Multi-UAV Area Exploration

Ahmed Daw
German University in Cairo
Cairo, Egypt

Nabil Hesham
German University in Cairo
Cairo, Egypt

Amr Mahgoub
German University in Cairo
Cairo, Egypt

Mohamed Ashraf
German University in Cairo
Cairo, Egypt

Mohamed Yasser
German University in Cairo
Cairo, Egypt

Abdelrahman Manea
German University in Cairo
Cairo, Egypt

***Abstract*—This report explores multi-UAV area exploration optimization using metaheuristic algorithms, including Simulated Annealing (SA), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Teaching-Learning-Based Optimization (TLBO). These algorithms are applied to optimize point to point UAV flight paths, minimize exploration time, maximize coverage, and ensure constraint compliance. Comparative performance analyses indicate PSO's consistency, GA's superior path quality, and population-based techniques' reliability over non-population approaches. The study highlights algorithm fine-tuning's significance and outlines future work on dynamic obstacle handling and reinforcement learning integration.**

***Index Terms*—Unmanned Aerial Vehicles, Optimization, Metaheuristics, Fitness Function, UAV Path Planning, Genetic Algorithm, Simulated Annealing, Particle Swarm, Teaching Learning Based Optimization.**

I. Introduction

Unmanned Aerial Vehicles and Unmanned Ground Vehicles Area Exploration has been extensively researched, in terms of optimizing fuel & energy consumption, and increasing the explored area in a minimal time frame, Ropero et al, 2019, proposed a solution to solve the energy constraint problem in UAVs and the UGV functionality problem, by the cooperation of UAVs and UGVs, where the UAVs are the robotic system in charge of reaching target points, and the UGVs serve as charging stations, this approach allowed the researchers to increase travel distance, where their approach has been tested in multiple scenarios, demonstrating the ability to generate a coordinated plan between both agents [1].

II. Literature Review

The field of optimizing cooperative UAV area exploration has been extensively researched throughout the last decade, in terms of increasing explored area and reaching desired targets in minimal running time.

Xu et al, 2020 explored the field cooperative path planning for multi-UAV system as used in military and civil applications, such as target striking and regional surveillance, in order to increase the probability of task completion and minimizing the risk of the UAVs being captured, by proposing an optimized cooperative path planning for multi-UAV system optimizing a combination of fuel/energy consumption and risk of capture, under the constraints of exploration area and time, establishing a multi constraint objective optimization model, followed by the testing of multiple optimization algorithms, such as an improved version of the grey wolf optimization algorithm, where the algorithm is improved in three main aspects, population initialization, decoy factor updating and individual position updating, simulation results showcase that the algorithm has optimized the generated paths by lowering the path cost, while reaching faster convergence when compared to other tested techniques [2].

In their research, Qiming et al, 2021 have evaluated the use of different meta-heuristic optimization techniques in solving the UAV swarm search task. In their research, the goal was to detect targets as fast as possible, while using the least number of UAVs, in another simulation environment, the goal was to maximize the search area with the UAV swarm. The search area was divided into discrete cells of MxN size and the used UAV model was assumed to be the basic UAV model by treating each UAV as a particle in a 2-Dimensional space. Some of the constraints taken into consideration were the curve angle θ, flight speed v, flight altitude h.

In their evaluation, the researchers tested the Genetic Algorithm (GA), Ant Colony (AC), Particle Swarm (PSA), with other optimizing algorithms, in GA, when minimizing the time was taken as the objective function, the total task time was reduced by 65%, however, the algorithm needed a certain number of UAVs and showed linear increase in energy consumption as the number of UAVs increases [3].

In their article, Ramasamy et al, 2022 have explored parameter tuning using genetic algorithm with a set of constraints such as fuel limitations of the UAVs and the speed limitations of the UGVs, reducing the time and/or fuel consumption were important in assessing the validity of the algorithms, taking into consideration the resource and terrain constraints. The paper compared between the Genetic Algorithm and the Bayesian Optimization, despite the fact that the GA provided relatively similar results to BO, it required 3 times more local-search optimization and required 6 times the computation time [4].

In their paper, Wu et al 2020 discussed the same topic of cooperative UAV-UGV exploration, but in surveillance applications. In this paper, the path planning problem was formulated to be a 0-1 optimization problem, in which the on-off states of the discrete points are to be optimized. A hybrid algorithm, combining the Estimation of Distribution Algorithm (EDA) and the Genetic Algorithm (GA) was proposed to solve the problem. The goal is to minimize the required time by the vehicles to complete a circular path, where the objective function can be expressed as shown in equation 1.

$$J = max(\frac{D_u}{N_u * v_u}, \frac{D_g}{N_g * v_g}) \tag{1}$$

Where D_u & D_g are the lengths of the circular paths, and N_u & N_g are the number of drones and UGVs and v_u & v_g are their velocities.

The decision variables are the on-off states of the active points, where each point can either be "open" (1) or "closed" (0), and open points must be covered during exploration.

The constraints for UAVs in this paper lied in avoiding collisions with buildings, which are represented as inaccessible grids, also, the vehicles must cover all assigned 2D grids to ensure complete coverage, where the on-off states must satisfy the coverage constraint. In their results, they validated the superiority and rationality of the proposed hybrid algorithm, where the workload can be balanced between the UAVs and UGVs, moreover, the cooperative planning yielded better results than using either systems alone, by significantly reducing the total surveillance time, the suggested algorithm combined the strengths of EDA and GA's strengths, resulting in minimizing the time of task completion, while increasing adaptability since the path is being assessed during runtime rather than before runtime, increasing flexibility in real-world applications [5].

III. Problem Formulation

The project aims to develop an efficient algorithm for optimizing flight paths of multiple UAVs navigating a defined area. The UAVs must reach designated targets while avoiding restricted zones. The optimization focuses on minimizing exploration time, maximizing coverage efficiency, and ensuring targets are reached without entering prohibited areas. The primary goal is to minimize exploration time, improve coverage, and reduce penalties for missed targets and restricted zone violations, with a greater emphasis on exploration time. Simulations show UAVs prioritize rapid convergence to targets.

A. Decision Variables

The decision variables consist of the velocities of the UAVs in the x and y directions, in m/s, as well as their positions in x and y in m, a solution x_i is represented as follows:

$$x_i = [v_{xi}, v_{yi}, p_{xi}, p_{yi}] \tag{2}$$

where:

- v_{xi}, v_{yi} are the velocities in x and y directions (in ms^{-1})
- p_{xi}, p_{yi} are the positions in x and y (in m)

B. Constraints

The constraints are handled through a nonlinear constraint function, where 2 constraints are checked as follows:

- **Restricted Area:** For every UAV at every step, the euclidean distance between the current UAV position and the center of the restricted area is calculated, if the distance is less than the radius + some buffer distance defined in the code, a penalty is added to the solution.
- **Safe distance:** The minimum safe distance, d_{safe} is defined in the code, if the distance between 2 UAVs is less than the defined distance, the penalty is added in a different fashion, it is added in terms of a variable called number of violations.
- **Maximum UAV velocity:** The maximum UAV velocity was pre-defined in the code, and before populating any v_x or v_y into any of the algorithms solutions, the velocity is capped to be between the inequality:

$$-v_{max} \leq v_{UAV} \leq v_{max} \tag{3}$$

C. Objective Function

The cost function aims to maximize the explored area while minimizing the time taken, subject to penalties for constraint violations (e.g., communication and collision avoidance).

$$ObjectiveFunction = \alpha * T_{exploration} - \gamma * C_{eff} + \delta * Penalty_{targets} \tag{4}$$

Where:

- α, γ and δ are weight parameters for the time of exploration, Coverage efficiency and designated targets penalty.
- $T_{exploration}$ is a measure of the total time taken by UAVs to reach their designated targets, represented as a sum of the time taken by each UAV based on their velocity and the distance covered. $T_{exploration}$ is calculated by:

$$T_{exploration} = \frac{100}{v_{UAV}} \tag{5}$$

  where 100 is the maximum time of exploration, and v_{UAV} is the speed of each UAV, faster UAVs explore the targets in less time
- C_{eff} is a ratio between the area covered by the UAVs with the total area of the map, computing for the effective area explored. It can be calculated as follows:

$$C_{eff}(i) = \frac{C_{UAV}(i)}{Area} \tag{6}$$

  where $C_{eff}(i)$ is calculated as a sum of position components of each UAV, and $Area$ is predefined from the map dimensions
- $Penalty_{targets}$ is a penalty term that accounts for the distance between the UAV positions and their targets, encouraging the UAVs to stay close to the targets, where it can be calculated as follows:

$$penalty(i) = min_j ||p_{UAV}(i) - t_{target}(j)|| \tag{7}$$

  where $||p_{UAV}(i) - t_{target}(j)||$ is the euclidean distance between a UAV i and target j, the penalty is accounted

to be the smallest distance to any of the targets, so UAVs that are far from the targets have larger penalties.

METHODOLOGY

D. Simulated Annealing

The Simulated Annealing (SA) algorithm, inspired by metal annealing, was implemented to optimize UAV flight paths. SA has advantages like faster convergence but can yield locally optimal solutions due to its random acceptance of worse solutions. In our implementation, the algorithm runs multiple times to find the best step for each UAV, ensuring all targets are reached while adhering to constraints.

The process begins with random initialization of a solution, where UAV positions (px, py) are predefined, and velocities (vx, vy) are randomized within $[-v\_UAV\_max, v\_UAV\_max]$. The SA generates a new solution by perturbation, scaled by a step size of 1, then checks for constraints and penalties. If the new solution is better or accepted by probability, it's evaluated for the best solution overall and stored. The UAVs' positions are updated, and the temperature adjusts according to the cooling schedule. The algorithm also includes a function assigning unique targets to each UAV, addressing the issue of UAVs reaching only 1-2 targets.

The algorithm succeeded in allowing the UAVs to reach their targets, the used parameters were: α cooling = 0.8, max iterations per run = 60, $T_{initial}$ = 30, T_{final} = 10, iterations per temp change = 20.

E. Genetic Algorithm

The Genetic Algorithm (GA) follows Charles Darwin's theory of evolution, through survival of the most fit individuals, allowing them to pass their traits to the next generations, in our problem, we followed a similar manner, in initializing a random population to start our GA run, and as iterations progress, the fitness value decreases showcasing the evolution.

The GA runs multiple times (n generations) to generate a single point in the path. Each generation consists of an m x 4 matrix of chromosomes representing UAV data (vx, vy, px, py). A random population is generated initially, and fitness values are calculated. The best chromosomes are carried over to the next generation, while the others undergo crossover and mutation. Crossover uses arithmetic recombination with a constant interpolation factor (α). Mutation introduces noise to the worst individuals and some offspring based on the mutation ratio (GA.MutationRatio) and noise scale (GA.NoiseScale). Feasibility checks and constraints handling are applied similarly to earlier methods.

The GA parameters that were used to run the standard deviation tests were, population size = 20, number of generations = 60, elitism ratio = 0.2, crossover ratio = 0.7, mutation ratio = 0.1, with $\alpha = 0.7$, and noise scale = 0.2.

F. Particle Swarm Optimization

Particle Swarm Optimization (PSO) is inspired by the behavior of bird flocks searching for food or migrating. Each bird represents a particle, and the group forms a swarm. Like Genetic Algorithms (GA), PSO is a population-based, nature-inspired iterative technique. However, PSO is generally more computationally efficient than GA [6].

In order to implement PSO into our problem, we followed the following approach, starting off by initializing the swarm size and the maximum iterations per PSO run, with the cognitive and social weights for velocity calculation, then initializing a random swarm to be our birds' velocities in the first run, where the positions are defined as the initial positions of the UAVs, with initialization of the global and personal bests, the neighborhood topology used was the synchronous star topology, to check all the bird's neighbors to check for global best. The inertia weight term w is set to be variable, with $w_i = 0.99$ and $w_f = 0.6$, to create a balance between exploration and exploitation for the swarm during a run, the main loop starts with calculating the fitness value for all birds in the swarm, and checking for constraints, which were explained in section 2.1, if any of the constraints were violated, a scaled penalty term would be added to the fitness value of that bird, then the personal bests of the birds are checked and updated, as well as the global bests in the swarm, to update the velocity, we first extract the position and velocity terms in our decision variables, for each UAV, the cognitive and social terms of the velocity are calculated, which are then used to update the UAV positions.

The swarm is then updated with the new positions and velocities, when the number of iterations exceeds the maximum defined number, the swarm is then reinitialized, to avoid rapid convergence, since the personal and global best positions are set to be straight line very early on, so the entire swarm will exploit the same positions every time, a small portion of the swarm is randomized every 10 iterations, and the entire swarm is re-initialized after every point, which introduces necessary diversity to explore more promising solutions, without the fear of rapid convergence to sub-optimal solutions.

The parameters used to run the performance indices tests were: swarm size = 20, maximum iterations per run = 60, social and cognitive weights = 1.5, and a dynamic inertia weight of initial weight = 0.99 and a final weight of 0.6.

G. Teaching Learning Based Optimization

Teaching Learning Based Optimization (TLBO), is a meta-heuristic population based optimization algorithm, which was introduced by Rao, et al in 2011, mimicking the behavior of students and teachers learning in a classroom, where the population is considered to be as a class, and each solution is considered to be a student in the class, the algorithm optimizes the solutions over 2 phases, the teaching phase and the learning phase, where the teaching phase relies on the most fit solution (teacher) to improve the average fitness of the classroom, where all solutions gain knowledge from their teacher, and the learning phase relies on the behavior of students learning from

a partner, if the partner's information helps the student improve its fitness, it will learn from it, and if not, the student's fitness will move away from the partner [7]. Our implementation of the algorithm was mostly similar in approach to what was proposed in the literature, however, in the main while loop, which runs until all UAVs reach their targets, the fitness value gets evaluated, before the teaching phase, to select the initial teacher, after the teaching phase, to check if the teacher has changed, and to update the global best teacher, as well after the learning phase, to check if the learning phase generated a new better solution than the current best, then after completing the teaching and learning phases for the set number of iterations, the global best teacher is used to move the UAVs to the next step. The parameters used to assess the performance indices were, classroom size = 20, maximum iterations per run = 60, with all the problem specific parameters being kept unchanged.

IV. Results & Performance Indices

In order to compare the performance of all the algorithms, we ran the algorithms using the same parameters 15 times, to calculate the standard deviation, mean fitness and the best fitness during those 15 runs, the table shows the results:

	Standard Deviation	Mean Fitness	Best Fitness
SA	26.9377	111.5836	85.4256
GA	6.7540	76.62864	72.5574
PSO	4.38347	89.10830	81.8662
TLBO	12.26142	137.69886	122.1275

Based on the table above, the PSO showed the lowest standard deviation, showing that every solution of the PSO is close to the other, due to the reduced effect of randomness on the solutions, on the contrary to the SA which heavily relies of random values to optimize, the GA showed the best mean fitness, and the best fitness during testing, which is a result of the dependency of elitism in influencing the solutions of the rest of the population. The TLBO, however, showed a standard deviation being a middle ground between the performance of the PSO, GA and the SA, in our implementation, the TLBO relied on randomness in generation of new solutions after every run, which might have caused this increase in standard deviation, and the higher values of mean and best overall fitness recorded.

The figures above show the overall convergence curves for the SA, GA and PSO, Figure 1 shows the fastest global convergence, where it took around 5500 iterations for all the UAVs to reach their targets, where the GA took around 13600 iterations and the PSO took 36000 iterations, however, the PSO, in less than 2000 iterations reached the lowest value of all the other algorithms, showing its power in exploitation. From the above figures, it is clear that the population based algorithms took longer to converge compared to the SA, since the solutions are computed for multiple cells at once, instead of checking one solution as done in the SA. The TLBO, shown in figure 4 reached its global minimum in around 33000 iterations, which is similar to the PSO's performance in this run, however the TLBO reached a worse fitness value when compared to all the other algorithms, due to the increased

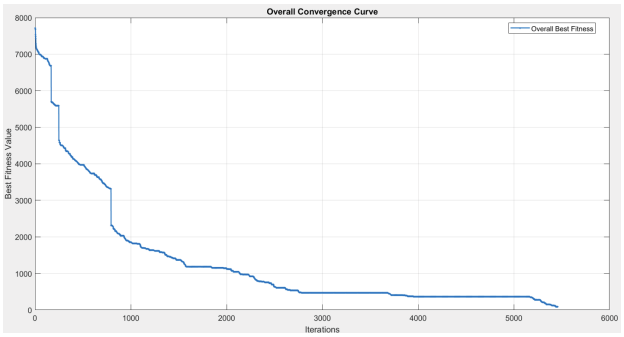

Fig. 1. Overall SA convergence curve

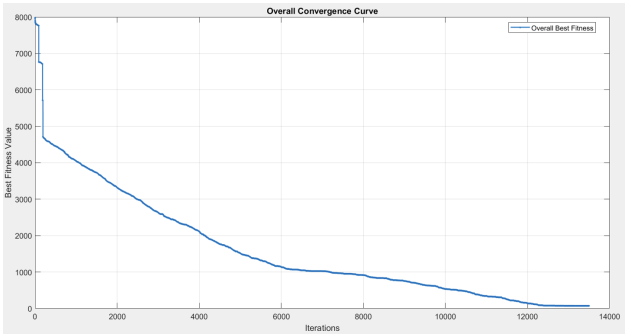

Fig. 2. Overall GA convergence curve

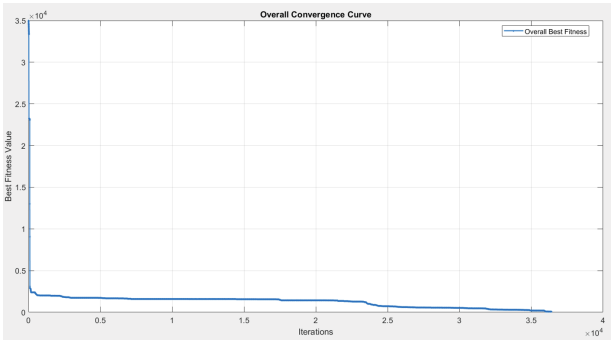

Fig. 3. Overall PSO convergence curve

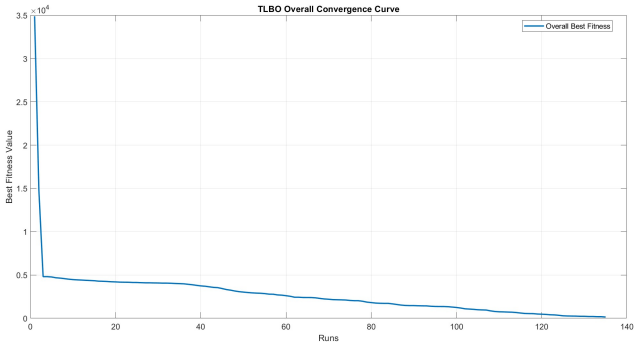

Fig. 4. Overall TLBO convergence plot

randomization of new population, and inaccurate position updates where the implementation does not create a balance between exploration and exploitation.

V. Conclusion

Overall, the algorithms showed varying success rates in minimizing the fitness function based on our problem formulation, with the PSO being the most consistent, in terms of performance indices, while the GA was the best performing in terms of quality of produced path. Population based techniques proved that their outputs are much more reliable, compared to other meat-heuristic techniques, such as the SA. Fine tuning the algorithm specific parameters is crucial to an algorithms success, as shown in the test cases in the previous chapter, also, a solid problem formulation can make or break an algorithm, since the problem formulation is the base of an optimization problem.

For future work, We would like to explore the effect of adding multiple static obstacles, as well the possibility of dynamic obstacles, alongside with tuning the problem formulation to reflect more realistic optimization scenarios, furthermore, testing more algorithms and using reinforcement learning algorithms to solve our optimization problem.

References

[1] F. Ropero, P. Muñoz, and M. D. R-Moreno, "Terra: A path planning algorithm for cooperative ugv–uav exploration," *Engineering Applications of Artificial Intelligence*, vol. 78, pp. 260–272, 2019.

[2] C. Xu, M. Xu, and C. Yin, "Optimized multi-uav cooperative path planning under the complex confrontation environment," *Computer Communications*, vol. 162, pp. 196–203, 2020.

[3] Z. Qiming, W. Husheng, and F. Zhaowang, "A review of intelligent optimization algorithm applied to unmanned aerial vehicle swarm search task," in *2021 11th International Conference on Information Science and Technology (ICIST)*, pp. 383–393, IEEE, 2021.

[4] S. Ramasamy, M. S. Mondal, J.-P. Reddinger, J. Dotterweich, J. Humann, M. Childers, and P. Bhounsule, "Heterogenous vehicle routing: comparing parameter tuning using genetic algorithm and bayesian optimization," pp. 104–113, 06 2022.

[5] Y. Wu, S. Wu, and X. Hu, "Cooperative path planning of uavs & ugvs for a persistent surveillance task in urban environments," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4906–4919, 2020.

[6] H. K. Thakkar, H. Shukla, and P. K. Sahoo, "Chapter 2 - metaheuristics in classification, clustering, and frequent pattern mining," in *Cognitive Big Data Intelligence with a Metaheuristic Approach* (S. Mishra, H. K. Tripathy, P. K. Mallick, A. K. Sangaiah, and G.-S. Chae, eds.), Cognitive Data Science in Sustainable Computing, pp. 21–70, Academic Press, 2022.

[7] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems," *Computer-aided design*, vol. 43, no. 3, pp. 303–315, 2011.