently represented in several ways due to the exceptional properties of fractional calculus[13]. There are several definitions of fractional calculus in the literature, but three of them most commonly used. These definitions are: Grunwald-Letnikov, Riemann-Liouville and Caputo which are used to define fractional order integrator and derivative operators, \( aD_t^p \) [31].

In a comparison to the PID controller, the FOPID controller which displayed in Fig. 6 is distinguished by two extra control parameters in which the orders of the integral part \( \lambda \) and derivative part \( \mu \) are non-integer. As a result, in FOPID controller design, the fractional calculus is included into PID controller to give greater flexibility in controller design and to improve dynamic performance. Parameters \( \lambda \) and \( \mu \) primarily impact the integral and differential links of the controller. The \( \lambda \) and \( \mu \) ranges are adjusted based on the order of the system. \( \lambda \) mostly influences the system’s steady accuracy and adjustment time, whereas \( \mu \) primarily affects overshoot and the stability of a closed loop system.

![Fig. 6. Structure of FOPID controller](image)

The schematic illustration of the PID and FOPID controllers on the \((\lambda, \mu)\) plane is included in Fig. 7. The P, PI, PD, and PID controllers are obviously four points on the plane, but the FOPID controller can have any value on the plane. If \( \lambda \) and \( \mu \) are arbitrarily chosen, then FOPID controller will cover the entire plane [32]. With the inclusion of two degrees of freedom, the FOPID provides greater flexibility in the area of control in constructing PID controllers and allows for better adjustment of the control system’s dynamics. It also enhances the control system’s robustness, accuracy and stability [33]. However, as the number of parameters to be adjusted grows, the system optimization issue gets more difficult. It is desirable to build a systematic algorithm for FOPID optimization in order to achieve the certain performance.

The expressions of the FOPID controller are stated below:

The output of the FOPID \((P^{1+D^\mu})\) controller is defined by Eq. (22):

\[
U(t) = K_p e(t) + K_i D_t^{-\lambda} e(t) + K_D D_t^\mu e(t)
\]

(22)

The transfer function of the FOPID is also given by Eq. (23):

\[
G_{FOPID}(s) = K_p + K_i S^{-\lambda} + K_D S^\mu
\]

(23)

![Fig. 7. Schematic representation of FOPID controller](image)

In actuality, the values of \((\lambda, \mu)\) can’t be precisely attained or implemented in real-time systems, because they constitute an infinite dimensional linear filter [34]. A band-limit FOC (Fractional order control) implementation is critical in practice finite-dimensional approximation of FOC across a suitable frequency range. There are various techniques for obtaining the approximate s-transfer function of a fractional order differentiator and integrator [12]. In [35], Oustaloup proposed an approximation that makes use of a recursive distribution of zeros and poles. A \( 5^{th} \) order Oustaloup’s recursive approximation is performed for \((\lambda, \mu)\) operators within the selected frequency band of \( \omega \in \{10^{-2}, 10^2\} \) rad/s. The high-order filter presented in Eq. (24) [36]:

\[
G_f(s) = s^\mu = K \prod_{n=0}^{N} \frac{s + \omega_{\mu}}{s + \omega_{\mu}^{n+1}}
\]

(24)

Where the poles \( \omega_{\mu} \), zeros \( \omega_{\mu}^{n} \), and gain \( K \) of the filter are obtained from the following formula:

\[
\omega_{\mu} = \omega_1 \left( \frac{\omega_{\mu}}{\omega_1} \right)^{k+1}, \quad \omega_{\mu}^{n} = \omega_b \left( \frac{\omega_{\mu}}{\omega_b} \right)^{k+1}
\]

(25)

Where \( \alpha \) is the order of fractional operator, \( \omega \in \{\omega_1, \omega_2, \omega_3, \omega_4, \omega_5\} \) is the filter frequency band, \((2N + 1)\) is the order of filter.

5.2.1. Optimal Design of the (FOPID) Controller

The (FOPID) controller tuning parameters are \( K_p, K_i, K_D, \mu \) and \( \lambda \). The searching space of each parameter as follow:

\[
K_p \in [0 \ 1000] , \quad K_i \in [0 \ 1000] , \quad K_D \in [0 \ 1000], \quad \mu \in [0 \ 2], \quad \lambda \in [0 \ 2]
\]

Further, use a new searching area \( K_p \in [0 \ 400] \) used in the case of multi objective function \( I_2 \) with (PPSO), (GSA-PSO) and (ES-PSO) strategies.

4.3. Fuzzy-PID (FPID) Controller

Fig. 8 shows the structure of the proposed Fuzzy-PID controller. This structure combines Fuzzy-PD and Fuzzy-PI controllers beside \( K_p \) and \( K_d \) as input scaling factors and \( K_{pi} \) and \( K_{pd} \) as output scaling factors. ‘Mamdani-type’ FLC the (e) and the rate change of error (de/dt) as inputs and compute single output \( (Ku) \). The FLC input and output variables are both represented by seven gaussian membership function \{NB, NM, NS, ZO, PS, PM, PB\}, the seven-membership function are obviously referring to negative big, negative medium, negative small, zero, positive small, positive medium, positive

ISSN (Print): 2456-6411 | ISSN (Online): 2456-6403
DOI: 10.46565/jreas.2022.v07i01.001
JREAS, Vol. 07, Issue 01, Jan 22
big, respectively. The Gaussian membership function is utilized because of its several advantages, including smooth functions, non-zero at all locations, and the provision of actual information at all points [29]. For both inputs and output, the range of inputs (e and de/dt) and output (Ku) fuzzy variables defined as follow: 

\[ e \in [-1600, 1600], \quad de \in [-800, 800] \text{ and } Ku \in [-10, 10]. \]

Fuzzy rules used to fire output (Ku) are demonstrated in Table 2. To determine the crisp output, the center of gravity defuzzification approach is used.

### Table 2: Fuzzy rules of Ku

| e  | NB | NM | NS | ZO | PS | PM | PB |
|----|----|----|----|----|----|----|----|
| de |    |    |    |    |    |    |    |

**4.3.1. Optimal Design of the (FPID) Controller**

The (FPID) controller tuning parameters are \( K_e, K_d, K_{P_I} \) and \( K_{P_D} \). The searching space area of each parameter as follow:

\[ K_e \in [0, 1000], \quad K_d \in [0, 100], \quad K_{P_I} \in [0, 100], \quad K_{P_D} \in [0, 500]. \]

**4.4. Self-Tuning Fuzzy PID (STF-PID) Controller**

Conventional PID Controllers performances depend upon the precision of system models and parameters. Usually, a precise non-linear model of actual DC motor is pretty hard to estimate. As a result, many researchers today are interested in applying intelligent control techniques which could deal with uncertain dynamic and non-linearity within the system and supply desired response. The Self-Tuning Fuzzy PID (STF-PID) controller characterized in Fig. 9, is a result of combining the PID controller with the fuzzy logic control approach. By utilizing the novel controller, it is possible to execute online automatic tuning of the PID controller parameters and keep making the selected control parameters more regional [2]. FLC transformed classical PID to an adaptive controller.

In the suggested strategy, the “Mamdani-type” FLC structure, utilizes the (e) and the rate change of error (de/dt) as inputs and compute three outputs \( (K_p, K_i, \text{ and } K_d) \) as shown in Fig. 10. Further, \( N \) value adjusted manually. The dual inputs and three outputs are well-defined using fuzzy sets, and each fuzzy set defined by seven fuzzy subsets \{NB, NM, NS, ZO, PS, PM, PB\}. The seven subsets are defined by generalized bell membership function and referring to negative big, negative medium, negative small, zero, positive small, positive medium, positive big, respectively. According to the summary of related technical knowledge and practical operation, the corresponding fuzzy rules are used to fire \( K_p, K_i \) and \( K_d \) outputs are shown in Table 3, Table 4 and Table 5. [37]. The input and output have quite a nonlinear relationship, especially for the proportional and derivative gains. Furthermore, the values of the proportional, integral, and derivative gains increase with increasing error amplitude and rate change of error. Moreover, for lower error and change-in-error levels, these values are steadily reduced. The center of gravity defuzzification technique is carefully chosen to compute crisp output [38].

### Table 3: Fuzzy rules of \( K_p \)

| e  | NB | NM | NS | ZO | PS | PM | PB |
|----|----|----|----|----|----|----|----|
| de |    |    |    |    |    |    |    |

Fig. 8. Structure of Fuzzy-PID controller

Fig. 9. Structure of (STF-PID) controller

Fig. 10. Structure of Mamdani-type FLC controller

The range of inputs and outputs fuzzy sets defined as follow:

\[ e \in [-1, 1], \quad de \in [-1, 1], \quad K_p \in [0, 100], \quad K_i \in [0, 600], \quad K_d \in [0, 550]. \]

| e  | NB | NM | NS | ZO | PS | PM | PB |
|----|----|----|----|----|----|----|----|
| de |    |    |    |    |    |    |    |

Table 3 Fuzzy rules of \( K_p \)
Table 4 Fuzzy rules of $K_p$

| $e$ | NB | NM | NS | ZO | PS | PM | PB |
|-----|-----|-----|-----|-----|-----|-----|-----|
| NB  | NB  | NB  | NM  | NM  | NS  | ZO  | ZO  |
| NM  | NB  | NB  | NM  | NS  | NS  | ZO  | ZO  |
| NS  | NB  | NM  | NS  | NS  | ZO  | PM  | PM  |
| ZO  | NM  | NM  | NS  | ZO  | PS  | PM  | PM  |
| PS  | NM  | ZO  | PS  | PS  | PS  | PM  | PB  |
| PM  | ZO  | ZO  | PS  | PS  | PS  | PB  | PB  |
| PB  | ZO  | ZO  | PS  | PM  | PM  | PB  | PB  |

Table 5 Fuzzy rules of $K_d$

| $e$ | NB | NM | NS | ZO | PS | PM | PB |
|-----|----|----|----|----|----|----|----|
| NB  | PS | NS | NM | NB | NB | NM | PS |
| NM  | PS | NS | NB | NM | NM | NS | ZO |
| NS  | ZO | NS | NM | NM | NS | NS | ZO |
| ZO  | ZO | NS | NS | NS | NS | NS | ZO |
| PS  | ZO | ZO | ZO | ZO | ZO | PS | PB |
| PM  | PB | NS | PS | PS | PS | PB | PB |
| PB  | PB | PM | PM | PS | PS | PB | PB |

4.5. Self-Tuning Fuzzy Fractional Order PID Controller

FOPID controller is the enhancement version of traditional PID controller, in addition the PID controller is an exceptional case of FOPID controllers. Compared to FOPID, STF-FOPID is a robust controller which doesn’t change its response for disturbance appearance. Self-Tuning Fuzzy FOPID (STF-FOPID) controller represented in Fig. 11, is the improved version of the FOPID which is a combination of fuzzy control theory with FOPID controller. (STF-FOPID) output performance have less settling time and free overshoot in comparison to PID and FOPID. (STF-FOPID) is an adaptive controller which may perform online tuning of FOPID controller parameters automatically and may deal with uncertain dynamic and non-linearity within the system and provide desired response.

In the suggested control strategy, the ‘Mamdani-type’ FLC utilizes the $(e)$ and the rate of the change of error $(de/dt)$ as inputs and compute three outputs $(K_p$, $K_i$, $K_d)$. Further, $(\lambda, \mu)$ values adjusted manually. The dual inputs and three outputs are defined using fuzzy sets as shown in Fig. 12:

![Fig. 11. Structure of (STF-FOPID) controller](image)

The range of inputs and outputs fuzzy sets defined as follow: $e \in [-7.5, 7]$, $de \in [-17, 12]$, $K_p \in [700]$, $K_i \in [0, 1.13]$, $K_d \in [0, 1.072]$. Fuzzy rules used to fire $K_p, K_i$ and $K_d$ outputs are shown in Table 3, Table 4 and Table 5. The center of gravity defuzzification method is chosen to calculate the crisp output.

4.6. Variable Coefficient PID Controller (Nonlinear-PID)

The classical PID controller given by Eqs. (19) and (20) have constant parameters. These parameters are proportional gain $K_p$, integral gain $K_i$, derivative gain $K_d$. Regardless of the system error signal, these parameters have the same value during whole operation. As a result, it is difficult with the use of constant parameters to interfere independently between the transient and steady states. The study's main goal is to find an answer to the question of what type of controller structure should be developed to enhance the transient state without impacting the steady-state response [13]. Therefore, it’s probable to establish (V-PID) controller structure as displayed in Fig. 13. The expressions for the V-PID controller which is represented in Fig. 13 are stated below:

The output of the variable PID (V-PID) controller is defined by Eq. (26):

$$U(t) = K_p e(t) + K_i \int_0^t e(t) dt + K_d \left( \frac{de(t)}{dt} + \mu \right)$$  \hspace{1cm} (26)

The transfer function of the PID (V-PID) is also given by Eq. (27):

$$G_{V-PID}(S) = K_p + K_i \frac{1}{S} + K_d \left( \frac{NS}{S+N} \right)$$ \hspace{1cm} (27)

![Fig. 13. Structure of (V-PID) controller](image)

It is obvious that the Eqs. (26) and (27) are obtained from Eqs. (19) and (20), which is a classical PID controller.
(\(K_p\)) gains of Eqs. (26) and (27) contrast from the classical PID ones with the gains symbolized by \(K_p, K_i\) and \(K_d\). These gains depend upon the current values of the errors. Thereby, the values of the controller parameters change in regard to the system error. This new V-PID approach allows to enhance system’s responses in both transient and steady-state conditions. The variable coefficients \(K_p, K_i\) and \(K_d\) are defined by Eqs. (28), (29) and (30) as follow:

\[
K_p = c_1|e(t)| + c_2 \\
K_i = c_3|e(t)| + c_4 \\
K_d = c_5|e(t)| + c_6
\]

Where, \(|e(t)|\) is the absolute error of the system with controller. Besides, \(c_1\) through \(c_6\) and \(N\) are the new tuning parameters of the V-PID controller for varying coefficients.

4.6.1. Optimal Design the of (V-PID) Controller

The (V-PID) controller tuning parameters are \(c_1, c_2, c_3, c_4, c_5, c_6\) and \(N\). The limits of searching space area of each parameter are defined as follow:

\[
c_1 \in [0 \; 300], \quad c_2 \in [0 \; 100], \quad c_3 \in [0 \; 100], \quad c_4 \in [0 \; 700], \quad c_5 \in [0 \; 100], \quad c_6 \in [0 \; 300], \quad N \in [0 \; 100]
\]

4.7. Variable Coefficient Fractional Order PID Controller

The FOPID controller illustrated in Eqs. (22) and (23) have constant coefficients which are proportional gain \(K_p\), integral gain \(K_i\) and derivative gain \(K_d\). These parameters have the same value during whole operation in spite of the system. As a consequence, it is impossible to interfere with the transient and steady-state responses separately. V-FOPID controller is established to enhance the transient while maintaining the steady-state response unaffected. In addition to providing far more freedom in constructing FOPID controllers, it also allows to fine-tune the dynamics of the control system. The V-FOPID approach grows the system’s robustness and stability [13]. The expressions for the V-FOPID controller which demonstrated in Fig. 14 are stated below as follows:

The output of the V-FOPID (\(V - PI^\lambda D^\mu\)) controller is defined by Eq. (31):

\[
U(t) = K_p e(t) + K_i D e(t) + K_d D^\mu e(t)
\]

The transfer function of the V-FOPID (\(V - PI^\lambda D^\mu\)) is given by Eq. (32):

\[
\dot{G}_{V\text{-FOPID}}(S) = K_p + K_i S^{-\lambda} + K_d S^\mu
\]

It is clearly seen that the Eqs. (31) and (32) are obtained from Eqs. (22) and (23), which is a classical FOPID controller. Nevertheless, the proportional, integral and derivative gains of Eqs. (31) and (32) traditional FOPID ones with the gains denoted by \(K_p, K_i\) and \(K_d\). These gains values are dependent on the contemporary values of the system’s exact error. Thus, the controller parameters’ values vary in relation to the system’s exact error. This novel fractional order PID technique allows to enhance system’s responses in both transient and steady-state conditions. The variable coefficients \(K_p, K_i\) and \(K_d\) are defined as in Eqs. (28), (29) and (30) as follows:

4.7.1. Optimal Design of the (V-FOPID) Controller

V-FOPID controller tuning parameters are \(c_1, c_2, c_3, c_4, c_5, c_6, \mu\) and \(\lambda\). The limits of searching space area of each parameter are as the following:

\[
c_1 \in [0 \; 300], \quad c_2 \in [0 \; 200], \quad c_3 \in [0 \; 700], \quad c_4 \in [0 \; 300], \quad c_5 \in [0 \; 400], \quad c_6 \in [0 \; 200], \quad \mu \in [0 \; 1], \quad \lambda \in [0 \; 1]
\]

5. Optimal Design Based on Evolutionary Techniques

Many evolutionary algorithms are reported in literature for tuning of PID controller parameters in many applications. In this research, Particle Swarm Optimization (PSO) [14], Adaptive Weighted Particle swarm optimization (AWPSO), Adaptive Acceleration Coefficients Particle Swarm Optimization (AACPSO), Modified Adaptive Accelerated Coefficients Particle Swarm Optimization (MAACPSO) [15], Crazy Particle Swarm Optimization (C-PSO) [16], Phasor Particle Swarm Optimization (PPSO) [17], Gravitational Search Algorithm with Particle Swarm Optimization (GSA-PSO) [18], and Eagle Strategy with Particle Swarm Optimization (ESPSC) [19], are proposed for optimal tuning of PID, FPID, V-PID, FOPID and V-FOPID controllers in Satellite Control System.

Performance Index measured using Integral Based Objective Functions and Dynamic Performance Indices Based Objective Functions [20]. The basics of these proposed performance index and evolutionary algorithms are as follow:

5.1. Formulation of Objective Function

During the optimal design of controller, the most crucial step is to pick the most proper objective function. Time domain objective functions are often categorized into two groups: Integral based objective functions and dynamic performance indices based objective functions [20].

Integral based objective functions commonly utilized in literature are: IAE (Integral of Absolute Error), ITAE (Integral of Time Absolute Error), ISE (Integral of Squared Error) and ITSE (Integral of Time Squared Error). For these objective functions, the difference between the system output and reference signal is denoted by the error. All of those integral based objective functions have advantages and disadvantages. As an example, since IAE and ISE criteria are autonomous of time, the obtained results have relatively small overshoot but a long settling time. On the other hand, ITAE and ITSE can overcome this disadvantage but, they can't provide a desirable stability margin.

The second category of the time domain objective functions is relied on the performance indices of the system dynamic output. These functions usually include the maximum overshoot (\(M_p\)), rising time (\(t_r\)), settling time (\(t_s\)), and steady-state error (\(E_{ss}\)) [20]. In this study, integral based objective function (\(J_1\)) (single objective function) given by Eq. (33) and dynamic performance indices based objective function (\(J_2\)) (multi objective function) given by Eq. (34) are used for controller design.
5.2. Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is one among the contemporary meta-heuristic population-based stochastic optimizations, firstly introduced by Kennedy and Eberhart in 1995. It performs an evolutionary search to identify near optimal or optimal solutions. The PSO behavior is inspired from the searching strategy for optimal food sources by the bird swarms. PSO evolutionary-based search procedure in which particles were initially randomly generated in a consistent manner across the search space, and velocity is also randomly initialized. These particles' positions ($X$) vary throughout time. During each iteration, each particle adjusts its position based on its own experience ($P_{best}$), as well as the experience of global neighborhoods particles ($G_{best}$), where each particle is connected to and can obtain information from every other particle in the swarm. Essentially, two PSO algorithms have been established. These two PSO algorithms namely Local Best ($P_{best}$) and Global Best ($G_{best}$), which differ in the size of their neighborhoods [39].

At every iteration, PSO algorithm updates the whole swarm through updating the velocity and position of each particle in the swarm by the following Eqs. (37) and (38):

$$V_{i}^{(iter+1)} = \omega V_{i}^{(iter)} + C_{1}r_{1}(P_{best_{i}}^{(iter)} - X_{i}^{(iter)}) + C_{2}r_{2}(G_{best_{i}}^{(iter)} - X_{i}^{(iter)})$$

$$X_{i}^{(iter+1)} = X_{i}^{(iter)} + V_{i}^{(iter+1)}$$

Where:
- $N$: Number of particles in swarm ($i = 1: N$).
- $iter$: Iteration number.
- $V_{i}^{(iter)}$: Velocity of particle ($i$) at iteration ($iter$).
- $X_{i}^{(iter)}$: Position of particle ($i$) at iteration ($iter$).
- $P_{best_{i}}^{(iter)}$: Personal best position of particle ($i$) at iteration ($iter$).
- $G_{best_{i}}^{(iter)}$: Global best position in the swarm at iteration ($iter$).
- $\omega$: Inertia weight factor within the range [0,1].
- $C_{1}$, $C_{2}$: Acceleration coefficients, usually are equal 2.
- $r_{1}$, $r_{2}$: Random numbers within the range [0,1].
- $V_{i}^{(iter+1)}$: Updated velocity of particle ($i$) at next iteration ($iter + 1$).
- $X_{i}^{(iter+1)}$: Updated position of particle ($i$) at next iteration ($iter + 1$).

5.2.1. Pseudocode of PSO Algorithm

Let $f: R^n \rightarrow R$ be the fitness function that must be minimized. The function takes a candidate solution of a vector of $N$ real numbers and produces a real number as output that indicates the fitness function value. The goal is to find the global minimal $x^*$ Fig. 15, [14].

**Step 1. Initialization**

For each particle $i = 1: N$, do

(a) Initialize the particle's position with a uniformly distribution as $X_i(0) \sim U[LB, UB]$, where LB and UB represent the lower and upper bounds of the search space.
(b) Initialize $p_{best}$ to its initial position: $P_{best_{i}}(0) = X_{i}(0)$.
(c) Initialize $g_{best}$ to the minimal value of the swarm: $G_{best}(0) = \arg \min_{X_{i}}[f]$.
(d) Initialize velocity: $V_i \sim [-|LB - UB|, |UB - LB|]$.

**Step 2. Repeat until a termination criterion is met**

For each particle $i = 1: N$, do

(a) Pick random numbers: $r_1, r_2 \sim (0, 1)$.
(b) Update particle's velocity using Eq. (37).
(c) Update particle's position using Eq. (38).
(d) If $\text{fitness}[X_{i}^{(iter)}] < \text{fitness}[P_{best_{i}}^{(iter)}]$ then
   i. Update the best-known position of particle $i$: $P_{best_{i}}^{(iter)} = X_{i}^{(iter)}$
   ii. If $\text{fitness}[X_{i}^{(iter)}] < \text{fitness}[G_{best_{i}}^{(iter)}]$ then
       Update the swarm's best-known position: $G_{best}^{(iter)} = X_{i}^{(iter)}$
(e) $iter \leftarrow (iter + 1)$.

**Step 3. Output $G_{best_{i}}^{(iter)}$ that holds the best-found solution.**

Fig. 15. Pseudocode of PSO algorithm

5.2.2. Advantages of PSO over Another Heuristic Techniques

PSO is one of the modern heuristic algorithms. It is a population-based search algorithm similar to other heuristic optimization techniques. PSO has many advantages as follows [40,41]:

1. PSO is a derivative-free technique like other heuristic optimization techniques.
2. PSO differs from other heuristic optimization approaches in that its principle and code implementation are both simple.
3. PSO less dependent of initial conditions compared to other methods, indicating that the convergence procedure of algorithm is robust.
4. PSO can produce good solutions fast and with more steady convergence characteristics.
5. PSO has the flexibility to be integrated with other optimization algorithms to create hybrid tools.
6. Unlike many other evolutionary approaches, PSO has fewer parameters to adjust. Also, PSO is capable of escaping local minima.
7. PSO is easy to implement since it uses basic mathematics operations.
8. PSO doesn’t need a good initial solution to begin its iterative process.

5.3. Adaptive Weighted Particle Swarm Optimization (AWPSO)

PSO performance in multi-objective optimization problems can be improved using the Adaptive Weighted Particle Swarm Optimization (AWPSO) approach[42]. The following two terms: inertia weight ($\omega$) and acceleration factor ($\alpha$) are utilized to construct adaptive weighted PSO. The inertia weight function is to balance between global and local explorations, it controls the previous velocities' effect on the new velocity. Larger inertia weights allow for more exploration of the search space, whereas lesser inertia weights mean that the search will be more concentrated and restricted to a smaller area of the overall search space. As the number of iterations grows, the acceleration factor value will increase. This modification can help to improve the global search capability and also to flee from the local optimum. The position formula for AWPSO is the same formula used for PSO as Eq. (38). For AWPSO, the velocity equation is given by Eq. (39).

$$V_{i}^{(iter+1)} = \omega V_{i}^{(iter)} + \alpha^{(iter)} (C_{1}r_{1}(P_{best_{i}}^{(iter)} - X_{i}^{(iter)}) + C_{2}r_{2}(G_{best_{i}}^{(iter)} - X_{i}^{(iter)}))$$

$$X_{i}^{(iter+1)} = X_{i}^{(iter)} + V_{i}^{(iter+1)}$$
The inertia weight formula is as in Eq. (40) which makes \( \omega \) value changes randomly from \((\omega_o, 1)\)
\[
\omega = \omega_o + r(1 - \omega_o) \tag{40}
\]
The acceleration factor formula is as in Eq. (41) which makes \( \alpha \) changes exponentially from \( \alpha_o \) to \((1 + \alpha_o)\).
\[
\alpha^{\text{iter}} = \alpha_o + \frac{\beta^\text{iter} - \alpha_o}{\beta_{\text{iter}} - \alpha_o} \tag{41}
\]
Where:
\[
\omega_o: \text{ The initial positive constant in the interval chosen from } [0,0.5] \\
\omega_{\text{max}}: \text{ Inertia Weight Factor value changes randomly from } \omega_o \text{ to } 1 \\
\alpha_{\text{iter}}: \text{ Is the initial positive constant in the interval } [0.5, 1].
\]
\[
\alpha_c: \text{ Determines the maximum value of inertia weight factor} \\
\alpha_{\text{min}}: \text{ Minimum value of inertia weight factor} \\
\alpha_{\text{max}}: \text{ Maximum value of inertia weight factor} \\
\alpha_o: \text{Initial value of inertia weight factor} \\
\omega_{\text{min}}, \omega_{\text{max}}: \text{ Determined with respect to } (\omega_{\text{min}}, \omega_{\text{max}}) \text{ by Eq. (45)} \\
C_{10}, C_{20}: \text{ Initial values of acceleration coefficients} \\
C_{1}^{(\text{iter})}, C_{2}^{(\text{iter})}: \text{ Adaptive acceleration coefficients given by Eqs. (46) and (47)}
\]
\[
\beta^\text{iter} = \beta_{\text{iter}} \text{ Determined based on the fitness value of } (G_{\text{best}}, \text{ and } P_{\text{best}}), \text{ at iteration } (\text{iter}), \text{ given by Eq. (49)}
\]
\[
\beta_{\text{iter}} = \frac{P_{\text{iter}}}{G_{\text{iter}}} \tag{49}
\]

5.4. Adaptive Accelerated Coefficients Based PSO (AACPSO)

The time-varying inertia weight (TVIW) which is relied on Eq. (42) can find a suitable solution much faster. However, due to a lack of diversity towards the end of the search, its capacity to fine tune the optimal solution is limited. Most experts agree that in PSO, problem-based adjustment of parameters may be a crucial aspect in locating the most accurate and efficient solution [41]. New studies have developed to improve PSO Algorithms, by includingTV-Varying Acceleration Coefficients (TVAC) in which \( C_{1} \) and \( C_{2} \) vary linearly with time as demonstrated in Eq. (39), [44]. As a result of using adaptive coefficients, the cognitive component decreases while the social component increases as search progresses [45].

\[
\alpha_{\text{iter}} = \omega_{\max} \frac{\beta^\text{iter} - \alpha_o}{\beta_{\text{iter}} - \alpha_o} \tag{42}
\]

To deal with the inertia weight and acceleration factors, a new approach called Adaptive Accelerated Coefficients (AAC) will be proposed for the PSO algorithm implementation [46]. The new strategy aims to vary acceleration coefficients and inertia weight exponentially over time within the range of their minimum and maximum limits. The choice of the exponential function is justified by the growing or decreasing speed of such a function to speed up the algorithm’s convergence process and encourage greater exploration of the search space. Furthermore, \( C_{1} \) and \( C_{2} \) vary adaptively according to the fitness value of \((G_{\text{best}})\) and \((P_{\text{best}})\), and Eq. (39) becomes as Eq. (43) [47]:

\[
V_{i}^{(\text{iter} + 1)} = \omega^{(\text{iter})} V_{i}^{(\text{iter})} + C_{1}^{(\text{iter})} r_1 (P_{\text{best}}^{(\text{iter})} - X_{i}^{(\text{iter})}) + C_{2}^{(\text{iter})} r_2 (G_{\text{best}}^{(\text{iter})} - X_{i}^{(\text{iter})}) \tag{43}
\]

\[
\omega^{(\text{iter})} = \omega_{o} e^{-\alpha_{\text{iter}}^\text{iter}} \tag{44}
\]

\[
\alpha_{\text{iter}} = \frac{1}{\beta_{\text{iter}}} \ln \left( \frac{\omega_{\text{max}}}{\omega_{\text{min}}} \right) \tag{45}
\]

\[
C_{1}^{(\text{iter})} = C_{10} e^{-\alpha_{\text{iter}} r_1 (P_{\text{best}}^{(\text{iter})} - X_{i}^{(\text{iter})})} \tag{46}
\]

\[
C_{2}^{(\text{iter})} = C_{20} e^{-\alpha_{\text{iter}} r_2 (G_{\text{best}}^{(\text{iter})} - X_{i}^{(\text{iter})})} \tag{47}
\]

\[
\beta_{\text{iter}} = \frac{P_{\text{iter}}}{G_{\text{iter}}} \tag{49}
\]

Where:
\[
\omega^{(\text{iter})}: \text{Adaptive inertia weight factor given by Eq. (44)}
\]

\[
\alpha_c: \text{ Determines with respect to values of } (C_{10}, C_{20}) \text{ by Eq. (48)}
\]

\[
P_{\text{iter}}: \text{The mean value of the best position } (P_{\text{best}}), \text{ related to all particles at current iteration } (\text{iter})
\]

\[
R_{\text{iter}}: \text{Determined based on the fitness value of } (G_{\text{best}}, \text{ and } P_{\text{best}}), \text{ at iteration } (\text{iter}), \text{ given by Eq. (49)}
\]

5.5. Modified Adaptive Accelerated Coefficients PSO(MAACPSO)

Modified Adaptive Accelerated Coefficients Based PSO (MAACPSO) technique is similar to (AACPSO) technique but it only differs that (MAACPSO) calculate the acceleration coefficient \( C_{1}^{(\text{iter})}, C_{2}^{(\text{iter})} \) through the Eq. (51) [15]:

In this study we assumed the \( C_{i} = 4.5 \) in Eq. (50), which is the sum of accelerated coefficients \( (C_{1}^{(\text{iter})}, C_{2}^{(\text{iter})}) \), to make MAACPSO algorithm get the optimal solution faster than in AACPSO and get better results.

\[
C_{1}^{(\text{iter})} + C_{2}^{(\text{iter})} = C_{10} \tag{50}
\]

\[
C_{2}^{(\text{iter})} = 4.5 - C_{1}^{(\text{iter})}, \text{ and } C_{1}^{(\text{iter})} = C_{10} e^{-\alpha_{\text{iter}} \beta_{\text{iter}}} \tag{51}
\]

5.6. Crazy Particle Swarm Optimization (C-PSO)

Crazy PSO structure is based on how a swarm of birds or fish moves to hunt food. A flock of birds will distribute or fly together to determine the best food resources. The movement that’s taking place which affects on the direction wherein the swarm will move towards the best food supply.. As a result of the exploring process, the most effective food source is discovered by one of the birds within the group. In the elementary PSO algorithm, a swarm contain \( i \) particles are impacted by the following three parameters: inertia, local best position and global best position [48]. Because numbers \( r_1 \) and \( r_2 \) are achieved at random manner, they could be excessively large or small. Random numbers with excessively high values will impact on local and global experiences, therefore pushing the particles away from the local solution. also, if these two random numbers are excessively small, the two searching experiences will not be properly utilized to decrease the convergence rate. A new approach proposed by Roy & Ghoshal in 2008 [16], this approach called a novel Crazy Particle Swarm Optimization. A craziness operator is used in Crazy PSO to ensure that each particle has a craziness probability, which helps to keep the particle population diverse. The velocity and position of each particle changes according to the Eqs. (52) and (59), [49].

\[
V_{i}^{(\text{iter} + 1)} = r_2 \text{sign}(r_2) V_{i}^{(\text{iter})} + (1 - r_2) C_{1}^{(\text{iter})} r_1 (P_{\text{best}}^{(\text{iter})} - X_{i}^{(\text{iter})}) + (1 - r_2) C_{2}^{(\text{iter})} (G_{\text{best}}^{(\text{iter})} - X_{i}^{(\text{iter})}) \tag{52}
\]
Prior to performing position update, particle velocity is crazed by Eq. (54):
\[V_i^{(iter+1)} = V_i^{(iter+1)} + P_c(r_4) \text{sign}(r_4) * V_i^{(craziness)}\]

Where \(P_c(r_4)\) and \(\text{sign}(r_4)\) are defined respectively as Eqs. (55) and (56):
\[
P_c(r_4) = \begin{cases} 1 & r_4 \leq P_{\text{craz}} \\ 0 & r_4 > P_{\text{craz}} \end{cases}
\]
\[
\text{sign}(r_4) = \begin{cases} -1 & r_4 \leq 0.5 \\ 1 & r_4 > 0.5 \end{cases}
\]

\(P_{\text{craz}}\) and \(\omega^{(iter)}\) are defined respectively as in Eqs. (57) and (58):
\[
P_{\text{craz}} = \omega_{\text{min}} - e^{\left(-\frac{\omega^{(iter)}}{\omega_{\text{max}} - \omega_{\text{min}}}\right)} (\text{Maxiter} - \text{iter}) + \omega_{\text{min}}
\]
\[
\omega^{(iter)} = (\omega_{\text{max}} - \omega_{\text{min}}) * \frac{(\text{Maxiter} - \text{iter})}{\text{Maxiter}} + \omega_{\text{min}}
\]

\(r_1, r_2, r_3, r_4\) : Are uniform distributed random vector within \[0,1\]
\(V_i^{(craziness)}\) : Is a random parameter which is uniformly chosen from the interval \([V_i^{\text{min}}, V_i^{\text{max}}]\)
\(P_{\text{craz}}\) : Is a predefined probability of craziness given by Eq. (57).
\(\omega^{(iter)}\) : Time-varying inertia weight given by Eq. (58).

The next position is update based on Eq. (59):
\[
X_i^{(iter+1)} = X_i^{(iter)} + V_i^{(iter+1)}
\]

### 5.7 Phasor Particle Swarm Optimization (PPSO)

The adaptive PSO algorithms update the weighted factor and acceleration coefficients at every iteration. This update caused increased optimization effectiveness despite the increase in problem dimension, which resulted in an increase in the degree of complexity and limitation of actual life problems. Therefore, a novel straightforward adaptive model for PSO, referred to as phasor particle swarm optimization (PPSO) is implemented as in [17]. It employs a phase angle \(\theta\), to model the particle control parameters, the technique draws its inspiration from phasor theory in mathematics. The using of phase angle \(\theta\) theorem converts the PPSO algorithm into a self-adaptive, trigonometric, balanced and nonparametric meta-heuristic method with simpler computations [17].

### 5.7.1 Working Mechanism of (PPSO)

Unlike other PSO techniques, the control parameters of PPSO are modeling using phase angle \(\theta\). PPSO transforming these parameters into a function of \(\theta\), to accomplish different optimization techniques. In order to achieve this goal, every particle has been defined through a one-dimensional phase angle \(\theta_i\) in order to achieve that, a magnitude vector \(X_i\) based on the phase \(\theta_i\) can be used to represent every particle \(i\) as in \((X_i = |X_i|e^{j\theta_i})\). By using the phase angles of the particles, PPSO produces adaptive search characteristics. This technique maintains a balance between global and local search, resulting in an algorithm that is both adaptive and nonparametric in structure. The phase angle varying prevent the algorithm from an early convergence to a local best solution by fast or slowing rise of \(P_{\text{best}}(\theta_i^{(iter)})\) and \(G_{\text{best}}(\theta_i^{(iter)})\), in the same or opposite direction [17].

Fig. 16 demonstrates the flow of the PPSO algorithm, which really is analogous to the other PSO algorithms. As a preliminary step of PPSO, a large number of N particles (initial population) \(X_i = |X_i|e^{j\theta_i}\) for \(i = 1:N\) are formed randomly inside the dimensional space that contains the problem, each particle has a unique phase angle \(\theta_i\). This initialization step produced as follows: through uniform distribution \(\theta_i^{(iter)} = U(0,2\pi)\), and with initial velocity limit \((V_{\text{max},i}^{(iter)})\). Next, the following Eqs. (60) is used to update velocity of each particle at every iteration as follows:
\[
V_i^{(iter+1)} = (\cos(\theta_i^{(iter)})^2\sin(\theta_i^{(iter)})/N) V_i^{(iter)} + \cos(\theta_i^{(iter)})^2\sin(\theta_i^{(iter)}) N
\]

5.8. Gravitational Search Algorithm with PSO (GSA-PSO)

E. Rashedi proposed the GSA algorithm, a new heuristic optimization method, in 2009 [50]. The fundamental physical theory from which GSA draws its inspiration is Newton's principle, according to which every particle in the cosmos is attracted to every other particle [51]. GSA are frequently viewed as a collection of agents (possible solutions) whose mass is proportional to their fitness function value. All masses are attracted to one another during the iteration process by the gravitational forces that exist between them, the greater the gravitational pull. As a result, heavier masses that are closer to the global optimum tend to attract smaller masses [50].

A new hybrid population-based algorithm (GSA-PSO) is developed with the mixture of Particle Swarm Optimization (PSO) and Gravitational Search Algorithm (GSA) [18]. To summarize, the primary idea behind GSA-PSO is to combine PSO's capacity to global search \(G_{\text{best}}\) with GSA's local search ability. To combine these algorithms Eq. (74) is proposed [18]:

\[
\frac{X_i^{(iter+1)} = X_i^{(iter)} + V_i^{(iter+1)}}{\text{Eq. (74)}}
\]
5.8.1. Working Mechanism of (GSA-PSO)

GSA-PSO algorithm is composed of the following steps as outlined [18]:

Step 1. Load System Parameters
Load objective function, parameters of antenna system, initial condition and specifies upper and lower limits.

Step 2. (Generate Initial Population.)
Consider a system in which there are N agents. All agents in search space are initially set up randomly. Each agent is taken into consideration as a possible solution.

Step 3. (Evaluate Fitness of Each Agent and Update Best Fitness).
While (Iter < MaxIter), do
Evaluate: fitness\(_i\)\(_{(\text{Iter})}\) = fitness\(_i\)\(_{(\text{Iter})}\), and fitness\(_{(\text{Gbest\_Iter})}\) = min\left\{\text{fitness}\(_i\)\(_{(\text{Iter})}\)\right\} (64)

Step 4. (Update (GSA-PSO) Algorithm Parameters).
Update gravitational constant, best fitness, worst fitness and inertia mass should be updated after each iteration in the following manner:
For the search accuracy to be controlled, the gravitational constant \(G_{(\text{Iter})}\) is established at the starting of the algorithm and is decreased with each iteration. Hence, the gravitational constant \(G_{(\text{Iter})}\) which is a function of initial value of gravitational constant \(G_0\), descending coefficient \(\alpha\) and iteration number is given by Eq. (65) as follows:
\[
G_{(\text{Iter})} = G_0 \times e^{-\alpha \cdot \text{Iter} / \text{MaxIter}}
\] (65)
Where:
\(G_{(\text{Iter})}\) : Gravitational constant.
The active gravitational mass (\(M_a\)), passive gravitational mass (\(M_p\)) and inertial mass of agent \(i\) (\(M_i\)), are calculated through fitness evaluation. According to Newton's laws of motion and attraction, a heavier mass moves more slowly. Consequently, a greater mass in GSA represents the most effective solution. All masses are assumed to be equal as in Eq. (66) to update the inertia mass \(M_i\) as in Eq. (68):
\[
M_{ai} = M_{pi} = M_{ii} = M_i
\] (66)
\[
m_{i(\text{Iter})} = \frac{\text{fitness}_{i(\text{Iter})} - \text{worst}_{(\text{Iter})}}{\text{best}_{(\text{Iter})} - \text{worst}_{(\text{Iter})}}
\] (67)
\[
M_{i(\text{Iter})} = \frac{m_{i(\text{Iter})}}{\sum_{j=1}^{N} m_{j(\text{Iter})}}
\] (68)
n\(\text{fitness}_{i(\text{Iter})}\) : The fitness value of agent \(i\) at iteration, \(\text{Iter}\).
n\(\text{worst}_{(\text{Iter})}\) : Worst fitness value in the current population at iteration, \(\text{Iter}\).
n\(\text{best}_{(\text{Iter})}\) : Best fitness value in the current population at iteration, \(\text{Iter}\).
n\(m_{i(\text{Iter})}\) : Mass of agent \(i\) at iteration, \(\text{Iter}\).
\(M_{i(\text{Iter})}\) : Inertial mass of the agent \(i\) at iteration, \(\text{Iter}\). For minimization problem, the \(\text{best}_{(\text{Iter})}\) and \(\text{worst}_{(\text{Iter})}\) fitness values are defined as in Eqs. (69) and (70):
\[
\text{best}_{(\text{Iter})} = \min \text{fitness}_{i(\text{Iter})}, i \in [1, N]
\] (69)
\[
\text{worst}_{(\text{Iter})} = \max \text{fitness}_{i(\text{Iter})}, i \in [1, N]
\] (70)

Step 5. (Calculation of the Gravitational Force and Total Force).
The gravitational force, \(F_{ij}\) , acting on agent \(i\) from agent \(j\) at iteration, \(\text{Iter}\), can be computed as follows by Eq. (71):
\[
F_{ij}^{\text{Iter}} = G_{(\text{Iter})} \frac{m_{i(\text{Iter})} \cdot m_{j(\text{Iter})}}{R_{ij}^{\text{Iter}} + \varepsilon} \left( X_{j}^{\text{Iter}} - X_{i}^{\text{Iter}} \right)
\] (71)
The total force \(F_{i}^{\text{Iter}}\) that acts on agent \(i\), among all agents is calculated as the following Eq. (72):
\[
F_{i}^{\text{Iter}} = \sum_{j=1}^{N} \text{rand}_{j} \cdot F_{ij}^{\text{Iter}}
\] (72)
Where:
\(R_{ij}^{\text{Iter}}\) : Euclidian distance between two agents \(i\) and \(j\).
\(\varepsilon\) : Small constant number
\(X_{i}^{\text{Iter}}\) : Position of agent \(i\), at iteration, \(\text{Iter}\).
\(X_{j}^{\text{Iter}}\) : Position of agent \(j\), at iteration, \(\text{Iter}\).
\(\text{rand}_{j}\) : Random number in the interval [0,1].
\(N\) : Population size

Step 6. (Calculate the Acceleration of Agents).
The acceleration of an agent is inversely proportional to its mass, and the result force is proportional to its acceleration,
consequently the acceleration of each agent \((a_{c_i}^{(iter)})\) can be calculated using Eq. (73):

\[
a_{c_i}^{(iter)} = \frac{F_i^{(iter)}}{M_i^{(iter)}}
\]

Step7. (Calculate Agents’ Velocity and Update Its Position).

The next velocity of agents is calculated as follow by Eq. (74):

\[
V_i^{(iter+1)} = \omega \ast V_i^{(iter)} + C_1 \ast r_1 \ast a_{c_i}^{(iter)} + C_2 \ast r_2 (G_{best}^{(iter)} - X_i^{(iter)})
\]

The next position of agents is updated as follow by Eq. (75):

\[
X_i^{(iter+1)} = X_i^{(iter)} + V_i^{(iter+1)}
\]

Where:

- \(G_{best}^{(iter)}\) : Global best solution at iteration, \((Iter)\).
- \(\omega\) : Weighting function, take as rand \((0,1)\)
- \(C_1, C_2\) : Weighting factors.

Step8. (Stopping criterion).

Check stopping criterion:

If the stopping criterion is reached, \((Iter = MaxIter)\) go to Step9.

else go to Step 3.

Step9. (Output Optimal Solution).

Finally, if (GSA-PSO) meets an end criterion, it will be terminated and the most effective agents will be printed.

The flowchart of the (GSA-PSO) algorithm is represented in Fig. 17.

Some observations are made to demonstrate how effective (GSA-PSO) is. The quality of solutions (fitness) is addressed in the updating mechanism in (GSA-PSO). The agents near good solutions try to attract the other agents which are exploring the search space. When all agents are close to a good solution, they move very slowly. In this case, the \((G_{best})\) assist them to exploit the global best. GSA-PSO use memory \((G_{best})\) to store the best solution has found so far, so it is reachable anytime. Each agent can observe the best solution so far and tend toward it. The abilities of global search and local search may be modified by varying \((C_1, C_2)\) [18].

5.9. Eagle Strategy with Particle Swarm Optimization (ES-PSO)

Eagle strategy (ES) is a metaheuristics optimization approach inspired by the foraging behaviors of eagles such as golden eagles which created by Xin-She Yang and Suash Deb [52]. The two-stage strategy of an eagle’s foraging behavior as follows: Let us start by supposing the Lévy walk is performed by an eagle. Once it finds a prey (promising solution) it switches to a chase strategy (local search stage). Second, the chase strategy can be defined as an intensive local search. Searching intensively in a small area using any optimization technique such as Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Artificial Bee Colony, Genetic Algorithm (GA), Simulated Annealing (SA), Cuckoo Search Algorithm (CSA), Flower Pollination Algorithm (FPA), and Firefly Algorithm (FA)Flower Pollination Algorithm (FPA), and Firefly Algorithm (FA) [53]. This method permits the combination of the advantages of many algorithms to achieve superior results [54].

Eagle strategy with particle swarm optimization (ES-PSO) technique is a metaheuristic technique that iteratively for the optimal solution, (ES-PSO) is a two-stages strategy: global search stage and local search stage [19]. Firstly, global search uses Lévy flight walks to explore the whole search space, once a good solution is identified, the search shifts to a local search stage where the PSO algorithm does an intensive local search, then the procedure re-starts with a fresh global search within a new searching space area. The global search and local search stages starts to iterate until meeting criterion, in this work stopping criterion is maximum number of iterations \((MaxIter)\), each iteration will be performed in a new searching area. In reality, several algorithms are frequently utilized at various stages. So long as the benefits of these many algorithms are combined, it should produce superior results. ES-PSO employs Lévy walks to better explore the global search space. Lévy distribution is given by Eq. (76) [54]:

\[
\alpha_{c_i}^{(iter)} = \frac{F_i^{(iter)}}{M_i^{(iter)}}
\]

\[
V_i^{(iter+1)} = \omega \ast V_i^{(iter)} + C_1 \ast r_1 \ast a_{c_i}^{(iter)} + C_2 \ast r_2 (G_{best}^{(iter)} - X_i^{(iter)})
\]

\[
X_i^{(iter+1)} = X_i^{(iter)} + V_i^{(iter+1)}
\]
\[ L(s) = \frac{\Gamma(\lambda) \sin(\pi \lambda/2)}{\lambda^{\lambda/2 + 1}} \]  

(76)

Where:
- \( L(s) \): Lévy distribution function.
- \( \Gamma(\lambda) \): Standard Gamma function.
- \( \lambda \): Gamma function parameter.
- \( s \): Step length.
- \( \lambda = 2 \): Lévy walks become the Cauchy Distribution.
- \( \lambda = 3 \): Lévy walks become Brownian Motion as a special case.

We used the PSO parameters that are often used in other applications [55,56], where \( C_1 = C_2 = 2 \), \( \omega_{\text{min}} = 0.4 \), and \( \omega_{\text{max}} = 0.9 \); then we set \( \lambda = 2 \). There are two critical conditions: \( \Gamma \to \infty \) and \( \Gamma \to 0 \). If \( \Gamma \to \infty \), the velocity of particles can’t be reduced and particles and the distance between them is large. Whereas, if \( \Gamma \to 0 \), then the particles are short sighted, so particles are going to be trapped in a restricted space because their velocities are so small.

5.9.1. Working Mechanism of (ES-PSO)

Steps of the proposed (ES-PSO) method have been explained as follows:

**Step1.** (Load System Parameters),
Load objective function and loads parameters of antenna system and initial condition and specifies upper and lower limits

**Step2.** (Generate Initial Population Randomly)
Initial particles are determined randomly via uniform distribution to locate their initial positions, then the initial velocities of particles are constituted.

**Step3.** (Global Search Stage)

**White** (Iter < MaxIter) do
Performing random global search using Lévy Flight by Eq. (77):
\[ X_i^{(\text{iter+1})} = X_i^{(\text{iter})} + aL(s,\lambda) \]  

and set \( (\lambda = 1.5, \alpha = 1, \text{and step length}, s, \text{set as } s = 5) \), then Evaluate fitness and update best fitness in global stage by Eq. (78):

**If** \( \text{fitness}(X_i^{(\text{iter+1})}) < \text{fitness}(X_i^{(\text{iter})}) \)
**Best global fitness = fitness}(X_i^{(\text{iter+1})})
\[ Gbest_i = X_i^{(\text{iter+1})} \]  

(78)

**else**, \( \text{Best global fitness = fitness}(X_i^{(\text{iter})}) \)
\[ Gbest_i = X_i^{(\text{iter})} \]

Update Best Position in Global Search Stage by (79):
**Best global position = Gbest_i**

(79)

Then, find a promising solution (Best global position) go to **Step4.**

**Step4.** (Switch Between Global Search and Local Search Stages)
The global and local search stages are controlled by switching parameter \( p \) such as in Eq. (80). Besides, establish a random number (In this paper set \( p = 0.2 \))

**If** \( p < \text{rand} \)
**Switch to local search stage (go to Step5.)**

**else,** Switch to global search stage (go to Step

**Step5.** (Intensive Local Search Stage)
In intensive local search stage, search around a promising solution, Calculate new velocity and position of each particle via Eqs. (81) and (83) as follows:
\[ V_i^{(\text{iter+1})} = \omega^{(\text{iter})}V_i^{(\text{iter})} \]
\[ + C_1r_1(Pbest_i^{(\text{iter})} - X_i^{(\text{iter})}) \]
\[ + C_2r_2(Gbest_i^{(\text{iter})} - X_i^{(\text{iter})}) \]  

(81)

Where:
\[ \omega^{(\text{iter})} = \omega_{\text{max}} \frac{\text{iter}(\omega_{\text{max}}-\omega_{\text{min}})}{\text{Maxiter}} \]  

(82)

The new position of each particle is updated as follow by Eq. (83):
\[ X_i^{(\text{iter+1})} = X_i^{(\text{iter})} + V_i^{(\text{iter+1})} \]  

(83)

Then, evaluate new fitness and update best fitness in local search stage by Eq. (84) as follows:

**If** \( \text{fitness}(X_i^{(\text{iter+1})}) < \text{fitness}(Pbest_i^{(\text{iter})}) \)
**Best local fitness = fitness}(X_i^{(\text{iter+1})})
\[ Pbest_i = X_i^{(\text{iter+1})} \]  

(84)

**else,** \( \text{Best local fitness = fitness}(Pbest_i^{(\text{iter})}) \)
\[ Pbest_i = X_i^{(\text{iter})} \]

Update Best Position in Local Search Stage by Eq. (85):
**Best local position = Pbest_i**

(85)

**Step6.** (Update Global Best Fitness and Global Best Position in the Overall (ES-PSO) Strategy)
The global best fitness and global best position in (ES-PSO) strategy updated through the Eq. (86):

**If** \( \text{Best local fitness} < \text{Best global fitness} \)
**Global fitness = Best local fitness**
**Global position = Best local position**  

(86)

**else,** \( \text{Global fitness = Best global fitness} \)
**Global position = Best global position**

(Update Optimal Solution in The Strategy):
**Optimal Solution = Global position**

**Step7.** (Update Iteration).
\[ \text{Iter} = \text{Iter} + 1 \]

**Step8.** (Stopping Criterion).
Check stopping criterion:

**If** the stopping criterion is reached, \( \text{Iter} = \text{MaxIter} \) go to **Step9.**
else go to Step 3.

**Step 9.** (Output Optimal Solution).

Finally, if (ES-PSO) meets an end criterion, it will be terminated and the most effective particles will be printed.

$p$ is the only parameter that controls the transition between local search and global search. $p$ determines when to perform
iteration process. The algorithm utilized for the global exploration should have enough randomness in order to explore the search space distinctly and effectively. This procedure is usually sluggish at first, but it should speed up as the system converges (or no better solutions can be found after a certain number of iterations). On the other hand, the algorithm used for the intensive local exploitation must be an efficient local optimizer. The goal is to find the local best solution as fast as possible while using the minimum amount of function evaluations. This stage must be fast and efficient [54]. The flowchart of the (ES-PSO) methodology, which summarized the strategy steps is represented in Fig. 18.

The ES-PSO strategy significantly outperforms the other metaheuristic methods. ES-PSO, in contrast to other techniques, ES-PSO uses the parameter p until it converges as close to the optimal solution as possible, then turned from global search mode to local search mode. In addition, the uniform distribution function in MATLAB is used to generate a random initial population. Consequently, fitness value changes from worst to best as you move from local to global search, you can not only locate an effective search but also switch to a different search region within the solution space [19].

5.1. Results and Discussions

The results are performed systematically in MATLAB/Simulink environment at the computer including a core-i5 1.8 GHz CPU and 4 GB RAM. The results are obtained from both integral based objective function (J1) (single objective function) given by (33) and dynamic performance indices based objective function (J2) (multi objective function) given by (34).

Many evolutionary techniques are proposed for optimal tuning of controller parameters. In this research, Particle Swarm Optimization (PSO), Adaptive Weighted Particle swarm optimization (AWPSO), Adaptive Acceleration Coefficients Particle Swarm Optimization (ACACPSO), Modified Adaptive Acceleration Coefficients Particle Swarm Optimization (MACACPSO), Crazy Particle Swarm Optimization (C-PSO), Phasar Particle Swarm Optimization (PPSO), Gravitational Search Algorithm with Particle Swarm Optimization (GSA-PSO) and Eagle Strategy with Particle Swarm Optimization (ES-PSO) are proposed for optimal tuning of Proportiona n-Integral-Derivative (PID), Fuzzy PID (FPID), Variable Coefficient PID (V-PID), Fractional Order PID (FOPID) and Variable Coefficient Fractional Order PID (V-FOPID) controllers. The numbers of population size and maximum iterations are chosen as 25 and 100, respectively for all optimization algorithms. Also, Self-Tuning Fuzzy PID (STF-PID) controller and Self Tuning Fuzzy FOPID (STF- FOPID) are proposed for satellite tracking control systems.

The settling time (t_s), rise time (t_r), overshoot percentage (O%) are measured by MATLAB/Simulink for 10 sec of simulation running. Additionally, plots the tracking process, steady state error and control action signal to compare between proposed control strategies and evolutionary techniques.

The results obtained from each objective function used to compare between PSO, AWPSO, AACPSO, MACACPSO, C-PSO, PPSO, GSA-PSO, ES-PSO techniques to select the best technique in terms of overshoot, steady state error, control action and settling time. Also, compare between single objective function and multi objective function. Finally collect the best results obtained from PID, FOPID, FPID, STF-PID, STF-FOPID, V-PID and V-FOPID controllers to compare between the performance in terms of the least overshoot, the minimum steady state error, the lowest control action and the fastest settling time.

| Table 6 Results obtained from satellite tracking system with optimal FOPID controller based on different evolutionary optimization |
|---|
| FOPID controller for single objective function: \( J_1 = \int_0^\infty |e(t)| \, dt \) |
| Algorithm | \( K_p \) | \( K_i \) | \( K_d \) | \( \mu \) | \( \lambda \) | \( OS \) (%), \( t_r \) (s), \( t_s \) (s) |
| PSO | 0.77 | 0.16 | 0.93 | 1.5 | 31.6 | 0.0 | 0.4 |
| AWPSO | 0.10 | 0.17 | 0.87 | 1.6 | 16.2 | 0.2 | 0.4 |
| O | 0.81 | 0.80 | 1.56 | 1.4 | 32.0 | 0.3 | 0.3 |
| AACPSO | 0.75 | 0.15 | 0.55 | 2.1 | 4.5 | 0.4 | 0.7 |
| MAACPSO | 0.80 | 0.17 | 0.65 | 1.2 | 1.4 | 0.6 | 0.3 |
| C-PSO | 0.77 | 0.17 | 0.45 | 1.2 | 1.4 | 0.8 | 0.3 |
| P-PSO | 0.78 | 0.18 | 0.57 | 2.1 | 4.5 | 0.4 | 0.7 |
| GSA-PSO | 0.80 | 0.17 | 0.65 | 1.2 | 1.4 | 0.6 | 0.3 |
| SO | 0.81 | 0.18 | 0.55 | 2.1 | 4.5 | 0.4 | 0.7 |
| PSO | 0.75 | 0.15 | 0.55 | 2.1 | 4.5 | 0.4 | 0.7 |

| Table 7 Results obtained from satellite tracking system with optimal PID controller based on different evolutionary optimization techniques. |
|---|
| PID controller for single objective function: \( J_1 = \int_0^\infty |e(t)| \, dt \) |
| Algorithm | \( K_p \) | \( K_i \) | \( K_d \) | \( N \) | \( OS \) (%), \( t_r \) (s), \( t_s \) (s) |
| PSO | 792.1 | 1000 | 620.0 | 59.3 | 53.2 | 40.14 | 0.96 | 0.8 |
| 68 | 49 | 4 | 8 | 79 |
| AWPSO | 1000 | 761.7 | 57.7 | 42.56 | 0.06 | 0.8 |
| 02 | 9 | 5 | 30 |
| AACPSO | 1000 | 589.0 | 49.9 | 39.67 | 0.07 | 0.7 |
| 0 | 16 | 4 | 6 | 35 |
Fig. 21. System steady state error with PID controller

Fig. 22. System steady state error with PID controller

6.1. PID Controller

Results obtained from satellite tracking system with optimal PID controller based on different evolutionary techniques represented in Table 7.

For single objective function, the PID controller achieved the lowest rise time when tuned by (MAACPSO). Also, the minimum overshoot percentage achieved when tuned PID by (GSA-PSO). However, the fastest settling time achieved when tuning PID using (ES-PSO). For multi objective function, the PID controller achieved the lowest rise time when tuned by (MAACPSO). The minimum overshoot percentage achieved when tuned PID by (C-PSO). However, the fastest settling time achieved when tuning PID using ES-PSO. It clearly that, the (ES-PSO) achieved the fastest settling time in addition to the rise time, steady state error and overshoot percentage very closed to the best results that achieved by other tuning algorithms for both objective functions.

The plots of tracking system response, steady state error and controller output signal for PID controller represented in Fig. 19 through Fig. 24.

The above plots concluded that, the (PPSO), (GSA-PSO) and (ES-PSO) obtained the best results compared to other optimization techniques. The multi objective function achieved high smooth tracking capability with smooth steady state error and lower control action than single objective function for the early running time which implies that the tracking process very fast and smooth with minimum deviations from the desired position at earlier running time, leading to a highly continuous communications between earth station and desired satellite. Also, it clearly seen that the control action values at beginning of running in the case of multi objective function are less than case of single objective function which implies that the actuator doesn’t reach its saturation limits.

6.2. Fractional Order PID (FOPID) Controller

Results obtained from satellite tracking system with optimal FOPID controller based on different evolutionary optimization techniques represented by Table 6. When the new searching range \( K_p \in [0, 400] \) is used with multi objective function \( J_2 \) for (PPSO), (GSA-PSO) and (ES-PSO) optimization methods, it gives results better than the standard searching range \( K_p \in [0, 1000] \).

For single objective function the FOPID controller achieved the lowest rise time when tuned by (C-PSO). The minimum overshoot percentage achieved by tuning FOPID controller using (MAACPSO), also the lowest control action achieved by (PSO). The least steady state error and the fastest settling time achieved when tuning parameters of fractional order PID controller using (ES-PSO) strategy. For multi objective function the FOPID controller doesn’t achieve desired...
performance for the standard searching range \(K_p \in [0 \text{ to } 1000]\) with (PPSO), (GSA-PSO) and (ES-PSO) methods. So, it's important to find a solution to achieve the predicted performance by tuned acceleration coefficient of the algorithms or changing the searching range of tuned parameters. A new searching range for proportional gain of FOPID controller \((K_p \in [0 \text{ to } 400])\) is used with (PPSO), (GSA-PSO) and (ES-PSO) which lead to improve the performance of FOPID in terms overshoot percentage, control action and settling time. By neglecting the old results of (PPSO, GSA-PSO, ES-PSO), the FOPID achieved the lowest rise time when tuned by (AACPSO). The minimum overshoot percentage achieved by tuning FOPID using (PPSO_N). However, the new searching range improves the settling time, overshoot percentage and control action, the sum of steady error during running time increased compared to the old range. It’s clearly that the (ES-PSO_N) achieved the fastest settling time, in addition to the rise time, steady state error and overshoot percentage very closed to the best results that achieved by other tuning algorithms for each objective functions.

The plots of tracking system response, steady state error and controller output signal for FOPID controller are represented in Fig. 25 to Fig. 30.

The above plots concluded that, the (PPSO), (GSA-PSO) and (ES-PSO) obtained the best results compared to other optimization techniques. The multi objective function achieved high smooth tracking capability with smooth steady state error and lower control action than single objective function for the early running time which implies that the tracking process very fast and smooth with minimum deviations from the desired position at earlier running time leading to a highly continuous communications between earth station and desired satellite. Also, it clearly seen that the control action values at beginning of running in the case of multi objective function are less than case of single objective function which implies that the actuator doesn’t reach its saturation limits. For multi objective function by applying the new searching range on (PPSO), (GSA-PSO) and (ES-PSO), the tracking process highly improved which implies that the FOPID could track the desired satellite trajectory very fast with smooth tracking process and minimum error values at start of tracking process. Also, achieved faster settling time and minimum overshoot percentage values.

### 6.3. Fuzzy PID (FPID) Controller

Results obtained from satellite tracking system with optimal fuzzy PID (FPID) controller based on evolutionary techniques displayed in Table 8

| Algorithm  | \(K_e\) | \(K_d\) | \(K_{PI}\) | \(K_{PD}\) | \(OS\) (%) | \(t_r\) (s) | \(t_s\) (s) |
|------------|--------|--------|--------|--------|---------|--------|--------|
| PSO        | 1000   | 74.9   | 100    | 500    | 10.01   | 0.16   | 0.70   |
| MAACP      | 1000   | 76.5   | 100    | 600    | 3.79    | 0.15   | 0.55   |
| SO         | 5      | 5      | 9      |        |         |        |        |
| P-PSO      | 992.   | 78.0   | 4.97   | 539    | 4.92    | 0.16   | 0.43   |
| GSA-       | 1000   | 82.8   | 95.7   | 500    | 3.13    | 0.17   | 0.41   |

For single objective function the PID controller achieved the minimum overshoot percentage and the lowest control by (GSA-PSO). Also, the lowest rise time and the fastest settling time achieved when tuning PID using (ES-PSO). For multi objective function the PID controller results are nearly the same for all tuning algorithms expect the (PSO) has some different results in terms of settling time and control action. The system has no overshoot because it will never reach to desire position, because there is a steady state error. It’s clearly
that the (ES-PSO) achieved the fastest settling time in addition to the rise time, steady state error and overshoot percentage very closed to the best results that achieved by other tuning algorithms for single objective functions. The plots of tracking system response, steady state error and controller output signal for FPID controller are illustrated in Fig. 31 to Fig. 36.

The above plots shown in Fig. 31 to Fig. 36 concluded that, the (PPSO), (GSA-PSO) and (ES-PSO) obtained the best results compared to (PSO and MAACPSO). The single objective function achieved smooth tracking capability with less overshoot percentage and control action than PID controller. Also, FPID controller achieved faster settling time compared to PID controller. However, the steady state error values increased. The control action values at beginning of simulation are less than case of PID controller, which implies that the actuator doesn’t reach its saturation limits smooth. The multi objective couldn’t improves the results because of the nature of fuzzy logic controller, which implies a free overshoot controller. FPID controller suitable for application in which the control action is very limited and the small deviations from the desired position acceptable.

6.4. Self-Tuning Fuzzy PID (STF-PID) Controller

Results obtained from satellite tracking system with (STF-PID) controller based on fuzzy logic control strategy presented in Table 9

Table 9

Results obtained from satellite tracking system with (STF-PID) controller

| N | OS (%) | t_r (s) | t_s (s) |
|---|---|---|---|
| 150 | 0.49 | 0.181 | 0.302 |
| 600 | 0.49 | 0.181 | 0.302 |
| 550 | 0.49 | 0.181 | 0.302 |
| 400 | 0.49 | 0.181 | 0.302 |
| 900 | 0.49 | 0.181 | 0.302 |

6.5. Self-Tuning Fuzzy Fractional Order PID (STF-FOPID) Controller

Results obtained from satellite tracking control system with (STF-FOPID) controller based on fuzzy logic control strategy represented in Table 10

Table 10

Results obtained from satellite tracking system with (STF-FOPID) controller

| K_p | K_i | K_d | μ | λ | OS (%) | t_r (s) | t_s (s) |
|---|---|---|---|---|---|---|---|
| 1.072 | 1.13 | 0.81 | 0.124 | 0.215 |
| 1.072 | 1.13 | 0.81 | 0.124 | 0.215 |
| 1.072 | 1.13 | 0.81 | 0.124 | 0.215 |
| 1.072 | 1.13 | 0.81 | 0.124 | 0.215 |
| 1.072 | 1.13 | 0.81 | 0.124 | 0.215 |

(STF-PID) achieved better performance in terms of overshoot percentage, steady state error and control action than (STF-FOPID). However, (STF-FOPID) controller could achieve faster rise time and settling time with overshoot nearly double overshoot values, which cause to increase the sum of steady state error to double as it compared to (STF-PID), but the results still excellent compared to PID and FOPID. Also, self-tuning based fuzzy logic controller is a robust (Adaptive) controller which does not vary its response for disturbance or non-linearity appearance.

The plots of tracking system response, steady state error, derivate of error signal and controller output signal for (STF-PID) controller and (STF-FOPID) controller represented in Fig. 37 to Fig. 40.

Although, (STF-FOPID) achieved faster settling time than (STF-PID), it’s consumed large control action at the beginning of tracking process compared to (STF-PID) controller. Also, the (STF-FOPID) didn’t achieved desired response compared to the (FOPID). The study predicts that, the settling time will be less than (0.11 sec) with overshoot percentage less than (0.1%). To achieve desired performance the fractional order (λ, μ) operators must be self-tuned by fuzzy logic strategy. Fuzzy rules of fractional derivate (μ) operator can take as fuzzy rules of (K_1) as in Table 4, and fuzzy rules of fractional integrator (λ) operators can take as fuzzy rules of (K_0) as in Table 5.

For single objective function the (V-PID) controller achieved the minimum overshoot percentage when tuned by (PSO). Also, the lowest rise time and the fastest settling time achieved when tuning (V-PID) controller parameters using (ES-PSO). For multi objective function the PID controller achieved the minimum overshoot percentage when tuned by (PSO). The lowest rise time and the fastest settling time achieved when tuning (V-PID) controller parameters using (ES-PSO).
fast and smooth with deviations from the desired position during all tracking process. Variable coefficient controllers improve the transient state without affecting the steady-state response.

The plots of tracking system response, steady state error and controller output signal of the (V-PID) controller are shown in Fig. 41 to Fig. 46.

![Fig. 37. System response with STF-PID and STF-FOPID controller](image)

![Fig. 38. System error with STF-PID and STF-FOPID controller](image)

![Fig. 39. System error derivate with STF-PID and STF-FOPID controller](image)

![Fig. 40. Output signal of STF-PID and STF-FOPID controller](image)

6.6. Variable Coefficient PID (V-PID) Controller

Results obtained from satellite tracking system with optimal variable coefficient PID (V-PID) controller based on different evolutionary optimization techniques are displayed in Table 11.

| Algo | C_1 | C_2 | C_3 | C_4 | C_5 | C_6 | N | OS | Er | Tr | Ts |
|------|-----|-----|-----|-----|-----|-----|---|----|----|----|----|
| PSO  | 101 | 301 | 30  | 58  | 5.8 | 0.1 | 0. | 0 | 45 | 31 | 31 |
| MA   | 10  | 34  | 14  | 30  | 37  | 17  | 0.1| 0  | 3  | 1  | 5  |
| ACP  | 0   | 7.2 | 0   | 7.6 | 0   | .3  | 31 | 0  | 42 | 42 | 42 |
| SO   | 9   | 0   | 1   |     |     |     |    |    |    |    |    |
| PPS  | 56  | 14  | 97  | 30  | 41  | 11  | 0.1| 0  |    |    |    |
| O    | .8  | 0   | 9.9 | 89  | 0   | .3  | 31 | 16 | 41 |    |    |
| GSA  | 1   | 63  | 41  | 29  | 22  | 76  | 8.4| 0  |    |    |    |
| -PSO | .7  | 2.7 | 8.2 | 3.0 | 4.8 | .4  | 8  | 97 | 40 |    |    |
| ES   | 1   | 10  | 61  | 16  | 27  | 29  | 79 | 13 | 0  |    |    |
| PSO  | 0   | 9.6 | 50  | 7.0 | 7.4 | .3  | 92 | 82 | 36 |    |    |

![Fig. 41. System response with V-PID controller (J_1)](image)

![Fig. 42. System response with V-PID controller (J_2)](image)

![Fig. 43. System steady state error with V-PID controller (J_1)](image)

![Fig. 44. System steady state error with V-PID controller (J_2)](image)

![Fig. 45. Output signal of V-PID controller (J_1)](image)

![Fig. 46. Output signal of V-PID controller (J_2)](image)

The multi objective function achieved high smooth tracking capability with smooth steady state error and lower control action than single objective function for the early running time which implies that the tracking process very fast and smooth with minimum deviations from the desired position at earlier running time leading to a highly continuous communications between earth station and desired satellite.

6.7. Variable Coefficient Fractional Order PID (V-FOPID) Controller
For single objective function the (V-FOPID) controller achieved lowest rise time and the fastest settling time when tuned by (PSO). However, the minimum overshoot percentage could achieve by tuning PID using (ES-PSO). For multi objective function the PID controller achieved lowest rise time when tuned by (PSO). The fastest settling time achieved when tuning (V-PID) controller parameters using (GSA-PSO). However, the minimum overshoot percentage could achieve by tuning PID using (ES-PSO). It’s clearly that, the (ES-PSO) achieved the minimum steady state error and overshoot percentage, in addition to the rise time and settling time are very closed to the best results that achieved by other tuning algorithms for single objective functions.

### Table 12

Results obtained from satellite tracking system with optimal (V-FOPID) controller based on different evolutionary optimization

| Algorithm | $C_1$ | $C_2$ | $C_3$ | $C_4$ | $C_5$ | $C_6$ | $\mu$ | $\lambda$ | OS (%) | $t_r$ (sec) | $t_s$ (sec) |
|-----------|------|------|------|------|------|------|------|------|--------|----------|----------|
| PSO       | 1    | 9.4  | 70   | 1    | 40   | 20   | 1    | 1.39  | 0      | 3        | 0        |
|           |      | 0.9  | 0    | 0    | 0    | 0    | 3    | 0.09  | 3      | 0        | 3        |
|           |      | 4    | 6    | 8    | 3    | 1    | 1.28 | 0     | 0      | 0        | 0        |
| MA        | 82   | 1    | 70   | 20   | 40   | 20   | 1    | 1.24  | 0      | 9        | 0        |
| ACP       | 0.6  | 0    | 0.9  | 0    | 0    | 0    | 1    | 0.09  | 3      | 0        | 3        |
| SO        | 9    | 0    | 0    | 0    | 0    | 0    | 0    | 7     | 0      | 1        | 1        |
| PPS       | 4    | 20   | 70   | 11   | 39   | 20   | 1    | 1.24  | 0      | 9        | 0        |
| O         | 91   | 0    | 0    | 6.5  | 6    | 0    | 6.3  | 0.06  | 3      | 0.08     | 2        |
|           |      | 28   | 9    | 1    | 1     | 1     | 1    |      |        |          |          |
| GSA       | 13   | 10   | 46   | 18   | 40   | 20   | 1    | 1.17  | 0      | 6        | 0        |
|           |      | 3.6  | 8.8  | 0    | 0    | 0    | 0.07 | 3     | 0       | 1.01     | 1        |
| PSO       | 11   | 90   | 14   | 28   | 3    |      |      |       |        |          |          |
| ES-ACP    | 30   | 17   | 31   | 30   | 40   | 20   | 1    | 1.3   | 0      | 9        | 0        |
| PSO       | 7    | 24   | 04   | 9    | 3    |      |      |       |        |          |          |

### V-FOPID controller for multi objective function: $J_2 = \int_0^\infty \omega_1 |e(t)| dt + \omega_2 \times OS\% + \omega_3 \times (t_s - t_r)$

| Algorithm | $C_1$ | $C_2$ | $C_3$ | $C_4$ | $C_5$ | $C_6$ | $\omega_1$ | $\omega_2$ | $\omega_3$ | $J_2$ (
|-----------|------|------|------|------|------|------|--------|--------|--------|-----------|
| PSO       | 1    | 20   | 70   | 1    | 40   | 20   | 1     | 1      | 2.9     | 0        | 0        |
|           |      | 0    | 0    | 0    | 0    | 0    | 0.07  | 7      | 0.08    | 2        | 3        |
|           |      |      |      |      |      |      | 0.05  | 0.02   | 0.03    |          |          |
| MA        | 59   | 1    | 70   | 1    | 34   | 20   | 1    | 1      | 1.1     | 0        | 0        |
| ACP       | 0.1  | 0    | 2    | 0    | 1    | 11   | 1    | 11     | 12      | 2        | 2        |
| SO        | 8    | 36   | 36   | 36   | 36   | 36   | 36    | 36     | 36      | 36       |
| PPS       | 1    | 20   | 70   | 1    | 37   | 20   | 1    | 1.9    | 0       | 0        | 0        |
| O         | 0    | 0    | 1    | 0    | 0    | 6    | 0.06  | 6      | 0.08    | 1        | 1        |
|           |      |      |      |      |      |      | 21    | 69     | 5       | 4        | 4        |
| GSA       | 32   | 3    | 40   | 23   | 37   | 19   | 1    | 1.9    | 0       | 0        | 0        |
|           |      | 0.71 | 5.0  | 4.9  | 0.07 | 6    | 0.08  | 6      | 0.08    | 1        | 1        |
| PSO       | 83   | 63   | 31   | 55   | 08   | 99   | 21    | 4      | 3       | 9        | 9        |

### Overall Discussion

The plots of tracking system response, steady state error and controller output signal of the (V-PID) controller represented in Fig. 47 to Fig. 52.

For single objective function the control action curve of (PSO) and (MAACPSO) is the same for both techniques The (GSA-PSO) and (ES-PSO) obtained the best results for both objective functions, but their results nearly equal for single and multi-objective functions.
achieved (0.08 sec) rise time whereas, when tuned PID using (ES-PSO) based on multi objective function the rise time increased to (0.21 sec). PID controller tuned by PSO based on single objective function achieved (0.87 sec) settling time and when tuned PID by (ES-PSO) achieved (0.50 sec) settling time. Whereas, when tuned PID using (ES-PSO) based on multi objective function the settling time decreased to (0.37 sec). It concluded that (ES-PSO) reduced the overshoot percentage, settling time and control action compared to PSO algorithm. Also, multi objective function has a magical effect in reducing the overshoot percentage, settling time and control action as compared with single objective function. The (FOPID) controller tuned by (ES-PSO) achieved (13 %) overshoot (0.03 sec) rise time. (0.21 sec) settling time. The FOPID controller reduces the overshoot, rise time, and settling time to (50 %) as it compared with PID controller. However, the FOPID improves the performance up to (50 %) compared to PID, increases the consumed control action. The FOPID achieved smooth and accurate tracking process during all process, especially at the first second of tracking as it compared PID controller. (STF-PID) controller compared with PID, the (STF-PID) controller achieved best performance as it combining the advantage of classical PID and fuzzy logic controller producing an adaptive controller which can deal with uncertain dynamic and non-linearity in the system. (STF-FOPID) achieved overshoot (0.81 %), rise time (0.12 sec) and settling time (0.21 sec). (STF-FOPID) is the improved version of the FOPID. Compared to PID and FOPID, (STF-FOPID) is a robust controller which does not vary its response for disturbance. The (V-PID) controller tuned by (ES-PSO) achieved (13 %) overshoot percentage, (0.08 sec) rise time and (0.36 sec) settling time. Whereas, when tuned (V-PID) using (ES-PSO) based on multi objective function, the overshoot decreased to (1.7 %) and settling time decreased to (0.19 sec). The variable coefficient PID controller is free overshoot controller and faster settling time compared to traditional PID.

Variable Coefficient Fractional Order PID (V-FOPID)

Table 13

| Optimal Controller (ES-PSO) | OS (%) | t_r (s) | t_s (s) |
|-----------------------------|--------|--------|--------|
| PID (ES-PSO_ J_1)           | 40.14  | 0.068  | 0.879  |
| PID (ES-PSO_ J_2)           | 25.45  | 0.088  | 0.505  |
| FOPID (ES-PSO_ J_1)         | 0.10   | 0.217  | 0.370  |
| FOPID (ES-PSO_ J_2)         | 19.05  | 0.038  | 0.217  |
| V-PID (ES-PSO_ J_1)         | 0.82   | 0.101  | 0.176  |
| V-PID (ES-PSO_ J_2)         | 0.49   | 0.181  | 0.302  |
| V-PID (ES-PSO_ J_2)         | 0.81   | 0.124  | 0.215  |
| V-PID (ES-PSO_ J_2)         | 13.92  | 0.082  | 0.367  |
| V-PID (ES-PSO_ J_2)         | 1.70   | 0.117  | 0.199  |
| V-PID (ES-PSO_ J_2)         | 1.31   | 0.083  | 0.137  |

6. Conclusion

The objective of this paper to design optimal controller to improve tracking process of multiple mission satellite ground station. The satellite tracking system based on geared DC servo motor designed and controlled in MATLAB/SIMULINK. controller is established to enhance the transient state without affecting the steady-state response. The (V-FOPID) tuned by (ES-PSO) achieved (13 %) overshoot and (0.13 sec) settling time. Whereas, when tuned (V-FOPID) using (ES-PSO) based on multi objective function achieved (0.52 %) overshoot and (0.14 sec) settling time. The (V-FOPID) is a free overshoot controller which could achieve fast, smooth, accurate tracking process with minimum deviation from desired position compared to other control strategies in addition to, enhances the transient state without affecting the steady-state response.

The plots of tracking system response, steady state error and controller output signal of proposed controllers represented in Fig. 53 to Fig. 55.

The best tracking process achieved by (V-FOPID) controller tuned using (ES-PSO), as well as, FOPID controller tuned by (ES-PSO) based on multi objective function, (V-PID) tuned by (ES-PSO) based on multi objective function, Self-Tuning Fractional Order PID (STF-FOPID) controller and Self-Tuning PID (STF-PID) controller respectively. The best control action achieved by PID and (V-PID) controllers. The FOPID, V-FOPID and (STF-FOPID) controllers consumed large control action at the beginning of tracking process to provide the desired performance in terms of overshoot percentage and settling time.

Many control strategy are used to control the system such as: Optimal PID controller, optimal Fuzzy PID controller, optimal variable coefficient PID controller, optimal variable coefficient fractional order PID controller, these controllers parameters have been fine-tuned by Particle Swarm Optimization (PSO), Adaptive Weighted Particle Swarm Optimization (AWPSO),
Adaptive Coefficients Particle Swarm Optimization (ACAPSO), Modified Adaptive Accelerated Coefficients Particle Swarm Optimization (MAACAPSO), Crazzy Particle Swarm Optimization (C-PSO), Phasor Particle Swarm Optimization (PPSO), Gravitational Search Algorithm with Particle Swarm Optimization (GSA-PSO) and Eagle Strategy with Particle Swarm Optimization (ES-PSO). The Performance Index has been computed using Integral Based Objective Functions and Dynamic Performance Indices Based Objective Functions. Self-Tuning Fuzzy FOPID (STF-FOPID) controller and Self Tuning Fuzzy FOPID (STF-FOPID) are proposed for satellite tracking control system. The system's response is analyzed, and the outcomes of various control strategies are measured and compared with those of other strategies. The (PPSO), (GSA-PSO) and (ES-PSO) obtained the best performance results compared to the other techniques. For single objective function PID controller tuned by (PSO) achieved settling time (0.87 sec) and overshoot percentage (40.1%), when tuned by (ES-PSO) achieved settling time (0.50 sec) and overshoot percentage (25.4%), multi objective function improves the performance of PID tuned by (ES-PSO) to achieved settling time (0.37 sec) and overshoot percentage (0.10%). In order to improve the performance of tracking system, FOPID controller is established to achieve settling time (0.17 sec) and overshoot percentage (0.82%) when tuned by (ES-PSO) based on multi objective function. Generally, it is difficult to find an accurate non-linear model of actual DC motor. As a result, Self-Tuning Fuzzy PID (STF-PID) and Self-Tuning Fuzzy fractional order PID (STF-FOPID) controllers are established to deal with the system's uncertain dynamics and non-linearity while still delivering the desired response. (STF-PID) controller could achieved settling time (0.30 sec) and overshoot percentage (0.49%), and (STF-FOPID) controller achieved settling time (0.21 sec) and overshoot percentage (0.81%). PID and FOPID controller have constant coefficient which are proportional gain $K_p$, integral gain $K_i$, derivative gain $K_d$, controllers have constant coefficient throughout the operation regardless of the system parameter. As a result, it's impossible to interfere with the transient and steady states separately. Variable Coefficient PID (V-PID) controller and Variable Coefficient Fractional Order PID (V-FOPID) controller are established to enhances the transient state without affecting the steady-state response. For multi objective function variable coefficient PID (V-PID) controller tuned by (ES-PSO) achieved settling time (0.19 sec) and overshoot percentage (1.7%). Variable coefficient fractional order PID controller tuned by (ES-PSO) could achieved settling time (0.13 sec) and overshoot percentage (1.3%) using single objective function, and for multi objective function the (V-FOPID) tuned by (ES-PSO) achieved settling time (0.14 sec) and overshoot percentage (0.52%).

7. Further Work
Further work may focus on designing adaptive control techniques based on fuzzy logic control as follow: design a self-tuning fuzzy PID(5-10) controller, to perform online tuning of $(K_p, K_i, K_d, N)$ parameters of PID controller. Adding filter coefficient to be auto tuning will improve the performance. Design a self-tuning fuzzy fractional order PID (STF-FOPID) controller, to perform online tuning of $(K_p, K_i, K_d, \lambda, \mu)$ parameters of FOPID controller leading to enhance the controller's performance and increase its robustness. However, the fuzzy system become more complex as we need five fuzzy system with five table of rules to fire five $(K_p, K_i, K_d, \lambda, \mu)$ outputs. Design a self-tuning fuzzy variable coefficient PID (STF-V-PID) controller, to perform online tuning of $(c_1, c_2, c_3, c_4, c_5, \epsilon_p, \epsilon_i, \epsilon_d, N)$ parameters of (V-PID) controller. This new structure will enhance the steady-state response while leaving the transient response untouched. As well as, design a self-tuning fuzzy variable coefficient fractional order PID(5-10) controller, to perform online tuning of $(c_1, c_2, c_3, c_4, c_5, \epsilon_p, \epsilon_i, \epsilon_d, \lambda, \mu)$ parameters of (V-FOPID) controller. This controller structure will be created to enhance transient state response without affecting steady state response. Besides the ability to handle system dynamics and non-linearity, it also provides an opportunity for better control system dynamics adjustment, which implies significantly increased robustness and stability of system. The Combination of the (V-FOPID) with a fuzzy control strategy can be used to automatically tune the controller's parameters online, improving the select ability of the control parameters. However, FLC transformed (V-FOPID) to an adaptive controller, the complexity off the system increases. There is a need to find the proper rules to fire the outputs of eight fuzzy systems. Finally, an adaptive neuro fuzzy inference system (ANFIS) will be used to train the rules of eight fuzzy system based on the running data of (V-FOPID) controller, the values of $(C_1, C_2, C_3, C_4, C_5, \epsilon_p, \epsilon_i, \epsilon_d, \lambda, \mu)$ parameters of (V-FOPID) controller replaced by the eight trained fuzzy logic systems.

References

[1] M. Shweta, S. Waghmare, M. Payal, S. Pathak, M. Parag, V. Meshram, M. Vidyas, B. Pawar, Satellite Dish Positioning System, IJBIST-International J. Innov. Res. Sci. Technol. 4 (2017). https://www.ijjestr.org.
[2] T. Van Hoi, N.X. Truong, B.G. Duong, Satellite Tracking Control System Using Fuzzy PID Controller, VNU J. Sci. Math. – Phys. 31 (2015) 36–46.
[3] A. Uthman, S. Sudin, Antenna azimuth position control system using PID controller & state-feedback controller approach, Int. J. Electr. Comput. Eng. 8 (2018) 1539–1550. https://doi.org/10.11591/ijece.v8i3.pp1539-1550.
[4] S.S. O. K.L. Ratnakar, G. Sivasankaran, Precision Control of Antenna Positioner Using P and PI Controllers, Int. J. Eng. Sci. Innov. Technol. 4 (2015) 234–245.
[5] H.I. Okumus, E. Sahin, O. Akyazi, Antenna azimuth position control with fuzzy logic and self-tuning fuzzy logic controllers, ELECO 2013 - 8th Int. Conf. Electr. Electron. Eng. (2013) 477–481. https://doi.org/10.1109/eleco.2013.6713888.
[6] T. Yamaoka, A. Ming, T. Kida, C. Kanamori, M. Satoh, Accuracy improvement of ship mounted tracking antenna for satellite communications, Nikkon Kikai Gakkai Ronbunshu, C Hen/Transactions Japan Soc. Mech. Eng. Part C. 73 (2007) 170–176. https://doi.org/10.1299/kikai.73.170.
[7] A. Mahmood, A. Abdulla, I. Mohammed, Helicopter Stabilization Using Integer and Fractional Order PID Controller Based on Genetic Algorithm, (2020). https://doi.org/10.4108/eaic.28-6.2020.2297914.
[8] J. Kim, K. Cho, C. Jang, Fuzzy control of data link antenna control system for moving vehicles, in: International Conference on Control, Automation, and Systems (ICCAS), KINTEX, Gyeonggi-Do, Korea, 2005: pp. 525–528.
[9] M. Ahmed, S. Bahari, M. Noor, M. Khar, B. Hassan, A. Bi, C. Soh, A Review of Strategies for Parabolic Antenna Control, Aust. J. Basic Appl. Sci. Aust. J. Basic Appl. Sci. 8 (2014) 135–148. www.ajbasweb.com.
[10] L.A. Alool, P.K. Kihato, S.J. Kamau, DC Servomotor-based Antenna Positioning Control System Design using Hybrid PID-LQR Controller, Eur. Int. J. Sci. Technol. 5 (2016) 17–31. www.eijst.org.uk.
[11] L. A. Alwal, P. K. Kihato, S. I. Kamau, Design of Neuro-Fuzzy System Controller for DC ServomotorBased Satellite Tracking System, IOSR J. Electr. Electron. Eng. 11 (2016) 89–102. https://doi.org/10.9790/1767-11040389102.
[12] P. Kumar, S. Chatterjee, D. Shah, U.K. Saha, S. Chatterjee, On comparison of tuning method of FOPID controller for controlling field controlled DC servo motor, Cogent Eng. 4 (2017). https://doi.org/10.1080/23319196.2017.1357875.
[13] O. Aydoğdu, M. Koçkemaz, Optimal design of a variable coefficient fractional order PID controller by using heuristic optimization algorithms, Int. J. Adv. Comput. Sci. Appl. 10 (2019) 314–321. https://doi.org/10.14569/IJACSA.2019.0100341.
[14] Y. Zhang, S. Wang, G. Ji, A Comprehensive Survey on Particle Swarm Optimization Algorithm and Its Applications, Math. Probl. Eng. 2015 (2015). https://doi.org/10.1155/2015/5931256.
analysis and fuzzy fractional-order PID parameter optimization for primary frequency modulation of a pumped storage unit based on a multi-objective gravitational search algorithm, Energies, 13 (2019). https://doi.org/10.3390/en13010137.

[37] Q. Gao, K. Li, Y. Hou, R. Hou, C. Wang, Balancing and positioning for a gun control system based on fuzzy fractional order proportional-integral-derivative strategy, Adv. Mech. Eng. 8 (2016) 1–9. https://doi.org/10.1177/1687814016639054.

[38] M. Al-Dhaifallah, N. Kanagaraj, K.S. Nisar, Fuzzy Fractional-Order PID Controller for Fractional Model of Pneumatic Pressure System, Math. Probl. Eng. 2018 (2018). https://doi.org/10.1155/2018/5787871.

[39] D. Bratron, J. Kennedy, Defining a standard for particle swarm optimization, Proc. 2007 IEEE Swarm Intell. Symp. SIS 2007. (2007) 120–127. https://doi.org/10.1109/SIS.2007.368035.

[40] K.Y. Lee, J.B. Park, Application of particle swarm optimization to economic dispatch problem: Advantages and disadvantages, 2006 IEEE PES Power Syst. Conf. Exp. PSCE 2006 - Proc. (2006) 186–192. https://doi.org/10.1109/PSCE.2006.296295.

[41] J.G. Vlachogiannis, K.Y. Lee, Economic load dispatch - A comparative study on heuristic optimization techniques with an improved coordinated aggregation-based PSO, IEEE Trans. Power Syst. 24 (2009) 991–1001. https://doi.org/10.1109/TPWRS.2009.206524.

[42] G. Vidya, S. Sarathambekai, K. Umamaheswari, S.P. Yamunadevi, Task scheduling using adaptive weighted particle swarm optimization with adaptive weighted sum, Procedia Eng. 38 (2012) 3056–3063. https://doi.org/10.1016/j.proeng.2012.06.356.

[43] N.K. Bahgat, M.I. El-Sayed, M.A.M. Hassan, F.A. Bendary, Load Frequency Control in Power System via Improving PID Controller Based on Particle Swarm Optimization and ANFIS Techniques, Int. J. Syst. Dyn. Appl. 3 (2014) 1–24. https://doi.org/10.4018/jisa.2014070101.

[44] K.T. Chaturvedi, M. Pandit, L. Srivastava, Self-organizing hierarchical particle swarm optimization for nonconvex economic dispatch, IEEE Trans. Power Syst. 23 (2008) 1079–1087. https://doi.org/10.1109/TPWRS.2008.926455.

[45] J.B. Park, K.S. Lee, J.R. Shin, K.Y. Lee, A particle swarm optimization for economic dispatch with nonsmooth cost functions, IEEE Trans. Power Syst. 20 (2005) 34–42. https://doi.org/10.1109/TPWRS.2004.831275.

[46] S. Ahmed, B. Tarek, N. Djemai, Economic dispatch resolution using adaptive acceleration coefficients based PSO considering generator constraints, 2013 Int. Conf. Control, Decis. Inf. Technol. CoDT.2013. (2013) 212–217. https://doi.org/10.1109/CoDT.2013.6689546.

[47] M.A. Abido, N.A. Al-Ali, Multi-objective differential evolution for optimal power flow, POWERENG 2009 - 2nd Int. Conf. Power Eng. Energy Electr. Drives Proc. (2009) 101–106. https://doi.org/10.1109/POWERENG.2009.4915212.

[48] N.A. Firdausanti, Iramah, On the comparison of crazy particle swarm optimization and advanced binary ant colony optimization for feature selection on high-dimensional data, in: Procedia Comput. Sci., Elsevier 2019. pp. 639–646. https://doi.org/10.1016/j.procs.2019.11.167.

[49] B. Xue, M. Zhang, W.N. Browne, Particle swarm optimization for feature selection in classification: A multi-objective approach, IEEE Trans. Cybern. 43 (2013) 1666–1671. https://doi.org/10.1109/TSMCB.2012.2227469.

[50] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: A Gravitational Search Algorithm, Inf. Sci. (Ny). 179 (2009) 2232–2248. https://doi.org/10.1016/j.ins.2009.03.004.

[51] I. Newton, In [experimental] philosophy particular propositions are inferred from the phenomena and afterwards rendered general by induction, Principia, n.d.

[52] X.S. Yang, S. Deb, Eagle strategy using Lévy walk and firefly algorithms for stochastic optimization, Stud. Comput. Intell. 284 (2010) 101–111. https://doi.org/10.1007/978-3-642-12538-6_9.

[53] K. Santhosh, R. Neela, Optimal Placement of Distribution Generation on Micro-Grid Using Eagle Strategy With Particle Swarm Optimizer, JREAS, Vol. 07, Issue 01, Jan 22.
[54] X.S. Yang, S. Deb, X. He, Eagle strategy with flower algorithm, Proc. 2013 Int. Conf. Adv. Comput. Commun. Informatics, ICACCI 2013. (2013) 1213–1217. https://doi.org/10.1109/ICACCI.2013.6637350.

[55] S. Cheng, M.Y. Chen, Multi-objective reactive power optimization strategy for distribution system with penetration of distributed generation, Int. J. Electr. Power Energy Syst. 62 (2014) 221–228. https://doi.org/10.1016/j.ijepes.2014.04.040.

[56] Q. Liu, J. Yue, An improved mind evolutionary algorithm for reactive power optimization, in: Proc. 32nd Chinese Control Conf., 2013; pp. 8048–8051.