Navigation Variable-based Multi-objective Particle Swarm Optimization for UAV Path Planning with Kinematic Constraints

Thi Thuy Ngan Duong [1,2], Duy-Nam Bui [2], Manh Duong Phung [3*]

[1]Department of Electrical Engineering, Ulsan National Institute of Science and Technology, UNIST-gil 50, Ulsan, 44919, Korea.
[2]Faculty of Electronics and Telecommunications, Vietnam National University, 144 Xuan Thuy, Hanoi, 100000, Vietnam.
[3]Undergraduate Faculty, Fulbright University Vietnam, 105 Ton Dat Tien, Ho Chi Minh City, 700000, Vietnam.

*Corresponding author(s). E-mail(s): duong.phung@fulbright.edu.vn;
Contributing authors: nganduong@unist.ac.kr; duynam@ieee.org;

Abstract

Path planning is essential for unmanned aerial vehicles (UAVs) as it determines the path that the UAV needs to follow to complete a task. This work addresses this problem by introducing a new algorithm called navigation variable-based multi-objective particle swarm optimization (NMOPSO). It first models path planning as an optimization problem via the definition of a set of objective functions that include optimality and safety requirements for UAV operation. The NMOPSO is then used to minimize those functions through Pareto optimal solutions. The algorithm features a new path representation based on navigation variables to include kinematic constraints and exploit the maneuverable characteristics of the UAV. It also includes an adaptive mutation mechanism to enhance the diversity of the swarm for better solutions. Comparisons with various algorithms have been carried out to benchmark the proposed approach. The results indicate that the NMOPSO performs better than not only other particle swarm optimization variants but also other state-of-the-art multi-objective and metaheuristic optimization algorithms. Experiments have also been conducted with real UAVs to confirm the validity of the approach for practical flights. The source code of the algorithm is available at https://github.com/ngandng/NMOPSO.

Keywords: Unmanned aerial vehicle (UAV), path planning, multi-objective optimization, particle swarm optimization

1 Introduction

Path planning is an essential problem for unmanned aerial vehicle (UAV) applications because it determines the flight path from the starting position to the destination that a UAV needs to follow to complete its mission [1, 2]. The path should be optimal in certain criteria such as shortest length or minimal energy consumption. It also needs to meet constraints imposed by the kinematic model and safe operation of the UAV [3]. Some objectives and constraints, however, may contradict leading to the non-existence

of a single global optimal path. Path planning techniques therefore need to balance those requirements to obtain best possible solutions.

In the literature, A*, sampling-based algorithms, and artificial potential field (APF) methods are among the most popular for path planning [4]. A* uses heuristics to guide its search in finding the shortest path between the start and goal positions [5, 6]. By maintaining a cost function originating from the start node, A* extends a path one edge at a time until the goal is reached. However, A* is limited in scalability as its discretization of the search space causes the number of cells to increase rapidly with the size of the space.

The APF method does not discretize the search space but defines it as a potential field formed by surrounding objects [7–9]. The UAV is then modeled as a particle traveling in the field, attracted by the target and repelled by the obstacles. As the result, the UAV will move toward the goal along a smooth path while avoiding obstacles. The path generated, however, are not optimized and degraded when the complexity of the environment increases.

The sampling-based method, on the other hand, uses randomization to expand a tree representing the path until it reaches the goal position [7, 10]. This approach guarantees to find a path to the goal if such a path exists. For example, the rapidly-exploring random trees (RRT) algorithm samples the search space randomly in a way biased toward the large unexplored areas [11]. As time passes, the algorithm explores more areas and eventually finds the route to the goal. However, the RRT algorithm does not optimize the length of the path during the search process. Its variants such as RRT* [12] can shorten the path but more computation is required.

Recently, nature-inspired optimization techniques have been used in UAV path planning due to their ability to produce optimal solutions [13–15]. These techniques use cost functions to formulate path planning as an optimization problem and then solve it with nature-inspired algorithms like the firefly algorithm (FA) [16], genetic algorithm (GA) [17, 18], artificial bee colony algorithm (ABC) [19], particle swarm optimization (PSO) [20–23], and ant colony optimization (ACO) [24]. These algorithms consider a path as a candidate solution and then use swarm intelligence to improve it. The type of swarm intelligence varies depending on the phenomena the algorithm relies on. The GA uses mutation and crossover operators. The DE also uses mutation but combines it with differential evolution. On the other hand, the ACO uses pheromones and randomization to direct the search process. Other algorithms, such as the ABC, FA, and PSO utilize the social cognition behavior of a swarm to explore the solution space.

Among nature-inspired algorithms, the PSO is often referred to as an efficient method capable of achieving optimal solutions with a high convergence rate. It is also less sensitive to initial conditions and can be adapted to various environment structures [25, 26]. The PSO obtains those features by balancing the personal experience of each individual and the experience of the whole swarm to find potential regions in the solution space. Several variants of PSO have been introduced for path planning like the discrete PSO (DPSO) [20], hybrid PSO [21], quantum-behaved PSO (QPSO) [27], and spherical vector-based PSO (SPSO) [22]. Most of the algorithms, however, are single objectives in which they combine objectives in a single cost function via a weighted sum. While this approach is simple to implement, combining multiple objectives leads to a cost function whose optimality does not represent the optimality of individual objectives. In practice, most objectives do not simultaneously reach their optimal values. Some of them are even contradicting that require a different approach called multi-objective optimization.

In the multi-objective optimization direction, several techniques have been introduced for path planning, such as the multi-objective firefly algorithm (MOFA) [28], multi-objective PSO (MOPSO) [3, 29], and nondominated sorting genetic algorithm II (NSGA-II) [30]. These algorithms use a similar mechanism called the non-domination principle to direct solutions toward Pareto optimal regions. In particular, NSGA-II uses a crowded-comparison operator with two attributes: non-domination rank and crowding distance to spread solutions across potential regions [31]. MOFA uses the search method from the Firefly algorithm to find non-dominated solutions and then carries out a migration procedure to maintain the diversity of the Pareto front [32]. In MOPSO, non-dominated solutions are used to guide the exploration of the solution space. Two factors including a position update mechanism and a mutation operator are used to promote diversity [33]. Multi-objective optimization has the advantage of finding best possible solutions that form a set of non-dominated solutions called the Pareto front. Current algorithms, however, have not incorporated constraints imposed by the UAV kinematics into the search process. Since those constraints are essential to generating flyable paths, it is important to include them in the algorithm.

In this study, a new algorithm named navigation variable-based multi-objective PSO (NMOPSO) is proposed to generate flyable and Pareto-optimal paths for UAVs. First, a set of objective functions and navigation variables are defined to incorporate requirements and constraints for optimal UAV operation. The multi-objective PSO is then used to find a set of non-dominated solutions that best fit the objective functions. A mutation mechanism is introduced to avoid premature convergence and improve the search performance. Four scenarios were created based on real digital elevation model (DEM) maps of different complexity to evaluate the performance of the proposed approach. Our contributions in this work include (i) the proposal of the NMOPSO that includes kinematic constraints and multiple objectives to generate Pareto optimal paths for the autonomous operation of UAVs; (ii) the introduction of an adaptive mutation operator to improve the swarm performance; (iii) the derivation of a formula inspired by the Denavit-Hartenburg representation to convert navigation variables to Cartesian coordinates for efficient path searching and evaluation; (iv) the implementation of experiments with real UAVs to confirm the validity of the NMOPSO for practical operation.

The rest of this paper is structured as follows. Section 2 introduces the kinematic model and definition of objective functions. Section 3 presents the proposed algorithm, NMOPSO. Section 4 provides comparison and experiment results. A conclusion is drawn in section 5 to end the paper.

2 Problem Formulation

This section presents the kinematic model of the UAV and objective functions defined for the path planning problem.

2.1 Kinematic model and constraints

Consider the UAV as a point moving in the environment. According to [34], its kinematic equations are described as follows:

$$\begin{cases} \dot{x} = V\cos\alpha\cos\beta \\ \dot{y} = V\cos\alpha\sin\beta \\ \dot{z} = V\sin\alpha \end{cases}, \tag{1}$$

where $[x, y, z]^T$ represents the position of the UAV; V is the linear velocity; α and β are respectively the climbing and turning angle. Due to physical limits, the velocity and angles of the UAV are subject to the following constraints:

$$\begin{cases} V_{\min} \le V \le V_{\max} \\ |\Delta\alpha| = |\theta| \le \theta_{\max} \\ |\Delta\beta| = |\psi| \le \psi_{\max} \end{cases}, \tag{2}$$

where $V_{\min}$ and $V_{\max}$ are respectively the minimum and maximum linear velocities and $\psi_{\max}$ and $\theta_{\max}$ are respectively the maximum variations of turning and climbing angles. It is important to incorporate those constraints into the path planning algorithm so that feasible paths are generated for the UAV to follow.

2.2 Objective functions for path planning

Requirements for the path are formulated via the definition of objective functions. The functions are inspired by our previous work [22] but have been modified and normalized to the range $[0, 1]$ to suit the multi-objective approach.

2.2.1 Path length

In autonomous flights, a UAV is pre-loaded with a list of waypoints it needs to travel through. A flight path p_i thus can be described by a set of n waypoints, $p_i = \{P_{i1}, ..., P_{in}\}$, each includes three components, $P_{ij} = (x_{ij}, y_{ij}, z_{ij})$, as shown in Figure 1. Denote $\left\|\overrightarrow{P_{ij}P_{i,j+1}}\right\|$ as the Euclidean distance between two

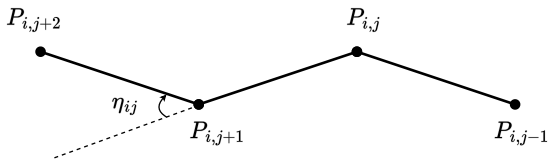


Figure 1: Illustration of a flight path and its variables

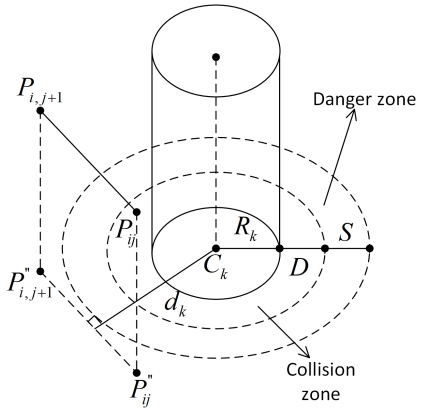


Figure 2: Obstacle avoidance

waypoints. The cost associated with the path length is then formulated as follows:

$$F_1 = \begin{cases} 1 - \dfrac{\left\|\overrightarrow{P_{i1}P_{in}}\right\|}{\sum_{j=1}^{n-1}\left\|\overrightarrow{P_{ij}P_{i,j+1}}\right\|}, & \text{if } \left\|\overrightarrow{P_{ij}P_{i,j+1}}\right\| \geq R_{\min} \\ \infty, & \text{otherwise} \end{cases} \tag{3}$$

where $R_{\min}$ is the minimum distance between two waypoints. Finding the shortest path is equivalent to finding p_i that minimizes F_1 in (3).

2.2.2 Collision avoidance

A flyable path should guide the UAV to avoid obstacles. In this work, an obstacle k is modeled as a cylinder with center C_k and radius R_k, as shown in Figure 2. Let d_k be the distance from the center of obstacle k to path segment $P_{ij}P_{ij+1}$, D be the UAV size, and S be the safe distance from the UAV to the obstacle. The objective function for safe operation of the UAV is expressed as follows:

$$F_2 = \frac{1}{K(n-1)} \sum_{j=1}^{n-1} \sum_{k=1}^{K} \mathcal{T}_k \left(\overrightarrow{P_{ij}P_{ij+1}}\right), \tag{4}$$

where K is the number of obstacles in the working area and $\mathcal{T}_k$ is calculated as:

$$\mathcal{T}_k \left(\overrightarrow{P_{ij}P_{ij+1}}\right) = \begin{cases} 0, & \text{if } d_k \geq D + R_k + S \\ 1 - \dfrac{d_k - D - R_k}{S}, & \text{if } D + R_k < d_k < D + R_k + S \\ \infty, & \text{otherwise} \end{cases} \tag{5}$$

Equation (5) implies that if a path segment is outside the danger zone (see Figure 2), it incurs no additional cost to the objective function. However, if a path segment falls within the danger zone, its cost

is inversely proportional to its distance to the obstacle. In case the path segment is within the collision zone, an infinite cost will be added to indicate a collision.

2.2.3 Flight altitude

During operation, it is preferable that the UAV flies at a stable altitude to minimize its energy consumption. Let $h_{\max}$ and $h_{\min}$ respectively be the maximum and minimum relative flight altitudes and h_{ij} be the present altitude of the UAV. Let $h_{\text{mean}} = \dfrac{h_{\max} + h_{\min}}{2}$ be the preferable relative altitude for the flight. The objective function for the flight altitude is defined as:

$$F_3 = \frac{1}{n} \sum_{j=1}^{n} \mathcal{H}_{ij}, \tag{6}$$

where

$$\mathcal{H}_{ij} = \begin{cases} \dfrac{2\,|h_{ij} - h_{\text{mean}}|}{h_{\max} - h_{\min}}, & \text{if} \quad h_{\min} \leq h_{ij} \leq h_{\max} \\ \infty, & \text{otherwise} \end{cases} \tag{7}$$

2.2.4 Smoothness

Apart from maintaining the altitude, a flight path should also minimize variations in the turning angle of the UAV as they are directly proportional to energy consumption. As illustrated in Figure 1, the turning angle η_{ij} is the angle between two consecutive path segments, $\overrightarrow{P_{ij}P_{i,j+1}}$ and $\overrightarrow{P_{i,j+1}P_{i,j+2}}$, and is computed as:

$$\eta_{ij} = \arctan\left(\frac{\left\| \overrightarrow{P_{ij}P_{i,j+1}} \times \overrightarrow{P_{i,j+1}P_{i,j+2}} \right\|}{\overrightarrow{P_{ij}P_{i,j+1}} \cdot \overrightarrow{P_{i,j+1}P_{i,j+2}}} \right). \tag{8}$$

Since this angle represents changes in the direction of the UAV along the flight path, the smooth cost F_4 is defined as:

$$F_4 = \frac{1}{n-2} \sum_{j=1}^{n-2} \frac{|\eta_{ij}|}{\pi}, \tag{9}$$

where π is the maximum turning angle used for the normalization.

3 Multi-objective optimization approach

Given objective functions F_1 to F_4, an optimal path would simultaneously minimize all of them. Such a path, however, does not exist since those functions do not minimize at the same point. Some functions are even contradicting such that the decrease of one function leads to the increase of another. To overcome it, most studies combine those functions into a single objective function using a weighted sum [22, 27]. While that approach is simple to implement, it is difficult to choose the right weight for each function and maintain the optimality of the obtained solution. In this work, we propose to use multi-objective optimization.

3.1 Pareto-optimal solution

Let X be a path generated for the UAV. The path planning algorithm needs to find path $\hat{X}$ that simultaneously minimizes all cost functions F_i defined in Section 2.2,

$$\hat{X} = \arg\min \left[F_1, F_2, F_3, F_4 \right]. \tag{10}$$

According to the multi-objective optimization theory, there may not exist the optimal solution $\hat{X}$, but only the solution X^* that is the best fit for all F_i. That solution is called the Pareto-optimal solution defined as follows:

Definition 1. A solution $X \in \mathcal{X} \subset \mathbb{R}^4$ is **non-dominated** with respect to $\mathcal{X}$ if there is no other solution $X' \in \mathcal{X}$ such that $F_i(X') \leq F_i(X)$ for every $i = 1, .., 4$ and $F_j(X') < F_j(X)$ for at least one j.
Definition 2. Let $\mathcal{F}$ be the feasible region. A solution $X^* \in \mathcal{F} \subset \mathbb{R}^4$ is considered **Pareto-optimal** if it is non-dominated with respect to $\mathcal{F}$.
Definition 3. The Pareto optimal set $\mathcal{P}^*$ is a set of non-dominated solutions defined by $\mathcal{P}^* = \{X^* \in \mathcal{F} | X^* \text{ is Pareto-optimal}\}$.

According to these definitions, a Pareto-optimal solution is one where no other solution can improve one objective without deteriorating at least one other objective. Unlike a single optimal solution, multiple Pareto-optimal solutions can exist, forming the Pareto optimal set or Pareto front. Several methods, such as weighting, lexicographic, and goal programming, can be used to find Pareto-optimal solutions [35]. In this work, the particle swarm optimization method is chosen due to its efficiency and robustness [36, 37].

3.2 Muli-objective particle swarm optimization

Particle swarm optimization (PSO) is a popular technique that was originally developed to solve single objective optimization problems using swarm intelligence. In PSO, a swarm of particles is first initialized so that the position of each particle represents a candidate solution. A cost function is then used to evaluate the fitness of those particles. Each particle of the swarm then moves in accordance to its best position and the best position of the swarm to improve its fitness until an optimal solution is obtained or the maximum number of iterations is reached.

Let x_i^t and v_i^t be the position and velocity of particle i at iteration t, respectively. Denote $x_{pbest,i}^t$ as the best position of particle i and x_{gbest}^t as the best position of the swarm at iteration t. The movement of particle i is described by the following equations:

$$\begin{aligned} v_i^{t+1} &= w v_i^t + c_1 r_1 (x_{pbest,i}^t - x_i^t) + c_2 r_2 (x_{gbest}^t - x_i^t), \\ x_i^{t+1} &= x_i^t + v_i^{t+1} \end{aligned} \tag{11}$$

where w is the inertial weight, c_1 and c_2 are respectively the cognitive and social coefficients, and r_1 and r_2 are random numbers drawn from the uniform distribution in the range of $[0, 1]$.

When using PSO for multi-objective optimization, it is important to control the particles' distribution so that they can find non-dominated solutions. The particles should evolve under the guidance of non-dominated particles called leaders to spread across multiple potential regions. A popular approach is to define a repository to store non-dominated solutions and then use them as leaders [38]. Each particle then selects a leader from the repository based on a crowd measure as follows.

Let $\mathcal{P}$ be the set of non-dominated solutions in the repository. A hypergrid is first established to allocate each particle in $\mathcal{P}$ to a hypercube as shown in Figure 3. The coordinate of each particle is determined by their objective value [39]. Specifically, the lower bound, G_i^L, and upper bound, G_i^U, of the hypergrid in the dimension representing F_i is determined as:

$$G_i^L = \min_{x \in \mathcal{P}} F_i(x) - \epsilon_i \tag{12}$$

$$G_i^U = \max_{x \in \mathcal{P}} F_i(x) + \epsilon_i \tag{13}$$

where ϵ_i is the padded grid length computed based on the number of grid divisions, M, as:

$$\epsilon_i = \frac{1}{2(M-1)} \left(\max_{x \in \mathcal{P}} F_i(x) - \min_{x \in \mathcal{P}} F_i(x) \right). \tag{14}$$

The coordinate of the hypercube containing particle x in dimension F_i is then computed by:

$$c_i = \left\lfloor M \frac{F_i(x) - G_i^L}{G_i^U - G_i^L} \right\rfloor, \tag{15}$$

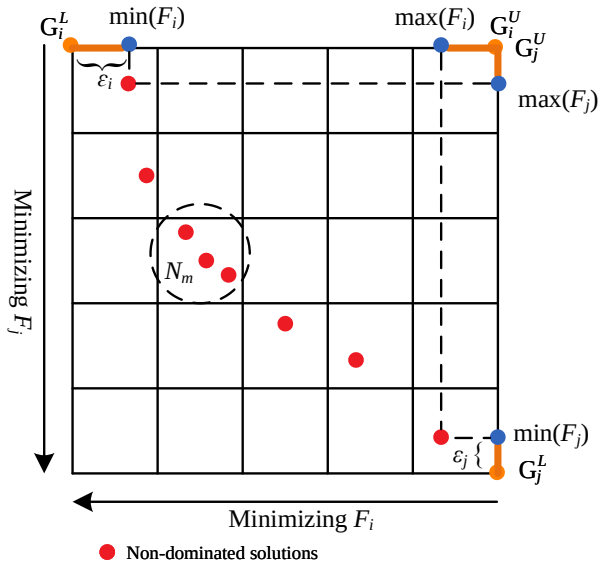


Figure 3: Illustration of a hypergrid and non-dominated solutions

where $\lfloor \cdot \rceil$ is the notation for rounding to the nearest integer. Let N_m be the number of particles located in hypercube m. The crowd measure of that hypercube is computed as:

$$\gamma_m = e^{-\kappa N_m}, \tag{16}$$

where κ is a scaling coefficient. A leader for each particle of the swarm is then selected from the hypergrid in a random fashion with the selection probability proportional to the crowd measure:

$$p_m = \frac{\gamma_m}{\sum_{l=1}^{\mathcal{L}} \gamma_l}, \tag{17}$$

where $\mathcal{L}$ is the number of hypercubes. Denote $x_{lbest,i}^t$ as the position of the selected leader for particle i. Equations for multi-objective PSO are written as:

$$\begin{aligned} v_i^{t+1} &= w v_i^t + c_1 r_1 (x_{pbest,i}^t - x_i^t) + c_2 r_2 (x_{lbest,i}^t - x_i^t) \\ x_i^{t+1} &= x_i^t + v_i^{t+1}. \end{aligned} \tag{18}$$

It can be seen that the main difference between the MOPSO and the original PSO is the replacement of x_{gbest}^t by $x_{lbest,i}^t$ to diverge particle movements and look for multiple Pareto-optimal solutions. To further enhance the MOPSO, two improvements are introduced: one relates to the navigation variables, and the other involves a mutation mechanism.

3.3 Navigation variables for particle position representation

When using PSO for path planning, the position of each particle represents a candidate solution, which is a flight path. For path p_i with n waypoints $P_{ij} = [x_{ij}, y_{ij}, z_{ij}]^T$, position X_i representing that path is expressed as:

$$X_i = (x_{i1}, y_{i1}, z_{i1}, x_{i2}, y_{i2}, z_{i2}, ..., x_{in}, y_{in}, z_{in}). \tag{19}$$

The use of Cartesian coordinates as in (19), however, does not incorporate maneuverable properties of the UAV into the path. It is therefore not efficient in finding non-dominated solutions or suitable to include kinematic constraints for feasible flight paths.

Inspired by the operation of robot manipulators, we address this issue by considering a flight path as an articulated chain consisting of n path segments. Each segment is described by a set of parameters

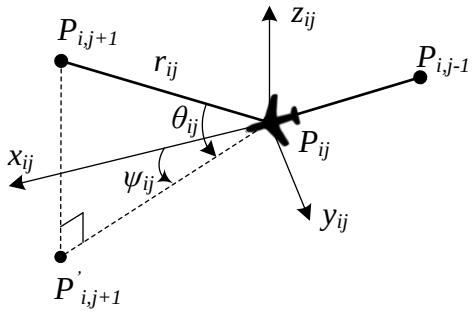


Figure 4: Illustration of variables for particle representation

similar to the Denavit–Hartenberg parameters [40] including the climbing angle θ, turning angle ψ, and length r. The end position of each segment then can be determined by the multiplication of transformation matrices representing the pose of previous path segments.

To implement this idea, at each waypoint P_{ij}, a coordinate frame attached to the UAV located at that point is defined, as shown in Figure 4. The x-axis points out of the UAV's front and is coincident with the line connecting $P_{i,j-1}$ and P_{ij}. The y-axis is directed out of the left side of the UAV and the z-axis is perpendicular to the x and y axes and is directed upward. Three new variables, (r, θ, ψ), are then defined as follows:

(i) r_{ij} is the distance between two consecutive nodes of path segment $\overrightarrow{P_{ij}P_{i,j+1}}$, $r_{ij} = \left\|\overrightarrow{P_{ij}P_{i,j+1}}\right\|$;

(ii) ψ_{ij} is the angle between two sequential path segments, $\overrightarrow{P_{i,j-1}P_{ij}}$ and $\overrightarrow{P_{ij}P'_{i,j+1}}$, where $P'_{i,j+1}$ is the projection of $P_{i,j+1}$ on plane $Ox_{ij}y_{ij}$;

(iii) θ_{ij} is the angle between path segments $\overrightarrow{P_{ij}P_{i,j+1}}$ and $\overrightarrow{P_{ij}P'_{i,j+1}}$.

We call (r, θ, ψ) navigation variables. Position Γ_i representing path p_i in the navigation space then can be expressed as:

$$\begin{cases} \Gamma_i = (r_{i1}, \theta_{i1}, \psi_{i1}, r_{i2}, \theta_{i2}, \psi_{i2}, ..., r_{in}, \theta_{in}, \psi_{in}) \\ \quad |\theta_{ij}| \leq \theta_{max} \quad \forall j \in \{1, .., n\} \\ \quad |\psi_{ij}| \leq \psi_{max} \quad \forall j \in \{1, .., n\} \end{cases} \tag{20}$$

Different from Cartesian coordinates in (19), the use of navigation variables allows to include maneuverable properties of the UAV to the particles' position so that they can better explore the solution space. More importantly, kinematic constraints on the climbing and turning angles described in (2) can be directly included in the algorithm by limiting the range of those angles as in (20). As a result, the solution space is significantly narrowed down to increase the possibility of finding feasible and non-dominated solutions.

During the optimization process, it is necessary to convert path Γ_i in the navigation space (20) to its equivalent X_i in the Cartesian space (19) to evaluate its fitness. This can be done by using transformation matrices. Specifically, the UAV's movement from waypoint P_{ij} to waypoint $P_{i,j+1}$ can be divided into three sub-movements:

(i) Rotate an angle ψ_{ij} about the z_{ij}-axis;

(ii) Rotate an angle θ_{ij} about the y_{ij}-axis;

(iii) Translate r_{ij} units along the x_{ij}-axis.

Hence, the transformation matrix between the frames located at two consecutive waypoints is computed as follows:

$$T^j_{i,j+1} = R_z(\psi_{ij}) R_y(\theta_{ij}) M_x(r_{ij}), \tag{21}$$

where R_y and R_z are respectively the transformation matrices representing the rotation about the y and z-axis, and M_x is the translation matrix about the x-axis.

Let T^0_{iS} be the transformation matrix representing the start position and orientation of the UAV with respect to the inertial frame. The transformation matrix describing the position and orientation of frame

$(Oxyz)_{ij}$ with respect to the inertial frame is given by:

$$T_{ij}^{0} = T_{iS}^{0} T_{ij}^{S}, \tag{22}$$

where T_{ij}^{S} is computed as:

$$T_{ij}^{S} = T_{i1}^{S} T_{i2}^{1} \cdots T_{ij}^{j-1}. \tag{23}$$

Finally, the Cartesian coordinates of waypoint P_{ij} can be obtained from the navigation variables as:

$$\tilde{P}_{ij} = T_{ij}^{0} \tilde{I}, \tag{24}$$

where $\tilde{P}_{ij} = [P_{ij}\ 1]^T$ and $\tilde{I} = [0\ 0\ 0\ 1]^T$.

3.4 Region-based mutation

When searching non-dominated solutions, certain particles of the swarm can be trapped in local optima, which affects the swarm's performance. For MOPSO, the chance of particles being trapped is higher due to their spread across multiple regions of the search space [38]. We overcome this problem by introducing an adaptive mutation mechanism so that the level of mutation is proportional to the crowding distance of particles. For a random particle x_i at iteration t, the mutation is conducted as follows:

$$x_{ij}^{t} = x_{ij}^{t} + \mathcal{N}_{ij} G^{t} x_{pbest,i}^{t}, \tag{25}$$

where j is a randomly selected component of x_i, $\mathcal{N}_{ij}$ is a random variable having the Gaussian distribution with zero mean and unit variance, and G is a mutation gain.

To better fit the multi-objective optimization problem, G is adjusted based on the distribution of non-dominated solutions in the Pareto optimal set. Since the spread of those solutions is proportional to the number of occupied hypercubes in the hypergrid, it can be used as a parameter to adjust G. When the number of occupied hypercubes is small, they indicate the dense concentration of particles over those regions. A large gain value is therefore needed to drive the particles to new regions. In contrast, when the particles are widely distributed, a small gain value will help to better explore those regions. Denote N_r^t as the number of occupied hypercubes at iteration t. The mutation gain is determined as follows:

$$G^{t} = \tanh\left(\frac{\Delta}{N_r^t}\right), \tag{26}$$

where $\tanh(\cdot)$ represents the hyperbolic tangent and Δ is a pre-defined constant. This constant is set equal to the number of hypercubes occupied when the swarm is initialized.

3.5 Implementation of the NMOPSO for UAV path planning

The implementation of the NMOPSO is presented in Algorithm 1 together with the equations used. Compared to the original PSO, additional steps related to the hypergrid, leader selection, navigation variables, and mutation are added to find Pareto-optimal solutions. The algorithm stops when the maximum number of iterations is reached.

4 Results

To evaluate the performance of the proposed algorithm, a number of comparisons and experiments have been conducted with details below.

4.1 Scenario setup

Evaluations are conducted using real digital elevation model (DEM) maps of two areas on Christmas Island, Australia, each with distinct terrain structures [41], as shown in Figures 7 and 8. Each map is then augmented with obstacles to form four scenarios of different levels of complexity as shown in Figure 8.

Algorithm 1: Pseudocode for the NMOPSO algorithm

```
/* Initialization                                                    */
Get the search map and initial information ;
Initialize swarm parameters w, c1, c2 ;
Initialize grid dimension M, scaling coefficient κ, mutation coefficient Δ ;
foreach particle i in swarm do
    Generate random path Γ_i^0 ;
    Get particle's position X_i^0 from Γ_i^0 ;
    Evaluate fitness F_j(X_i^0) of particle i ;
    Set pbest_i for particle i based on the fitness ;
end
Initialize repository P ;
Create a hypergrid and find grid location c_i for solutions in P ;   /* Eq. 15 */
/* Search                                                            */
while not max_iteration do
    foreach particle i in swarm do
        Select a leader from P ;                                    /* Eq. 17 */
        Calculate velocity v_i^t and update new position Γ_i^t ;    /* Eq. 18 */
        Map Γ_i^t to X_i^t ;                                        /* Eqs. 21-24 */
        Update fitness F_j(X_i^t) ;                                 /* Eqs. 3-9 */
        Update pbest_i ;
        Apply mutation ;                                            /* Eq. 25 */
    end
    Update P;
    Update the hypergrid ;                                          /* Eq. 15 */
end
Set Pareto optimal set P* = P ;
Generate paths from non-dominated solutions in P* ;
```

Comparisons are then carried out on these scenarios with pre-defined start and goal positions. Parameters of NMOPSO in all comparisons are chosen as follows: the number of path nodes $n = 10$; $w = 1$ with the damping rate of 0.98; $c_1 = 1.5$ and $c_2 = 1.5$; the number of grid divisions $M = 7$; the scaling coefficient $\kappa = 2$; the mutation coefficient $\Delta = 5$; and the turning and climbing angle limits $\theta_{max} = \psi_{max} = \pi/4$.

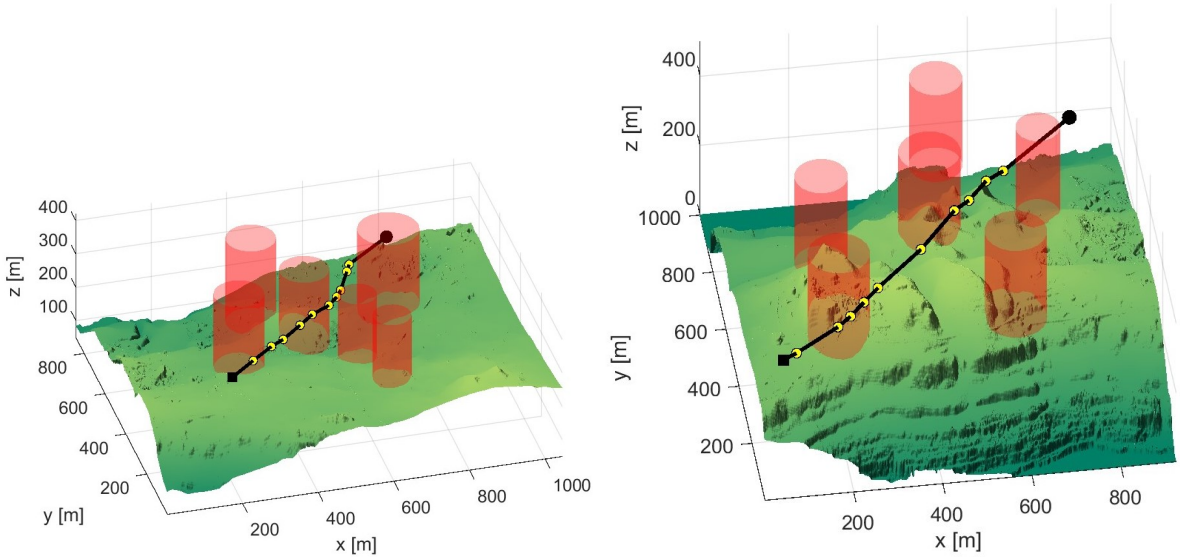
4.2 Path planning results

Figures 5 and 6 show the paths generated for scenarios 1 and 4. It can be seen that all paths are collision-free and successfully reach the goal positions. The side view of those paths shows that they adapt to the terrain structure but sharp changes in altitude are suppressed due to kinematic and smoothness constraints. The paths thus are feasible for the UAV to follow. Note that the paths shown in Figures 5 and 6 are just some among many non-dominated solutions in the Pareto Front obtained by the algorithm. Each solution dominates in certain objectives. Therefore, some non-dominated solutions may be preferred over others depending on the application.

4.3 Comparison with other PSO variants

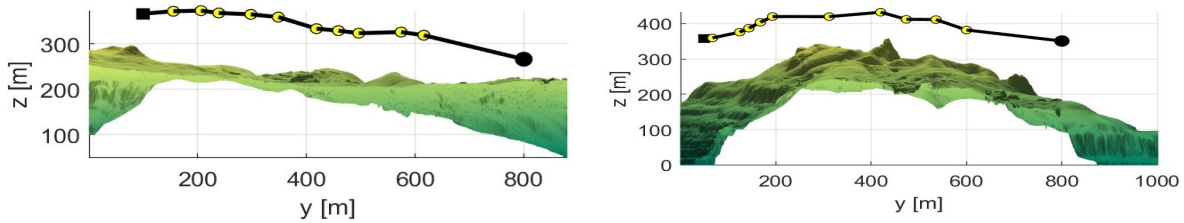
In this evaluation, the NMOPSO is compared with other single-objective PSO variants including the original PSO, quantum-behaved PSO (QPSO) [42], and angle-encoded PSO (θ-PSO) [43]. Their parameters are chosen to be the same as the NMOPSO. However, the cost function is a weighted sum of the objectives as follows:

$$F = \sum_{i=1}^{4} w_i F_i, \tag{27}$$



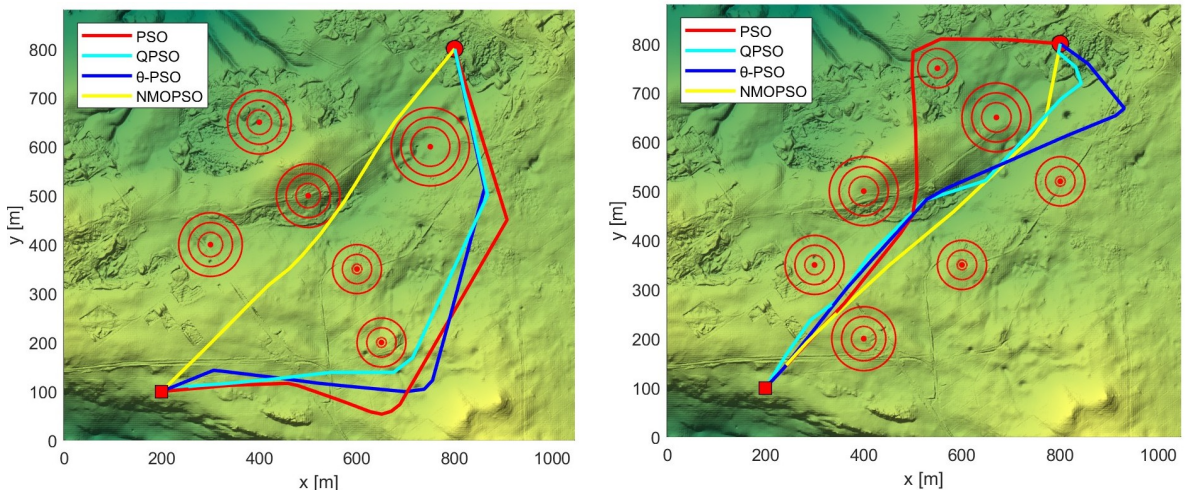
(a) Scenario 1 (b) Scenario 4

Figure 5: 3D view of the paths generated by the NMOPSO in scenarios 1 and 4



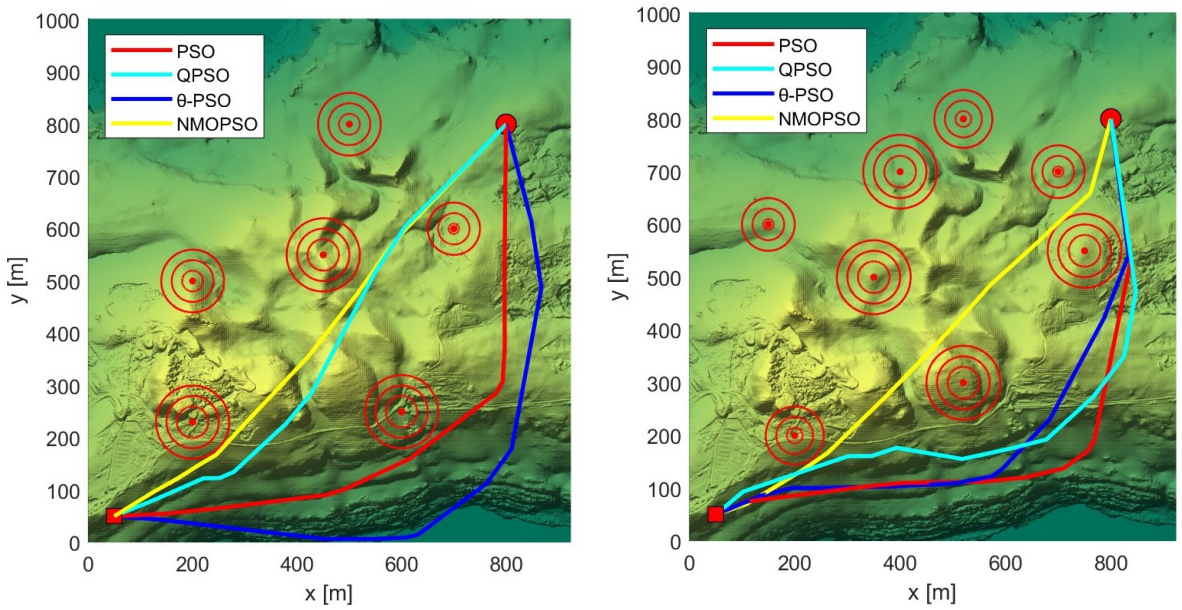
(a) Scenario 1 (b) Scenario 4

Figure 6: Side view of the paths generated by the NMOPSO in scenarios 1 and 4



(a) Scenario 1 (b) Scenario 2

Figure 7: Top view of the paths generated by the PSO-based algorithms in simple elevation maps



(a) Scenario 3

(b) Scenario 4

Figure 8: Top view of the paths generated by the PSO-based algorithms in complex elevation maps

where w_i is a weighting coefficient. In our implementation, $w_i = 1$ since the objective functions are already normalized to $[0, 1]$.

Figures 7 and 8 show the paths generated by the algorithms. It can be seen that all algorithms are able to generate collision-free paths to the goal. The NMOPSO, however, produces the shortest and smoothest paths in all scenarios. This result can be further confirmed in Table 1, which shows the values of the objective functions corresponding to the generated paths. It can be seen that the NMOPSO generates multiple Pareto optimal solutions. Some of them dominate certain objectives, while others have reasonable values across all objectives. They together outperform other PSO variants in most objectives.

4.4 Comparison with other metaheuristic algorithms

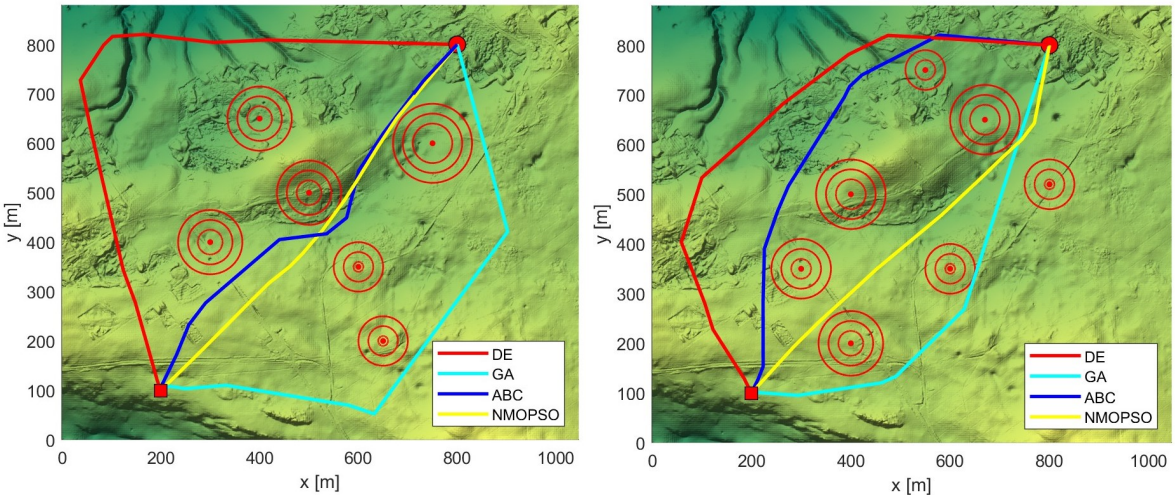
Comparisons with other popular metaheuristic algorithms, including the differential evolution (DE), genetic algorithm (GA), and artificial bee colony (ABC), have been conducted to further evaluate the performance of the NMOPSO. Those algorithms are implemented based on [44], [45], and [46], respectively. The generated paths are shown in Figure 9 and 10. It can be seen that all algorithms are able to generate feasible paths. The NMOPSO, however, introduces the shortest paths with small turning angles. Table 1 shows the values of the objective functions corresponding to the paths generated by those algorithms. It can be seen that the ABC gives average results across all objectives. The DE produces good results for F_2 and F_3, but the path length is not optimized. The GA produces poor results in complex scenarios due to its node reduction mechanism. The NMOPSO introduces the best overall performance reflected via its lowest cost for F_1 and F_4 in most scenarios. The cost associated with the flight altitude is not as good as the DE due to kinematic constraints. However, it ensures the paths generated are feasible for the UAV to follow. In addition, the generation of multiple non-dominated solutions gives the NMOPSO capabilities to fulfill different needs of applications.

4.5 Comparison with other multi-objective optimization algorithms

Comparisons with other multi-objective optimization algorithms, including the original multi-objective particle swarm optimization (MOPSO) algorithm [38], non-dominated sorting genetic algorithm (NSGA-II) [31], and Pareto envelope-based selection algorithm II (PESA-II) [47], have been conducted to evaluate the non-dominated solutions generated by the NMOPSO. Metrics used for comparisons include the

Table 1: Comparison result with single-objective algorithms

Scens.	Objective	Algorithm								
		NMOPSO			ABC	DE	GA	PSO	QPSO	θ-PSO
1	F1	**0.0127**	0.0164	0.0382	0.1247	0.3991	0.2691	0.2709	0.2542	0.2782
	F2	0.0262	0.0254	**0**	0.0005	**0**	0.0011	**0**	0.0002	4.09E-9
	F3	0.289	0.086	0.0691	0.176	0.0154	0.2774	**0.0003**	0.1553	0.0004
	F4	**0.036**	**0.036**	0.0593	0.1272	0.0842	0.1932	0.0775	0.1065	0.0852
2	F1	0.0577	0.0348	**0.0256**	0.1767	0.3675	0.1311	0.1188	0.1224	0.1546
	F2	**0**	**0**	0.0235	0.0003	**0**	0.0041	4.31E-5	0.0011	0.0002
	F3	0.1807	0.1855	0.3469	0.166	0.0207	0.3124	**0.0032**	0.1525	0.0218
	F4	0.0473	0.0487	**0.0328**	0.137	0.0891	0.1363	0.1152	0.1637	0.1199
3	F1	0.0261	0.0457	**0.0189**	0.2736	0.3081	0.2103	0.3167	0.1647	0.3502
	F2	0.0147	**0**	0.0194	1.33E-6	**0**	0.0028	7.42E-7	0.0006	**0**
	F3	0.0941	0.1185	0.1597	0.1666	0.0164	0.2662	6.55E-5	0.1186	**5.86E-5**
	F4	0.0505	0.0628	**0.0356**	0.1454	0.0871	0.2462	0.0858	0.1464	0.0867
4	F1	0.2133	**0.0519**	0.1437	0.2706	0.234	0.3284	0.2962	0.2322	0.3034
	F2	0.0145	0.0059	**0**	8.33E-6	1.28E-7	**0**	6.06E-5	0.0006	2.51E-7
	F3	0.0876	0.1605	0.0865	0.097	0.025	0.2747	**0.0002**	0.499	0.0005
	F4	0.0942	**0.0745**	0.148	0.1151	0.0922	0.2178	0.0885	0.1298	0.0923

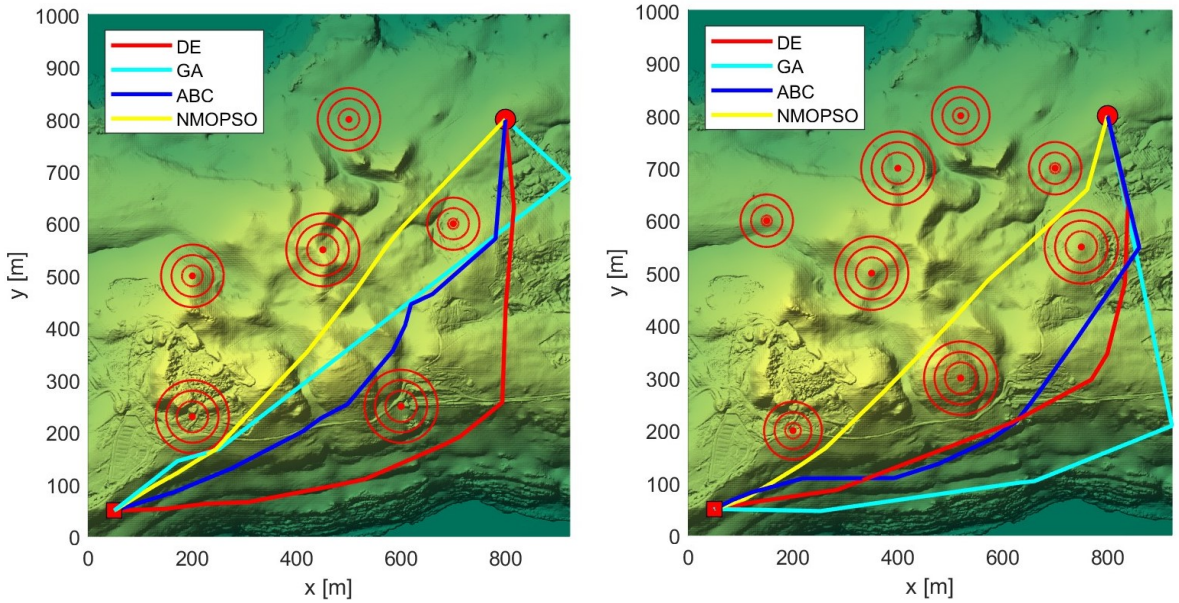


(a) Scenario 1 (b) Scenario 2

Figure 9: Top view of the paths generated by the NMOPSO and other meta-heuristic algorithms

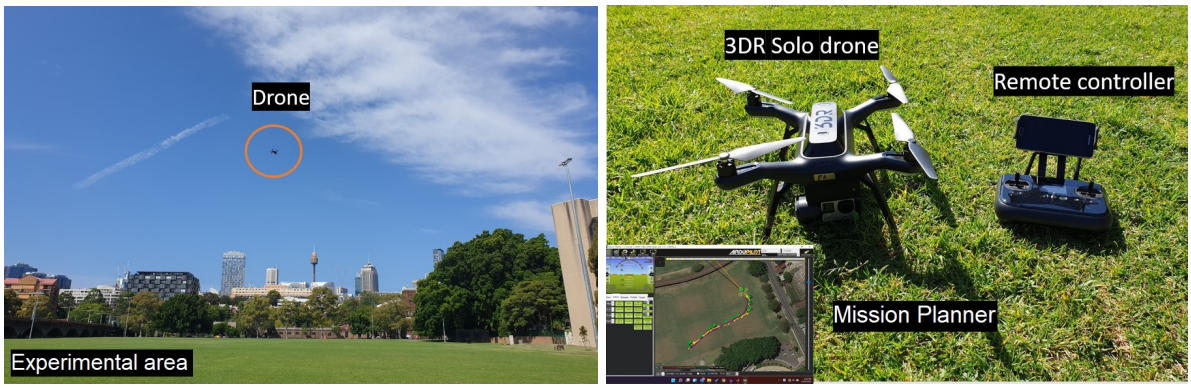
maximum value (max), minimum value (min), mean value (mean), and standard deviation (std) of each objective on the obtained Pareto front. A new metric called solution distribution, s_d, is also used to evaluate the diversity of non-dominated solutions. It is defined as:

$$s_d = \frac{n_p}{n_o}, \tag{28}$$



(a) Scenario 3 (b) Scenario 4

Figure 10: Top view of the paths generated by NMOPSO and other nature-inspired algorithms in complex elevation maps



(a) Experimental environment (b) 3DR Solo drone and Mission Planner

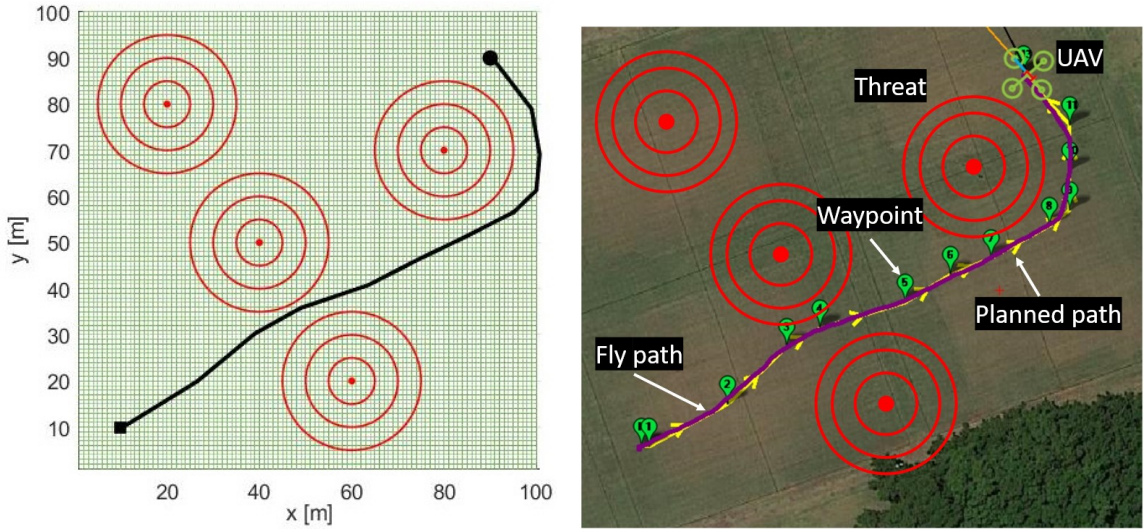
Figure 11: The UAV used and experimental environment

where n_p is the number of non-dominated solutions found and n_o is the number of cells occupied by those non-dominated solutions. A small value of s_d thus indicates a good distribution of solutions, which is expected for the multi-objective problem.

The comparison result is shown in Table 2. It can be seen that the proposed NMOPSO outperforms other algorithms in objectives F_1 and F_4. Particularly in objective F_1, NMOPSO demonstrates the best optimization capability across all three metrics: max, min, and mean. For objectives F_2 and F_3, PESA-II yields good mean values. However, when considering the max, min, and std metrics in scenarios 2, 3, and 4, along with the result for other objectives, it is evident that PESA-II produces less diverse solutions. This is also reflected in its highest s_d value. NSGA-II provides reasonable results but is not dominant in any particular objective. Similarly, MOPSO only achieves average results due to the local minimum issue. NMOPSO resolves this issue through the proposed adaptive mutation mechanism. This mechanism also enhances the solution distribution reflected via the best s_d values of NMOPSO in most scenarios.

Table 2: Comparison result for multi-objective algorithms

Scens.	Alg.	F1				F2				F3				F4				s_d
		Max	Min	Mean	Std	Max	Min	Mean	Std	Max	Min	Mean	Std	Max	Min	Mean	Std	
1	NMOPSO	0.0615	0.0107	**0.0178**	0.0099	0.0434	0.0000	0.0141	0.0119	0.6742	0.0642	0.2691	0.1621	0.0663	0.0243	**0.0376**	0.0092	**1.1354**
	NSGA-II	0.3104	0.0325	0.1225	0.0802	0.0500	0.0000	0.0068	0.0091	0.7786	0.0172	0.2654	0.2391	0.1683	0.0690	0.1028	0.0247	2.2644
	PESA-II	0.6089	0.5317	0.5630	0.0232	0.0098	0.0000	**0.0020**	0.0028	0.2639	0.1755	**0.2120**	0.0257	0.5047	0.3566	0.4189	0.0447	2.8368
	MOPSO	0.3734	0.2035	0.2826	0.0408	0.0306	0.0000	0.0087	0.0083	0.1632	0.0071	0.0550	0.0485	0.2765	0.1308	0.1771	0.0350	1.1831
2	NMOPSO	0.0472	0.0249	**0.0314**	0.0050	0.0293	0.0000	0.0092	0.0085	0.6585	0.0869	0.2741	0.1464	0.0551	0.0289	**0.0372**	0.0060	**1.0861**
	NSGA-II	0.2918	0.0569	0.1464	0.0671	0.0421	0.0000	0.0058	0.0081	0.7905	0.0184	0.2325	0.2410	0.1888	0.0721	0.1158	0.0331	2.2087
	PESA-II	0.6254	0.5711	0.5856	0.0130	0.0026	0.0000	**0.0006**	0.0008	0.2505	0.1959	0.2159	0.0134	0.4146	0.3286	0.3548	0.0234	3.3209
	MOPSO	0.2085	0.1880	0.1955	0.0049	0.0314	0.0000	0.0089	0.0083	0.0475	0.0139	**0.0274**	0.0089	0.1416	0.1075	0.1185	0.0070	1.0865
3	NMOPSO	0.0759	0.0119	**0.0234**	0.0143	0.0388	0.0000	0.0109	0.0106	0.6107	0.0748	0.2947	0.1596	0.0860	0.0306	**0.0449**	0.0118	**1.1827**
	NSGA-II	0.2930	0.0505	0.1298	0.0661	0.0435	0.0000	0.0062	0.0083	0.7436	0.0342	0.2277	0.2110	0.2011	0.0697	0.1168	0.0361	2.2316
	PESA-II	0.4817	0.4442	0.4648	0.0114	0.0059	0.0000	**0.0022**	0.0019	0.2863	0.1905	0.2514	0.0290	0.4012	0.3503	0.3763	0.0166	3.0891
	MOPSO	0.2276	0.1752	0.1911	0.0146	0.0188	0.0000	0.0049	0.0046	0.0629	0.0119	**0.0316**	0.0129	0.1420	0.0709	0.0933	0.0187	1.2853
4	NMOPSO	0.3204	0.0797	**0.1619**	0.0593	0.0303	0.0000	0.0067	0.0079	0.7717	0.0773	0.3245	0.1875	0.1626	0.0695	**0.0975**	0.0210	**1.1567**
	NSGA-II	0.3981	0.1075	0.2419	0.0715	0.0342	0.0000	**0.0040**	0.0062	0.8137	0.0436	0.2889	0.2288	0.2463	0.0843	0.1452	0.0438	2.2546
	PESA-II	0.5497	0.4952	0.5223	0.0179	0.0047	0.0000	0.2311	0.0015	0.2696	0.1944	0.2311	0.0228	0.4366	0.3500	0.3854	0.0274	3.0496
	MOPSO	0.2745	0.1848	0.2367	0.0245	0.0252	0.0000	0.0072	0.0068	0.2239	0.0240	**0.0776**	0.0496	0.1828	0.0890	0.1223	0.0221	1.2374



(a) Top view of the planned path generated by NMOPSO in MATLAB

(b) The planned path (yellow) and the actual flight paths (purple) in experiments

Figure 12: Experimental results

4.6 Experimental validation

To verify the validity of the proposed algorithm in generating paths for practical flights, experiments have been carried out with a real UAV named 3DR Solo. This UAV can be programmed to fly automatically via a ground control station software called Mission Planner, as depicted in Figure 11b. The experimental area is located on a flat terrain at the latitude and longitude of $(-33.876399, 151.192293)$. It has the size of 100×100 m^2 and is augmented with four obstacles, as shown in Figure 11a. This information, along with the UAV's starting and goal locations, is input into the NMOPSO algorithm implemented in MATLAB to generate a planned path comprising a list of waypoints, as shown in Figure 12a. Those waypoints are then converted to geographic coordinates and uploaded to the 3DR Solo drone via Mission Planner to fly.

Figure 12b shows the flight result in which the yellow line represents the planned path and the purple line represents the actual flight path. Their overlap implies that the path generated by NMOPSO is feasible for the drone to follow. Since the flight path does not intersect the threat circles, the path is safe for drone operation. The results thus demonstrate the validity of NMOPSO for practical flights.

4.7 Discussion

Using a multi-objective approach, the NMOPSO generates a set of non-dominated solutions that can meet various application requirements. For example, the third solution for Scenario 2 in Table 1 minimizes the energy consumption as it represents a short and smooth path, while the first and second solutions prioritize safety. The NMOPSO also allows new objectives to be added as additional dimensions of its hypercube. The algorithm is, therefore, scalable and suitable for complex tasks. Besides, the inclusion of kinematic constraints to the problem helps narrow down the solution space. The NMOPSO takes advantage of this by utilizing navigation variables to speed up the process of finding non-dominated solutions. The multi-objective approach, however, faces the challenge of finding a large number of non-dominated solutions, which requires a balance between exploration and exploitation. Some of our enhancements to the NMOPSO, such as utilizing a repository to store non-dominated solutions and implementing a region-based mutation mechanism, can address this issue, but at the cost of adding additional computation requirements.

5 Conclusion

In this work, a new path-planning algorithm, NMOPSO, has been introduced to generate Pareto optimal paths for UAVs considering their kinematic constraints. Several mechanisms such as navigation variables, fitness evaluation, and adaptive mutation have been integrated into the algorithm to better explore the solution space for non-dominated solutions. Comparison results show that the NMOPSO outperforms other PSO variants and state-of-the-art metaheuristic optimization algorithms in most criteria including path length, safety and smoothness. In addition, experiments with paths generated for real flights have been conducted. The overlap between the planned and actual flight paths confirms the validity of our approach for practical UAV operations.

Conflict of interest

On behalf of all authors, the corresponding author states that there is no conflict of interest.

Data availability statement

The data generated in this study is available from the authors on reasonable request.

Financial interests

The authors have no relevant financial or non-financial interests to disclose.

References

[1] Wang, R.-B., Wang, W.-F., Geng, F.-D., Pan, J.-S., Chu, S.-C., Xu, L.: UAV path planning in mountain areas based on a hybrid parallel compact arithmetic optimization algorithm. Neural Computing and Applications, 1–16 (2023) https://doi.org/10.1007/s00521-023-08983-2

[2] Puente-Castro, A., Rivero, D., Pazos, A., Fernandez-Blanco, E.: A review of artificial intelligence applied to path planning in UAV swarms. Neural Computing and Applications, 1–18 (2022) https://doi.org/10.1007/s00521-021-06569-4

[3] Zhen, X., Enze, Z., Qingwei, C.: Rotary unmanned aerial vehicles path planning in rough terrain based on multi-objective particle swarm optimization. Journal of Systems Engineering and Electronics **31**(1), 130–141 (2020) https://doi.org/10.21629/JSEE.2020.01.14

[4] Zhao, Y., Zheng, Z., Liu, Y.: Survey on computational-intelligence-based UAV path planning. Knowledge-Based Systems **158**, 54–64 (2018) https://doi.org/10.1016/j.knosys.2018.05.033

[5] Mandloi, D., Arya, R., Verma, A.K.: Unmanned aerial vehicle path planning based on A* algorithm and its variants in 3d environment. International Journal of System Assurance Engineering and Management **12**(5), 990–1000 (2021)

[6] Zhang, Z., Wu, J., Dai, J., He, C.: A novel real-time penetration path planning algorithm for stealth UAV in 3D complex dynamic environment. IEEE Access **8**, 122757–122771 (2020) https://doi.org/10.1109/ACCESS.2020.3007496

[7] Fan, J., Chen, X., Wang, Y., Chen, X.: UAV trajectory planning in cluttered environments based on PF-RRT* algorithm with goal-biased strategy. Engineering Applications of Artificial Intelligence **114**, 105182 (2022) https://doi.org/10.1016/j.engappai.2022.105182

[8] Pan, Z., Zhang, C., Xia, Y., Xiong, H., Shao, X.: An improved artificial potential field method for path planning and formation control of the multi-UAV systems. IEEE Transactions on Circuits and Systems II: Express Briefs **69**(3), 1129–1133 (2022) https://doi.org/10.1109/TCSII.2021.3112787

[9] Jayaweera, H.M., Hanoun, S.: A dynamic artificial potential field (D-APF) UAV path planning technique for following ground moving targets. IEEE Access **8**, 192760–192776 (2020) https://doi.org/10.1109/ACCESS.2020.3032929

[10] Sababha, B.H., Al-mousa, A., Baniyounisse, R., Bdour, J.: Sampling-based unmanned aerial vehicle air traffic integration, path planning, and collision avoidance. International Journal of Advanced Robotic Systems **19**(2), 17298806221086431 (2022) https://doi.org/10.1177/17298806221086431

[11] Kothari, M., Postlethwaite, I.: A probabilistically robust path planning algorithm for UAVs using rapidly-exploring random trees. Journal of Intelligent & Robotic Systems **71**(2), 231–253 (2012) https://doi.org/10.1007/s10846-012-9776-4

[12] Karaman, S., Walter, M.R., Perez, A., Frazzoli, E., Teller, S.: Anytime motion planning using the RRT*. In: 2011 IEEE International Conference on Robotics and Automation, pp. 1478–1483 (2011). https://doi.org/10.1109/ICRA.2011.5980479

[13] Li, H., Liu, X., Huang, Z., Zeng, C., Zou, P., Chu, Z., Yi, J.: Newly emerging nature-inspired optimization - algorithm review, unified framework, evaluation, and behavioural parameter optimization. IEEE Access **8**, 72620–72649 (2020) https://doi.org/10.1109/ACCESS.2020.2987689

[14] Duan, H., Li, P.: UAV Path Planning, pp. 99–142. Springer, Berlin, Heidelberg (2014). https://doi.org/10.1007/978-3-642-41196-0_4

[15] Sharma, A., Shoval, S., Sharma, A., Pandey, J.K.: Path planning for multiple targets interception by the swarm of UAVs based on swarm intelligence algorithms: A review. IETE Technical Review **39**(3), 675–697 (2022) https://doi.org/10.1080/02564602.2021.1894250

[16] Cheng, L., Zhong, L., Zhang, X., Xing, J.: A staged adaptive firefly algorithm for UAV charging planning in wireless sensor networks. Computer Communications **161**, 132–141 (2020) https://doi.org/10.1016/j.comcom.2020.07.019

[17] Schacht-Rodríguez, R., Ponsart, J.-C., García-Beltrán, C.-D., Astorga-Zaragoza, C.-M., Theilliol, D., Zhang, Y.: Path planning generation algorithm for a class of UAV multirotor based on state of health of lithium polymer battery. Journal of Intelligent & Robotic Systems **91**(1), 115–131 (2018) https://doi.org/10.1007/s10846-018-0870-0

[18] Wu, Y., Wu, S., Hu, X.: Cooperative path planning of UAVs & UGVs for a persistent surveillance task in urban environments. IEEE Internet of Things Journal **8**(6), 4906–4919 (2021) https://doi.org/10.1109/JIOT.2020.3030240

[19] Xu, C., Duan, H., Liu, F.: Chaotic artificial bee colony approach to uninhabited combat air vehicle (UCAV) path planning. Aerospace Science and Technology **14**(8), 535–541 (2010) https://doi.org/10.1016/j.ast.2010.04.008

[20] Phung, M.D., Quach, C.H., Dinh, T.H., Ha, Q.: Enhanced discrete particle swarm optimization path planning for UAV vision-based surface inspection. Automation in Construction **81**, 25–33 (2017) https://doi.org/10.1016/j.autcon.2017.04.013

[21] Yu, Z., Si, Z., Li, X., Wang, D., Song, H.: A novel hybrid particle swarm optimization algorithm for path planning of UAVs. IEEE Internet of Things Journal, 1–1 (2022) https://doi.org/10.1109/JIOT.2022.3182798

[22] Phung, M.D., Ha, Q.: Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization. Applied Soft Computing **107**, 107376 (2021) https://doi.org/10.1016/j.asoc.2021.107376

[23] Shi, L., Xu, S.: UAV path planning with QoS constraint in device-to-device 5G networks using particle swarm optimization. IEEE Access **8**, 137884–137896 (2020) https://doi.org/10.1109/ACCESS.2020.3010281

[24] Morin, M., Abi-Zeid, I., Quimper, C.-G.: Ant colony optimization for path planning in search and rescue operations. European Journal of Operational Research (2022) https://doi.org/10.1016/j.ejor.2022.06.019

[25] Gaing, Z.-L.: Particle swarm optimization to solving the economic dispatch considering the generator constraints. IEEE Transactions on Power Systems **18**(3), 1187–1195 (2003) https://doi.org/10.1109/TPWRS.2003.814889

[26] Eberhart, R.C., Shi, Y.: Comparison between genetic algorithms and particle swarm optimization. In: Evolutionary Programming VII, pp. 611–616. Springer, Berlin, Heidelberg (1998). https://doi.org/10.1007/BFb0040812

[27] Fu, Y., Ding, M., Zhou, C.: Phase angle-encoded and quantum-behaved particle swarm optimization applied to three-dimensional route planning for UAV. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans **42**(2), 511–526 (2012) https://doi.org/10.1109/TSMCA.2011.2159586

[28] Hidalgo-Paniagua, A., Vega-Rodríguez, M.A., Ferruz, J., Pavón, N.: Solving the multi-objective path planning problem in mobile robotics with a firefly-based approach. Soft Computing **21**(4), 949–964 (2017) https://doi.org/10.1007/s00500-015-1825-z

[29] Zhang, Y., Gong, D.-w., Zhang, J.-h.: Robot path planning in uncertain environment using multi-objective particle swarm optimization. Neurocomputing **103**, 172–185 (2013) https://doi.org/10.1016/j.neucom.2012.09.019

[30] Ahmed, F., Deb, K.: Multi-objective optimal path planning using elitist non-dominated sorting genetic algorithms. Soft Computing **17**(7), 1283–1299 (2013) https://doi.org/10.1007/s00500-012-0964-8

[31] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation **6**(2), 182–197 (2002) https://doi.org/10.1109/4235.996017

[32] Yang, X.-S.: Multiobjective firefly algorithm for continuous optimization. Engineering with Computers **29**(2), 175–184 (2013) https://doi.org/10.1007/s00366-012-0254-1

[33] Reyes-Sierra, M., Coello, C.: Multi-objective particle swarm optimizers: A survey of the state-of-the-art. International Journal of Computational Intelligence Research **2**, 287–308 (2006) https://doi.org/10.5019/j.ijcir.2006.68

[34] Li, B., Zhang, J., Dai, L., Teo, K.L., Wang, S.: A hybrid offline optimization method for reconfiguration of multi-UAV formations. IEEE Transactions on Aerospace and Electronic Systems **57**(1), 506–520 (2021) https://doi.org/10.1109/TAES.2020.3024427

[35] Gunantara, N.: A review of multi-objective optimization: Methods and its applications. Cogent Engineering **5**(1), 1502242 (2018) https://doi.org/10.1080/23311916.2018.1502242

[36] Yen, G.G., Leong, W.F.: Dynamic multiple swarms in multiobjective particle swarm optimization. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans **39**(4), 890–911 (2009) https://doi.org/10.1109/TSMCA.2009.2013915

[37] Trivedi, V., Varshney, P., Ramteke, M.: A simplified multi-objective particle swarm optimization

algorithm. Swarm Intelligence **14**(2), 83–116 (2020) https://doi.org/10.1007/s11721-019-00170-1

[38] Coello Coello, C.A., Lechuga, M.S.: MOPSO: a proposal for multiple objective particle swarm optimization. In: Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600), vol. 2, pp. 1051–10562 (2002). https://doi.org/10.1109/CEC.2002.1004388

[39] Rostami, S., Neri, F.: Covariance matrix adaptation pareto archived evolution strategy with hypervolume-sorted adaptive grid algorithm. Integrated Computer-Aided Engineering **23**(4), 313–329 (2016) https://doi.org/10.3233/ica-160529

[40] Corke, P.I.: A simple and systematic approach to assigning denavit–hartenberg parameters. IEEE Transactions on Robotics **23**(3), 590–594 (2007) https://doi.org/10.1109/TRO.2007.896765

[41] Geoscience Australia: Digital Elevation Model (DEM) of Australia derived from LiDAR 5 Metre Grid. Commonwealth of Australia (Geoscience Australia) (2015). https://doi.org/10.26186/89644

[42] Sun, J., Feng, B., Xu, W.: Particle swarm optimization with particles having quantum behavior. In: Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753), vol. 1, pp. 325–3311 (2004). https://doi.org/10.1109/CEC.2004.1330875

[43] Hoang, V.T., Phung, M.D., Dinh, T.H., Ha, Q.P.: Angle-encoded swarm optimization for UAV formation path planning. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5239–5244 (2018). https://doi.org/10.1109/IROS.2018.8593930

[44] Storn, R., Price, K.: Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization **11**(4), 341–359 (1997) https://doi.org/10.1023/A:1008202821328

[45] Roberge, V., Tarbouchi, M., Labonte, G.: Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. IEEE Transactions on Industrial Informatics **9**(1), 132–141 (2013) https://doi.org/10.1109/TII.2012.2198665

[46] Karaboga, D., Basturk, B.: On the performance of artificial bee colony (ABC) algorithm. Applied Soft Computing **8**(1), 687–697 (2008) https://doi.org/10.1016/j.asoc.2007.05.007

[47] Corne, D.W., Jerram, N.R., Knowles, J.D., Oates, M.J.: PESA-II: Region-based selection in evolutionary multiobjective optimization. In: Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation. GECCO'01, pp. 283–290. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2001)