Computer Systems Science & Engineering
DOI: 10.32604/csse.2023.032737
Article

3D Path Optimisation of Unmanned Aerial Vehicles Using Q Learning-Controlled GWO-AOA

K. Sreelakshmy[1], Himanshu Gupta[1], Om Prakash Verma[1], Kapil Kumar[2], Abdelhamied A. Ateya[3] and Naglaa F. Soliman[4,*]

[1]Department of Instrumentation and Control Engineering, Dr. B. R. Ambedkar National Institute of Technology Jalandhar, Punjab, 144027, India
[2]Department of Industrial and Production Engineering, Dr. B. R. Ambedkar National Institute of Technology Jalandhar, Punjab, 144027, India
[3]Department of Electronics and Communication Engineering, Zagazig University, Zagazig, Sharqia, 44519, Egypt
[4]Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, Riyadh, 84428, Saudi Arabia
*Corresponding Author: Naglaa F. Soliman. Email: nfsoliman@pnu.edu.sa
Received: 28 May 2022; Accepted: 30 June 2022

Abstract: Unmanned Aerial Vehicles (UAVs) or drones introduced for military applications are gaining popularity in several other fields as well such as security and surveillance, due to their ability to perform repetitive and tedious tasks in hazardous environments. Their increased demand created the requirement for enabling the UAVs to traverse independently through the Three Dimensional (3D) flight environment consisting of various obstacles which have been efficiently addressed by metaheuristics in past literature. However, not a single optimization algorithms can solve all kind of optimization problem effectively. Therefore, there is dire need to integrate metaheuristic for general acceptability. To address this issue, in this paper, a novel reinforcement learning controlled Grey Wolf Optimisation-Archimedes Optimisation Algorithm (QGA) has been exhaustively introduced and exhaustively validated firstly on 22 benchmark functions and then, utilized to obtain the optimum flyable path without collision for UAVs in three dimensional environment. The performance of the developed QGA has been compared against the various metaheuristics. The simulation experimental results reveal that the QGA algorithm acquire a feasible and effective flyable path more efficiently in complicated environment.

Keywords: Archimedes optimisation algorithm; grey wolf optimisation; path planning; reinforcement learning; unmanned aerial vehicles

1 Introduction

Unmanned Aerial Vehicles (UAVs) are an evolving aerospace technology with immense potential for numerous applications. Even though formerly devoted for defence applications, the range of applications of UAVs have been extended to several commercial and domestic fields. Disaster management and

disaster zone mapping [1], construction [2], search and rescue, survey, security, and traffic surveillance [3] are some of the applications in which UAVs have been immensely utilised. The ever increasing usage of UAVs in day-to-day life applications requires the UAVs to autonomously navigate in a real world environment. The accessibility to 3D for traversing and the presence of obstacles make the path planning a perplexing task. The 3D path planning of UAV, aims to create a flyable path in a challenging environment. The path optimisation of UAV enables to detect a safe collision-free flight route.

The path planning of UAVs in 3D environment can be formulated as an optimisation problem, for which the fitness function can be designed and an optimisation technique can be devised to obtain the optimum solution path. The objective of path planning is to construct a collision-free flight route for the UAV from the start to the target position where it is deployed. Along with the collision avoidance, other constraints, such as energy consumption and time required to accomplish the desired path are also considered to frame the objective function. Throughout the literature, metaheuristic algorithms prove to be successful optimisation techniques for several engineering optimisation problems, due to their advantages such as ease of implementation and simplicity [4]. Several proficient metaheuristics have been utilised successfully for path optimisation as well. Furthermore, metaheuristics have been examined to perform better than graph based methods for UAV path planning [5]. The previous literature reveals the employability of metaheuristics for 2D (Two Dimensional) path planning however, for UAVs, it restrict the mobility in any one of the plane [6,7].

For 3D path planning of UAVs, Particle Swarm Optimisation (PSO) and Global Best path competition (GBPSO) have been exhausted in past [8,9]. For military environments containing radar guided Surface-to-Air Missiles (SAM), an Improved Particle Swarm Optimisation (IPSO) algorithm has also been employed [10]. However, PSO struggles with the problem of early convergence and trapping in local minima and therefore, to overcome these limitations, Rauch-Tung-Striebel (RTS) smoother and Metropolis criterion have been extensively incorporated [11]. Recently Double Dynamic Biogeography-Based Learning PSO has also been investigated to improve PSO for UAV path planning [12]. Further, being motivated by the efficacy of Grey Wolf Optimization (GWO), it has been exhaustively utilised with improved and hybrid versions, such as Markov Decision Process based GWO with Level Comparison (GWOLC) [13], Modified Symbiotic Organism Search (MSOS) based GWO [14] and hybrid Golden Eagle Optimiser-GWO [15] for optimal flight route mapping. Considering UAVs as an emerging technology for disaster management, a constrained Differential Evolution (DE) algorithm has been proposed for flight route optimisation [1]. Additionally, Improved Bat Algorithm (IBA) [16] and Glow-worm Swarm Optimisation (GSO) [17] have been employed to estimate the efficient route in static and dynamic environment, respectively. In addition to this, physics inspired optimisers such as Equilibrium Optimiser (EO) [18], Quantum-Entanglement Pigeon-inspired Optimisation (QEPO) [19], Multiverse optimiser (MVO) [20] and Gravitational Search Algorithm (GSA) [21] have been successfully employed for automatic optimal flight path generation in the 3D flight environment. Recently, other physics inspired algorithms have also been introduced such as Archimedes Optimisation Algorithm (AOA) [22], Flow Direction Algorithm (FDA) [23], Artificial Electric Field Algorithm (AEFA) [24], etc. However, the capabilities of these developed algorithms for optimal path planning have been seldom examined. Apart from metaheuristics, other Artificial Intelligent (AI) techniques such as Heuristic search, Artificial Neural Networks, Local search techniques have been studied extensively for UAV path planning [25].

Moreover, with the advent of Machine Learning (ML), enormous work has been extensively reported to elevate the optimisation capabilities of metaheuristics. In this sequence, Reinforcement Learning (RL) has been commonly utilised to control the exploration and exploitation behaviours [26]. The efficacy of RL based metaheuristic for UAV flight route planning has been validated to achieve better flying path and convergence speed [27]. Furthermore, as per No Free Lunch (NFL) [28] theorem, an optimisation algorithm promising good performance in terms of accuracy and convergence speed on one set of

problems does not ensure similar performance on other set of problems. This provides a possibility to develop a new and advance optimization algorithms for UAV path planning.

Therefore, the present work investigates the possibilities to develop a novel metaheuristic by employing AOA and GWO. Further, the RL based algorithm selection mechanism has been developed in order to reach the new horizons with proposed metaheuristics. In AOA, the capability to optimize any real time optimization problem has been controlled by transfer operator that maintains the global-local search balance. The transfer operator assures that the first 30% iterations are exploration, by updating the individual positions based on the position of a randomly selected individual. During the next phase, the exploitation, the positions of the individuals are updated utilising the position of the best individual. However, the low convergence speeds of AOA is the consequence of absence of exploitation during the first one-third iterations. Additionally, the previous literature comprehensively employed the use of GWO for UAVs path planning because of its higher convergence speed and capability to generate optimum path with obstacle [29,30]. The GWO estimates three best individuals, alpha, beta, and delta wolves and the positions of all the individuals are updated based on these best individuals. This ensures good exploitation capacity for the GWO algorithm however, it suffers by the premature convergence and stagnation at local minima [31].

To effectively address the above mentioned issues, the Q learning controlled GWO-AOA (QGA) has been proposed in the present work. The exploitation capabilities of GWO and the exploration abilities of AOA have been combined in the introduced algorithm. For maintaining the balance between global and local search, ε greedy Q learning has been employed [32]. The Q learning states and actions are labelled as AOA and GWO. The Q learning agent selects between GWO and AOA according to the Q table values to achieve maximum reward. The incorporation of Q agent for regulating the exploration and exploitation balance will provide higher convergence rates and prevents trapping in the local optimal solutions.

The performance of the developed QGA has been validated against 22 popular benchmark functions and then, employed to estimate the optimum path for UAVs. For this purpose, the results obtained for the benchmark functions have been compared with GWO, AOA, RL based GWO (RLGWO) [27], Enhanced AOA (EAOA) [33], and PSO [34]. Further, the popular statistical test such as Friedman test and Wilcoxon rank-sum test have also been executed on the benchmark results to endorse the efficacy of the compared metaheuristics. For path planning also, the developed QGA has been exhaustively compared with aforementioned metaheuristics under complex 3D environment. Summarising the main contribution of the present investigation are:

- Development of Q-controlled GWO-AOA (QGA) to enhance the exploration-exploitation balance.
- Development of algorithm selection mechanism to efficiently select the employed metaheuristics based on Q-State parameters.
- Exhaustive validation of developed QGA against other metaheuristics for various benchmark functions based on statistical tests.
- Examining the employability of QGA for real-time path-planning optimization problem of UAV in 3D environment.

The rest of the paper is organised as follows. Section 2 includes the brief description about Q learning, GWO, and AOA. The detailed description of the proposed QGA is presented in Section 3. The path planning problem statement is formulated in Section 4. The performance analysis of QGA over benchmark functions and the effectiveness of proposed algorithm for path planning in 3D flying field is presented in Section 5. Lastly, Section 6 concludes the work with discussions on future research.

2 Background

2.1 Q Learning

RL comprises four sub-elements, namely, policy, reward signal, a value function, and a model of the environment. RL depends on the interactivity between the learning agent and its environment. Reward signal is the feedback from the environment to the RL agent which quantifies this interactivity. Q learning is an off policy Temporal Difference (TD) method. Q indicates the action-value function. One of the main components of Q learning is the Q table. The reward is the reinforcement signal which guides the learning agent during the process. The main objective of the learning agent is to maximise the reward signal. The Q value is dependent on the reward signal. During each operation, the Q table is updated based on the Eq. (1). For each state-action pair there will be a corresponding Q table value ($Q(S_{itr},\ A_{itr})$). Each Q table update depends on the reward signal (*Reward*) from the environment. The goal of maximizing the reward at the end of the operations is ensured by performing an action with the highest Q value.

$$Q_{itr+1}(S_{itr},\ A_{itr}) = Q(S_{itr},\ A_{itr}) + \mu\left(Reward_{itr+1} + \gamma \max_{A} Q(S_{itr+1},\ A_{itr}) - Q(S_{itr},\ A_{itr})\right) \tag{1}$$

where, S is the state, A is the action, *itr* is the current iteration, μ indicates the learning rate, and γ denotes the discount factor.

2.2 Archimedes Optimisation Algorithm (AOA)

The Archimedes Optimisation Algorithm (AOA) is based on Archimedes' principle that defines the relation between buoyant forces or upthrust and the weight of the displaced fluid. According to Archimedes' principle, if a body is completely or partially immersed in a liquid, the net upward force, also known as the buoyant force acting on the body will be equal to the weight of the liquid displaced by the body. If the weight of the displaced liquid is greater than the upward force, the body will be completely immersed in the liquid whereas, if the weight of the body is equal to the weight of the displaced liquid, the body will be floating in the liquid. The initial population of AOA consists of various bodies with different densities and volumes. Each of these bodies will try to achieve an equilibrium state with neutral buoyancy. This can be mathematically represented by Eq. (2), which is further expanded to Eq. (3)

$$F_{liq} = F_b \tag{2}$$

$$\rho_{liq} V_{liq} acn_{liq} = \rho_b V_b acn_b \tag{3}$$

where, ρ_{liq}, V_{liq}, and acn_{liq} are the density, volume and acceleration of the liquid, respectively. Similarly, ρ_b, V_b, and acn_b are the density, volume and acceleration of the object floating in the liquid, respectively.

To obtain the acceleration of the body, Eq. (3) is rearranged as shown in Eq. (4)

$$acn_b = \frac{\rho_{liq} V_{liq} acn_{liq}}{\rho_b V_b} \tag{4}$$

If other forces such as collision with other bodies are affecting the body under consideration, Eqs. (2)–(3) are modified as Eqs. (5)–(6) respectively.

$$W_{liq} - W_{eb} = F_b \tag{5}$$

$$\rho_{liq} V_{liq} acn_{liq} - \rho_{eb} V_{eb} acn_{eb} = \rho_b V_b acn_b \tag{6}$$

where, ρ_{eb}, V_{eb}, and acn_{eb} denotes the density, volume, and acceleration of the external object, W_{liq} is the weight of the liquid displaced and W_{eb} is the weight of the external object.

The initial position (P_{bi}) and acceleration (acn_{bi}) of the i th body immersed in the liquid are generated using Eqs. (7)–(8). The density and volume of each body is initialised randomly in the range [0, 1].

$$P_{bi} = lb_b + r_1(ub_b - lb_b) \tag{7}$$

$$acn_{bi} = lb_b + r_2(ub_b - lb_b) \tag{8}$$

where, r_1 and r_2 indicates random numbers in the range [0, 1], lb_b and ub_b are the lower bounds and upper bounds of the search space, respectively.

The density factor and transfer operator helps in maintaining exploration-exploitation balance in AOA. During first 30% of the iterations, AOA is in global search followed by the local search. The exploration and exploitation phases are defined using transfer operator (TR) defined by Eq. (9). Density factor (df) represents the exploration to exploitation shift and is calculated by Eq. (10).

$$TR = exp\left(\frac{itr - Max}{Max}\right) \tag{9}$$

$$df = exp\left(\frac{itr - Max}{Max}\right) - \left(\frac{itr}{Max}\right) \tag{10}$$

where, itr and Max depicts the current iteration and maximum number of iterations, respectively.

During each iteration the density (ρ_{bi}) and volume (V_{bi}) of each body are updated using Eqs. (11)–(12).

$$\rho_{bi}(itr+1) = \rho_{bi}(itr) + r_3(\rho_{alpha} - \rho_{bi}(itr)) \tag{11}$$

$$V_{bi}(itr+1) = V_{bi}(itr) + r_4(V_{alpha} - V_{bi}(itr)) \tag{12}$$

where, r_3 and r_4 corresponds to the random numbers [0, 1], ρ_{alpha} and V_{alpha} represents the best density and volume of the body obtained from the previous iteration, respectively.

Further, if $TR \leq 0.5$, exploration is performed and, the acceleration and position are updated using the previously selected random body, respectively. This acceleration update mechanism is represented by Eq. (13) where, ρ_{br}, V_{br} and acn_{br} denotes the density, volume and acceleration of the randomly selected body respectively.

$$acn_{bi}(it+1) = \frac{\rho_{br} + V_{br} acn_{br}}{\rho_{bi}(itr+1) V_{bi}(itr+1)} \tag{13}$$

Then, the acceleration, acn_{bi} is normalised using Eq. (14), where, u and l are constants.

$$norm\_acn_{bi}(itr+1) = u \times \frac{acn_{bi}(itr+1) - min(acn)}{\max(acn) - min(acn)} + l \tag{14}$$

The position of i th individual during the exploration stage is updated using Eq. (15).

$$P_{bi}(itr+1) = P_{bi}(itr) + c_1 \times r_5 \times norm\_acn_{bi}(itr+1) \times df \times (P_{br} - P_{bi}(itr)) \tag{15}$$

where, c_1 is a constant, r_5 is a random number, and P_{br} is the position of the selected random object.

Moreover, if $TR > 0.5$, the algorithm will perform local search (exploitation). The acceleration is updated using the Eq. (16) and then normalised by employing Eq. (14).

$$acn_{bi}(itr+1) = \frac{\rho_{alpha} + V_{alpha} acn_{alpha}}{\rho_{bi}(itr+1) V_{bi}(itr+1)} \tag{16}$$

where, ρ_{alpha}, V_{alpha}, and acn_{alpha} indicates the density, volume, and acceleration of the best body obtained so far, respectively.

Afterwards, if P_{alpha} is the position of the body with best fitness function then, the position of the i th body is updated by Eq. (17) considering c_2 and r_6 as constant and random number, respectively.

$$P_{bi}(itr+1) = P_{alpha}(itr) + Flag \times c_2 \times r_6 \times norm\_acn_{oi}(itr+1) \times df \times (TF \times P_{alpha} - P_{bi}(itr)) \tag{17}$$

Additionally, based upon the constant (c_3 and c_4) and random value (r_7), TF and $Flag$ are also regularly estimated by Eqs. (18)–(20).

$$TF = c_3 \times TR \tag{18}$$

$$Flag = \begin{cases} +1, & if\ F \leq 0.5 \\ -1, & otherwise \end{cases} \tag{19}$$

$$F = 2 \times r_7 - c_4 \tag{20}$$

2.3 Grey Wolf Optimisation (GWO)

The Grey Wolf Optimisation (GWO) is inspired from the hunting behaviours and social hierarchy of grey wolf packs. Grey wolves live in groups called packs and maintain a strict social hierarchy among the group. The topmost leader of the group is called the α wolf, followed by leader groups, the β, and the δ wolves, and the rest of the pack is called as the ω wolves. The candidate solutions (wolf population) is updated based on the leader groups which represent the best, the second best, and the third best solutions, respectively using Eqs. (21)–(28).

$$P_{bi}(itr+1) = (P_1 + P_2 + P_3)/3 \tag{21}$$

$$P_1 = P_{alpha}(itr) - A_1 D_{alpha} \tag{22}$$

$$P_2 = P_{beta}(itr) - A_2 D_{beta} \tag{23}$$

$$P_3 = P_{delta}(itr) - A_3 D_{delta} \tag{24}$$

$$D_{\alpha} = |C_1 P_{alpha}(itr) - P_{oi}(itr)| \tag{25}$$

$$D_{\beta} = |C_2 P_{beta}(itr) - P_{oi}(itr)| \tag{26}$$

$$D_{\delta} = |C_3 P_{delta}(itr) - P_{oi}(itr)| \tag{27}$$

$$C_k = 2 r_{ck}$$

$$A_k = 2 a_t . r_{tk} - a_t \tag{28}$$

where, a_t is linearly decreased from 2 to 0 during each iteration, r_{ck} is a random number which gives C_k, $k = 1, 2, 3$, r_{tk} is another random number used to calculate the parameter A_k, $P_{bi}(itr)$ is the position of i th candidate solution at the current iteration itr, $P_{bi}(itr+1)$ is the updated position. P_{bi} is given by the mean of P_1, P_2, and P_3, which depends on the positions of the leader group, P_{alpha}, P_{beta}, and P_{delta}, respectively.

3 The Q Learning-Controlled GWO-AOA

GWO and AOA are both population based algorithms. In the case of AOA, the exploration and exploitation behaviour is selected by *TR* which leads to slow convergence rate. In GWO, extremely efficient exploitation capabilities are witnessed because of the dependence on the leader group only. However, this also reduces the population diversity which lead to premature convergence and premature local minima. Therefore, the GWO and AOA are combined to obtain more efficient global-local capabilities. But, selecting a particular optimization algorithm at different stages is very challenging task and therefore, the artificially intelligent, Q learning is employed. Additionally, premature convergence and stagnation at local optima can also be avoided since, the Q agent maintains the balance between exploration and exploitation. The Q agent makes selection based on the Q table values, such that it enables to increase the reward as the iterations progress. The reward is assigned to the agent when the performance is improved. The accumulated performance along with reward decides the change of state or action selection.

The Q learning is a value based RL technique where, Q denotes the action-value function. In this work, two actions and states are defined, AOA and GWO. The Q table contains the Q values for each state-action pair. The training agent and the environment are represented by the individuals of the population and the search space, respectively. During each operation, the Q table is updated using Eq. (1) and the exploration-exploitation balance is maintained using ε greedy policy. As per policy, exploration operation is performed with a probability, $1-\varepsilon$, and exploitation with a probability, ε. The exploitation operation forces the Q agent to attain maximum reward by choosing the current best selection, whereas exploration indicates a random selection. The flowchart for the algorithm is illustrated in Fig. 1.

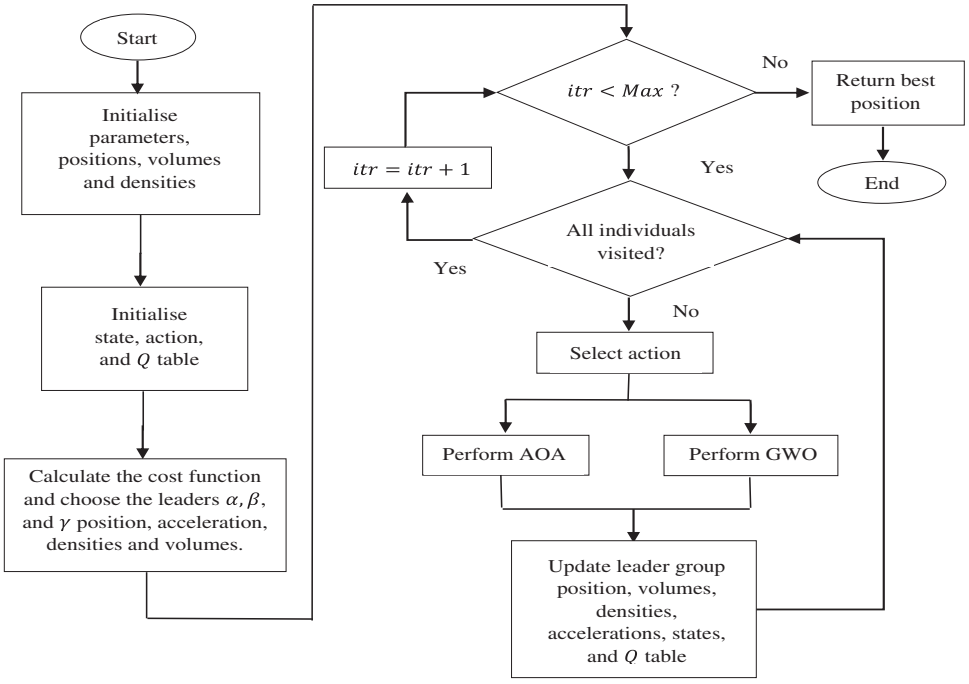

Figure 1: Flowchart for QGA

The agent decides the next action based on the Q table values, which are further depending on the reward signal. By selecting the action with the maximum Q value during the exploitation operation, the agent maximizes its reward. When an operation results in a better cost function value, a positive reward is

assigned, otherwise, the Q agent is penalised by a negative reward. If *Reward* indicate the reward signal achieved by the agent from the environment then the reward assignment is represented by Eq. (29).

$$Reward = \begin{cases} +1, & if\ cost\ function\ is\ improved \\ -1, & otherwise \end{cases} \tag{29}$$

Pseudocode 1 The QGA

Set the parameters
Set the states and actions
Set the Q table
Initialize the population
Initialize density, volume and acceleration
Calculate the cost function
Choose the leader group, α, β, and δ
Set $itr = 0$
while $itr < Max$ **do**
    ε greedy policy derived from Q selects the action
    for every P_{bi} in population **do**
        Update densities and volumes
        switch action
            Case 1: AOA
                if $TR \leq 0.5$
                    Update position using Eqs. (13)–(15)
                else
                    Update position using Eqs. (16)–(17)
                end if
            Case 2: GWO
                Update position using Eqs. (21)–(28)
        end switch
    end for
    Update cost function
    Get reward using Eq.(29)
    Update P_{alpha}, P_{beta}, and P_{delta}
    Update the Q table
$itr = itr + 1$
end while
Return best position

4 The Path Planning

4.1 Problem Formulation and Cost Function

The present work attempts to formulate the NP Hard problem such as UAV path planning in 3D environment as an optimisation problem and metaheuristics is found as the most effective approach to solve such problems. The prime objective of path planning for UAV is to find an optimal flyable path from source to destination without collisions in minimum time and energy. To fulfill these requirements, the cost function is formulated based on [14] and [27]. Then, an initial population of candidate paths is generated and by employing an optimisation algorithm, the feasible path is achieved as per the designed cost function. The shortest path between the start and target position have M waypoints, which are initialised with coordinates, (x_{ml}, y_{ml}, z_{ml}), where $ml = 0, 1, 2, \ldots, M$ then, the population individuals for optimisation with $2M$ dimension is initialised as shown in Eq. (30).

$$P_i = [y_{i1}, y_{i2}, \ldots y_{iM}, z_{i1}, z_{i2}, \ldots z_{iM}] \tag{30}$$

where, y_{ik} and z_{ik} are the y and z coordinates of the k th waypoint of the i th individual. Initially, the population individuals, P_i, are generated randomly and then, the QGA is employed to optimise the cost function during each iteration.

The cost function to be minimised for the optimal flight trajectory from start to target position is formulated considering the collisions with the obstacles in the environment, energy consumption, and travel time. The collision cost, $F_{collision}$ is defined to indicate the collision with any obstacles in the flying environment. The energy consumption will be low for the path with shortest length. The shortest possible collision-free path will also have minimum travel time. Therefore, the cost function, F_{cost}, is computed as the weighted sum of the collision cost, energy cost, and the deviation cost. Hence, F_{cost} is represented as shown in Eq. (31),

$$F_{cost} = p_1 F_{energy} + p_2 F_{collision} + p_3 F_{deviation} \tag{31}$$

Since the length of the path is directly proportional to the energy consumption, the energy cost (F_{energy}) is estimated as the length of the path. When the generated flight path passes through the obstacles present in the environment, a collision cost is evaluated. The flight path with length, l, is divided into M, number of waypoints from start to target positions. For calculating the collision cost, each path segment is again divided into 5 equidistant points, and each of these path segments is examined whether, they are passing through the obstacle or not. If the path segment of length, l_m is colliding with an obstacle j as shown in Fig. 2 then, $F_{collision}$ is calculated by Eqs. (32)–(33), otherwise, $f_{collision,l_m}$ is assigned as zero (Eq. (33)). Here, $s^m_{0.25,j}$ is the length from the centre of the obstacle to the second equidistant point on the path segment, r_j is the base radius of the obstacle and N is the maximum number of obstacles.

$$F_{collision} = \int_0^l f_{collision} dl \tag{32}$$

$$f_{collision,l_m} = \frac{l_m}{5} \sum_{j=1}^{N} \frac{1}{5} (|s^m_{0,j} - r_j| + |s^m_{0.25,j} - r_j| + |s^m_{0.5,j} - r_j| + |s^m_{0.75,j} - r_j| + |s^m_{1,j} - r_j|) \tag{33}$$

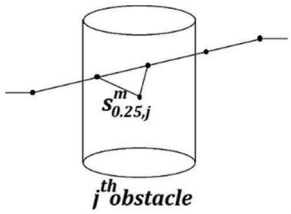

Figure 2: Calculating the collision cost

The deviation cost, $F_{deviation}$, is calculated using Eqs. (34)–(35)

$$F_{deviation} = \int_0^l f_{deviation} \tag{34}$$

$$f_{deviation} = \sqrt{(y_m - y_{ml})^2 + (z_m - z_{ml})^2} \tag{35}$$

where, y_m and z_m are the y and z coordinates respectively of the m th waypoint of obtained path, y_{ml} and z_{ml} are the y and z coordinates respectively corresponding to the m th waypoint on the shortest path between the start and the target position. Further, to generate the flyable path for UAVs, the obtained path is smoothen by employing cubic spline curves [35,36].

4.2 Simulation Environment

A real world 3D UAV flight environment will contain several threats. Hence, in order to prove the efficiency of the developed algorithm, a simulation environment that resembles the real world civilian environment is required. Therefore, for simulating the results a 3D flight environment consisting of different kinds of obstacles is created. The 3D environment constitutes a Gaussian obstacle, cone, hemispheres, and cylinders of various sizes. The configurations of the simulated obstacles are presented in Tab. 1. The proposed algorithm is utilised to minimise the cost function to obtain an optimal flight path. Finally, all the simulation work are accomplished on the MATLAB 2019a programming platform installed in the Windows-10 operating system with an AMD Ryzen 5 (2.10 GHz) processor and 8GB RAM.

Table 1: Obstacle configurations

Obstacle	Base center	Radius (m)	Height (m)
Gaussian	(0, 0, 0)	-	7
Cone	(−17, 7, 0)	2.5	10
Hemispheres	(6, 7, 0)	5	-
	(20, 16, 0)	5	-
Cylinders	(12, 0, 0)	3.5	8
	(22, 0, 0)	5	12
	(25, 25, 0)	2.3	9
	(15, 8, 0)	2	10
	(10, 15, 0)	4	8

5 Results and Discussions

5.1 Benchmark Function Analysis

The performance of the developed QGA is evaluated on 22 benchmark functions as mentioned in Supplementary A. The obtained simulated results are compared with GWO, AOA, RLGWO, EAOA, and PSO. The results are analysed for 20 runs, with the maximum number of iterations limited to 1000. The population size is chosen as 30 for all the algorithms for fair comparison. Further, the statistical performance parameters such as Mean and Standard Deviation are computed using the obtained simulated results from 20 runs and tabulated in Tab. 2. Moreover, for the estimation of the rank and overall performance validation, Friedman test is performed and average rank based on the test is also rendered in Tab. 2. The obtained results reveals that QGA is able to provide optimal or near optimal mean values compared with GWO, AOA, PSO, EAOA, and RLGWO for 11 benchmark functions. GWO is resulting in optimal average solutions for 9 benchmark functions, whereas, RLGWO provided optimal average results for 5 only. Therefore, QGA outperforms others by minimum margin of approximately 22%. This is also validated by the Friedman test which estimates the average rank of QGA as 2.31 that is lower than 7.60% by its nearest counterpart. Conclusively, the QGA is ranked 1 as per the Friedman test among the 6 algorithms under consideration.

Table 2: Mean and standard deviation of the optimized cost function values for benchmark functions

Function	Parameter	RLGWO	GWO	AOA	QGA	PSO	EAOA
F1	Mean	9.98E − 34	9.95E − 59	7.82E − 14	**1.00E − 72**	1.04E − 06	4.36E + 04
	SD	4.23E − 33	6.79E − 59	3.67E − 13	4.49E − 72	3.63E − 06	1.36E + 04
F2	Mean	1.47E − 14	1.10E − 34	3.92E − 05	**3.75E − 42**	8.87E + 00	2.47E + 11
	SD	6.06E − 14	2.48E − 34	1.84E − 04	1.68E − 41	8.29E + 00	8.12E + 11
F3	Mean	6.15E − 03	8.77E − 16	2.92E − 25	**8.86E − 53**	1.53E + 03	9.63E + 04
	SD	2.74E − 02	1.41E − 14	1.37E − 24	3.49E − 52	2.84E + 03	3.36E + 04
F4	Mean	1.56E − 04	1.44E − 14	3.84E − 23	2.37E − 29	1.29E + 00	**1.03E − 175**
	SD	3.94E − 04	2.60E − 14	1.78E − 22	7.31E − 29	7.79E − 01	0.00E + 00
F5	Mean	2.80E + 01	**2.69E + 01**	2.70E + 01	2.75E + 01	2.25E + 04	1.67E + 08
	SD	2.74E + 00	4.75E − 01	8.80E − 01	8.03E − 01	4.00E + 04	7.63E + 07
F6	Mean	2.89E + 00	**5.75E − 01**	2.96E + 00	1.51E + 00	4.95E + 02	5.04E + 04
	SD	6.41E + 00	3.40E − 01	1.03E + 00	4.49E − 01	2.21E + 03	1.36E + 04
F7	Mean	2.21E − 03	8.41E − 04	5.02E − 02	**7.39E − 04**	7.01E − 01	2.47E + 01
	SD	2.61E − 03	3.44E − 04	1.53E − 01	4.28E − 04	3.00E + 00	2.82E + 01
F8	Mean	1.72E + 01	**5.00E − 02**	9.36E + 01	1.14E − 01	1.44E + 02	3.56E + 02
	SD	3.30E + 01	2.33E − 14	3.67E + 01	5.12E − 01	2.90E + 01	5.63E + 01
F9	Mean	3.28E − 06	1.62E − 14	1.33E − 01	**6.93E − 15**	3.13E + 00	2.00E + 01
	SD	1.46E − 05	2.15E − 15	6.23E − 01	2.60E − 15	3.96E + 00	7.69E − 01
F10	Mean	6.01E − 02	2.58E − 03	8.57E − 04	**1.52E − 03**	4.61E + 00	4.50E + 02
	SD	2.26E − 01	2.46E − 03	4.02E − 03	4.93E − 03	2.03E + 01	1.20E + 02
F11	Mean	7.75E − 02	**3.56E − 02**	1.97E + 00	8.78E − 02	2.08E − 01	3.97E + 08
	SD	4.30E − 02	3.30E − 02	4.53E + 00	4.36E − 02	4.55E − 01	2.04E + 08

(Continued)

Table 2 (continued)

Function	Parameter	RLGWO	GWO	AOA	QGA	PSO	EAOA
F12	Mean	1.18E + 00	**5.35E − 01**	3.60E + 00	1.21E + 00	4.20E − 01	7.38E + 08
	SD	3.28E − 01	2.12E − 01	2.40E + 00	2.71E − 01	1.38E + 00	4.45E + 08
F13	Mean	4.96E + 00	5.06E + 00	4.91E + 00	6.21E + 00	**1.20E + 00**	4.64E + 00
	SD	4.62E + 00	4.88E + 00	4.35E + 00	5.22E + 00	6.11E − 01	3.05E + 00
F14	Mean	8.10E − 04	1.33E − 03	3.88E − 03	4.01E − 04	6.81E − 03	4.12E − 03
	SD	4.39E − 04	9.38E − 03	7.43E − 03	**2.88E − 04**	9.13E − 03	4.52E − 03
F15	Mean	**−1.03E + 00**	−1.03E + 00	−9.82E−01	**−1.03E + 00**	**−1.03E + 00**	−1.02E + 00
	SD	2.40E − 09	6.04E − 02	2.36E − 03	1.20E − 09	2.28E − 16	1.90E − 02
F16	Mean	**3.98E − 01**	3.98E − 01	3.87E − 01	**3.98E − 01**	**3.98E − 01**	4.39E − 01
	SD	2.21E − 07	0.00E + 00	3.74E − 02	2.49E − 07	0.00E + 00	4.19E − 02
F17	Mean	**3.00E + 00**	**3.00E + 00**	2.86E + 00	**3.00E + 00**	**3.00E + 00**	3.35E + 00
	SD	1.06E − 05	0.00E + 00	0.00E + 00	5.21E − 06	1.04E − 15	7.82E − 01
F18	Mean	−3.86E + 00	−3.86E + 00	−3.68E + 00	**−3.86E + 00**	−3.74E + 00	−3.71E + 00
	SD	1.67E − 03	2.40E − 03	2.42E − 03	3.03E − 03	2.96E − 01	1.33E − 01
F19	Mean	−3.28E + 00	**−3.25E + 00**	−3.08E + 00	−3.28E + 00	−3.20E + 00	−2.12E + 00
	SD	6.85E − 02	7.64E − 02	9.81E − 02	5.93E − 02	1.12E − 01	5.07E − 01
F20	Mean	−8.76E + 00	−9.14E + 00	−5.39E + 00	−7.35E + 00	−6.76E + 00	−2.40E + 00
	SD	2.21E + 00	1.56E + 00	3.76E + 00	2.60E + 00	3.57E + 00	1.91E + 00
F21	Mean	**−1.04E + 01**	**−1.04E + 01**	−4.45E + 00	−8.81E + 00	−7.07E + 00	−2.98E + 00
	SD	4.48E − 04	4.81E + 00	2.98E + 00	2.50E + 00	3.21E + 00	1.61E + 00
F22	Mean	**−1.05E + 01**	**−1.05E + 01**	−4.67E + 00	−1.00E + 01	−7.02E + 00	−2.61E + 00
	SD	7.02E − 04	2.26E − 04	3.40E + 00	1.66E + 00	4.00E + 00	1.50E + 00
Average rank		2.95	2.5	3.77	2.31	4.0	5.45
Total rank		3	2	4	1	5	6

Further, the optimum value for each benchmark are computed against the simulated runs and tabulated in Tab. 3 which is also used to select the convergence curves of the respective benchmark. The QGA is able to provide optimal values for 14 functions. Hence, for 63.60% of the benchmark, the QGA provides an optimal or near optimal solutions for the considered 22 benchmark functions; whereas, GWO, AOA, RLGWO, PSO and EAOA achieved the optimum values for 59%, 59%, 50%, 50% and 18.18% functions, respectively. Therefore, the QGA dominates all the other comparison algorithm by a significant margin of 4.6%, 4.6%, 13.07%, 13.60% and 45.42%, respectively.

The *P* values evaluated for Wilcoxon rank-sum test are denoted in Tab. 4. The *P* value less than 0.05 indicates the statistical significance of QGA compared to the other algorithms. When compared with PSO, the QGA is achieving statistically significant results for 14 functions. For 12 benchmark functions, the QGA provides statistically significant results when compared with GWO. Except for the benchmark functions, F3, F4, F13, F19, and F20, the cost function values achieved by QGA is statistically significant

than AOA. While comparing with EAOA, the QGA attains statistically significant cost function values for all the considered functions except the benchmark function F13. The comparison with RLGWO shows that for 12 functions, the QGA is achieving statistically significant results. Therefore, it is perceived that according to Wilcoxon rank-sum test also, the QGA exceedingly dominates all the other employed algorithms.

Table 3: Best values of cost function obtained for benchmark functions

Function	RLGWO	GWO	AOA	QGA	PSO	EAOA
F1	4.11E − 48	3.79E − 61	**1.33E − 171**	8.12E − 127	6.24E − 18	2.12E + 04
F2	4.23E − 30	7.23E − 36	**7.49E − 91**	6.77E − 70	2.00E − 02	4.22E + 01
F3	3.08E − 13	1.41E − 19	**5.87E − 114**	1.58E − 100	1.36E + 00	3.90E + 04
F4	3.10E − 10	1.39E − 15	3.11E − 50	4.97E − 77	3.36E − 01	**3.78E − 211**
F5	2.63E + 01	2.61E + 01	2.60E + 01	2.62E + 01	**1.81E + 01**	1.55E + 07
F6	4.90E − 01	**1.56E − 05**	8.37E − 01	5.01E − 01	4.22E − 13	2.67E + 04
F7	6.33E − 04	5.33E − 04	7.30E − 03	**2.66E − 04**	1.56E − 02	6.04E + 00
F8	**0.00E + 00**	**0.00E + 00**	4.63E + 01	**0.00E + 00**	8.95E + 01	2.42E + 02
F9	2.22E − 14	1.51E − 14	4.00E − 14	**4.44E − 15**	1.15E − 06	1.82E + 01
F10	**0.00E + 00**	**0.00E + 00**	5.55E − 16	**0.00E + 00**	8.11E − 11	2.72E + 02
F11	3.48E − 02	6.60E − 03	9.10E − 02	3.40E − 02	**2.39E − 14**	1.02E + 08
F12	9.62E − 01	1.76E − 01	2.17E + 00	8.14E − 01	**4.35E − 12**	4.71E + 07
F13	**9.98E − 01**	**9.98E − 01**	**9.98E − 01**	**9.98E − 01**	**9.98E − 01**	1.00E + 00
F14	3.07E − 04	**3.08E − 04**	**3.08E − 04**	**3.08E − 04**	3.07E − 04	3.07E − 04
F15	**−1.03E + 00**	**−1.03E + 00**	**−1.03E + 00**	**−1.03E + 00**	**−1.03E + 00**	−1.**03E + 00**
F16	**3.98E − 01**	**3.98E − 01**	**3.98E − 01**	**3.98E − 01**	**3.98E − 01**	**3.98E − 01**
F17	**3.00E + 00**	**3.00E + 00**	**3.00E + 00**	**3.00E + 00**	**3.00E + 00**	**3.00E + 00**
F18	**−3.86E + 00**	**−3.86E + 00**	**−3.86E + 00**	**−3.86E + 00**	−3.85E + 00	−3.69E + 00
F19	**−3.32E + 00**	**−3.32E + 00**	**−3.32E + 00**	**−3.32E + 00**	**−3.32E + 00**	−2.91E + 00
F20	**−1.02E + 01**	**−1.02E + 01**	**−1.02E + 01**	**−1.02E + 01**	**−1.02E + 01**	−8.66E + 00
F21	**−1.04E + 01**	**−1.04E + 01**	**−1.04E + 01**	**−1.04E + 01**	**−1.04E + 01**	−5.84E + 00
F22	**−1.05E + 01**	**−1.05E + 01**	**−1.05E + 01**	**−1.05E + 01**	**−1.05E + 01**	−7.74E + 00

Table 4: Wilcoxon rank-sum test

Function	QGA *vs.* PSO	QGA *vs.* GWO	QGA *vs.* AOA	QGA *vs.* EAOA	QGA *vs.* RLGWO
F1	1.01E–07	6.8E–08	0.008352	6.8E–08	6.77E–08
F2	6.8E–08	6.8E–08	0.031517	6.8E–08	6.8E–08
F3	6.8E–08	6.8E–08	0.113551	6.8E–08	6.8E–08
F4	6.8E–08	6.8E–08	0.076431	6.8E–08	6.8E–08
F5	0.243594	0.147847	0.000758	6.8E–08	0.903116
F6	1.2E–06	1.58E–06	2.6E–05	6.8E–08	0.967635

(Continued)

Table 4 (continued)

Function	QGA *vs.* PSO	QGA *vs.* GWO	QGA *vs.* AOA	QGA *vs.* EAOA	QGA *vs.* RLGWO
F7	6.8E−08	0.06388	6.8E−08	6.8E−08	0.009786
F8	1.51E−08	0.499262	1.51E−08	1.51E−08	2.32E−06
F9	4.37E−08	2.56E−07	4.19E−08	4.37E−08	4.31E−08
F10	2.25E−07	0.573704	5.21E−06	1.51E−08	0.000466
F11	0.060111	0.000179	1.58E−06	6.8E−08	0.560852
F12	1.2E−06	1.43E−07	1.06E−07	6.8E−08	0.579218
F13	1.93E−07	0.281366	0.439459	0.903112	0.860405
F14	0.132627	0.029084	0.013273	1.43E−07	0.000563
F15	8.01E−09	2.78E−07	1.51E−08	6.8E−08	0.001349
F16	8.01E−09	8.01E−09	1.13E−08	6.8E−08	8.29E−05
F17	4.91E−08	2.99E−08	2.99E−08	6.8E−08	0.839232
F18	0.388414	0.175637	0.001944	6.8E−08	0.285305
F19	0.283263	0.143627	0.635038	6.8E−08	0.049864
F20	0.421973	0.207791	0.187628	1.57E−06	0.189523
F21	0.559911	0.19835	1.99E−05	9.13E−07	0.776391
F22	0.594823	0.009473	0.000196	9.17E−08	0.096196

Finally, the performance of all the employed algorithms is also compared based on convergence plots and sample of these are presented in Fig. 3. It is very obvious from these plots, the QGA shows the faster convergence compared to others and therefore, the adaptive algorithm selection mechanism based on Q learning enables faster convergence without compromising accuracy.

5.2 Path Planning Results

The developed algorithm is employed to generate 3D flight path in the simulated environment. The maximum number of iterations utilised is 1500, with a population size of 50, for all the algorithms. The results are compared with GWO, AOA, EAOA, RLGWO, and PSO. The Fig. 4 provides the top view of the simulated environment and the generated paths. The 3D view of the created paths in the flight environment is depicted in Fig. 5. The simulated results confirmed that QGA is providing the optimal flight path. From these figures, it is witnessed that only QGA, RLGWO and AOA are able to avoid the collision while estimating the optimum path however, both RLGWO and AOA are consuming higher fuel. Therefore, only QGA successfully estimates the optimum flyable path with all the considered constraints. Not only PSO, EAOA and GWO fails to avoid collision but also the length of the traced path is higher compared to the path generated by QGA.

By comprehensive analysis of the convergence curves shown in Fig. 6, it is quite evident that though QGA is converging slower as compared to GWO, EAOA, and PSO but, the QGA is able to achieve collision free optimal flight path. Also, the final optimized objective function values achieved by QGA is significantly lower than that of EAOA, PSO, and GWO.

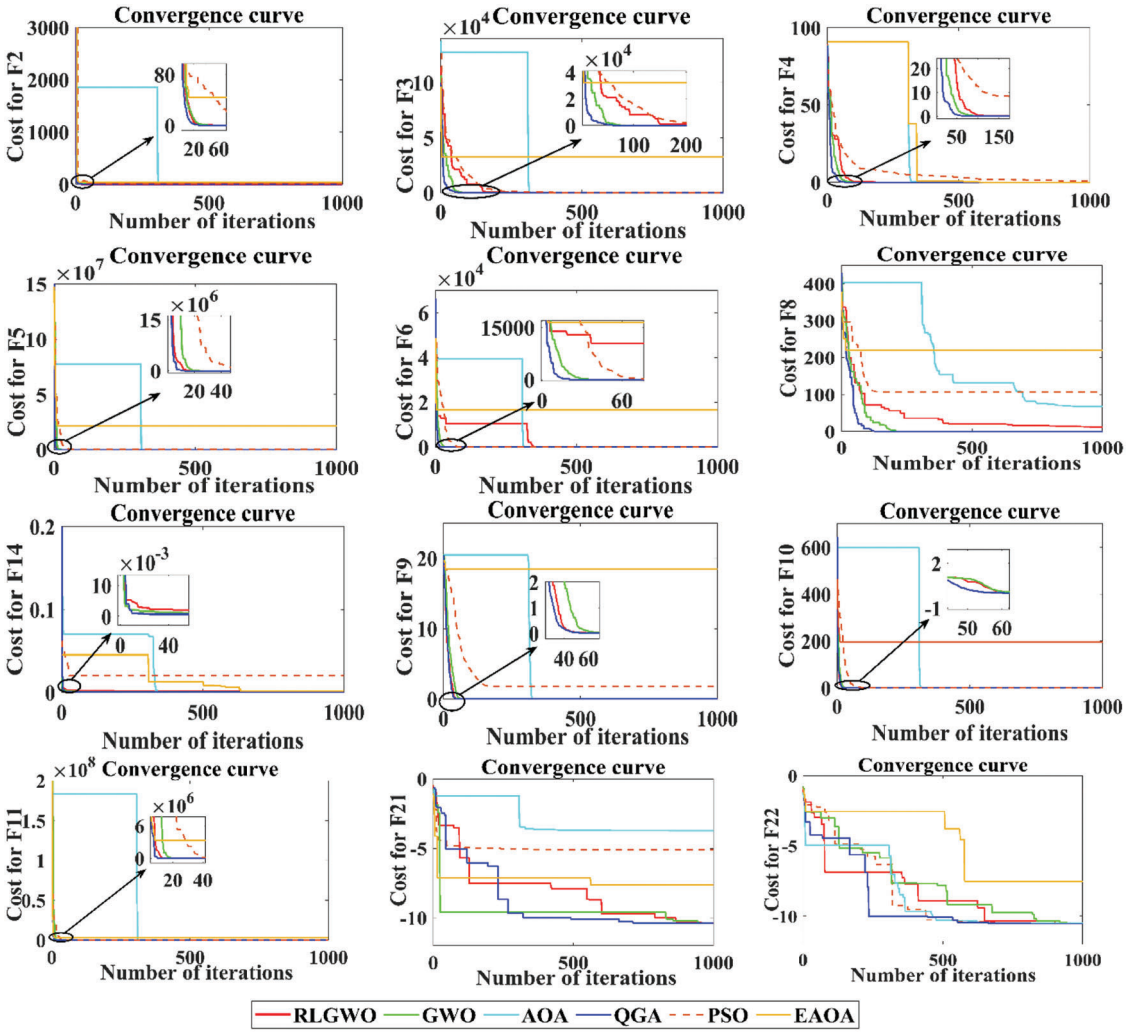

Figure 3: Convergence curves for selected benchmark functions

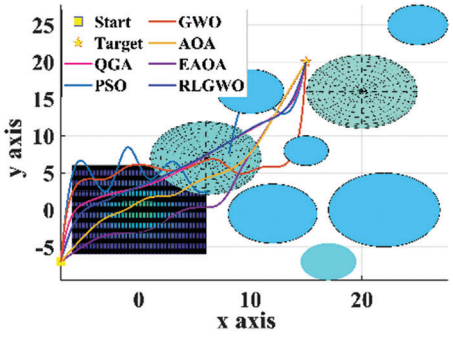

Figure 4: 2D top view of the paths

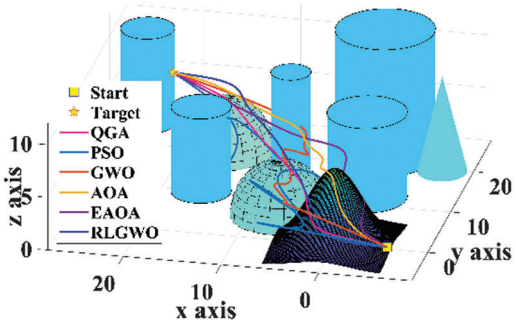

Figure 5: 3D view of the paths

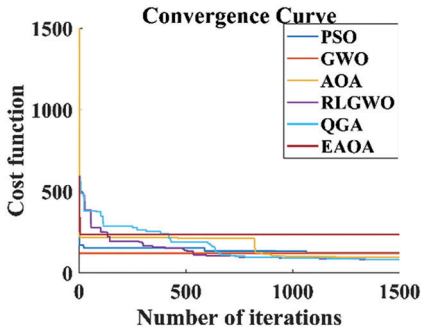

Figure 6: Convergence curve for path planning

Tab. 5 demonstrate the ability to generate the optimal flyable path for UAVs by the developed QGA with others based on the statistical means. All the employed algorithms are used to estimate the path for 20 runs. The obtained results reveals that the developed QGA achieved the optimum cost value function which is approximately 28.50%, 26.83%, 11.00%, 60.90% and 5.80% lesser than PSO, GWO, AOA, EAOA and RLGWO, respectively. Similarly, Mean value and Standard Deviations are estimated as 4.31% and 2.51% lesser than its nearest competitor (RLGWO).

Table 5: Cost function values for UAV path planning

S. No.	Algorithms	Best cost function value	Mean Value	Standard Deviation
1	PSO [34]	145.44	200.87	6.62E-01
2	GWO [37]	142.13	180.52	5.24E-02
3	AOA [22]	116.84	180.46	8.51E-01
4	EAOA [33]	265.90	296.40	5.02E-02
5	RLGWO [27]	110.40	123.59	6.38E-02
6	QGA	104.00	118.26	6.22E-02

6 Conclusions

In the present work, a novel Q learning control GWO-AOA (QGA) has been proposed and validated on 22 different benchmark functions based on convergence curves, Freidman test, and Wilcoxon rank-sum test. Additionally, the employability of QGA has been investigated for path planning of UAVs in three dimensional environment. The experimental results reveals that the adaptive algorithm selection mechanism based on Q learning enables QGA to achieve better exploration-exploitation capabilities

because of which it dominates all the compared algorithms for both benchmark evaluations and path planning of UAVs. Moreover, the cubic-spline curves has been utilized to smooth the generated flight path for various adopted algorithms. The employability and superiority of the compared algorithms has been comprehensively endorsed on the basis various statistical results and convergence curve analysis. Though, the developed QGA achieved optimum cost for both benchmark and path planning of UAVs, it is not the fastest one which open the window for the future research.

Acknowledgement: The authors extend their appreciation to Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R66), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Funding Statement: This research was funded by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R66), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] X. Yu, C. Li and J. F. Zhou, "A constrained differential evolution algorithm to solve UAV path planning in disaster scenarios," *Knowledge-Based Systems*, vol. 204, pp. 106209, 2020.

[2] M. C. Tatum and J. Liu, "Unmanned aircraft system applications in construction," *Procedia Engineering*, vol. 196, no. June, pp. 167–175, 2017.

[3] H. Gupta and O. P. Verma, "Monitoring and surveillance of urban road traffic using low altitude drone images: A deep learning approach," *Multimedia Tools and Applications*, vol. 81, no. 14, pp. 19683–19703, 2021.

[4] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[5] S. Ghambari, J. Lepagnot, L. Jourdan and L. Idoumghar, "A comparative study of meta-heuristic algorithms for solving UAV path planning," in *Proc. of the 2018 IEEE Symposium Series on Computational Intelligence*, SSCI 2018, Banglore, India, pp. 174–181, 2019.

[6] M. Cakir, "2D path planning of UAVs with genetic algorithm in a constrained environment," in *6th Int. Conf. on Modeling, Simulation, and Applied Optimization 2015*, ICMSAO, Istanbul, Turkey, 2015.

[7] T. Turker, O. K. Sahingoz and G. Yilmaz, "2D path planning for UAVs in radar threatening environment using simulated annealing algorithm," in *2015 Int. Conf. on Unmanned Aircraft Systems*, ICUAS 2015, Denver, Colorado, USA, pp. 56–61, 2015.

[8] C. Huang and J. Fei, "UAV path planning based on particle swarm optimization with global best path competition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 32, no. 6, pp. 1859008, 2018.

[9] S. Shao, Y. Peng, C. He and Y. Du, "Efficient path planning for UAV formation via comprehensively improved particle swarm optimization," *ISA Transactions*, vol. 97, pp. 415–430, 2020.

[10] J. J. Shin and H. Bang, "UAV path planning under dynamic threats using an improved PSO algorithm," *International Journal of Aerospace Engineering*, vol. 2020, pp. 17, 2020.

[11] X. Wu, W. Bai, Y. Xie, X. Sun, C. Deng *et al.,* "A hybrid algorithm of particle swarm optimization, metropolis criterion and RTS smoother for path planning of UAVs," *Applied Soft Computing Journal*, vol. 73, pp. 735–747, Dec. 2018.

[12] Y. Ji, X. Zhao and J. Hao, "A novel UAV path planning algorithm based on double-dynamic biogeography-based learning particle swarm optimization," *Mobile Information Systems*, vol. 2022, pp. 23, 2022.

[13] W. Jiang, Y. Lyu, Y. Li, Y. Guo and W. Zhang, "UAV path planning and collision avoidance in 3D environments based on POMPD and improved grey wolf optimizer," *Aerospace Science and Technology*, vol. 121, pp. 107314, 2022.

[14] C. Qu, W. Gai, J. Zhang and M. Zhong, "A novel hybrid grey wolf optimizer algorithm for unmanned aerial vehicle (UAV) path planning," *Knowledge-Based Systems*, vol. 194, no. 105530, pp. 105530, 2020.

[15] L. Ji-Xiang, L. -J. Yan, Z. -M. Cai, J. -S. Pan, X. -K. He *et al.,* "A new hybrid algorithm based on golden eagle optimizer and grey wolf optimizer for 3D path planning of multiple UAVs in power inspection," *Neural Computing and Applications*, vol. 34, pp. 11911–11936, 2022.

[16] X. Zhou, F. Gao, X. Fang and Z. Lan, "Improved bat algorithm for UAV path planning in three-dimensional space," *IEEE Access*, vol. 9, pp. 20100–20116, 2021.

[17] U. Goel, S. Varshney, A. Jain, S. Maheshwari and A. Shukla, "Three dimensional path planning for UAVs in dynamic environment using glow-worm swarm optimization," *Procedia Computer Science*, vol. 133, pp. 230–239, 2018.

[18] A. -D. Tang, T. Han, H. Zhou and L. Xie, "An improved equilibrium optimizer with application in unmanned aerial vehicle path planning," *Sensors*, vol. 21, no. 5, pp. 1814, 2021.

[19] S. Li and Y. Deng, "Quantum-entanglement pigeon-inspired optimization for unmanned aerial vehicle path planning," *Aircraft Engineering and Aerospace Technology*, vol. 91, no. 1, pp. 171–181, 2019.

[20] G. Jain, G. Yadav, D. Prakash, A. Shukla and R. Tiwari, "MVO-Based path planning scheme with coordination of UAVs in 3-D environment," *Journal of Computational Science*, vol. 37, pp. 101016, 2019.

[21] H. Xu, S. Jiang and A. Zhang, "Path planning for unmanned aerial vehicle using a mix-strategy-based gravitational search algorithm," *IEEE Access*, vol. 9, pp. 57033–57045, 2021.

[22] F. A. Hashim, K. Hussain, E. H. Houssein, M. S. Mabrouk and W. Al-Atabany, "Archimedes optimization algorithm: A new metaheuristic algorithm for solving optimization problems," *Applied Intelligence*, vol. 51, no. 3, pp. 1531–1551, 2021.

[23] H. Karami, M. Valikhan, S. Farzin and S. Mirjalili, "Flow direction algorithm (FDA): A novel optimization approach for solving optimization problems," *Computers and Industrial Engineering*, vol. 156, pp. 107224, 2021.

[24] A. Yadav, "AEFA : Artificial electric field algorithm for global optimization," *Swarm and Evolutionary Computing*, vol. 48, no. March, pp. 93–108, 2019.

[25] S. Aggarwal and N. Kumar, "Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges," *Computer Communications*, vol. 149,. Elsevier B.V., pp. 270–299, Jan. 01, 2020.

[26] N. Mazyavkina, S. Sviridov, S. Ivanov and E. Burnaev, "Reinforcement learning for combinatorial optimization: A survey," *Computers and Operations Research*, vol. 134, no. August 2020, pp. 105400, 2021.

[27] C. Qu, W. Gai, M. Zhong and J. Zhang, "A novel reinforcement learning based grey wolf optimizer algorithm for unmanned aerial vehicles (UAVs) path planning," *Applied Soft Computing Journal*, vol. 89, no. pp. 106099, 2020.

[28] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computing*, vol. 1, no. 1, pp. 67–82, 1997.

[29] W. E. I. Zhang, S. A. I. Zhang, F. Wu and Y. Wang, "Path planning of UAV based on improved adaptive grey wolf optimization algorithm," *IEEE Access*, vol. 9, pp. 89400–89411, 2021.

[30] G. Huang, Y. Cai, J. Liu, Y. Qi and X. Liu, "A novel hybrid discrete grey wolf optimizer algorithm for multi-UAV path planning," *Journal of Intelligent Robotic Systems*, vol. 103, no. 3, pp. 1–18, 2021.

[31] H. Mohammed and T. Rashid, "A novel hybrid GWO with WOA for global numerical optimization and solving pressure vessel design," *Neural Computing and Applications*, vol. 32, no. 18, pp. 14701–14718, 2020.

[32] M. Wunder, M. Littman and M. Babes, "Classes of multiagent Q-learning dynamics with ε-greedy exploration," in *ICML 2010-Proc., 27th Int. Conf. on Machine Learning*, Haifa, Israel, pp. 1167–1174, 2010.

[33] A. S. Desuky, S. Hussain, S. Kausar, A. Islam and L. M. El Bakrawy, "EAOA : An enhanced archimedes optimization algorithm for feature selection in classification," *IEEE Access*, vol. 9, no. February 2022, pp. 120795–120814, 2021.

[34] H. Gupta, S. Kumar, D. Yadav, O. P. Verma, T. K. Sharma *et al.,* "Data analytics and mathematical modeling for simulating the dynamics of COVID-19 epidemic—a case study of India," *Electronics*, 2021 January 8; vol. 10, no. 2, pp. 127, 2021.

[35] A. Ravankar, A. A. Ravankar, Y. Kobayashi, Y. Hoshino and C. C. Peng, "Path smoothing techniques in robot navigation: State-of-the-art, current and future challenges," *Sensors (Switzerland)*, vol. 18, no. 9, pp. 1–30, 2018.

[36] M. S. M. Altaei and A. H. Mahdi, "Cubic spline based path planning for UAV," *International Journal of Computer Science and Mobile Computing*, vol. 8, no. 12, pp. 80–95, 2019.

[37] S. Mirjalili, S. M. Mirjalili and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, no. 2014, pp. 46–61, 2014.

Supplementary A

Function name	Function number	Function	Global optimum
Sphere	F1	$F_1(x) = \sum_{n=1}^{N} x_n^2$	0
Schwefel 2.22	F2	$F_2(x) = \sum_{n=1}^{N} \lvert x_n \rvert + \prod_{n=1}^{N} \lvert x_n \rvert$	0
Schwefel 1.2	F3	$F_3(x) = \sum_{i=1}^{N} \left(\sum_{n=1}^{i} x_n^2 \right)$	0
Schwefel 2.21	F4	$F_4(x) = \sum_{n=1}^{N} 100(x_{n+1} - x_n^2) + (x_n)^2$	0
Rosenbrock	F5	$F_5(x) = \sum_{n=1}^{N} 100(x_{n+1} - x_n^2)^2 + (x_n)^2$	0
Step	F6	$F_6(x) = \sum_{n=1}^{N} (x_n + 0.5)^2$	0
Quartic	F7	$F_7(x) = \sum_{n=1}^{N} n x_n^4 + rand[0,\ 1]$	0
Rastrigin	F8	$F_8(x) = \sum_{n=1}^{N} (x_n^2 - 10\cos(2\pi x_n) + 10)$	0
Ackley	F9	$F_9(x) = -20\exp\left(-0.2\sqrt{\frac{1}{N}\sum_{n=1}^{N} x_n^2} \right) - exp\left(\frac{1}{N}\sum_{n=1}^{N} \cos(2\pi x_n) \right)$	0
Greiwank	F10	$F_{10}(x) = \frac{1}{4000}\left(\sum_{n=1}^{N} (x_n - 100)^2 \right) - \left(\prod_{n=1}^{N} cos\left(\frac{x_k - 100}{\sqrt{k}} \right) \right) + 1$	0
Penalized	F11	$F_{11}(x) = \frac{\pi}{N}\left\{ 10\sin(\pi y_1) + \sum_{n=1}^{N-1} (y_n - 1)^2 [1 + 10sin^2(\pi y_{n+1})] + (y_N - 1)^2 \right\} + \sum_{n=1}^{N} u(x_n,\ 10,\ 100,\ 4)$ $y_n = 1 + \frac{x_n + 1}{4}$ $u(x_n,\ a,\ k,\ m) = \begin{cases} k(x_n - a)^m & x_n > a \\ 0 & -a < x_n < a \\ k(-x_n - a)^m & x_n < -a \end{cases}$	0

(Continued)

(continued)

Function name	Function number	Function	Global optimum
Penalized 2	F12	$F_{12}(x) = 0.1\left\{sin^2(3\pi x_1) + \sum_{n=1}^{N}(x_n-1)^2[1+sin^2(3\pi x_n+1)] + (x_N-1)^2[1+sin^2(2\pi x_n)]\right\} + \sum_{n=1}^{N} u(x_n,\ 5,\ 100,\ 4)$	0
Foxholes	F13	$F_{13}(x) = \left[\frac{1}{500} + \sum_{n=1}^{25}\frac{1}{n+\sum_{a=1}^{2}(x_n - A_{na})^6}\right]^{-1}$	0.998
Kowalik	F14	$F_{14}(x) = \sum_{n=1}^{11}\left[a_n - \frac{x_1(b_n^2+b_n x_2)}{b_n^2+b_n x_3+x_4}\right]$	3.08E-04
Six Hump Camel Back	F15	$F_{15}(x) = \left(4-2.1x_1^2+\frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4+4_2^2)x_2^2$	−1.0316
Branin	F16	$F_{16}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1-\frac{1}{8\pi}\right)cosx_1 + 10$	0.3979
Goldstein-Price	F17	$F_{17}(x) = \sum_{n=1}^{N}[1+(x_1+x_2+1)^2(19-14x_1+13x_1^2-14x_2+6x_1x_2+3x_2^2)]$	3
Hartmann 3	F18	$F_{18}(x) = -\sum_{n=1}^{4}\alpha_n exp\left[-\sum_{m=1}^{3}A_{nm}(x_m-C_{nm})^2\right]$, where $\alpha = (1.0,\ 1.2,\ 3.0,\ 3.2)^T$ $A = \begin{bmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix}$ $C = 10^{-4}\begin{bmatrix} 3689 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8828 \end{bmatrix}$	−3.86
Hartmann 6	F19	$F_{19}(x) = -\sum_{n=1}^{4}\alpha_n exp\left[-\sum_{m=1}^{6}B_{nm}(x_m-D_{nm})^2\right]$, where $\alpha = (1.0,\ 1.2,\ 3.0,\ 3.2)^T$ $B = \begin{bmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{bmatrix}$ $D = 10^{-4}\begin{bmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1104 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{bmatrix}$	−3.322

(continued)

Function name	Function number	Function	Global optimum
Shekel 5	F20	$F_{20}(x) = -\sum_{n=1}^{5}\left[\sum_{m=1}^{4}(x_m - P_{nm})^2 + \beta_n\right]^{-1}$, where $\beta = (0.1,\ 0.2,\ 0.2,\ 0.4,\ 0.4,\ 0.6,\ 0.3,\ 0.7,\ 0.5,\ 0.5)^T$ $P = \begin{bmatrix} 4 & 1 & 8 & 6 & 3 & 2 & 5 & 8 & 6 & 7.0 \\ 4 & 1 & 8 & 6 & 7 & 9 & 3 & 1 & 2 & 3.6 \\ 4 & 1 & 8 & 6 & 3 & 2 & 5 & 8 & 6 & 7.0 \\ 4 & 1 & 8 & 6 & 7 & 9 & 3 & 1 & 2 & 3.6 \end{bmatrix}$	−10.1532
Shekel 7	F21	$F_{21}(x) = -\sum_{n=1}^{7}\left[\sum_{m=1}^{4}(x_m - P_{nm})^2 + \beta_n\right]^{-1}$, where $\beta = (0.1,\ 0.2,\ 0.2,\ 0.4,\ 0.4,\ 0.6,\ 0.3,\ 0.7,\ 0.5,\ 0.5)^T$ $P = \begin{bmatrix} 4 & 1 & 8 & 6 & 3 & 2 & 5 & 8 & 6 & 7.0 \\ 4 & 1 & 8 & 6 & 7 & 9 & 3 & 1 & 2 & 3.6 \\ 4 & 1 & 8 & 6 & 3 & 2 & 5 & 8 & 6 & 7.0 \\ 4 & 1 & 8 & 6 & 7 & 9 & 3 & 1 & 2 & 3.6 \end{bmatrix}$	−10.4028
Shekel 10	F22	$F_{22}(x) = -\sum_{n=1}^{10}\left[\sum_{m=1}^{4}(x_m - P_{nm})^2 + \beta_n\right]^{-1}$, where $\beta = (0.1,\ 0.2,\ 0.2,\ 0.4,\ 0.4,\ 0.6,\ 0.3,\ 0.7,\ 0.5,\ 0.5)^T$ $P = \begin{bmatrix} 4 & 1 & 8 & 6 & 3 & 2 & 5 & 8 & 6 & 7.0 \\ 4 & 1 & 8 & 6 & 7 & 9 & 3 & 1 & 2 & 3.6 \\ 4 & 1 & 8 & 6 & 3 & 2 & 5 & 8 & 6 & 7.0 \\ 4 & 1 & 8 & 6 & 7 & 9 & 3 & 1 & 2 & 3.6 \end{bmatrix}$	−10.5363