Applied Soft Computing Journal 167 (2024) 112269

Contents lists available at ScienceDirect

Applied Soft Computing

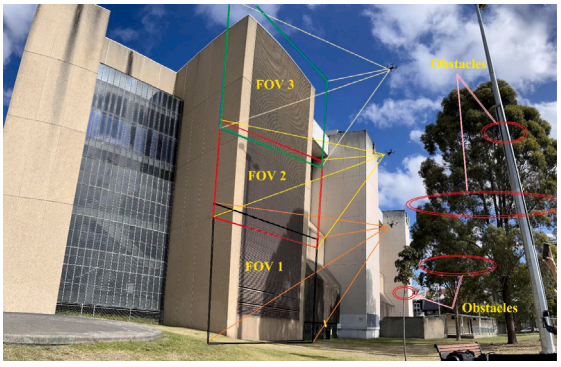
journal homepage: www.elsevier.com/locate/asoc

Fermat-Weber location particle swarm optimization for cooperative path planning of unmanned aerial vehicles

Lanh Van Nguyen [a,*], Ngai Ming Kwok [b], Quang Phuc Ha [a]

[a] School of Electrical and Data Engineering, University of Technology Sydney, Ultimo NSW 2007, Australia
[b] School of Engineering, Design and Built Environment, Western Sydney University, Penrith NSW 2751, Australia

GRAPHICAL ABSTRACT



Multi-UAV Building Inspection using the Fermat-Weber Location Particle Swarm Optimization algorithm for Cooperative Path Planning

ARTICLE INFO

Keywords:
UAV
Path planning
PSO
Fermat Weber location
Game theory

ABSTRACT

This paper presents an effective algorithm, called the Fermat-Weber location particle swarm optimization (FWL-PSO), developed for cooperative path planning of Unmanned Aerial Vehicles (UAVs). Initially, FWL-PSO is constructed by harnessing the Fermat-Weber optimality to identify potential solutions. Within the framework of FWL-PSO, a collection of high-performing particles is established, determined by their respective fitness scores. Following this, the Fermat-Weber location of these elite particles is calculated to supersede the traditional global best, thereby augmenting the learning strategy of the standard PSO. As a result, this method enables the evolution of information while encouraging search diversity. Subsequently, FWL-PSO is employed for handling the interactions of multiple UAVs. In this context, the path planning for a group of UAVs is formulated as a Nash game that incorporates all cooperative interdependencies and safety conditions. The algorithm is then integrated to solve the optimization problem for achieving the Nash equilibrium. To assess its efficacy, extensive simulations and experiments are conducted across a variety of path-planning scenarios. Comparative analyses between FWL-PSO and existing PSO variants underscore the enhanced efficiency and reliability of our proposed approach.

1. Introduction

The increasing prevalence of unmanned aerial vehicles (UAVs) across a myriad of applications has accentuated the importance of path planning for efficacious task execution in such domains as surveillance [1], agriculture [2], search and rescue [3], infrastructural inspections [4], among others. In this regard, the attainment of precise and

* Corresponding author.
E-mail addresses: vanlanh.nguyen@uts.edu.au (L.V. Nguyen), drnmkwok@hotmail.com (N.M. Kwok), quang.ha@uts.edu.au (Q.P. Ha).

https://doi.org/10.1016/j.asoc.2024.112269
Received 8 November 2023; Received in revised form 22 August 2024; Accepted 11 September 2024
Available online 23 September 2024

dependable path planning is paramount to guarantee their effective deployment. The process of UAV path planning involves numerous factors to be addressed. This procedure aims to determine optimal flight trajectories that not only guide the drones through complex environments but also take into account a number of interdependent factors such as obstacle avoidance, energy efficiency, mission objectives, and safety. In complex settings, such as urban landscapes or disaster areas, the formulation of paths that can circumvent obstacles, accommodate terrain variations, adapt to dynamic constraints, and optimize flight distance given time and energy limits is of utmost importance.

In the domain of UAV path planning, numerous strategies have been proposed. Traditional methodologies, such as the A* algorithm, employed in grid-based scenarios, and the Dijkstra algorithm, utilized for weighted graphs, demonstrate exceptional performance in specific contexts. Nevertheless, they confront scalability challenges as the dimension of the search space expands exponentially [5,6]. Conversely, sampling-based techniques, exemplified by the rapidly exploring random trees (RRT) and probabilistic roadmaps (PRM), exhibit proficiency in the generation of viable flight paths [7,8]. However, they occasionally encounter difficulties in accommodating the maneuver constraints of UAVs, potentially resulting in substantial deviations between the intended and realized flight paths. Meanwhile, methods such as visibility graphs and potential fields are capable of generating smooth and continuous flight paths, but they may grapple with the issue of local minima [9,10].

Unlike traditional methods, metaheuristic optimization methods provide effective solutions for tackling complex optimization problems. They have shown great potential in seeking a feasible solution under multiple constraints and objectives. A diverse range of algorithms has been developed for this purpose, including cuckoo search [11], genetic algorithm (GA) [12], differential evolution (DE) [13], and colony optimization [14]. Drawing inspiration from the collective behavior of birds and fish, particle swarm optimization (PSO) operates as a population-based algorithm that exhibits two critical aspects of swarm intelligence: cognitive and social cohesion [15]. These characteristics empower individual particles within the swarm to drive an optimization solution by combining personal experiences with collective insights, deviating from traditional evolutionary approaches. Therefore, compared to other alternative techniques, PSO exhibits superior performance, computational efficiency and robustness to initial conditions and variations in objectives, allowing for its flexible operation in many application environments through a streamlined set of parameters [16]. With its inherent swarm dynamics, PSO can be parallelized across multiple processors or computing clusters, effectively reducing the latency in offline and online path planning scenarios [17]. Capitalizing on these advantages, PSO has been employed in numerous applications, including UAV path planning.

Despite its commendable attributes, PSO exhibits a substantial drawback by showing a propensity for entrapment within local optima, wherein particles tend to converge towards the most favorable position identified within their entire group. In scenarios governed by multimodal functions, it is quite common for the favorable position to align with a local optimum rather than the global one, potentially resulting in suboptimal solutions. To tackle this issue, there have been some enhancements of the PSO algorithms, such as adaptively adjusting their acceleration coefficients [18] or particle neighborhood structures [19]. However, there may remain a possibility of inappropriate parameter tuning due to the underutilization of fitness information across the population. Auxiliary techniques to avoid local minima while enhancing population diversity include a chaotic and sharing-learning PSO addressing extended traveling salesman problems [20], a hybrid PSO utilizing cross-learning strategies to overcome premature convergence [21], and a vector-based PSO employing spherical vectors for effective UAV mission space exploration [22], among others. While search performance can be improved to some extent, these algorithms may face difficulties regarding convergence speed and solution accuracy.

To overcome the limitations mentioned above and also expand the scope of solution exploration, various learning strategies have been proposed. For example, in social learning particle swarm optimization (SL-PSO) [23], social learning principles are incorporated for suboptimal particles to gather knowledge from superior ones, while simultaneously adjusting their positions according to that of the mean. Other techniques have been developed using the level learning methodology. These include the level-based learning swarm optimizer (LLSO) [24], the two-phase learning-based swarm optimizer (TPLSO) [25], and the three-phase competitive swarm optimizer (TPCSO) [26]. Therein, particles are categorized based on their fitness values. Suboptimal particles receive guidance from others positioned within higher-performing levels. Conversely, particles situated in the highest-performing level are preserved and directly propagated to the ensuing generation.

In terms of search efficiency, the previously mentioned methods often rely on the selection of a better-performing particle to update those that are less effective, without a specific mechanism for harnessing the insights derived from the pool of superior particles. This limitation can potentially obscure valuable information inherent in elite particles, thus impeding the ability of the algorithm to approach an optimal solution. As a remedy, the present paper introduces a novel algorithm known as Fermat-Weber location particle swarm optimization (FWL-PSO). Here, the Fermat-Weber location, a geometric concept renowned for its advantageous properties in the optimization problem of transportation [27] or UAV-assisted IoT networks [28], is incorporated into our development. The FWL inclusion guides the algorithm towards the exploration of promising regions with enhanced convergence properties, ultimately elevating solution quality via establishing the Fermat-Weber location as a universal point of reference for the elite particles. This fosters a convergence towards a global solution with possibilities to reveal latent representations and interactions within the particle group.

In the context of multiple UAVs, cooperative path planning can be formulated as an optimization problem for individual vehicles under various constraints [29,30]. This involves computing optimal control signals for each UAV from the leader [29] or the centroid [30] of the entire group. Another technique, the artificial potential field (APF), is utilized for the path planning of multiple UAVs [31]. While capable of generating continuous paths subject to constraints, they may suffer from high computational latency or local minima. Notably, computational evolution algorithms have been applied to multi-UAV cooperative path planning, allowing optimal solutions in complex scenarios [32]. These algorithms typically involve multiple layers for individual UAV path planning and cooperation. Initially, each vehicle employs an evolutionary strategy to generate a feasible path for executing a task. These paths are then refined using a global cost function to ensure cooperation. While this method can produce smooth flying paths for the group without discretizing the workspace, it may converge to sub-optimal solutions if cooperative constraints and UAV maneuver tasks are not adequately addressed.

The main problem in multiple UAV cooperation, compared to path planning for a single UAV, lies in resolving conflicts and interactions among group members while achieving a common task under various individual and interconnected constraints. To this end, the game theory, a branch of mathematics focusing on studying conflicts and interactions of rational decision-makers, provides a promising tool for determining optimal strategies [33]. In the literature, most games can be categorized as either cooperative or non-cooperative. Cooperative games (CGs) involve multiple players with a shared goal, aiming to enhance their collective performance compared to individual efforts [34]. However, there is a trade-off of CGs in balancing the stability of player coalitions with overall system efficiency. In contrast, non-cooperative games (NCGs) feature players with distinct properties such as individual payoff functions, game procedures, intentions, and potential strategies, making them easier to proceed with than CGs. Among NCGs, the Nash game is widely utilized in scenarios where all players can

make simultaneous decisions in symmetric competitions, such as determining the maximum coverage value of UAVs in a multi-level and multi-dimensional assisted network [35]. In Nash games, the objective is to reach a solution known as the Nash equilibrium, whereby each player, while aware of others' strategies, can independently make decisions within a competitive environment. Motivated by these works, to address the complex dynamics inherent in UAV cooperative path planning, conceptualized here as a game, we propose integrating FWL-PSO into the game framework to identify its Nash equilibrium.

Addressing the cooperative path planning for multiple UAVs aimed at civil infrastructure inspection by using swarm optimization and game theory, the contributions of this work are threefold: (i) the development of a novel FWL-PSO algorithm for obtaining computationally efficient optimal solutions; (ii) the integration of FWL-PSO into game theory to obtain the Nash equilibrium for cooperative path planning of UAVs; and (iii) the demonstration of building inspection using multiple UAVs to enhance coverage over a large area in a difficult setting of the site. These achievements enable the efficient deployment of versatile UAVs capable of handling complex scenarios, greatly enhancing UAV-based task execution autonomy.

This paper is organized as follows. Section 2 develops the proposed FWL-PSO algorithm. Section 3 focuses on the integration of FWL-PSO with Nash game for cooperative path planning of UAVs. The UAV path planning problem is formulated in this section, focusing on cooperative UAV-based inspection tasks. Section 4 provides extensive simulation results, while Section 5 presents experimental on-field validation. Finally, Section 6 summarizes the findings of this study and draws its conclusions.

2. Fermat-Weber location particle swarm optimization

In many real-world applications, traditional methods often face difficulties due to the presence of multiple local optima. When a cost function has multiple peaks and valleys, it is quite difficult for an optimization algorithm to determine global optima, as they are often stuck in some local ones. To address this issue, heuristic and meta-heuristic approaches such as the PSO have gained popularity, owing to their capability of exploring the solution space more effectively to have a better chance of reaching the optimum. They need, however, to be equipped further with an effective tool to handle more complex problems of multi-modal optimization in a computationally-efficient manner. Here, the FWL-PSO algorithm is proposed for that purpose. The Fermat-Weber location principle aims to find out a point that minimizes the sum of distances to a given set of vertices. This principle can be applied to guide the search process in order to minimize a collective cost function, as formulated in this paper.

2.1. Fermat-Weber location

Given p vertices $X_1, X_2, \dots, X_p$ in $\mathbb{R}^n$, and weights $\eta_i > 0$, $i = 1, \dots, p$, associated with them, the Fermat Weber location, also known as the geometric median, is defined as the point that minimizes the weighted sum of distances from an arbitrary point $Y \in \mathbb{R}^n$, distinct from all vertices, to each vertex X_i:

$$\overset{*}{Y} = \underset{Y \in \mathbb{R}^n}{\operatorname{argmin}} \left\{ f(Y) = \sum_{i=1}^{p} \eta_i \|X_i - Y\|_2 \right\}, \tag{1}$$

where $\|\cdot\|_2$ represents the Euclidean norm. Accordingly, at the Fermat Weber location $\overset{*}{Y}$, we have [36]:

$$\nabla f(\overset{*}{Y}) = \sum_{i=1}^{p} \frac{\eta_i (X_i - \overset{*}{Y})}{\|X_i - \overset{*}{Y}\|_2} = 0. \tag{2}$$

This is equivalent to:

$$\overset{*}{Y} = \left(\sum_{i=1}^{p} \frac{\eta_i X_i}{\|X_i - \overset{*}{Y}\|_2} \right) \Big/ \left(\sum_{i=1}^{p} \frac{\eta_i}{\|X_i - \overset{*}{Y}\|_2} \right). \tag{3}$$

Despite being an easy-to-understand concept, it has been shown that an explicit expression of the geometric median to date is not available, for which the Weiszfeld algorithm can be employed to approximate $\overset{*}{Y}$ as [37]:

$$Y_{j+1} = \begin{cases} \left(\sum_{i=1}^{p} \frac{\eta_i X_i}{\|X_i - Y_j\|_2} \right) \Big/ \left(\sum_{i=1}^{p} \frac{\eta_i}{\|X_i - Y_j\|_2} \right), & \text{if } Y_j \notin \{X_1, X_2, \dots, X_p\} \\ X_i, & \text{if } Y_j = X_i, i = 1, 2, \dots, p, \end{cases} \tag{4}$$

where Y_j represents the estimate of the Fermat Weber location at iteration j in an iterative sequence from any point.

2.2. FWL-PSO development

The proposed FWL-PSO framework is described in detail as follows.

(i) *Representations:*

In the matrix-based framework of FWL-PSO, a population consisting of N particles is engaged in solving a D-dimensional problem. In this context, vectors are used to represent the individuals within the population. For an individual k within the population, the resulting representation in the optimization process spans D dimensions, as expressed by the vector:

$$X_i = [x_{i1}, x_{i2}, \dots, x_{iD}]^\top, \tag{5}$$

where $i = 1, 2, \dots, N$. To represent the entire population, an $N \times D$ matrix is used [38]:

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{ND} \end{bmatrix}. \tag{6}$$

In this representation, each row corresponds to an individual, and each column corresponds to a specific variable.

(ii) *Initialization:*

At first, the population position X and its velocity V are initialized randomly in the feasible domain of each dimension as

$$X = \mathbb{1}_N \times (X_U - X_L) \circ R_{N \times D} + \mathbb{1}_N \times X_L, \tag{7}$$

$$V = \mathbb{1}_N \times (V_U - V_L) \circ R_{N \times D} + \mathbb{1}_N \times V_L, \tag{8}$$

where $\circ$ denotes the element-wise Hadamard product, $\mathbb{1}_N$ is an all-ones column vector with N entries, X_U, X_L and V_U, V_L are respectively D-dimensional row vectors of the upper bound, lower bound of the position and velocity variables, and R is an $N \times D-$ matrix whose elements are randomly generated in $[0, 1]$.

Subsequently, the cost of all the individuals is evaluated and recorded in the N-dimensional vector J_X as

$$J_X = J(X). \tag{9}$$

In the initialization stage, the personal best position matrix, denoted as X_P, is directly set to be the same as the particle position matrix X. Similarly, the cost matrix associated with these personal best positions, J_P is also set identical to the cost vector J_X, i.e.,

$$X_P = X, \text{ and } J_P = J_X. \tag{10}$$

(iii) *Fermat-Weber Location Determination:*

After the initialization phase, the FWL-PSO algorithm selects the top p best-performing particles, approximately 5% of the total population. These particles are chosen through a sorting process and then stored in an elite pool, X_P^e. These standout particles are the most successful members of the group to provide valuable insights for finding the best solutions. Their combined knowledge helps them navigate the complexities of optimization more efficiently. The process of selecting the top p of the best-performing particles is outlined as follows:

$$[\tilde{J}_P, I] = sort(J_P), \tag{11a}$$

$$\tilde{X}_P = X_P(I), \tag{11b}$$

$$X_P^e = \tilde{X}_P(1, \dots, p). \tag{11c}$$

In Eq. (11a), the *sort* function is employed to yield the index vector I of the same size with J_P, signifying the arrangement of elements within J_P in an ascending order, resulting in $\tilde{J}_P$. Consequently, the personal best position matrix is rearranged into $\tilde{X}_P$ given in Eq. (11b). Meanwhile, Eq. (11c) describes the stored top p particles of the N-element population, along with their corresponding personal best cost values. The global best position X_G and its value J_G are directly established as the first row in $\tilde{X}_P$ and $\tilde{J}_P$, respectively. That is,

$$X_G = \tilde{X}_P(1, :), \text{ and } J_P = \tilde{J}_P(1, :). \tag{12}$$

A key feature of FWL-PSO is the determination of the Fermat-Weber location using the positions of the selected elite particles. This geometric centroid minimizes the aggregate distances between itself and the elite particles, effectively encapsulating their spatial arrangement. It serves as a reference for the cumulative influence of the elite particles, directing the swarm towards regions of a heightened promise, defined as:

$$\overset{*}{F} = \underset{F \in \mathbb{R}^D}{\operatorname{argmin}} \sum_{i=1}^{p} \eta_i \|X_P^e(i, :) - F\|_2, \tag{13}$$

where $F \in \mathbb{R}^D$ is any position in $\mathbb{R}^D$ of elite particles, and $\overset{*}{F}$ is the Fermat-Weber location minimizing the sum of distances between itself and the elite particles. Here, weight η_i represents the importance of the ith elite particle located at $X_P^e(i, :)$, and p is the total number of elite particles.

Let N_f be the number of iterations to estimate the Fermat-Weber location as per Eq. (4). At each F_j, $j = 1, \dots, N_f$, the next iteration F_{j+1} is computed based on positions of the current F_j and other elite particles. The estimated Fermat-Weber location is then updated accordingly, or remains unchanged if coinciding with one of the elite particle positions:

$$F_{j+1} = \begin{cases} \left(\sum_{i=1}^{p} \dfrac{\eta_i X_P^e(i, :)}{\|X_P^e(i, :) - F_j\|_2} \right) \Big/ \left(\sum_{i=1}^{p} \dfrac{\eta_i}{\|X_P^e(i, :) - F_j\|_2} \right), \\ \qquad \text{if } F_j \notin \{X_P^e(1, :), X_P^e(2, :), \dots, X_P^e(p, :)\} \\ X_P^e(i, :), \qquad \text{if } F_j = X_P^e(i, :), i = 1, 2, \dots, p. \end{cases} \tag{14}$$

In this process, F_1 can be initially set at any point within a feasible domain.

(iv) *Particle Movement and Update:*

Particle positions and velocities are iteratively updated to traverse the search space in pursuit of an optimal solution. The particle motion incorporates the interplay among personal best positions, collective information from elite particles, and spatial guidance from the Fermat-Weber location.

The particle velocity is updated using the equation below:

$$V(t+1) = c_0 \times V(t) + c_1 \times R_1 \circ (X_P(t) - X(t)) + c_2 \times R_2 \circ (\mathbb{1}_N \times F(t) - X(t)), \tag{15}$$

where $t = 1, 2, \dots, T_m$ is the current iteration running from 1 to its maximum T_m, c_0 is the inertia weight, c_1 and c_2 are the acceleration coefficients, R_1 and R_2 are two $N \times D$ matrices containing uniformly distributed random numbers, and F represents the FWL global best in its interactions with particles in the swarm. The particle velocities within each generation are limited by upper and lower bounds expressed by D-dimensional row vectors V_U and V_L, respectively. After updating, particles exceeding velocity limits are adjusted by the following equations:

$$L_{V_U} = \begin{cases} 1, & \text{if } V(t+1) > \mathbb{1}_N \times V_U \\ 0, & \text{otherwise.} \end{cases} \tag{16a}$$

$$L_{V_L} = \begin{cases} 1, & \text{if } V(t+1) < \mathbb{1}_N \times V_L \\ 0, & \text{otherwise.} \end{cases} \tag{16b}$$

$$\begin{aligned} V(t+1) = {} & V(t+1) \circ (\mathbb{1}_{N \times D} - L_{V_U} - L_{V_L}) \\ & + \mathbb{1}_N \times V_U \circ L_{V_U} + \mathbb{1}_N \times V_L \circ L_{V_L}, \end{aligned} \tag{16c}$$

where $\mathbb{1}_{N \times D}$ is an all-ones $N \times D$-matrix. The logic matrices L_{V_U} and L_{V_L} record the elements in V that exceed respectively the corresponding upper and lower bounds.

The particle position is now updated with the new velocity as follows:

$$X(t+1) = X(t) + V(t+1). \tag{17}$$

The new position is also constrained by its maximum X_U and minimum X_L remains within the search space by using similar adjustments:

$$L_{X_U} = \begin{cases} 1, & \text{if } X(t+1) > \mathbb{1}_N \times X_U \\ 0, & \text{otherwise.} \end{cases} \tag{18a}$$

$$L_{X_L} = \begin{cases} 1, & \text{if } X(t+1) < \mathbb{1}_N \times X_L \\ 0, & \text{otherwise.} \end{cases} \tag{18b}$$

$$\begin{aligned} X(t+1) = {} & X(t+1) \circ (\mathbb{1}_{N \times D} - L_{X_U} - L_{X_L}) \\ & + \mathbb{1}_N \times X_U \circ L_{X_U} + \mathbb{1}_N \times X_L \circ L_{X_L}, \end{aligned} \tag{18c}$$

where L_{X_U} and L_{X_L} are logical matrices corresponding to the upper and lower position bounds.

The swarm personal best X_P and its cost J_P are then recomputed, taking into account the relationship with the individual cost J_X, as follows:

$$L_J = \begin{cases} 1, & \text{if } J_X < J_P \\ 0, & \text{otherwise.} \end{cases} \tag{19a}$$

$$X_P = L_J \times \mathbb{1}_D \circ X + (\mathbb{1}_N - L_J) \times \mathbb{1}_D \circ X_P, \tag{19b}$$

$$J_P = L_J \circ J_X + (\mathbb{1}_N - L_J) \circ J_P. \tag{19c}$$

Then, the values of X_P^e, F, X_G, and J_G are also updated accordingly.

(v) *Convergence and Termination*:

The FWL-PSO algorithm runs through a series of iterative steps, progressively refining particle positions and fitness values while making fitness improvements across successive iterations. This ensures the algorithm consistently proceeds towards an optimal solution. A predetermined termination criterion is established for monitoring the algorithm convergence and computational throughput. The termination can take place when fitness values settle, indicating the algorithm convergence, or based on an iteration threshold to simply fix the number of iterations, upholding a balance between solution quality and computational time. Here, an iteration threshold is implemented to conclude the algorithm.

2.3. FWL-PSO implementation

For realization, Algorithm 1 presents the pseudocode for implementing the proposed FWL-PSO. The algorithm commences by initializing

encoded variables with random positions and velocities and then proceeds through a primary loop. Within this loop, it assesses fitness and updates particle dynamics based on the Fermat-Weber location within the pool of elite particles. Consequently, the algorithm capitalizes on spatial optimization and convergence advantages. Termination is triggered when the specified threshold T_m is reached. The resulting Fermat-Weber location of the elite pool is then the global best X_G with the associated cost function J_G.

Algorithm 1 FWL-PSO implementation

Input: Parameters: c_0, c_1, c_2, T_m, N, p;

Begin:

1. Initialize position X as Eq. (7);
2. Initialize velocity V as Eq. (8);
3. Evaluate cost J_X as Eq. (9);
4. Update personal best position X_P and personal best cost J_P as Eq. (10);
5. Sort J_P to find $\tilde{J}_P$, $\tilde{X}_P$, and X_P^e as Eq. (11);
6. Compute the Fermat-Weber location of the elite pool as Eq. (14);

for $t = 1 : T_m$ **do**

7. Update velocity as Eq. (15);
8. Address velocity exceeding bounds as Eq. (16);
9. Update position as Eq. (17);
10. Address position exceeding bounds as Eq. (18);
11. Evaluate cost J_X as Eq. (9);
12. Update X_P and J_P as Eq. (19);
13. Sort J_P to find $\tilde{J}_P$, $\tilde{X}_P$, and X_P^e as Eq. (11);
14. Compute the Fermat-Weber location of the elite pool as Eq. (14);

end for

15. Obtain the best solution $X_G = X_P^e(1,:)$; $J_G = \tilde{J}_P(1,:)$

Output: The best solution: X_G, J_G.

End

3. Game-based path planning for UAVs

3.1. Path planning problem formulation

The drone-based execution of complex tasks frequently necessitates the coordinated efforts of multiple UAVs operating as a cohesive team, such as in large-scale 3D mapping or extensive façade inspection for buildings or civil infrastructure. Collaborative control of a UAV group compared to individual UAVs can yield significant benefits in terms of efficiency, reliability, and adaptability [39]. The process of planning paths for multiple UAVs can be formulated as an optimization problem, as described in the following:

$$P_m(0) \xrightarrow[s.t.\ J(\Pi_m,\Pi_m^-)]{P_m(k)} P_m(end),\ m = 1, 2, \ldots, M, \tag{20}$$

where M represents the number of drones with $P_m(0)$ and $P_m(\text{end})$ representing respectively the initial and target poses of UAV$_m$. The path Π_m comprises a sequence of K waypoints in a 3D space $P_m(k) = (x_m(k), y_m(k), z_m(k))$, $k = 1, 2, \ldots, K$, which collectively define the flight trajectory for UAV$_m$. Meanwhile, Π_m^- corresponds to a set of paths taken by neighboring drones to UAV$_m$. The cost function of UAV$_m$ in a cooperative task, $J(\Pi_m, \Pi_m^-)$, consists of a single cost, $J_s(\Pi_m)$, and a cooperative cost, $J_c(\Pi_m, \Pi_m^-)$, determined by

$$J(\Pi_m, \Pi_m^-) = J_s(\Pi_m) + \beta J_c(\Pi_m, \Pi_m^-), \tag{21}$$

where β is a weighting factor.

3.1.1. Single-UAV cost

In Eq. (21), the single-UAV cost $J_s(\Pi_m)$ for an individual drone is computed as:

$$J_s(\Pi) = \omega_1 F_{sa} + \omega_2 F_{tr} + \omega_3 F_{ac} + \omega_4 F_{sm}, \tag{22}$$

where F_{sa} is the safety cost related to potential threats, F_{tr} stands for the traveling cost, F_{ac} represents the altitude constraint cost, and F_{sm} corresponds to the smoothness cost. The weight coefficients ω_i for $i = 1, 2, 3, 4$ are design parameters.

The safety cost F_{sa} is necessary to avoid collisions, based on mitigating the threat of colliding with obstacles. Consider $\mathcal{T}$ obstacles in the flying area, where $D_\tau(k)$ is the distance from the flight segment $\overline{P(k)P(k+1)}$ of a UAV with radius r_u to obstacle τ with radius r_τ at waypoint k. The safety cost is then computed as

$$F_{sa} = \sum_{k=1}^{K-1} \sum_{\tau=1}^{\mathcal{T}} S_\tau(k), \tag{23}$$

where

$$S_\tau(k) = \begin{cases} 0, & \text{if } D_\tau(k) > r_u + r_\tau \\ \infty, & \text{otherwise.} \end{cases} \tag{24}$$

The traveling cost F_{tr} accounts for the UAV path length, which is minimized to save flight time and energy. It is computed as

$$F_{tr} = \sum_{k=1}^{K-1} L(k), \tag{25}$$

where $L(k) = \|P(k+1) - P(k)\|$ denotes the length of the segment $\overline{P(k)P(k+1)}$.

The altitude constraint cost F_{ac} is essential to limit the UAV trajectory height for effective inspection. Let h_{min} and h_{max} be the minimum and maximum heights, respectively, and let $h(k)$ represent the relative height of the UAV with respect to the ground. The desired flying height is $\bar{h} = 0.5(h_{min} + h_{max})$. Accordingly, the altitude constraint cost is calculated as

$$F_{ac} = \sum_{k=1}^{K} H(k), \tag{26}$$

where

$$H(k) = \begin{cases} \left|h(k) - \bar{h}\right|, & \text{if } h_{min} \le h(k) \le h_{max}, \\ \infty, & \text{otherwise.} \end{cases} \tag{27}$$

The smoothness cost F_{sm} is included to limit sharp changes in UAV maneuvers such as turning or climbing, required for inspection coverage. It is computed as

$$F_{sm} = \beta_1 \sum_{k=1}^{K-2} \phi(k) + \beta_2 \sum_{k=1}^{K-1} \left|\varphi(k) - \varphi(k+1)\right|, \tag{28}$$

where $\phi(k)$ and $\varphi(k)$ are the turning and climbing angles, respectively, and β_1 and β_2 represent the corresponding curvature radii.

More details about the above individual cost functions can be found in [40,41]. All these costs share the unit of distance (meters). Here, we apply weighting coefficients ω_i for $i = 1, 2, 3, 4$ in the single-UAV cost to balance the contributions of the individual costs. These weighting factors allow for flexibility in tuning the influence of each cost function on the overall objective, ensuring that no single cost dominates the optimization process.

3.1.2. Cooperative-UAV cost

For civil infrastructure inspection, there is a growing interest in deploying UAVs as mobile sensors, given their distinct advantages over traditional static monitoring methods. The major role of the UAV in this context is to provide a comprehensive coverage of the area of interest and collect vital information, primarily through visual data capture. Here, the cooperative cost for the cooperative constraint, denoted as $J_c(\Pi_m, \Pi_m^-)$, is determined based on the specific cooperative task. In this

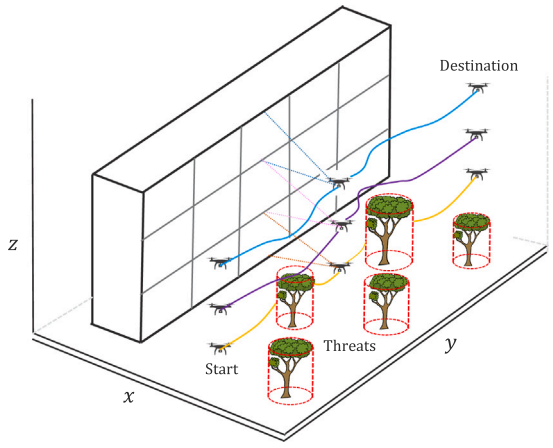


Fig. 1. Façade inspection using multiple UAVs.

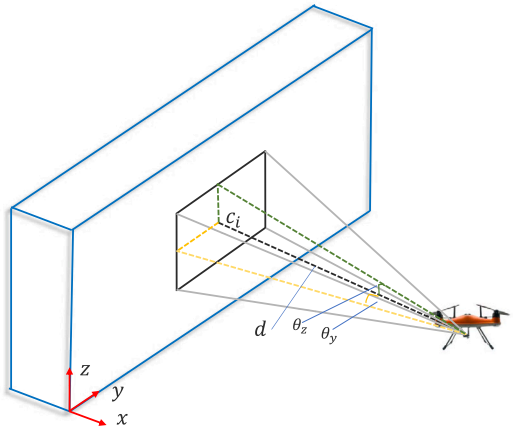


Fig. 2. UAV field of view.

work, our focus is on building façade inspection using multiple UAVs, as illustrated in Fig. 1. Consequently, the cooperative cost is derived as in the following.

Here, each UAV is equipped with a horizontally-facing camera, allowing it to cover a square Field of View (FOV), configuring a pyramid with a half-angle θ, as shown in Fig. 2. Therein, two specific cases for θ are designated as θ_z and θ_y respectively between the vertical medians of the side faces and the height. To be fully covered, a point q falls within the FOV of a UAV if it satisfies the following equations:

$$\|q - c_i\| \leq d \tan\theta, \tag{29}$$

where c_i denotes the projected position of the drone on the surface, while the pyramid height d signifies the distance from the UAV to the surface. The assigned tasks of the UAV are then to achieve maximal coverage within its FOV as well as a desired overlapping level to enhance the overall coverage efficiency.

For ease of reference, we establish the coordinate system xyz fixed to the building, with the z-axis oriented in an upward direction, as depicted in Fig. 2. The UAV flight planning is oriented along the y-axis, denoted as $y_m(k)$. As a result, the vertical FOV becomes the sole focus in the path planning problem, given the maximum distance d_{max} from which a captured surface can still adhere to the minimum target pixel resolution.

The cooperative-UAV cost $J_c(\Pi_m, \Pi_m^-)$ in Eq. (21) is derived from the UAV cooperation and inspection requirements, based on (i) the coverage height H_c, (ii) the overlapping height H_o, and (iii) the avoidance of intervehicle collisions.

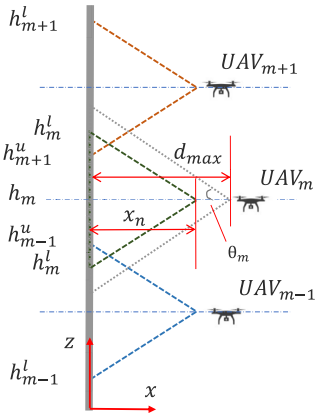


Fig. 3. Cooperative field of view.

(i) Cooperative coverage height: For effective inspection, the height $H_c(k)$ of a UAV trajectory from the group is, in turn, dependent on the maximum distance to remain in FOV d_{max} and the building height H_b, computed as:

$$H_c(k) = \sum_{m=1}^{M} [\max(0, \min(h^u_m, H_b) - \max(h^l_m, h^u_{m-1}))], \tag{30}$$

where h^u_m and h^l_m are respectively the upper and lower vertical coverage bounds of UAV_m, as depicted in Fig. 3, and $h^l_{m-1} = 0$ for $m = 1$. These bounds are obtained as,

$$h^u_m = \begin{cases} z_m + x_m \tan\theta_m, & \text{if } x_m \leq d_{max} \\ 0, & \text{if } x_m > d_{max}, \end{cases} \tag{31a}$$

$$h^l_m = \begin{cases} z_m - x_m \tan\theta_m, & \text{if } x_m \leq d_{max} \\ 0, & \text{if } x_m > d_{max}, \end{cases} \tag{31b}$$

where z_m and θ_m are the $z-$ coordinate and angle θ, associated with UAV_m, as shown Fig. 2.

(ii) Total overlapping height: To guarantee the completeness of the inspection coverage, there should be an overlapping in the FOV of each UAV. This height $H_o(k)$ is thus determined as:

$$H_o(k) = \sum_{m=1}^{M} \left| \mu(h^u_m - h^l_m) - \max(h^l_m - h^u_{m-1}, 0) \right|, \tag{32}$$

where μ is a number of desired overlapping percentages.

(iii) Avoidance of intervehicle collisions: An essential constraint to guarantee the safe execution of the search plan when UAVs collaborate as a team is to avoid any collision between them. The collision constraint is established to prevent potential collisions between the drones. We define $G_{col}(\Pi_m, \Pi_m^-)$, the total number of collisions in the solution, expressed as:

$$G_{col}(\Pi_m, \Pi_m^-) = \sum_{k=1}^{K} \sum_{m=1}^{M} \sum_{m'=m+1}^{M} (g_{col}(P_m(k), P_{m'}(k))), \tag{33}$$

where $g_{col}(P_m(k), P_{m'}(k))$ is a collision function that returns 1 when UAV_m and $\text{UAV}_{m'}$ collide at waypoint k and returns 0 otherwise. The solution (Π_m, Π_m^-) is considered feasible only when $G_{col}(\Pi_m, \Pi_m^-) = 0$, indicating that it avoids all potential collisions throughout the entire mission.

From (30), (32) and (33), the cooperative-UAV cost can be overall defined as

$$J_c(\Pi_m, \Pi_m^-) = \begin{cases} \beta_1 \sum_{k=1}^{K} [H_b - H_c(k)] + \beta_2 \sum_{k=1}^{K} H_o(k), \\ \qquad \text{if} \quad G_{col}(\Pi_m, \Pi_m^-) = 0, \\ \infty, \qquad \text{otherwise}, \end{cases} \tag{34}$$

where β_1 and β_2 are the coefficients of the total coverage and overlapping heights, respectively.

3.2. Nash game for cooperative path planning

In cooperative path planning for UAVs, the objective is to simultaneously minimize the cost function, denoted as $J_m(\Pi_m, \Pi_m^-)$, for all UAVs involved ($m = 1, 2, \dots, M$). This cost function includes numerous factors constraining UAV operations in their flight paths, including the cost for individual UAVs as defined in (22) and the cooperative coverage cost outlined in (34). During execution of inspection tasks, conflicts may emerge from the complex interdependencies within the path of an individual UAV, Π_m, as well as in relation to the paths of other UAVs within the team, Π_m^-. Consequently, there should be a balance for every UAV in the team to secure its overlapping coverage area, prevent collisions and optimize its flying route. Recognizing the effectiveness of game theory in resolving these conflicts and managing interactions [42], we develop a game-based cooperative guidance law consisting of two pivotal steps for multi-UAV path planning. Firstly, a Nash game is formulated to capture the strategic interactions among the UAVs. Then, the FWL-PSO algorithm is integrated to search for the Nash equilibrium, representing the optimal paths, wherein no UAV can unilaterally enhance its cost function without impacting the costs of the others. Further details of this method are presented in the following.

Let us consider each member of the UAV group as an individual decision-maker or player. These players formulate strategies, with the strategy for player UAV_m designated as its chosen path, denoted as Π_m. The effectiveness of their strategies is evaluated through a payoff mechanism, where the payoff for player UAV_m is quantified by its corresponding cost function, denoted as $J_m(\Pi_m, \Pi_m^-)$. Under the condition of shared information between the group from the communication links, each player is assumed to be aware of the strategies taken by their fellow players, and no player can enhance their payoff by unilaterally changing their plan alone due to the intrinsic symmetry in the players' roles [43]. From this stipulation, the Nash game can be formulated as,

$$G = (U, S, J), \tag{35}$$

where $U = (U_1, U_2, \dots, U_M)$ represents the set of players. The set $S = (S_1, S_2, \dots, S_M)$ stands for the strategy sets of players, where $S_m = (\Pi_{m_1}, \Pi_{m_2}, \dots, \Pi_{m_\Sigma})$ for $m = 1, 2, \dots, M$, represents all Σ_m strategies adopted by player U_m. The set $J = (J_1, J_2, \dots, J_M)$ represents the payoffs of all players. The Nash equilibrium is then defined as $\overset{*}{S} = (\overset{*}{\Pi}_1, \overset{*}{\Pi}_2, \dots, \overset{*}{\Pi}_M)$ satisfying the following condition:

$$\begin{aligned} &\forall \Pi_{m_\sigma} \in S_m,\ J_m(\overset{*}{\Pi}_m, \overset{*}{\Pi}_m^-) \le J_m(\Pi_{m_\sigma}, \overset{*}{\Pi}_m^-), \\ &\qquad m \in \{1, 2, \dots, M\},\ \sigma \in \{1, 2, \dots, \Sigma_m\}, \end{aligned} \tag{36}$$

where $\overset{*}{\Pi}_m^- = (\overset{*}{\Pi}_1, \dots, \overset{*}{\Pi}_{m-1}, \overset{*}{\Pi}_{m+1}, \dots, \overset{*}{\Pi}_M)$ represents the optimal strategy set of the rivals of player U_m, and for the UAV with player U_m, it is determined as:

$$\overset{*}{\Pi}_m = \underset{P_m}{\operatorname{argmin}}\, J_m(\Pi_m, \overset{*}{\Pi}_m^-). \tag{37}$$

3.3. FWL-PSO for Nash equilibrium

The problem of cooperative path planning for multiple UAVs is now rendered to the determination of the Nash equilibrium (37) for all $m = 1, 2, \dots, M$, as presented in the pseudocode of Algorithm 2. Initially, the current locations of the drones are taken as temporary opponent strategies, denoted as Π_m^-, for a player UAV_m of the group. Subsequently, the algorithm iteratively refines the strategy, Π_m, for each player by minimizing their respective cost function, J_m, over all strategies used by the opponents that are made available for the player. Once $\overset{*}{\Pi}_m(\Pi_m^-)$ is ascertained, opposing players, in turn, take their strategy by minimizing their cost function, denoted as J_m^-, while considering $\overset{*}{\Pi}_m(\Pi_m^-)$ and Π_m^-. The resulting strategy for an opposing player is then integrated into $\overset{*}{\Pi}_m(\Pi_m^-)$ to yield the optimal strategy for all neighboring players $\overset{*}{\Pi}_m(\overset{*}{\Pi}_m^-)$ at the optimal strategy of player U_m. This process is reiterated for all players participating in the game, yielding the Nash equilibrium strategies for all players as $\overset{*}{S} = (\overset{*}{\Pi}_1, \overset{*}{\Pi}_2, \dots, \overset{*}{\Pi}_M)$. This ensures that no player can enhance their outcome by independently altering their strategy, thus accomplishing the Nash equilibrium.

Algorithm 2 Nash equilibrium

for $m = 1 : M$ **do**

1. Fix Π_m^-;
2. $\overset{*}{\Pi}_m(\Pi_m^-) = \operatorname{argmin}_{\Pi_m} J_m(\Pi_m, \Pi_m^-)$;
3. Obtain $\overset{*}{\Pi}_m(\Pi_m^-)$;
4. $\overset{*}{\Pi}_m^- = \operatorname{argmin}_{\Pi_m^-} J_m^-(\overset{*}{\Pi}_m(\Pi_m^-), \Pi_m^-)$;
5. Obtain $\overset{*}{\Pi}_m^-$;
6. Substitute $\overset{*}{\Pi}_m^-$ into $\overset{*}{\Pi}_m(\Pi_m^-)$;
7. Obtain $\overset{*}{\Pi}_m(\overset{*}{\Pi}_m^-)$;

end for

8. Obtain $\overset{*}{S} = (\overset{*}{\Pi}_1, \overset{*}{\Pi}_2, \dots, \overset{*}{\Pi}_M)$;

It should be noted that in Algorithm 2, the optimization functions at steps 2 and 4 are tackled by employing the FWL-PSO method, described in Algorithm 1. After the application of FWL-PSO to solve the minimization of the cost functions for UAV path planning, the resulting flight paths of individual vehicles are obtained in the form of vectors that capture the movement of the UAV between waypoints. These vectors are represented within the inertial frame (xyz). For a UAV flight path denoted as i and comprising K nodes, the resulting representation encompasses $D = 3K$ dimensions, as delineated by the position vector $X_i = [x_{i1}, y_{i1}, z_{i1}, x_{i2}, y_{i2}, z_{i2}, \dots, x_{iK}, y_{iK}, z_{iK}]^\top$. In summary, these FWL-PSO and Nash game algorithms are merged into an overall algorithm for the proposed optimal framework of cooperative path planning for multiple UAVs with the pseudocode presented in Algorithm 3.

4. Simulation results

This section provides the simulation results obtained with the proposed algorithms, spanning from single UAV path planning to multi-UAV path planning scenarios.

4.1. Evaluation using digital elevation model (DEM) maps

The evaluation scenario used an authentic digital elevation model (DEM) map derived from LiDAR sensors, as detailed in [44]. We selected two areas of Christmas Island in Australia with different terrain structures as our areas of interest. We tailored them to create benchmarking scenarios, as depicted in Fig. 4. Within these scenarios, we positioned various threats, symbolized by red cylinders, at diverse altitudes.

To appraise the outperformance of the proposed path planning algorithm, extensive comparisons were conducted with other metaheuristic algorithms, including DE [45] and GA [46], as well as available PSO variants, including the canonical PSO [16], SPSO [22], LL-PSO [24], TPCSO [26], and ACVDEPSO [47]. All PSO variants utilized identical design parameters: $c_0 = 1$ with a damping rate of 0.98, $c_1 = 1.5$, and $c_2 = 1.5$. The swarm size was consistently set at $N = 500$ particles, and the maximum number of iterations was fixed at $T_m = 150$. The number of waypoints was equal to $K = 10$ for all cases. Each algorithm underwent a total of 50 runs. Their fitness average and

Algorithm 3 FWL-PSO and Nash game for UAV cooperative path planning

Begin:

1. Get search map and initial information of all UAVs in the group;
2. Set FWL-PSO parameters: c_0, c_1, c_2, T_m, N, p;

for $m = 1 : M$ **do**

3. Encode the flight path Π_m as a position vector X_m;
4. Generate random position X_m as (7) and velocity V_m as (8);

end for

for $m = 1 : M$ **do**

5. Calculate the cost value $J_{X_m} = J(X_m, X_m^-)$ as (9) and (21);
6. Obtain personal best X_{P_m} and J_{P_m} as (10);
7. Sort $\tilde{X}_{P_m}$ and select elite pool X_{Pm}^e as (11);
8. Update global best X_{G_m} and J_{G_m} as (12);
9. Calculate Fermat-Weber location F_m as (13);

end for

for $t = 1 : T_m$ **do**

for $m = 1 : M$ **do**

10. Update velocity $V_m(t+1)$ as (15) and (16);
11. Update position $X_m(t+1)$ as (17) and (18);
12. Fix the best position of the rival $X_{G_m}^-(t)$;
13. Calculate $J_{X_m}(t+1)$ (9) and (21);
14. Obtain personal best $X_{P_m}(t+1)$ and $J_{P_m}(t+1)$ as (19);
15. Re-sort $\tilde{X}_{P_m}(t+1)$ and update elite $X_{Pm}^e(t+1)$ as (11);
16. Update global best X_{G_m}, J_{G_m} as (12);
17. Update Fermat-Weber location $F_m(t+1)$ as (13);

end for

end for

18. Obtain the best strategy X_{G_m} and J_{G_m}, $m = 1, 2, ..., M$;
19. Obtain the Nash equilibrium: $X_G = (X_{G_1}, X_{G_2}, ..., X_{G_M})$.
20. Obtain the optimal path for each UAV$_m$, $m = 1, 2, ..., M$.

End

Table 1
FWL-PSO design parameters.

Parameter	Symbol	Value
Inertia weight	c_0	1
Inertia weight damping ratio	ω_{damp}	0.98
Acceleration coefficients	$[c_1,\ c_2]$	[1.5, 1.5]
Number of particles	N	500
Number of iterations	T_m	150
Number of running simulations	N_s	50
Number of waypoints	K	10
Planning cost weight factors	$[\omega_1, \omega_2, \omega_3, \omega_4]$	[10, 1, 1, 1]
Number of best-performing particles	p	25
Number of iterations to find FWL	N_f	25
FWL weight parameters	$\eta_i,\ i = 1, \ldots, p$	1

standard deviation values were computed for comparison. For FWL-PSO, we set $p = 0.05N = 25$ elite particles and $N_f = 25$ iterations. In this study, the weight parameter η_i was kept constant at 1 for all $i = 1, 2, \ldots, p$, signifying that all elite particles contributed equally in determining the Fermat-Weber location, exerting an equal influence on the search capabilities of the algorithm. The design parameters are listed in Table 1.

The top view of the paths generated by the various PSO variants is illustrated in Fig. 4, while the convergence of the best fitness values over iterations is shown in Fig. 5. All the algorithms are capable of producing paths that meet the requirements for distance traveled, obstacle avoidance, and smoothness. Notably, FWL-PSO exhibits a rapid convergence, achieving near-optimal solutions after only 35 iterations. In contrast, the other PSO variants require more iterations to converge to relatively good solutions. This outcome is further indicated in Table 2, listing the average and standard deviation of the fitness values obtained from running the algorithms for 50 repetitions. The proposed FWL-PSO delivers the best fitness values in terms of the mean and standard deviation in all cases, confirming its advantages over other techniques.

Table 2
Comparison of fitness values (all units in meters).

Algorithm	Scenario 1		Scenario 2		Scenario 3		Scenario 4	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
DE	5.174	0.072	5.256	0.059	5.243	0.020	5.322	0.023
GA	5.086	0.046	5.347	0.016	5.179	0.078	5.473	0.171
PSO	4.970	0.019	5.081	0.048	5.063	0.005	5.169	0.057
SPSO	4.960	0.013	4.987	0.020	5.037	0.007	5.111	0.008
LL-PSO	4.968	0.012	4.976	0.027	5.079	0.034	5.201	0.044
TPCSO	4.967	0.010	4.968	0.022	5.060	0.006	5.149	0.037
ACVDEPSO	4.946	0.012	4.931	0.015	5.045	0.005	5.098	0.006
FWLPSO	**4.937**	**0.002**	**4.901**	**0.002**	**5.033**	**0.001**	**5.088**	**0.005**

4.2. Building inspection using multiple UAVs

To assess the effectiveness of the proposed game-based FWL-PSO framework for UAV cooperative path planning, we consider the inspection of a building façade. It spans 100 m in length and 30 m in height. The operational space surrounding the building was defined with dimensions of 30 m × 100 m × 35 m, as depicted in Fig. 6. Note that the façade chosen here for simulation was after a real-world asset but there should be no specific restrictions on the building size given the available number of UAVs used for inspection. For obstacles in the environment, we modeled them as cylinders located at coordinates $(10, 20)$, $(20, 40)$, $(10, 60)$, and $(20, 80)$. These cylinders have heights of 10 m, 20 m, 20 m, and 10 m, respectively, and all share a common radius of 3 m. The initial positions of the UAV team were set at $[P_1(0),\ P_2(0),\ P_3(0)] = [10\ 10\ 10; 1\ 1\ 1; 5\ 15\ 25]$. The target locations were placed at $[P_1(end), P_2(end), P_3(end)] = [10\ 10\ 10; 99\ 99\ 99; 5\ 15\ 25]$.

The FWL-PSO algorithm was configured with parameters listed in Table 1. Here, for the individual cost function for each UAV, weight coefficients were selected as $[\omega_1,\ \omega_2,\ \omega_3,\ \omega_4] = [1,\ 1,\ 0.1,\ 0.1]$ to prioritize minimizing distance and avoiding obstacles while still considering altitude and smoothness. To account for complete coverage, the desired overlapping percentage set at $\mu = 15\%$ and $\beta_1 = \beta_2 = 1$. Each path was defined with a total of $K = 9$ waypoints, excluding the initial and target positions.

The simulation results shown in Fig. 6 illustrate the effectiveness of the integrated FWL-PSO with game theory for multi-UAV path planning in this inspection task. Notably, the vertical FOV of three UAVs adequately covers the entire height of the building throughout the flight, ensuring comprehensive monitoring and minimizing the need for additional travel. This optimized path planning not only minimizes the distance traveled but also successfully avoids both intervehicle and obstacle collisions, highlighting the capability of the system to ensure safe and cooperative multi-UAV missions. Moreover, the Nash equilibrium, a critical aspect of the planning coordination for a multi-UAV system, is reached after 35 iterations. This is illustrated in Fig. 7, which showcases the convergence of cost values over iterations. The swift and stable convergence is attributed to the FWL-PSO merits to

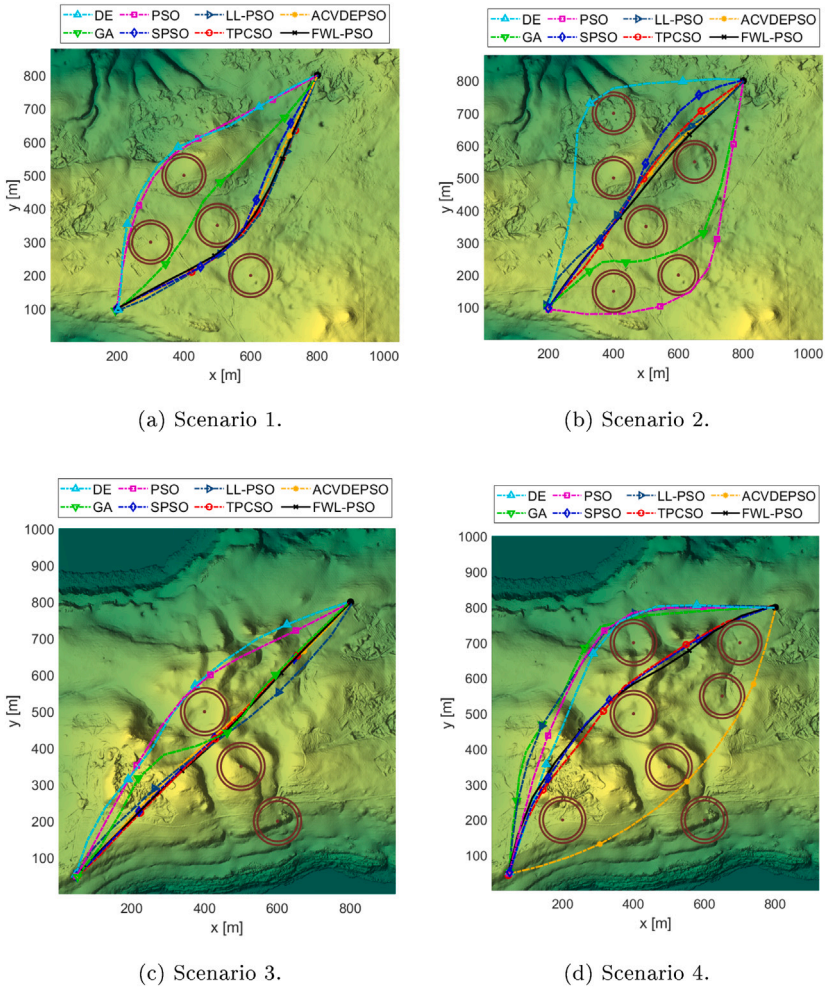


Fig. 4. Top view of generated paths.

obtain optimal solutions in the proposed framework. To further evaluate the performance of the approach for each UAV, Fig. 8 presents histograms of the final cost values derived from 50 simulation trials. They statistically reaffirm the effectiveness of the proposed approach for UAV path planning in civil infrastructure inspection.

5. Experimental validation

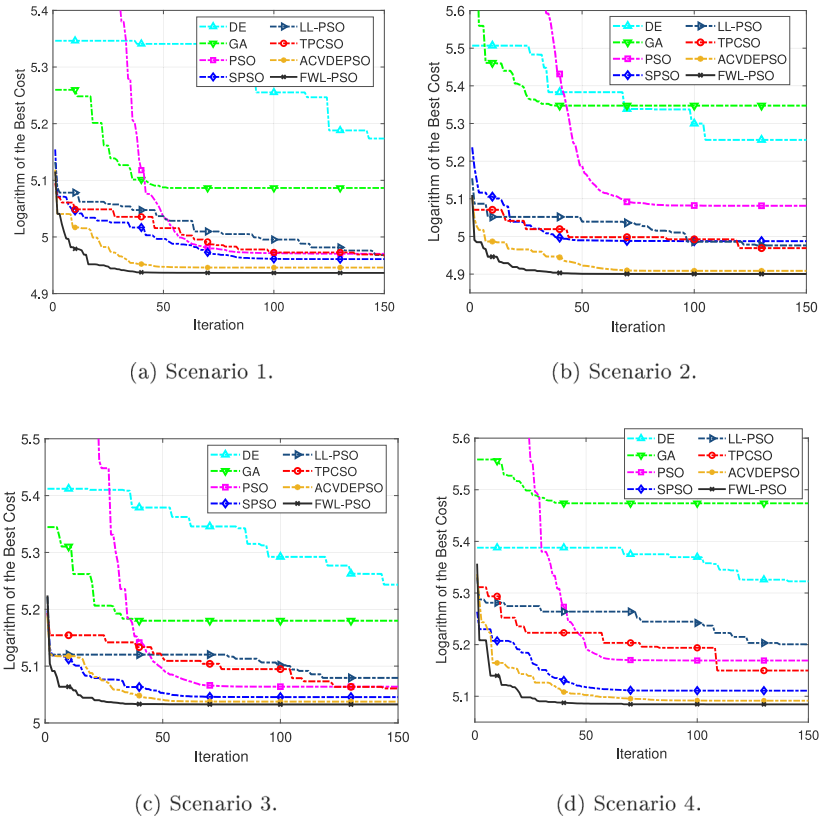
To validate the proposed FWL-PSO algorithm for cooperative UAV path planning in real-world conditions, field tests were conducted to thoroughly inspect a 30-meter-high building situated in Wentworth Park, New South Wales, Australia. The experimental work aims to demonstrate not only the practicality of the algorithm but also its ability to optimize inspection routes for low-cost UAVs.

5.1. Experimental setup

To establish a reference point within our Cartesian coordinate system, we selected the origin at coordinates −33.87658617°S, 151.19252566°E. For our inspection tasks, we employed three 3DR Solo drones equipped with GoPro Hero 4 cameras. These drones are equipped with dual Cortex M4 168 MHz processors, ensuring precise control. Additionally, they were supplied with an ARM Cortex A9 running the Arducopter flight operating system [30]. Autonomous flight planning and data analysis were conducted using the Mission Planner ground control station. The compact form factor and 4K resolution capabilities of the GoPro Hero 4 camera make it particularly well-suited for capturing detailed images in hard-to-reach locations during field tests. The initial positions of the drones concerning the origin were set as $P_1(0) = [10\ 1\ 5]$, $P_2(0) = [10\ 1\ 15]$, and $P_3(0) = [10\ 1\ 25]$. The corresponding target positions were chosen as $P_1(end) = [10\ 80\ 5]$, $P_2(end) = [10\ 80\ 15]$, and $P_3(end) = [10\ 80\ 25]$. We also determined the locations, heights, and radii of obstacles in the flying field, including nearby trees, light poles, and a public lavatory.

5.2. Field test results

By utilizing the FWL-PSO algorithm and Nash game framework for cooperative path planning, we obtained optimal waypoints for all UAVs. These coordinates were then converted into longitude, latitude, and altitude to execute our generated paths. The speed profiles of the UAVs were set with a reference speed of 1 m/s. Subsequently, we uploaded the waypoints and corresponding speeds to each drone through Mission Planner for autonomous execution. Fig. 9(a) shows the imported trajectory typically for UAV1.



(a) Scenario 1. (b) Scenario 2.

(c) Scenario 3. (d) Scenario 4.

Fig. 5. Best fitness values over iterations.

The software facilitated assigning waypoints, flight path planning, data logging, as well as streamlining the entire workflow. To illustrate the experiments, Fig. 9(b) depicts the three UAVs during the cooperative mission of inspecting the building façade. These UAVs adeptly navigated around obstacles while ensuring complete coverage of the entire building surface within their field of view by accurately tracking the paths generated by the proposed approach. Indeed, Fig. 10 presents the tracking performance in terms of the distance from the actual position to the desired generated path observed during the inspection with minor tracking errors less than 0.6 m. These slight errors mainly resulted from minor GPS positioning inaccuracies rather than any limitations in the tracking controller of the drone.

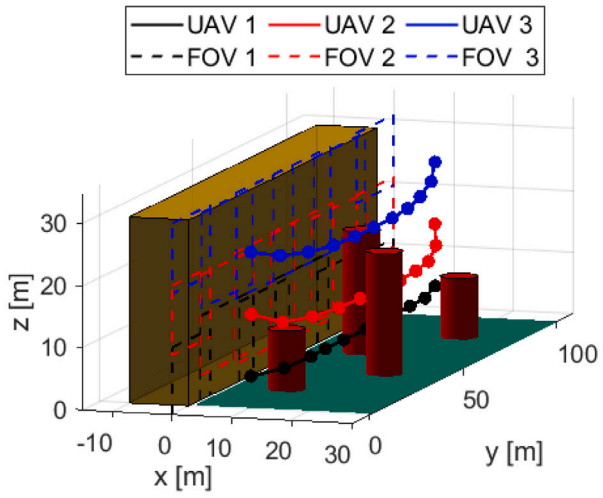
5.3. Discussion

Our experimental results have been consistent with the simulation results and, more importantly, demonstrated that a team of drones could successfully follow optimal paths generated from the proposed cooperative path planning framework, integrating the Fermat-Weber location with particle swarm optimization and the Nash game theory. Insightful results obtained underscore the effectiveness and practicality of the proposed approach. As demonstrated in the simulation and field tests for inspection of a building façade, the ability to consistently generate and execute safe paths resulting from the optimized cooperative planning can enhance the operational performance of multiple UAVs in civil infrastructure inspection as well as across various scenarios, for example, search and rescue, surveillance, maintenance and transportation.

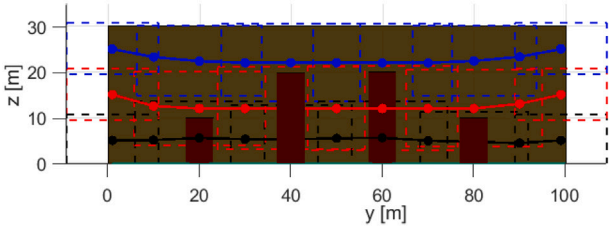
Here, from extensive comparison as indicated in Table 2 and validated by experiments, the determination of Fermat-Weber locations have significantly improve convergence and computational efficiency of the metaheuristic optimization. Moreover, by integrating the game theory principles, our algorithm facilitates cooperative decision-making among UAVs. This enables collaborative path optimization while considering potential conflicts and interactions. Such an approach improves path planning precision and adaptability, empowering UAVs to safely navigate through complex environments and obstacles. Furthermore, drawing upon the game theory in path planning is promising for analyzing strategic interactions between UAVs, ultimately ensuring the reliability of their cooperation in execution of more complicated tasks.

6. Conclusion

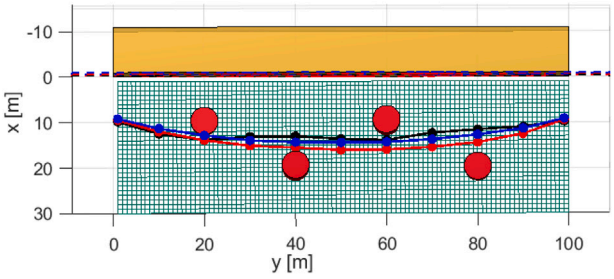
This paper has presented the development of a novel FWL-PSO algorithm for cooperative path planning of UAVs, utilizing the advantages of Fermat-Weber location principles to enhance computational efficiency and optimization convergence. The integration of FWL-PSO with game theory for multi-UAV cooperative path planning, leading to the swift attainment of Nash equilibria, demonstrates the ability to reach the optimal solutions in handling UAV interactions from group operation as well as the practicality of the proposed approach for cooperative path planning. Comparative evaluations have established the superiority of FWL-PSO performance when compared with the available PSO variants. The proposed approach significantly contributes to advancing multi-UAV path planning and demonstrates the potential of merging heuristic optimization methods with game theory to address more complicated cooperative scenarios with application to real-world tasks. Future works will focus on exploring UAV-based cooperative path planning in other scenarios in civil infrastructure automation and extending using the proposed framework to deal with a dynamic environment moving obstacles.



(a) 3D view.



(b) Front view.



(c) Top view.

Fig. 6. UAV paths and FOVs.

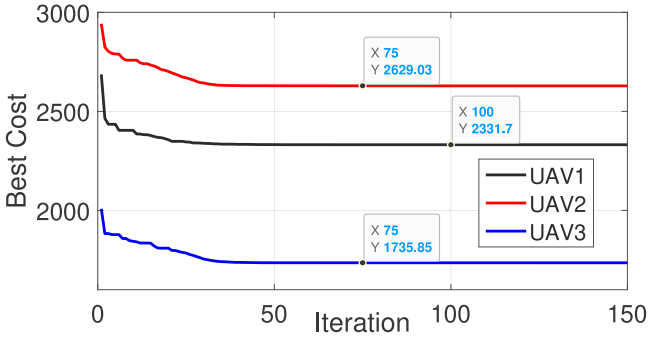


Fig. 7. Best fitness values over iterations.

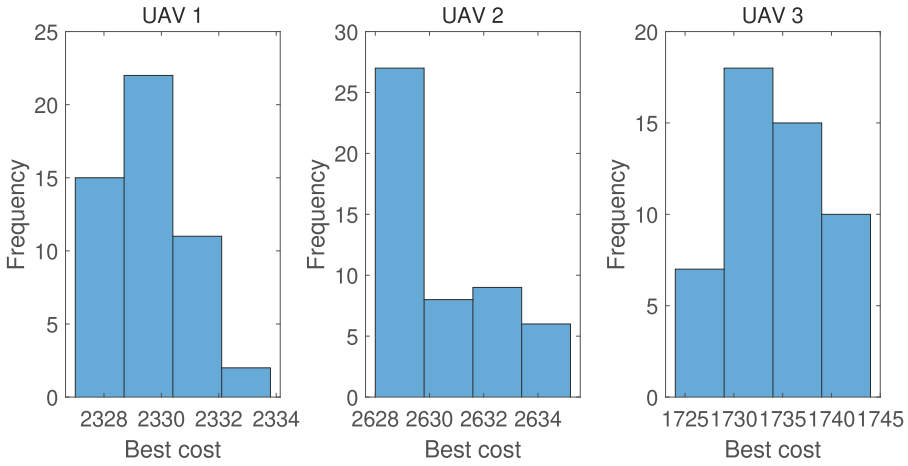
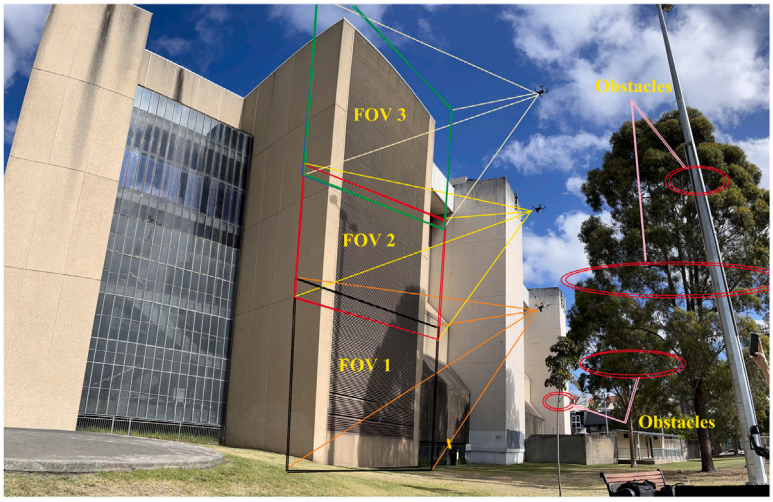


Fig. 8. Histogram of best fitness values.

(a) Imported trajectory for UAV 1.



(b) Cooperative UAVs during experiments.

Fig. 9. Experimental results.

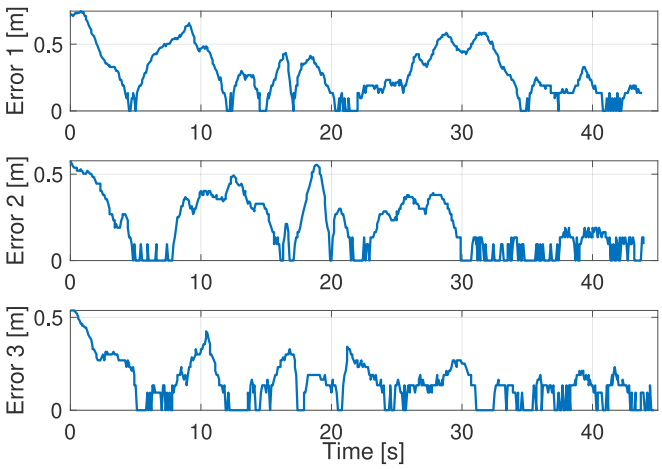


Fig. 10. Tracking performance.

CRediT authorship contribution statement

Lanh Van Nguyen: Writing – original draft, Software, Resources, Methodology, Formal analysis, Conceptualization. **Ngai Ming Kwok:** Writing – review & editing, Formal analysis. **Quang Phuc Ha:** Writing – review & editing, Supervision, Project administration, Methodology, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

The first author would like to thank the Vingroup Science and Technology Scholarship Program, managed by VinUniversity and sponsored by Vingroup, for Overseas Study for Master and Doctoral degrees.

References

[1] D.H. Stolfi, M.R. Brust, G. Danoy, P. Bouvry, A competitive predator–prey approach to enhance surveillance by UAV swarms, Appl. Soft Comput. 111 (2021) 107701, http://dx.doi.org/10.1016/j.asoc.2021.107701.

[2] Y. Xu, Z. Sun, X. Xue, W. Gu, B. Peng, A hybrid algorithm based on MOSFLA and GA for multi-UAVs plant protection task assignment and sequencing optimization, Appl. Soft Comput. 96 (2020) 106623, http://dx.doi.org/10.1016/j.asoc.2020.106623.

[3] R.W.L. Coutinho, A. Boukerche, UAV-mounted cloudlet systems for emergency response in industrial areas, IEEE Trans. Ind. Inform. 18 (11) (2022) 8007–8016, http://dx.doi.org/10.1109/TII.2022.3174113.

[4] K. Liu, B.M. Chen, Industrial UAV-based unsupervised domain adaptive crack recognitions: From database towards real-site infrastructural inspections, IEEE Trans. Ind. Electron. 70 (9) (2023) 9410–9420, http://dx.doi.org/10.1109/TIE.2022.3204953.

[5] G. Tang, C. Tang, C. Claramunt, X. Hu, P. Zhou, Geometric A-star algorithm: An improved A-star algorithm for AGV path planning in a port environment, IEEE Access 9 (2021) 59196–59210, http://dx.doi.org/10.1109/ACCESS.2021.3070054.

[6] N.L. Prasad, B. Ramkumar, 3-D deployment and trajectory planning for relay based UAV assisted cooperative communication for emergency scenarios using Dijkstra's algorithm, IEEE Trans. Veh. Technol. 72 (4) (2023) 5049–5063, http://dx.doi.org/10.1109/TVT.2022.3224304.

[7] P. Pharpatara, B. Hérissé, Y. Bestaoui, 3-D trajectory planning of aerial vehicles using RRT*, IEEE Trans. Control Syst. Technol. 25 (3) (2017) 1116–1123, http://dx.doi.org/10.1109/TCST.2016.2582144.

[8] Y. Lin, S. Saripalli, Sampling-based path planning for UAV collision avoidance, IEEE Trans. Intell. Transp. Syst. 18 (11) (2017) 3179–3192, http://dx.doi.org/10.1109/TITS.2017.2673778.

[9] L. Blasi, E. D'Amato, M. Mattei, I. Notaro, UAV path planning in 3-D constrained environments based on layered essential visibility graphs, IEEE Trans. Aerosp. Electron. Syst. 59 (3) (2023) 2359–2375, http://dx.doi.org/10.1109/TAES.2022.3213230.

[10] Z. Pan, C. Zhang, Y. Xia, H. Xiong, X. Shao, An improved artificial potential field method for path planning and formation control of the multi-UAV systems, IEEE Trans. Circuits Syst. II 69 (3) (2022) 1129–1133, http://dx.doi.org/10.1109/TCSII.2021.3112787.

[11] P.-C. Song, J.-S. Pan, S.-C. Chu, A parallel compact cuckoo search algorithm for three-dimensional path planning, Appl. Soft Comput. 94 (2020) 106443, http://dx.doi.org/10.1016/j.asoc.2020.106443.

[12] V. Roberge, M. Tarbouchi, G. Labonté, Fast genetic algorithm path planner for fixed-wing military UAV using GPU, IEEE Trans. Aerosp. Electron. Syst. 54 (5) (2018) 2105–2117, http://dx.doi.org/10.1109/TAES.2018.2807558.

[13] Y. Fu, M. Ding, C. Zhou, H. Hu, Route planning for unmanned aerial vehicle (UAV) on the sea using hybrid differential evolution and quantum-behaved particle swarm optimization, IEEE Trans. Syst. Man Cybern.: Syst. 43 (6) (2013) 1451–1465, http://dx.doi.org/10.1109/TSMC.2013.2248146.

[14] X. Yu, W.-N. Chen, T. Gu, H. Yuan, H. Zhang, J. Zhang, ACO-A*: Ant colony optimization plus A* for 3-D traveling in environments with dense obstacles, IEEE Trans. Evol. Comput. 23 (4) (2018) 617–631, http://dx.doi.org/10.1109/TEVC.2018.2878221.

[15] R.C. Eberhart, Y. Shi, J. Kennedy, Swarm Intelligence (Morgan Kaufmann Series in Evolutionary Computation), Morgan Kaufmann Publishers, 2001.

[16] R.C. Eberhart, Y. Shi, Comparison between genetic algorithms and particle swarm optimization, in: International Conference on Evolutionary Programming, 1998, pp. 611–616, http://dx.doi.org/10.1007/BFb0040812.

[17] M.D. Phung, C.H. Quach, T.H. Dinh, Q. Ha, Enhanced discrete particle swarm optimization path planning for UAV vision-based surface inspection, Autom. Constr. 81 (2017) 25–33, http://dx.doi.org/10.1016/j.autcon.2017.04.013.

[18] W. Liu, Z. Wang, Y. Yuan, N. Zeng, K. Hone, X. Liu, A novel sigmoid-function-based adaptive weighted particle swarm optimizer, IEEE Trans. Cybern. 51 (2) (2019) 1085–1093, http://dx.doi.org/10.1109/TCYB.2019.2925015.

[19] R. Vafashoar, H. Morshedlou, M.R. Meybodi, Bifurcated particle swarm optimizer with topology learning particles, Appl. Soft Comput. 114 (2022) 108039, http://dx.doi.org/10.1016/j.asoc.2021.108039.

[20] X. Guo, M. Ji, Z. Zhao, D. Wen, W. Zhang, Global path planning and multi-objective path control for unmanned surface vehicle based on modified particle swarm optimization (PSO) algorithm, Ocean Eng. 216 (2020) 107693, http://dx.doi.org/10.1016/j.oceaneng.2020.107693.

[21] B. Liang, Y. Zhao, Y. Li, A hybrid particle swarm optimization with crisscross learning strategy, Eng. Appl. Artif. Intell. 105 (2021) 104418, http://dx.doi.org/10.1016/j.engappai.2021.104418.

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

[23] G. Liu, X. Chen, R. Zhou, S. Xu, Y.-C. Chen, G. Chen, Social learning discrete particle swarm optimization based two-stage X-routing for IC design under intelligent edge computing architecture, Appl. Soft Comput. 104 (2021) 107215, http://dx.doi.org/10.1016/j.asoc.2021.107215.

[24] Q. Yang, W.-N. Chen, J.D. Deng, Y. Li, T. Gu, J. Zhang, A level-based learning swarm optimizer for large-scale optimization, IEEE Trans. Evol. Comput. 22 (4) (2018) 578–594, http://dx.doi.org/10.1109/TEVC.2017.2743016.

[25] R. Lan, Y. Zhu, H. Lu, Z. Liu, X. Luo, A two-phase learning-based swarm optimizer for large-scale optimization, IEEE Trans. Cybern. 51 (12) (2020) 6284–6293, http://dx.doi.org/10.1109/TCYB.2020.2968400.

[26] C. Huang, X. Zhou, X. Ran, Y. Liu, W. Deng, W. Deng, Co-evolutionary competitive swarm optimizer with three-phase for large-scale complex optimization problem, Inform. Sci. 619 (2023) 2–18, http://dx.doi.org/10.1016/j.ins.2022.11.019.

[27] A. Comăneci, M. Joswig, Tropical medians by transportation, Math. Program. (2023) 1–27, http://dx.doi.org/10.1007/s10107-023-01996-8.

[28] C. Liu, Y. Guo, N. Li, X. Song, AoI-Minimal task assignment and trajectory optimization in multi-UAV-assisted IoT networks, IEEE Internet Things J. 9 (21) (2022) 21777–21791, http://dx.doi.org/10.1109/JIOT.2022.3182160.

[29] A. Bemporad, C. Rocchi, Decentralized linear time-varying model predictive control of a formation of unmanned aerial vehicles, in: 2011 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 7488–7493, http://dx.doi.org/10.1109/CDC.2011.6160521.

[30] V.T. Hoang, M.D. Phung, T.H. Dinh, Q.P. Ha, System architecture for real-time surface inspection using multiple UAVs, IEEE Syst. J. 14 (2) (2020) 2925–2936, http://dx.doi.org/10.1109/JSYST.2019.2922290.

[31] J. Rao, C. Xiang, J. Xi, J. Chen, J. Lei, W. Giernacki, M. Liu, Path planning for dual UAVs cooperative suspension transport based on artificial potential field-A* algorithm, Knowl.-Based Syst. 277 (2023) 110797, http://dx.doi.org/10.1016/j.knosys.2023.110797.

[32] H.S. Yahia, A.S. Mohammed, Path planning optimization in unmanned aerial vehicles using meta-heuristic algorithms: A systematic review, Environ. Monit. Assess. 195 (1) (2023) 30, http://dx.doi.org/10.1007/s10661-022-10590-y.

[33] R.B. Myerson, Game Theory: Analysis of Conflict, Harvard Press, 1997.

[34] S. Goudarzi, M.H. Anisi, D. Ciuonzo, S.A. Soleymani, A. Pescapé, Employing unmanned aerial vehicles for improving handoff using cooperative game theory, IEEE Trans. Aerosp. Electron. Syst. 57 (2) (2020) 776–794, http://dx.doi.org/10.1109/TAES.2020.3021106.

[35] I.A. Nemer, T.R. Sheltami, A.S. Mahmoud, A game theoretic approach of deployment a multiple UAVs for optimal coverage, Transp. Res. A 140 (2020) 215–230, http://dx.doi.org/10.1016/j.tra.2020.08.004.

[36] A. Beck, S. Sabach, Weiszfeld's method: Old and new results, J. Optim. Theory Appl. 164 (2014) 1–40, http://dx.doi.org/10.1007/s10957-014-0586-7.

[37] Y. Vardi, C.-H. Zhang, The multivariate L1-median and associated data depth, Proc. Natl. Acad. Sci. USA 97 (4) (2000) 1423–1426, http://dx.doi.org/10.1073/pnas.97.4.1423.

[38] Z.-H. Zhan, J. Zhang, Y. Lin, J.-Y. Li, T. Huang, X.-Q. Guo, F.-F. Wei, S. Kwong, X.-Y. Zhang, R. You, Matrix-based evolutionary computation, IEEE Trans. Emerg. Top. Comput. Intell. 6 (2) (2022) 315–328, http://dx.doi.org/10.1109/TETCI.2020.3047410.

[39] L.V. Nguyen, I. Torres Herrera, T.H. Le, D.M. Phung, R.P. Aguilera, Q.P. Ha, Stag hunt game-based approach for cooperative UAVs, in: Proceedings of the 39th International Symposium on Automation and Robotics in Construction, IAARC Publications, 2022, pp. 367–374, http://dx.doi.org/10.22260/ISARC2022/0051.

[40] L. Van Nguyen, M.D. Phung, Q.P. Ha, Game theory-based optimal cooperative path planning for multiple UAVs, IEEE Access 10 (2022) 108034–108045, http://dx.doi.org/10.1109/ACCESS.2022.3213035.

[41] L.V. Nguyen, T.H. Le, I. Torres Herrera, N.M. Kwok, Q.P. Ha, Intelligent path planning for civil infrastructure inspection with multi-rotor aerial vehicles, Constr. Robot. 8 (2) (2024) 1–22, http://dx.doi.org/10.1007/s41693-024-00135-9.

[42] A. Ji, X. Xue, Q. Ha, X. Luo, M. Zhang, Game theory–based bilevel model for multiplayer pavement maintenance management, Autom. Constr. 129 (2021) 103763, http://dx.doi.org/10.1016/j.autcon.2021.103763.

[43] J. Liu, P. Yi, Predefined-time distributed Nash equilibrium seeking for noncooperative games with event-triggered communication, IEEE Trans. Circuits Syst. II 70 (9) (2023) 3434–3438, http://dx.doi.org/10.1109/TCSII.2023.3259483.

[44] Australia Geoscience 2015, Digital Elevation Model (DEM) of Australia Derived from LiDAR 5 metre Grid, Geoscience Australia, Canberra, 2015, http://dx.doi.org/10.26186/89644.

[45] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Glob. Optim. 11 (1997) 341–359, http://dx.doi.org/10.1023/A:1008202821328.

[46] V. Roberge, M. Tarbouchi, G. Labonté, Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning, IEEE Trans. Ind. Inform. 9 (1) (2012) 132–141, http://dx.doi.org/10.1109/TII.2012.2198665.

[47] C. Huang, X. Zhou, X. Ran, J. Wang, H. Chen, W. Deng, Adaptive cylinder vector particle swarm optimization with differential evolution for UAV path planning, Eng. Appl. Artif. Intell. 121 (2023) 105942, http://dx.doi.org/10.1016/j.engappai.2023.105942.