UAVs Path Planning by Particle Swarm Optimization based on Visual-SLAM Algorithm

Umair Ahmad Mughal, Ishtiaq Ahmad, and KyungHi Chang, *Senior Member, IEEE*

Abstract—Unmanned Aerial Vehicles (UAVs) have gained tremendous popularity due to its high mobility in various robotics platforms. We use a state-of-the-art visual simultaneous localization and mapping (VSLAM) method to trace the UAV poses while simultaneously constructing an incremental and progressive map of the surrounding environment. In this regard, a single UAV is first used to compose a map of the region of interest by utilizing the monocular vision-based approach. The constructed map is processed as an input mean for the optimization algorithm to plan multiple UAVs' optimum paths. We design a path planner based on particle swarm optimization (PSO) and propose a path updating mechanism based on region sensitivity (RS) to avoid sensitive areas if any hazardous events are detected while executing the final path. Moreover, we propose a dynamic fitness function (DFF) to evaluate the path planner's planning strategy considering various optimization parameters such as flight risk estimation, energy consumption, and operation completion time. The proposed planner attains the high fitness value and reaches the destination safely following the shortest path while avoiding all the unexpected hazardous events and restricted areas, which validate the effectiveness of our proposed PSO-VSLAM system as depicted by the simulation results.

Index Terms—Visual-SLAM, PSO, Path Planning, Autonomous Aerial Vehicles, UAV

1 Introduction

An autonomous aerial vehicle's ability to navigate in an unrecognized environment while simultaneously constructing a progressive map and localizing itself is a significant field of study in robotics. Research has been conducted on simultaneous localization and mapping (SLAM) due to its real-life practical applications [1]. The advancements in vision-based SLAM techniques estimate the robot's state and build a map as it moves around the terrain [2]. Many SLAM systems in the literature use multiple heterogeneous sensors such as Laser Range Finders (LRF), an inertial measurement unit (IMU), a GNSS receiver, a magnetometer, an optical flow sensor (OFS), a barometer, and a Light Detection and Ranging (LiDAR) [3], [4]. However, in recent years, single-camera SLAM systems have become very popular due to its light weight, low cost, and variety of applications in complex environment [5] [6]. In this respect, monocular visual-SLAM has been acknowledged vastly for UAVs' applications in enabling the fully autonomous system without the support of external positioning systems in various complex situations.

In particular, UAVs play a vital role in various civil applications due to their high-mobility, low maintenance cost, and ease of deployments. UAVs are widely being used for traffic monitoring, health services, search and rescue, security and surveillance, and delivery of goods [7]–[9]. Similarly, UAVs are also being taken as an enabler in the wireless communication network for public safety. While operating as a base station, UAVs provide excellent connectivity to the terrestrial base station to boost the wireless network's coverage, capacity, and efficiency [10]. Since all applications of the UAVs are dependent on path planning, therefore, it should be very accurate to fulfill the requirements of the application, as mentioned earlier. Consequently, it is inevitable.

Path planning algorithms are designed to achieve the optimal trajectory within a particular set of constraints (terrain constraints, collision avoidance, etc.) and objectives (energy consumption, flight risk, etc). Generally, UAVs fly through complex terrain. Hence, path planning is not only concerned with constraints and multiple objectives but also includes dealing with hazardous events that can happen suddenly on the path of the UAV. Therefore, we introduce region sensitivity (RS) to counter unconditional threats, enabling the UAV to sense an unstable region and optimize its path to reach the destination. This paper introduces a framework for optimal path planning based on maps created by the monocular-vision approach. A state-of-the-art visual-SLAM (VSLAM) algorithm tracks the camera's pose while simultaneously building an incremental map of the surrounding environment. The generated map is processed and serves as input to an optimization algorithm.

Particle swarm optimization (PSO) has been regarded as an efficient means of planning an optimum path. In the PSO framework, the particle is viewed as an integrated individual, representing a candidate solution. Thus, the global best particle is determined by the performance of all the particles. So, the best one selected from all the particles may not indicate that each particle's dimension is also the best. For example, with a particle located in D-dimensional space, most of the D dimensions have an excellent effect on

- *U. A. Mughal is with the Department of Electrical and Computer Engineering, Inha University, Incheon, South Korea, 22212.*
  E-mail: umairm9552@gmail.com
- *I. Ahmad is with the Department of Electrical and Computer Engineering, Inha University, Incheon, South Korea, 22212.*
  E-mail: ishtiaq001@gmail.com
- *K. H. Chang is with the Department of Electrical and Computer Engineering, Inha University, Incheon, South Korea, 22212.*
  E-mail: khchang@inha.ac.kr
  (Corresponding author: K. H. Chang)

the optimal problem. Still, others possibly perform with a low influence on the result. Suppose that a specific particle considered as a whole is selected as the best particle to participate in the iterative process. In that case, the standard PSO algorithm's search efficiency and solution accuracy will be restricted and decreased. When a particle represents a candidate path to solve the path planning problem, the PSO planner evaluates the candidate path by considering the whole path's quality instead of a single waypoint.

Compared to other metaheuristic search algorithms PSO framework has more simplicity in implementation and has less computational time. Moreover. it has more ability to solve nonlinear problems as compared to other heuristic algorithms like ant colony optimization (ACO), Genetic algorithm (GA), an evolutionary algorithm (EA). The GA is having an expensive computational cost because it is inherently discrete, i.e., it encodes to design the discrete variables, while the PSO is inherently continuous and can be modified easily to handle the discrete design variables. Therefore, we adopt PSO as it converges quickly in a dynamic environment.

1.1 Main Contributions

This paper aims to develop a system that derives the optimal paths for multiple UAVs to reach their destinations, even in a GPS-denied environment safely. We designed a two-step centralized system based on visual-SLAM to construct an incremental and progressive map of the surrounding environment. The constructed terrain map in the form of a points cloud is used as an input mean to the proposed multiple-path UAVs optimization planner. We implement the Canny and Harris detector simultaneously to stabilize the system in the least textured environment.

We proposed a dynamic fitness function (DFF), which contains multiple optimization indexes, such as flight risk estimation, energy consumption, operation completion time, and numerous constraints i.e. UAV constraints, which take care about the physical limitations of the UAVs, and environmental constraints to care about surrounding conditions, as a joint cost determinant, to plan the optimum path for multiple UAVs to reach the destination safely. We have also proposed a path-updating mechanism based on the RS to counter the unexpected hazardous events, enabling the UAV to sense an unstable region and optimize its path accordingly.

The proposed optimization planner utilizes the PSO to compute each path's fitness based on the RS and DFF. Considering all the aspects, make our proposed approach more practical in path planning for multiple UAVs.

1.2 Related Work

Research has been conducted in underwater, indoor, and outdoor environments using SLAM and PSO technology. In [11], the authors utilize active SLAM for robot path planning based on the deep reinforcement learning. The obstacle in the path are detected using the convolutional residual network. The proposed scheme uses Dueling DQN algorithm for obstacle avoidance, and at the same time, building the 2D map of the surrounding using the FastSLAM. Similarly, in [12] authors also use stereo vision-based active SLAM to localize, navigate, and define its surrounding environment. The cognitive-based adaptive optimization algorithm introduces to avoid the obstacles and complete the task effectively. In [11], [12] the main focus of the approaches is to the complete robot task while detecting and avoiding the obstacle in the environment.

The author in [13] proposes a visual-SLAM approach to construct the incremental map of the terrain for surveillance. The author introduces the Cognitive-based Adaptive Optimization (CAO) algorithm for path planning. The multi-robot system designed in [13] concentrates only on exploring the surrounding environment, and the robot team's arrangement is an optimization objective. The author did not consider the terrain limitation and other aspects of the environment. In contrast, we consider the environmental and physical limitations of the UAV as optimization parameters.

The monocular-inertial SLAM for path planning has been proposed in [14]. The author proposes the path planning algorithm, which works in the SLAM estimation loop for a vision-inertial system. The designed keyframe-SLAM system fuses the monocular camera's sensing cues and cues taken from the Inertial Measurement Unit (IMU). The authors only inspect the feasibility of their proposed work irrespective of considering the environmental and UAV physical challenges.

In [15], authors introduce PSO algorithm for a network of UAVs for the exploration task in a disaster scenario. The modified version of the PSO called dynamic-PSO for UAVs network (dPSO-U) propose to discover the disaster areas. Particularly, the UAVs follow the delay tolerant (DTN) networking approach for sharing the information. The solution works only to evaluate the best value combinations of the UAVs for effectively exploring the scene.

In [16], a team of UAVs is used to enhance the communication services and provide the best coverage followed by the 5G network. The UAVs act as bridge between the cellular base station and the users. The main focus of the designed approach is to locate the UAVs at the optimum position to maximize the communication coverage ratio. The authors propose per-Drone Iterated PSO (DI-PSO) scheme based on PSO algorithm, in which PSO is utilized to calculate the optimum position for each drone. However, the PSO design in approach individually evaluates the optimum position for each UAV, while in our approach, a UAV behaves as a PSO particle.

In [17], a team of UAVs use to perform the fire fighting task in a forest. The target spots are considered to be known before the mission start. The UAVs were assigned to the different fire spots using the auction-based algorithm. Afterwards, UAVs use the centralized PSO algorithm along with parametrization and time discretization (CPTD) algorithm to calculate the optimal trajectories towards the assigned fire spots. However, our algorithm is not centralized but distributed. Each UAV works as an individual body of the UAV network, providing the UAV facility to set its own rules to carry out the task. If one of the UAVs face a failure, the mission goals can still be performed.

In [18], an improved PSO algorithm is introduced for the real-time path planning of a single UAV. The work falls in the category of the low-level trajectory planning as it include to avoid the moving obstacle. There are mainly two improvements: (i) an adaptive strategy, which adjusts the inheritance parameters of the PSO algorithm (inertia, local and global best) to magnify the searching capability of the PSO algorithm; (ii) a chaos optimization mechanism, which increases the ability of the PSO algorithm to escape from the local maxima. However, the approach considers only a single UAV, and PSO implementation is used only to evaluate the trajectory following parameter of the PSO. This means the approach does not work for the multiple UAVs trajectories we propose in our algorithm.

Most relevant to this paper is the proposed Next-Best-View Planner (NBVP) [19]. It uses the RRT approach inside the planning loop. The visual data retrieve from the depth sensor using a tree node. While planning, the small segment of the best view is executed in each iteration, which allows adapting the trajectory towards the plan between the iterations as a new explored map. However, obtaining the best view from the RRT is a costly step, and it merely does proper scaling with the tree size.

Comparing with the work mentioned earlier, some authors use only SLAM technology for path planning [12] while some use only PSO algorithm for this cause [13-16]. Some use SLAM technology to combine optimization algorithms and other path planning mechanisms [9-11]. However, no work in the literature has done using SLAM and PSO for the multiple UAVs path planning. Moreover, they only focused on their approach's feasibility in all related work irrespective of the environmental constraints and UAV physical limitations. Therefore, considering the practical scenario, we have designed a system reflecting all aspects combining both VSLAM and PSO. Some of the environmental and physical traits of the surrounding have been studied in our previous work [20]. However, there are several challenges to be resolved; in this regard, we introduce dynamic fitness function (DFF), which includes multiple optimization parameters to take care of environmental constraints like terrain limitation, restricted area, collision avoidance, flight risk, Etc., similarly, UAV physical limitations like turning angle, flight slope, Etc. Moreover, we propose the region sensitivity (RS) to encounter any unexpected hazardous event in the UAV path. The proposed DFF and RS work in the optimization algorithm by feeding the realistic environment's information using a monocular vision-based SLAM approach.

1.2.1 DIFFERENCES IN SYSTEM METHODOLOGIES IN THE LITERATURE

The major differences in the system methodologies of the state-of-the-art contributions. In [21], the authors proposed the improved PSO (IPSO) for path planning of a robot. The authors study three approaches PSO, artificial potential field (APF), IPSO in two different environmental setups. Both setups have fixed obstacles and goal of the study is to avoid the obstacles. The major difference to our approach is that we have proposed the DFF considering the multiple optimization indexes. Moreover, our environment is dynamic and any unexpected event can happen. We proposed the RS mechanism to deal with it. Similarly, the input feeding in our approach is from the realistic environment using the Visual-SLAM approach while the study in [21] utilizes the equation for input information. In [22], the authors proposed the adaptive selection mutation constrained differential evolution algorithm to solve the UAV path planning in a disaster environment. The input environmental setup is different from our approach. The major difference to this study is the use different optimization indexes to measure the fitness value to reach the optimum solution. In [23], the author proposes the reference point-based multi-objective evolutionary algorithm for the UAV path planning. In this study, the UAV visits all the targets and return to the base while being monitored by the radar. The goal is to the minimizing the radar threat in a continuous terrain which is a different approach. Similarly, the author in [24] proposed hybrid grey wolf optimization approach for the UAV path planning. The authors identified the cost function as a weighted sum of threat cost, fuel cost, and cost offered due to deviation, which is a different from our joint cost determinant. Different philosophies and goals of the system designs in the literature [25] resulted in different system parameters.

2 VISUAL-SLAM FRAMEWORK

Vision-based SLAM systems mostly work on the corner features detector, for instance, Harris corner detector. The corner points are easily distinguishable so that the data association for the corner detector is relatively easy to extract. However, in non-textured environment, the corner detector is not able to detect the sufficient feature points. To address this problem, we introduce the fused mechanism of corner-edge points which utilize the edge points as well. The point detected on the edge segment termed as edge-point. Our method detect the corner as well as edge points to estimate the position of the camera by matching the 3D points of the next image. In this way, the trajectory of the camera and a 3D map are built.

It provides robustness for the estimation process due to large number of points detected even in the non-textured condition. It detect the edge point in case of long edge segments which are hard to detect. In addition to robustness, it provides the detailed representation of the object which increase the modeling process of surface detection and reconstruction.

2.1 Approach

Correspondence between the points may lead to multiple matches, including outliers. Random sampling consensus (RANSAC) [26] handles inliers, outliers, dividing data using perspective projection [27].The large matching errors are eliminated by the progressive sample consensus PROSAC algorithm [28]. The motion of the camera and correspondence of the points between the frames occur simultaneously. In the beginning, we estimate the trajectory of the camera with small detected points, and afterwards, we use a coarse-to-fine approach to refine the trajectory and feature point correspondence by progressively increasing the points. The overall approach to constructing a map using the visual-SLAM system can be seen in Figure 1.

2.1.1 KEYPOINT MATCHING

Keypoint matching is an essential operation in most computer-vision applications, encompassing Structure from Motion (SfM), Multi-view Stereo (MvS), image registration, and image retrieval. The process starts with the detection and description of keypoints and continues to the keypoint-matching. The detection step searches the locally distinctive and adequate keypoints, which can be easily observed (uniqueness) and easily detected in different images frame (reliability). The descriptor is usually a multi-dimensional vector that represents the detected keypoints in space. Finally, the descriptor in one image searched the approximate nearest neighbor descriptor in every other overlapping image and matched them. Therefore, the matched points indicate the keypoint in the image, which is projected onto the corresponding two images taken from the two different viewpoints. blueWe initially utilize features from the accelerated segment test (FAST) to determine the keypoints. Afterward, the edge points are detected utilizing the popular Canny edge detector [30] and Harris Corner detector [29]. The criteria for matching around the edge points is a normalized, small correlated window. The outliers are eliminated by the orientation of the gradient at the edge using the robust independent elementary features (BRIEF) descriptor. As they have the nominal computational complexity and more accuracy than other detectors and descriptors [31]. The keypoint matching with respect to scaling the image. The highest rate of matching is given by the ORB, which is a combination of FAST and BRIEF, is 49.5% in 0.02 second while least rate is noticed by the SIFT i.e. 31.8% in 0.25 second, while the SURF gives rate of 3.6.6% in 0.08 second.

2.1.2 KEYPOINT RECONSTRUCTION

A 3D point from consecutive images is calculated using the following equation:

$$P_e = (\frac{b(x_1 + x_r)}{2(x_1 - x_r)}, \frac{by}{(x_1 - x_r)}, \frac{bf}{(x_1 - x_r)})^T \tag{1}$$

where b indicates baseline, and f is the focal length, y=y_1=y_r, while (x_1, y_1) represents the points on one image, and (x_r, y_r) represents the point on the consecutive next image. We set $u = (x_1, y_1, x_r, y_r)$ and $P_c = S(u)$, and therefore, the covariance of the edge point (P_c) is calculated as

$$\Sigma_{P_e} = \frac{\delta S}{\delta u} \Sigma_u \frac{\delta S^T}{\delta u} \tag{2}$$

Now, we assume $\Sigma_u = diag(\sigma^2_{x_1}, \sigma^2_{y_r}, \sigma^2_{y_r})$ and, for the implementation, we take $\sigma_{x_r} = \sigma_{y_1} = \sigma_{y_r} = 0.5[Pixels]$. The correlation between $\sigma_{y_1} and \sigma_{y_r}$ is assumed to be very strong.

2.1.3 CAMERA MOTION ESTIMATION

The trajectory of the camera can be estimated by successfully matching the points from time t-1 to t when the points are reconstructed in frame $I_t - 1$, and the points are detected at frame I_t. There are two methods for point registration. One is a 3D–3D method for point matching in which the reconstructed 3D points up to frame $I_t - 1$ are taken and matched with the 3D points reconstructed at frame I_t. The other method for point registration is 3D–2D matching in which the reconstructed 3D points up to frame $I_t - 1$ are taken and matched with the detected 2D points in frame I_t. The latter method has mostly been taken in the literature, because the influence of the depth errors in registration accuracy is very small, which makes it more stable [33]. Let v_t be a camera pose at time t, where P^i_{t-1} is a i-th reconstructed 3D point at $t - 1$. Similarly, p^i_{t-1} is a point that was taken as a projection of P^i_{t-1} on the image at I_t. The point P^i_{t-1} is termed a map point because it is stored for map generation, and therefore, point p^i_{t-1} can be represented as $p^i_{t-1} = k(P^i_{t-1}, v_t)$, where k indicates the function of perspective projection:

$$\begin{aligned} K &= N_t^{-1}(P^i_{t-1} - M_t) \\ k(P^i_{t-1}, v_t) &= (f\frac{K_x}{K_z}, f\frac{K_y}{K_z})^T \end{aligned} \tag{3}$$

where M_t and N_t are the translation and rotation matrices of vector v_t. Let g^i_t is a point on the image corresponding to p^i_{t-1}, so the cost function, C, can be defined as

$$C(v_t) = \Sigma^n_{i=1} q(g^i_t, p^i_{t-1}) \tag{4}$$

where $q(g^i_t, p^i_{t-1})$ represents the penalty that depends on the Euclidean distance between points g^i_t and p^i_{t-1}. We use the perpendicular distance between the point p^i_{t-1} and the segment containing the point g^i_t in image [33]. We estimate the motion using pose vector v_t at time t, and the correspondence between the points from decreasing cost function $C(v_t)$. This can be achieved by utilizing the gradient descent method, setting the initial value of vector v_t to $v_t - 1$, and setting closest point g^i_t to its closest corresponding point, p^i_{t-1}, by calculating the Euclidean distance. This process of point matching repeats, which decreases $C(v_t)$, and the optimal pose vector v_t, and thus, point correspondences are achieved.

2.1.4 MAP CONSTRUCTION

We build an incremental 3D map of the environment based on camera pose vector v_t by transforming the 3D points into world coordinates from the camera coordinates. Let us take the camera coordinates and P^i_e as the i-th 3D point, so the location of this point in the camera coordinates can be represented as follows:

$$P^i = c(P^i_e, v_t) = N_t P^i_e + M_t \tag{5}$$

We integrate the identified 3D points based on their correspondences, which decreases the depth error. Based on the covariances, we integrate the location of all the identified 3D points. We take the average location of the identified 3D points between the keyframes, which increases the efficiency. The created 3D points indicate the map, and estimate the trajectory of the camera, between the keyframes.

2.1.5 CAMERA MOTION UPDATE

Camera motion is updated by extracting the keyframe from the sequence of images with interval d, and then, we refine the motion using the RANSAC algorithm between the keyframes. As can expected, the camera motion is relatively

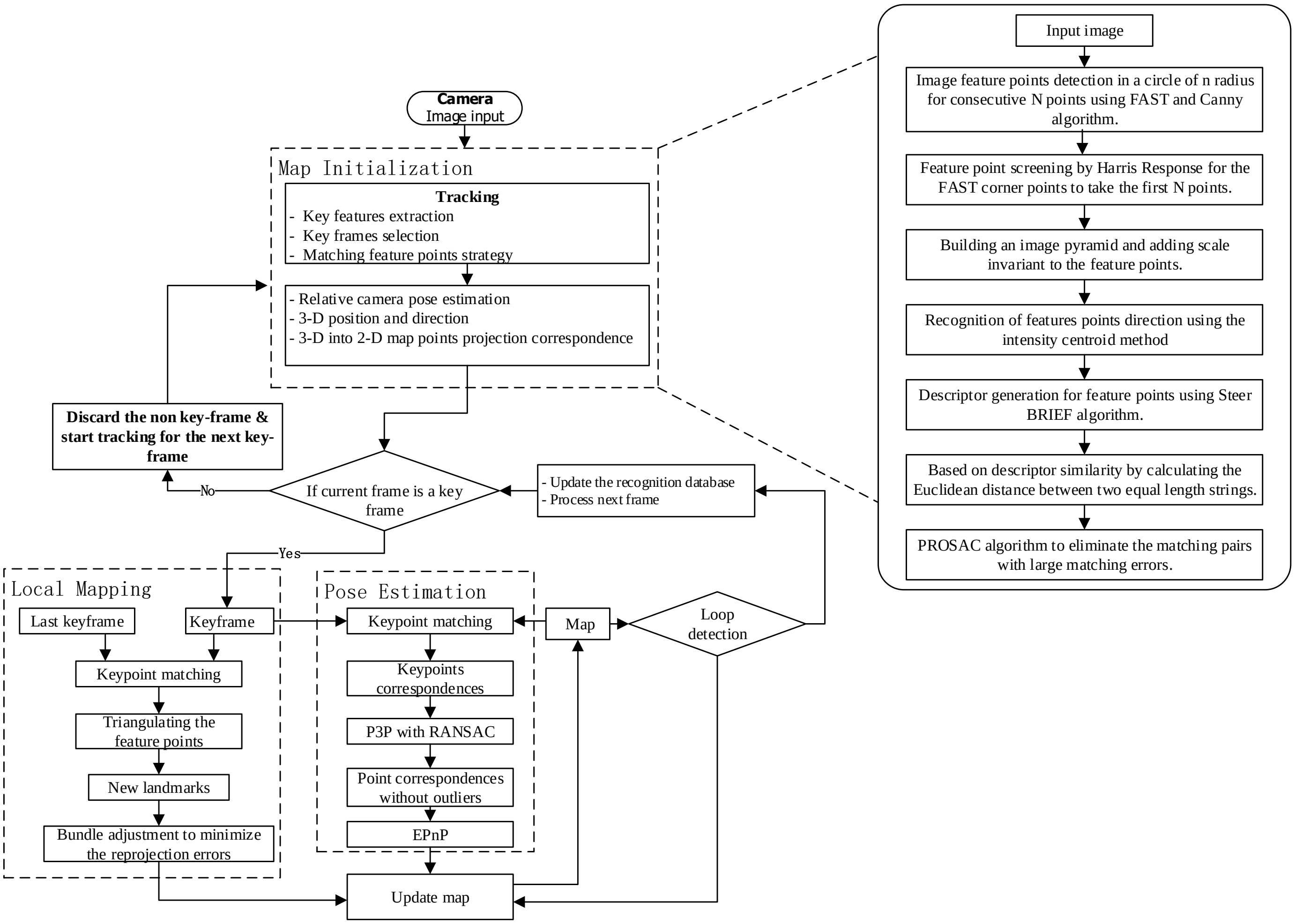

Camera
Image input
Map Initialization
Tracking
- Key features extraction
- Key frames selection
- Matching feature points strategy
- Relative camera pose estimation
- 3-D position and direction
- 3-D into 2-D map points projection correspondence
Discard the non key-frame & start tracking for the next key-frame
No
If current frame is a key frame
- Update the recognition database
- Process next frame
Yes
Local Mapping
Last keyframe
Keyframe
Keypoint matching
Triangulating the feature points
New landmarks
Bundle adjustment to minimize the reprojection errors
Pose Estimation
Keypoint matching
Keypoints correspondences
P3P with RANSAC
Point correspondences without outliers
EPnP
Map
Loop detection
Update map
Input image
Image feature points detection in a circle of n radius for consecutive N points using FAST and Canny algorithm.
Feature point screening by Harris Response for the FAST corner points to take the first N points.
Building an image pyramid and adding scale invariant to the feature points.
Recognition of features points direction using the intensity centroid method
Descriptor generation for feature points using Steer BRIEF algorithm.
Based on descriptor similarity by calculating the Euclidean distance between two equal length strings.
PROSAC algorithm to eliminate the matching pairs with large matching errors.

Fig. 1. Flowchart of map construction using Visual-SLAM.

large between the keyframes, so to avoid the local minima, we initialize the value of a keyframe to I_d from the estimated camera motion by each keyframe $I_t + 1$. Every 3D point P^i_{t-d} taken upto keyframe I_{t-d} is supposed to project onto keyframe I_t and match to the 3D point q^i_t in the image. Errors exist due to accumulation between the keyframes, which are reduced using the bundle adjustment, and therefore, the keyframe significantly converges to the global minima to update the motion of the camera in terms of 3D points.

Uncertainty is evaluated by calculating the covariance matrix of camera poses. We use $\overline{v}_t$ and Σ_{vt} to represent the mean and covariance, in which $\overline{v}_t$ is calculated from the keyframe, whereas Σ_{vt} is calculated with the following mechanism. Let s_t represents the vector of multiple points in the image at time t where w_t indicates the vector of 3D points, which are matched with s_t. We can indicate s_t as $s_t = h(w_t, v_t) + n_t$, where n_t is noise having zero mean and zero covariance, Σ_{nt}, and s_t can be obtained with the Taylor expansion, as follows:

$$s_t \approx k(\overline{w}_t, \overline{v}_t) + \frac{\delta k}{\delta w_t}(w_t - \overline{w}_t) + \frac{\delta k}{\delta v_t}(v_t - \overline{v}_t) + n_t \tag{6}$$

We can calculate the covariance of camera trajectory utilizing equation 6, as follows:

$$\begin{aligned} \Sigma_{vt} &= (J^T_{vt}(\Sigma_{nt} + J_{wt}\Sigma_{wt}J^T_{wt})^{-1}J_{vt})^{-1} \\ J_{vt} &= \frac{\delta k}{\delta v_t}(\overline{w}_t, \overline{v}_t), \quad J_{wt} = \frac{\delta k}{\delta wt}(\overline{w}_t, \overline{r}_t) \end{aligned} \tag{7}$$

where Σ_{wt} represents the covariance matrix of the 3D points that match s_t. The size of the Σ_{wt} depends on the number of 3D points and if the number of points is large, which makes Σ_{wt} computation intractable.

To deal with this problem, we ignore the correlation term in Σ_{wt} to simplify the computation. One interpretation is to ignore the correlation entity in Σ_{wt} so that the location of all the 3D points are independent of one another. But this interpretation marks the covariance as very small. Therefore, we assume that the location of all the 3D points that are reconstructed from the same frame have a strong correlation. To overcome the complexity, we divide all 3D points into two parts, w_a and w_b, where w_a indicates the 3D points reconstructed from the last keyframe, I_{t-d}, and w_b represents the reconstructed 3D points from the past keyframes, I_1 to I_{t-2d}. This is assumed based on the fact that all the 3D points are generated in the very last keyframe, more specifically I_{t-2d+1} to I_{t-d}. In the same flow, we again assume that 3D points in the same group have a strong correlation in their locations, while 3D points in separate groups have zero correlation in their locations. In this, we can approximate each group covariance to the mean covariance of all 3D points. Considering all the assumptions, we have the following:

$$J_{wt}\Sigma_{wt}J_{wt}^T = \frac{1}{|w_a|}\sum_{P\in wa} J_P\Sigma_P J_P^T + \frac{1}{|w_b|}\sum_{P\in wb} J_P\Sigma_P J_P^T$$
$$J_P = \frac{\delta k}{\delta P}(\overline{P}, \overline{v}_t) \tag{8}$$

where J_P and Σ_P represent the Jacobian and covariance matrix of a 3D point, respectively. This supposition decreases the computational complexity of the system.

2.1.6 MAP UPDATE

We construct the 3D map according to section 2.1.4. We fuse the matched 3D points with weights according to their covariances. The 3D point explained in section 2.1.4 can also be expressed as

$$P_t^i = c(P_{e,t}^i, v_t) \tag{9}$$

As mentioned above, we are ignoring correlation term Σ_{wt},and therefore, we calculate the covariance matrix of each 3D point. Let $\overline{P}_t^i$ and $\Sigma_{P_t^i}$ represent the mean and covariance of a 3D point, respectively. Using the Taylor expansion, we have the following:

$$P_t^i \approx c(\overline{P}_{e,t}^i, \overline{v}_t) + \frac{\delta c}{\delta P_{e,t}^i}(P_{e,t}^i - \overline{P}_{e,t}^i) + \frac{\delta c}{\delta v_t}(v_t - \overline{v}_t) \tag{10}$$

The covariance of 3D point P_t^i can be calculated using equation 10 as follows:

$$\overline{\Sigma}_{P_t^i} = \frac{\delta c}{\delta P_{e,t}^i}\Sigma_{P_{e,t}^i}\frac{\delta c}{\delta P_{e,t}^i}^T + \frac{\delta c}{\delta v_t}\Sigma_{v_t}\frac{\delta c}{\delta v_t}^T \tag{11}$$

We update the location and covariance of a 3D point by fusing equation 11 with the point at $t-d$, as follows:

$$\overline{P}_t^i = \overline{P}_{t-d}^i + \Sigma_{P_{t-d}^i}(\Sigma_{P_{t-d}^i} + \overline{\Sigma}_{P_t^i})^{-1}(P_t^i - \overline{P}_{t-d}^i)$$
$$\Sigma_{P_t^i} = (\Sigma_{P_{t-d}^i}^{-1} + \overline{\Sigma}_{P_t^i}^{-1})^{-1} \tag{12}$$

3 SWARM-BASED PATH PLANNING APPROACH

In this section, we introduce the proposed path planning scheme based on particle swarm optimization. The elevation map generated by the visual-SLAM algorithm is used as input terrain information for the optimization algorithm to plan the optimum path. The data set we used in our system is very diverse, and provides information on the terrain. There are multiple system constraints, which must be satisfied before planning the path from source to destination and meeting the multiple objectives we desire in order to obtain the maximum value. In this regard, we propose the DFF to derive the optimal trajectory of the UAVs while considering all the constraints and objectives of the system.

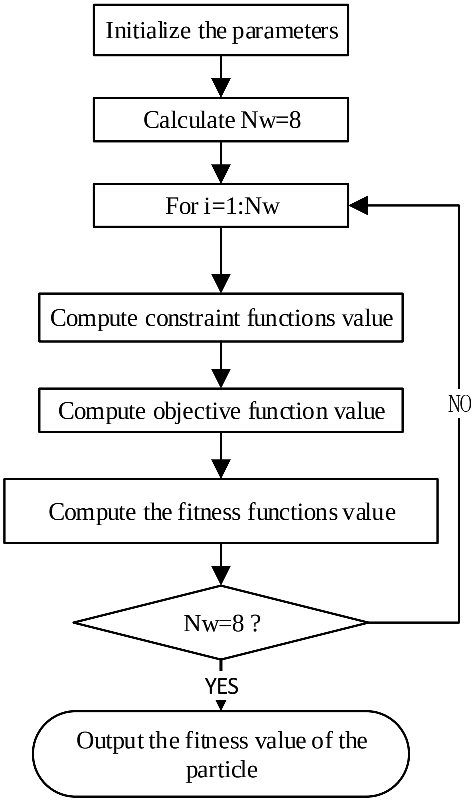

Initialize the parameters
Calculate Nw=8
For i=1:Nw
Compute constraint functions value
Compute objective function value
Compute the fitness functions value
Nw=8 ?
NO
YES
Output the fitness value of the particle

Fig. 2. Flowchart to compute dynamic fitness function

3.1 Working Principle of Particle Swarm Optimization

PSO is a heuristic search algorithm. It was first developed by Kennedy and Eberhart in 1995 to introduce a method for optimization of a nonlinear function [35]. PSO has been regarded as one of most effective optimization techniques. It is a nature-inspired set of computational methodologies to resolve complex real-world problems. PSO executes the search mechanism using a swarm of particles that update iteratively to seek the optimum solution. Each particle moves in accordance to its previous best particle in the group, as well as the global best particle in the swarm. Each particle updates its velocity and position according to information received from the previous velocity and best position achieved by any particle in the group, and the best position achieved by the global swarm.

3.2 PSO Formulation

The mathematical formulation for each particle's velocity and position are stated as follows. Let the total number of particles in a swarm be P, the total iterations is N, and the 3D dimension of each particle is D. Therefore, for the i^{th} particle, position x and velocity v can be represented as:

$$x_i = (x_{i1}, x_{i2}, ..., x_{iD})$$
$$v_i = (v_{i1}, v_{i2}, ..., v_{iD}) \tag{13}$$

The position for the best particle, $p_{i,best}$, in the group and the global best swarm particle, s_{best}, can be computed as follows:

$$\begin{aligned} p_{i,best} &= (p_{i1,best}, p_{i2,best}, ..., p_{iD,best}) \\ s_{best} &= (s_{1,best}, v_{2,best}, ..., v_{D,best}) \end{aligned} \tag{14}$$

Because p_{best} and s_{best} are termed cost values for PSO, once a cost function is defined, then the position and velocity are updated as follows:

$$\begin{aligned} x_{ij}^{t+1} &= x_{ij}^{t} + v_{ij}^{t+1} \\ v_{ij}^{t+1} &= \chi v_{ij}^{t} + ar_1(p_{ij,best} - x_{ij}^{t}) + br_2(s_{j,best} - x_{ij}^{t}) \end{aligned}$$

$$for \ \ i = 1,2,3,...,P \ \ j = 1,2,3,....,D \ \ t = 1,2,3,...,N \tag{15}$$

where a and b are the self-cognitive acceleration property and the social knowledge parameter of the swarm, respectively, which represent the inheritance characteristics of the personal particle and the whole swarm; r_1 and r_2 are random values in the range [0-1], and χ represents the inertia of an individual particle, which induces an effect on the velocity from one iteration to next iteration. The authors in [36] suggested optimum values of $a = b = 1.496$ and $\chi = 0.7298$ for PSO performance.

4 Proposed Dynamic Fitness Function

In order to derive the optimal trajectories, the DFF computes the fitness of the trajectory considering optimization parameters, which are divided into two groups, namely, objectives and constraints. The former consist of risk estimation, energy consumption, and operation completion time; the latter are further divided into two parts depending upon the UAV's physical constraints (flying slope and turning angle) and the physical limitations of the environment (region sensitivity, restricted areas, and terrain constraints). The working flow of the DFF can be observed in Figure 2. The DFF can be formulated as seen in equation :

$$DFF_{fitness} = F_{objectives} + F_{constraints} \tag{16}$$

where $F_{objectives}$ indicates the objectives function on which we focus to gain the maximum value, whereas $F_{constraints}$ indicates the UAV physical and environmental restrictions, which must be fulfilled before planning the trajectory.

4.1 Objectives Design

We have defined the above optimization parameters, and the objectives were designed to improve the quality of path planning. We have defined the objectives as weighted components of risk estimation, energy consumption, and operation completion time so that the formulation of the objectives can be represented as follows:

$$F_{objectives} = w_1 O_{RE} + w_2 O_{EC} + w_3 O_{OT} \tag{17}$$

where w_1, w_2, w_3 denote the weights of the objective components [34], which are chosen to derive the importance of each component while planning the path, and O_{RE}, O_{EC}, O_{OT} are functions from which values are taken in the range [0,1]. We aim to derive the optimum path with less risk, energy, and time.

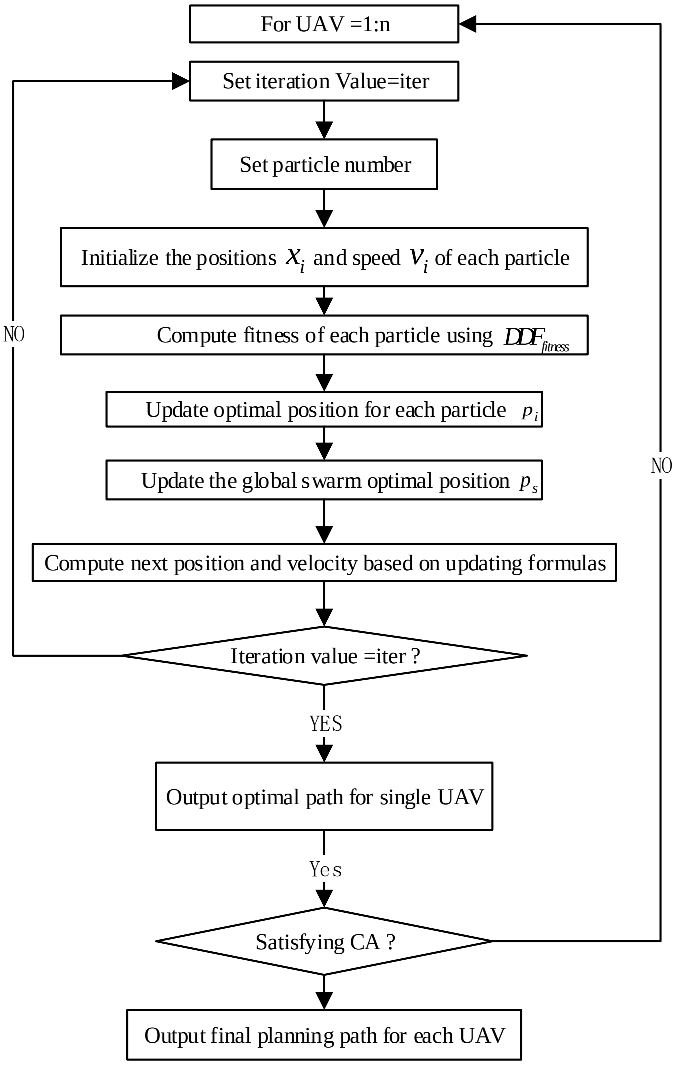

For UAV =1:n
Set iteration Value=iter
Set particle number
Initialize the positions X_i and speed V_i of each particle
Compute fitness of each particle using $DDF_{fitness}$
Update optimal position for each particle p_i
Update the global swarm optimal position p_s
Compute next position and velocity based on updating formulas
Iteration value =iter ?
NO
YES
Output optimal path for single UAV
Yes
Satisfying CA ?
NO
Output final planning path for each UAV

Fig. 3. Framework of the path planner

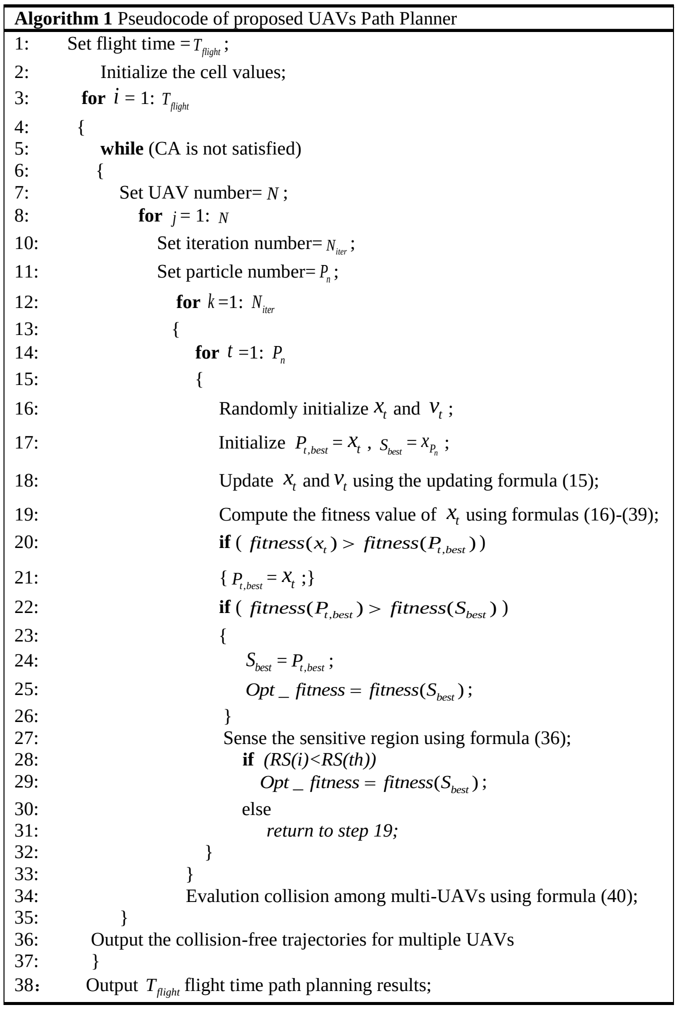
4.1.1 Risk Estimation

There are some restrictions that should be applied during flight. Small UAVs are very vulnerable to harsh weather conditions, like rain, snow, or strong winds, which can pose a high risk of damage to the UAVs. Similarly, the UAV altitude while executing the task should be reasonable, because at a higher altitude, the wind is stronger, which can increase the risk. Moreover, the camera sometimes cannot focus properly due to dense clouds that also put the UAV at risk. On the basis of the above risk situations, we define the following two types of risk.

4.1.1.1 Environmental Risk : The environment has a wide range of characteristics, and therefore, it is difficult to make a model that precisely measures the environmental risk. Therefore, for simplicity, an environmental value is generated randomly, $r_{i,i+t}^{e}$, that represents the risk from the $i-th$ waypoint to waypoint $(i+1)$. The summation of the risk values would be considered the environmental risk.

4.1.1.2 Altitude Risk : The altitude risk is actually an absolute difference in altitude between two waypoints, and therefore, we formulate altitude risk $r_{i,i+t}^{a}$ as follows:

$$r_{i,i+t}^{a} = k * (z_{i+1} - z_i) \tag{18}$$



Algorithm 1 Pseudocode of proposed UAVs Path Planner

```
1:   Set flight time = $T_{flight}$;
2:       Initialize the cell values;
3:     for $i$ = 1: $T_{flight}$
4:     {
5:       while (CA is not satisfied)
6:       {
7:         Set UAV number= $N$;
8:           for $j$ = 1: $N$
10:            Set iteration number= $N_{iter}$;
11:            Set particle number= $P_n$;
12:              for $k$ =1: $N_{iter}$
13:              {
14:                for $t$ =1: $P_n$
15:                {
16:                  Randomly initialize $x_t$ and $v_t$;
17:                  Initialize $P_{t,best} = x_t$ , $S_{best} = x_{P_n}$;
18:                  Update $x_t$ and $v_t$ using the updating formula (15);
19:                  Compute the fitness value of $x_t$ using formulas (16)-(39);
20:                  if ( $fitness(x_t) > fitness(P_{t,best})$ )
21:                  { $P_{t,best} = x_t$ ;}
22:                  if ( $fitness(P_{t,best}) > fitness(S_{best})$ )
23:                  {
24:                      $S_{best} = P_{t,best}$;
25:                      $Opt\_fitness = fitness(S_{best})$;
26:                  }
27:                  Sense the sensitive region using formula (36);
28:                    if (RS(i)<RS(th))
29:                      $Opt\_fitness = fitness(S_{best})$;
30:                    else
31:                        return to step 19;
32:                }
33:              }
34:              Evalution collision among multi-UAVs using formula (40);
35:        }
36:    Output the collision-free trajectories for multiple UAVs
37:    }
38:   Output $T_{flight}$ flight time path planning results;
```

Fig. 4. Pseudocode of the proposed path planner

where k represents a constant parameter for control. Since risk estimation depends on the location, and it increases or decreases according to the weather conditions at that particular location, as well as on the UAV's altitude at the same time during flight. Therefore, the total risk can be formulated as follows:

$$O_{RE} = \frac{\sum_{i=1}^{N_w - 1} RE_i}{maxRE} \tag{19}$$

$$\begin{gathered} RE_i = w_{ER} r^{e}_{i,i+t} + w_{AR} r^{a}_{i,i+t} \\ where \;\; w_{ER} + w_{AR} = 1 \end{gathered} \tag{20}$$

RE_i shows the total risk from the i-th waypoint to waypoint $(i + 1)$), while w_{ER} and w_{AR} are the weight factors of the environmental and altitude risks, respectively. N_w denotes the total number of waypoints from source to destination, and $maxRE$ is a normalized value of the risk, which can be computed as follows:

$$maxRE = (N_w - 1) * [w_{ER} * Z * w_{AR}(2 * maxr^e)] \tag{21}$$

where $maxr^e$ indicates the maximum value instigated by the environment risk, and Z is the altitude of the UAV during flight.

4.1.2 ENERGY CONSUMPTION

Fuel plays a pivotal role for UAVs while executing their tasks. Therefore, it is very important that the UAV must reach the destination before consuming all its energy. In this regard, a feasible path from source to destination with less energy consumption (EC) would be the priority. We assume that the velocity of the UAV remains constant during flight. We formulate EC as follows:

$$O_{EC} = \frac{\sum_{i=1}^{N_w - 1} FC_i}{maxFC} \tag{22}$$

$$FC_i = P_u * t_{i,i+1} \tag{23}$$

$$t_{i,i+1} = \frac{d_{i,i+1}}{v} \tag{24}$$

$$d_{i,i+1} = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2} \tag{25}$$

where FC_i represents the fuel burned in flying from the i-th waypoint to waypoint $(i + 1)$. P_u is the power of the UAV at velocity v, while $t_{i,i+1}$ is the total time taken by the UAV to fly from the i-th waypoint to waypoint $(1 + i)$; $d_{i,i+1}$ indicates the Cartesian distance of a flight from the i-th waypoint to waypoint $(1 + i)$, and $maxFC$ is a normalized value for fuel consumption, which can be formulated as follows:

$$\begin{gathered} maxFC = (N_w - 1) * P_u * \frac{d_{max}}{v} \\ where \;\; d_{max} = \sqrt{X^2 + Y^2 + Z^2} \end{gathered} \tag{26}$$

where X,Y,Z indicate the three dimensions of the UAV, i.e., the X-axis, Y-axis, and Z-axis, respectively, during flight time.

4.2 Constraints Design

The constraints are designed to optimize the feasible path for the flight. When a constraint is satisfied, it will be equal to 0; if not, a penalty will be given. Assigning negative penalty Q ensures that the path from source to destination is guaranteed feasible, because the fitness value of each optimized path is always positive. Considering the physical limitations of the UAV and the restrictions due to the environment, we can design the constraints as follows:

$$F_{Constraints} = UAV_{constraints} + Environment_{constraints} \tag{27}$$

4.2.1 UAV CONSTRAINTS

These constraints are due to the physical properties of a UAV during flight. The behavior offered by the UAV while maneuvering should be regarded as a priority, because it provides smoothness in flying to reach the destination. In this respect, we take care of the critical aspects of a UAV for turning angles as well as flying slope. Therefore, we can define the UAV constraints as follows:

$$UAV_{constraints} = TA + FS \tag{28}$$

TABLE 1
Simulation Parameters

Parameter	Value
No. of UAVs	2
Speed	10 m/sec
Power	20
Iteration number	32, 64, 128, 256, 512
Sensitivity threshold	10
Turning angle threshold	85^o
Gliding angle threshold	-30^o
Climbing angle threshold	30^o
Minimum distance threshold	0.2
Initial environmental risk	$1-5$
Flight time threshold	2
Grid size	20x20

4.2.1.1 Turning Angle: The turning angle explains the maneuverability of a UAV in the horizontal direction, i.e., the angle adopted during the flight from the previous and current directions. The turning angle should be less than the maximum tolerable threshold for turning, and therefore, we formulate the turning angle as follows:

$$TA = 0,\ \ TA = \sum_{i=2}^{N_w - 1} TA_i \tag{29}$$

$$where\ \ TA_i = \begin{cases} Q, & if\ \ \theta > \theta_{max} \\ 0, & otherwise \end{cases}$$

where θ ddefines the turning angle of the UAV in 3D directions (x_i, y_i, z_i), and θ_{max} maximum tolerable angle. The authors in [34] provided the formulation to calculate turning angle θ_i as follows:

$$\begin{aligned} &\theta = \arccos\left(\frac{(p_{x_i}, p_{y_i})(p_{x_{i+1}}, p_{y_{i+1}})^T}{\|(p_{x_i}, p_{y_i})\|_2\ \|(p_{x_{i+1}}, p_{y_{i+1}})\|_2}\right) \\ &p_{x_i} = x_i - x_{i-1} \\ &p_{x_{i+1}} = x_{i+1} - x_i \\ &p_{y_i} = y_i - y_{i-1} \\ &p_{y_{i+1}} = y_{i+1} - y_i \end{aligned} \tag{30}$$

where $\| x \|_2$ is a vector norm for a vector x.

4.2.1.2 Flying Slope: The flying slope defines the maneuverability of a UAV in the vertical direction, i.e., the slope adopted by the UAV while gliding and when climbing. The slope taken by the UAV during flight is along the horizontal direction from one waypoint to the next waypoint. Considering the maximum tolerable thresholds for gliding and climbing angles, the slope of a UAV can be formulated as follows:

$$FS = 0,\ \ FS = \sum_{i=2}^{N_w} FS_i$$

$$where\ \ FS_i = \begin{cases} Q, & if\ \ f_i \notin [\tan(\alpha_{max}), \tan(\beta_{max})] \\ 0, & otherwise \end{cases} \tag{31}$$

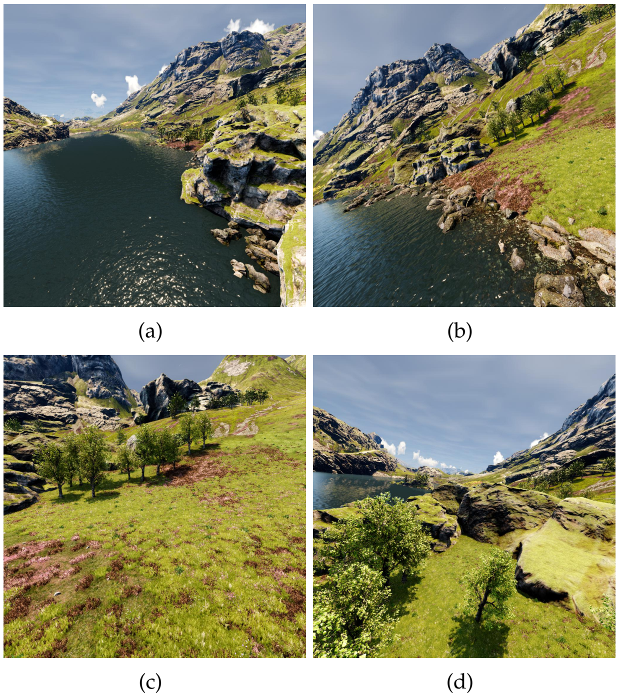

(a) (b) (c) (d)

Fig. 5. Flight path (sequence 0001)

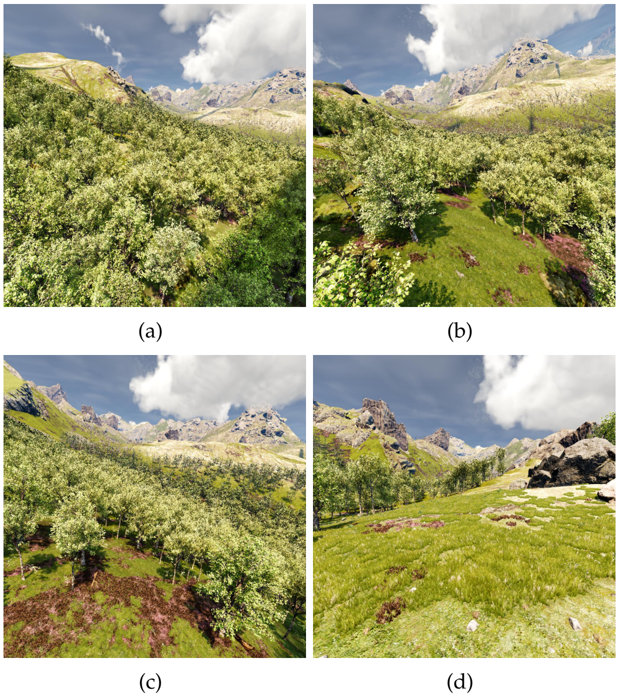

(a) (b) (c) (d)

Fig. 6. Flight path (sequence 0005)

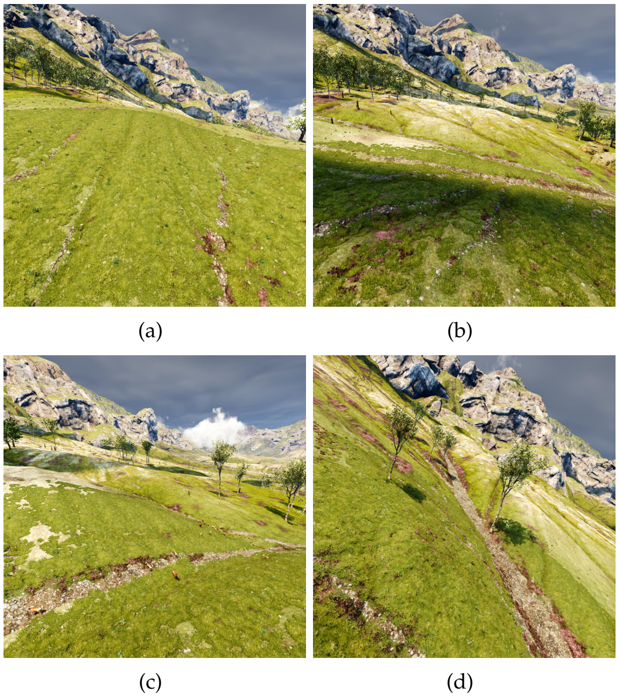

(a) (b) (c) (d)

Fig. 7. Flight path (sequence 0012)

where FS_i is the flying slope from one waypoint to the i-th waypoint; α_{max} and β_{max} represent the maximum tolerable gliding and climbing angles, and f_i can be formulated, according to [34], as follows:

$$f_i = \frac{z_i - z_{i-1}}{\|(x_i - x_{i-1}, y_i - y_{i-1})\|_2} \tag{32}$$

where f_i is the flying slope taken by the UAV from the i-th waypoint (x_i, y_i, z_i).

4.2.2 Environment Constraints

There are some restrictions due to the external environment, which the UAV must follow. We have to consider any restricted area where the UAV is not allowed to fly, like military bases, critical government institutions, etc. Therefore, the path to the destination should be designed to avoid these restricted areas. Similarly, there are certain situations where the UAV encounters an unexpected event, for instance, an unregistered aerial vehicle, flying birds, and toys flown by the general public. To deal with these situations, we introduce region sensitivity in our planning algorithm. This deals with randomly generated sensitive regions where the UAV senses a hazard and optimizes its path accordingly to avoid those regions and reach the destination safely. Moreover, the terrain offers constraints during flight. Therefore, constraints due to the environment can be formulated as follows. We divide the whole path into a 20x20 grid, and the UAV can sense four cells around itself.

$$Environment_{constraints} = RA + RS + TL + ML + CA \tag{33}$$

4.2.2.1 Restricted Area: There are some specific areas that UAVs are forbidden to fly through due to regulations, for example, cantonments, restricted government regions, etc., and therefore, the feasible path to reach the destination should be a legal one, avoiding those regions. For simplicity, we consider a restricted area to be a rectangle. Designation of a restricted area can be formulated as follows:

$$\begin{aligned} RA = 0, \;\; RA &= \sum_{i=1}^{N_w} RAC_i \\ with \;\; RAC_i &= \begin{cases} Q, & if\ waypoint\ in\ Range\ (x_r, y_r) \\ 0, & otherwise \end{cases} \end{aligned} \tag{34}$$

$$Range\ (x_r, y_r) = \{m_x \le x_r \le n_x\} \cap \{m_y \le y_r \le n_y\} \tag{35}$$

where m_x and n_x represent the lower and upper bounds, respectively, for x coordinates of the r-th restricted area at the i-th waypoint, whereas m_y and n_y indicate the lower and upper bounds, respectively, of y coordinates of the r-th restricted area at the i-th waypoint.

4.2.2.2 Region Sensitivity: There are some unconditional and unexpected events that can happen during flight. Therefore, there are randomly generated hazardous events in the path of a UAV from source to destination. The UAV senses the hazard and optimizes the path to avoid it. The sensitivity of a region at any instant during the flight can be formulated as follows:

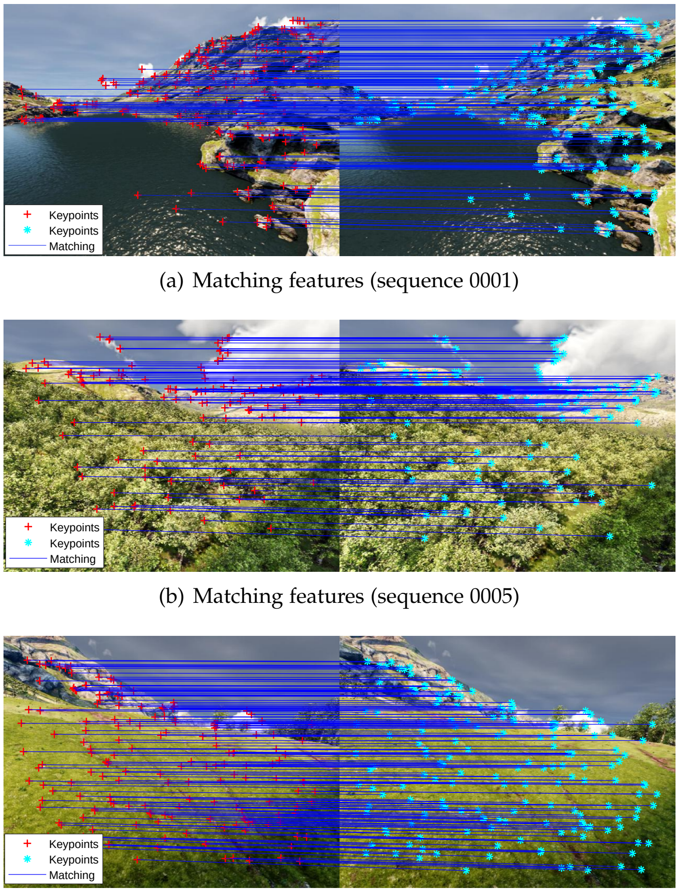
$$\begin{aligned} Rs = 0, \;\; Rs &= \sum_{i=1}^{N_w} Rs_i(t) \\ where \;\; Rs_i &= \sum_{cellx \in N(i)} v_{cellx}(t) \\ Rs_i &= \begin{cases} Q, & if\ Rs_i > Rs_{th} \\ 0, & otherwise \end{cases} \end{aligned} \tag{36}$$

where $Rs_i(t)$is the value for sensitivity at the i-th waypoint during the flight at time t, and $v_{cellx}(t)$ is the cell value at flight time t. $N(i)$ is the set of neighbor cells for the i-th waypoint. The UAV checks the values of the cells at every waypoint, and if any cell has a sensitivity value greater than the threshold, penalty Q will be given. It checks the values of the set of neighbor cells for $N(i)$ to avoid that region to satisfy the constraint.

4.2.2.3 Terrain Limits: During flight, a UAV should take into consideration the limitations of the terrain so that the UAV always flies above it and avoids collisions (for exmaple, with mountains). To adhere to a terrain constraint, the algorithm gives penalty Q to provide the feasible path. This constraint can be formulated as follows:

$$\begin{aligned} TL = 0, \;\; TL &= \sum_{i=1}^{N_w} TL_i \\ with \;\; TL_i &= \begin{cases} Q, & if\ z_i \le map(x_i, y_i) \\ 0, & otherwise \end{cases} \end{aligned} \tag{37}$$

where $map(x_i, y_i)$ is a function that returns the altitude of the terrain location at point (x_i, y_i), which finds the number of points inside that location.



(a) Matching features (sequence 0001)

(b) Matching features (sequence 0005)

(c) Matching features (sequence 0012)

Fig. 8. Image registration between consecutive scenes

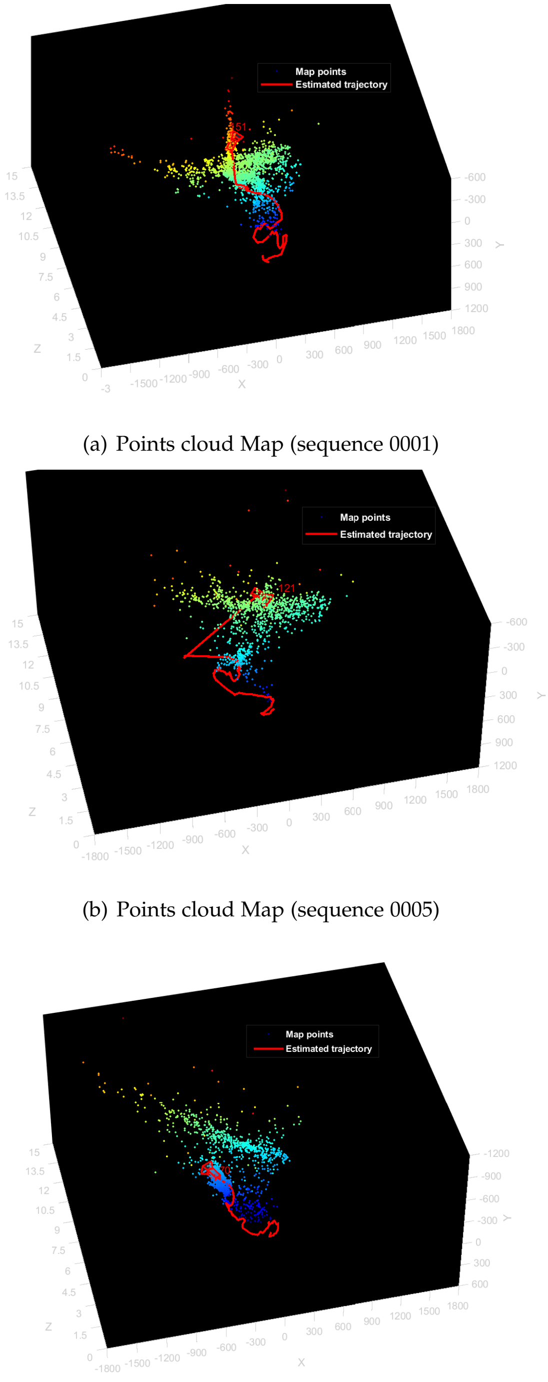
4.2.2.4 Map Limits: For a feasible path, the UAV must stay inside the mission space to avoid uncertainties. Therefore, the algorithm applies penalty Q to the points of a trajectory that are off the map limits. This constraint ensures the space of a mission can be formulated as follows:

$$
\begin{aligned}
ML &= \sum_{i=1}^{N_w} ML_i \\
with\ \ ML_i &= \begin{cases} 0, & Inmap(x_i, y_i) \\ Q, & otherwise \end{cases}
\end{aligned} \tag{38}
$$

$$
Inmap(x_i, y_i) = (x_l^m \leq x_i \leq x_u^m) \wedge (y_l^m \leq y_i \leq y_u^m) \tag{39}
$$

where x_l^m and x_u^m are the lower and upper bounds, respectively, for the x coordinate, and y_l^m and y_u^m are the lower and upper bounds, respectively, for the y coordinate. The minimum value to satisfy the map constraint is $ML = 0$.

4.2.2.5 UAV Collision Avoidance: In calculating the paths for multiple UAVs, the planner must ensure the UAVs do not come close to each other, which would increase the possibility of a collision while following their respective paths. In order to maintain a safe distance between them, the constraint can be formulated as follows:



(a) Points cloud Map (sequence 0001)

(b) Points cloud Map (sequence 0005)

(c) Points cloud Map (sequence 0012)

Fig. 9. Map construction by VSLAM to be used for path planning

$$
\begin{aligned}
CA &= \sum_{i=1}^{N_w^u} \sum_{j=1}^{N_w^v} CA_i \\
with\ \ CA_i &= \begin{cases} Q, & if\ \ d_{ij}^{uv} < d_{min} \\ Q, & otherwise \end{cases}
\end{aligned} \tag{40}
$$

$$
d_{min}^{uv} = \sqrt{(x_i^u - x_j^v)^2 + (y_i^u - y_j^v)^2 + (z_i^u - z_j^v)^2} \tag{41}
$$

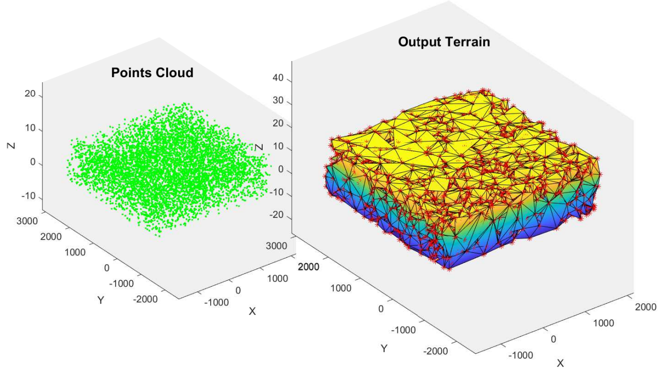


Fig. 10. Terrain representation from the points cloud of sequence 0012

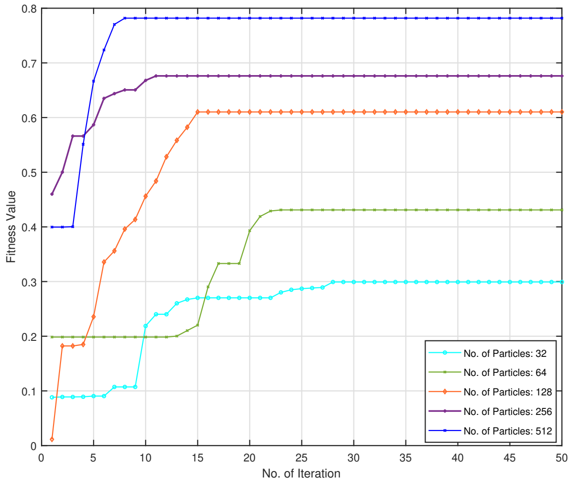


Fig. 11. Optimal fitness values for different numbers of optimization particles

where d_{min} is the minimum distance between the UAVs to avoid a collision, and d_{ij}^{uv} is the distance between the i-th waypoint and the j-th waypoint of the u-th UAV trajectory and the j-th UAV trajectory, respectively.

5 Operation of the Proposed Path Planner

In this section, we explain the working mechanism of the proposed multiple UAV–path planner, which is based on visual-SLAM, PSO, and the DFF explained in sections 2, 3, and 4, respectively. The proposed planner first utilizes the elevation map generated by visual-SLAM and fed into the PSO planning algorithm to derive the optimum trajectory for each UAV to the defined destinations, in which the DFF optimizes all the possible waypoint sequences to reach destinations considering all constraints and objectives, along with satisfying the collision avoidance condition. If all conditions are satisfied, the planner will output the optimum trajectory for each UAV to its destination, as shown in Figure 3.

TABLE 2
Distance and Time comparison with the conventional PSO algorithm

Parameter	Value
Distance covered by UAV1	3,062.4369 (m)
Distance covered by UAV2	3,065.0706 (m)
Travel time by UAV1	307.2542 (sec)
Travel time by UAV2	307.4481 (sec)

(a) Flight Dynamics of first flight using Proposed Algorithm

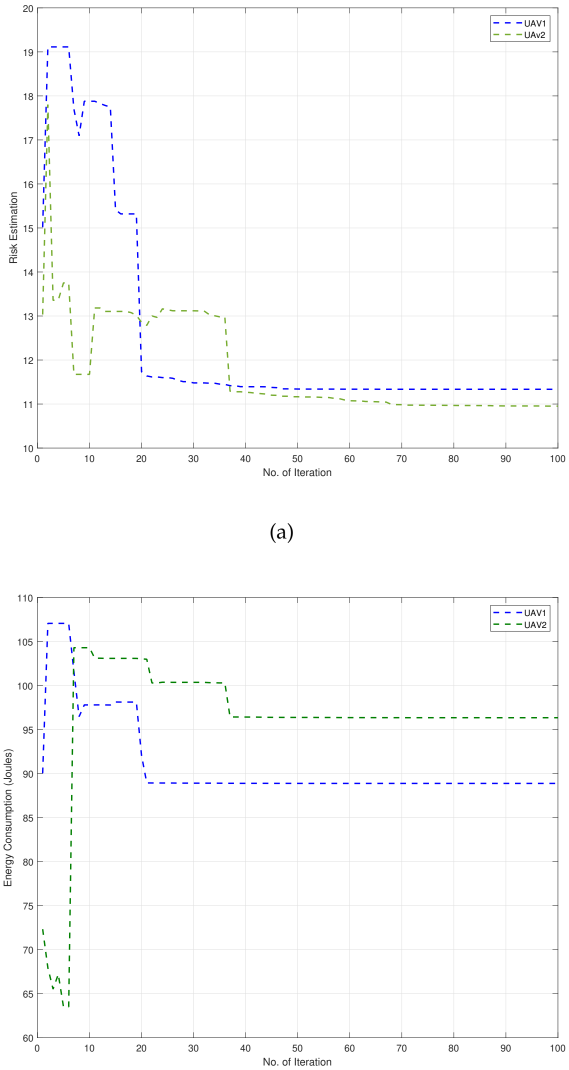
Parameter	Value
Distance covered by UAV1	5,571.9591 (m)
Distance covered by UAV2	5,452.2465 (m)
Travel time by UAV1	551.6796 (sec)
Travel Time by UAV2	549.7633 (sec)

(b) Flight Dynamics of first flight using conventional PSO Algorithm

In our proposed system, the path from source to destination consists of waypoints and line segments. We opted for an eight-waypoint trajectory-generation system. For clear understanding, we divided the whole operation area into cells. First of all, we determine the estimated flight time to the destination. Next, we initialize the PSO algorithm to plan the optimum path for each UAV, which can be seen in Figure 4 from step 5-33. In the quest to attain the optimum trajectory for each UAV, at first, the planner randomly generates the velocity and position vectors of particle P_N. Next, using equation (15), the velocity and position vectors of each particle are updated. After that, the proposed DFF is applied to the updated particle as shown the working flowchart of the DFF in Figure 2. Considering all the constraints and objectives, the DFF optimizes each particle and finally outputs the best particle, $p_{i,best}$ and the global best particle in the swarm, s_{best}, which is explained in section 4. The DFF output is based on the fitness value acquired by each particle. Then, we store the optimum path for the first UAV and set the iteration number to N_t. Before initializing the other UAVs, we aim to derive a collision-free path, and therefore, we check collision avoidance condition CA. If CA is satisfied, the planner outputs optimum trajectories for all UAVs; otherwise, it goes back to step 5 if the CA is not satisfied. Finally, when the flight time reaches T_{flight}, the planner will output the optimum paths for all UAVs to their respective destinations. The process of the proposed planner is represented in the pseudocode algorithm shown in Figure 4.

6 Simulation Results

In this section, we develop a Matlab-based operational environment to evaluate the working performance of the proposed two-step UAV path–planning system. The main simulation parameters are listed in Table 1. In our implementation, we used a data set [37] that was collected by a monocular camera installed at the quad-copter, in different environments. The data set is publicly available, and more details can be found at $midair.ulg.ac.be$. The data set was



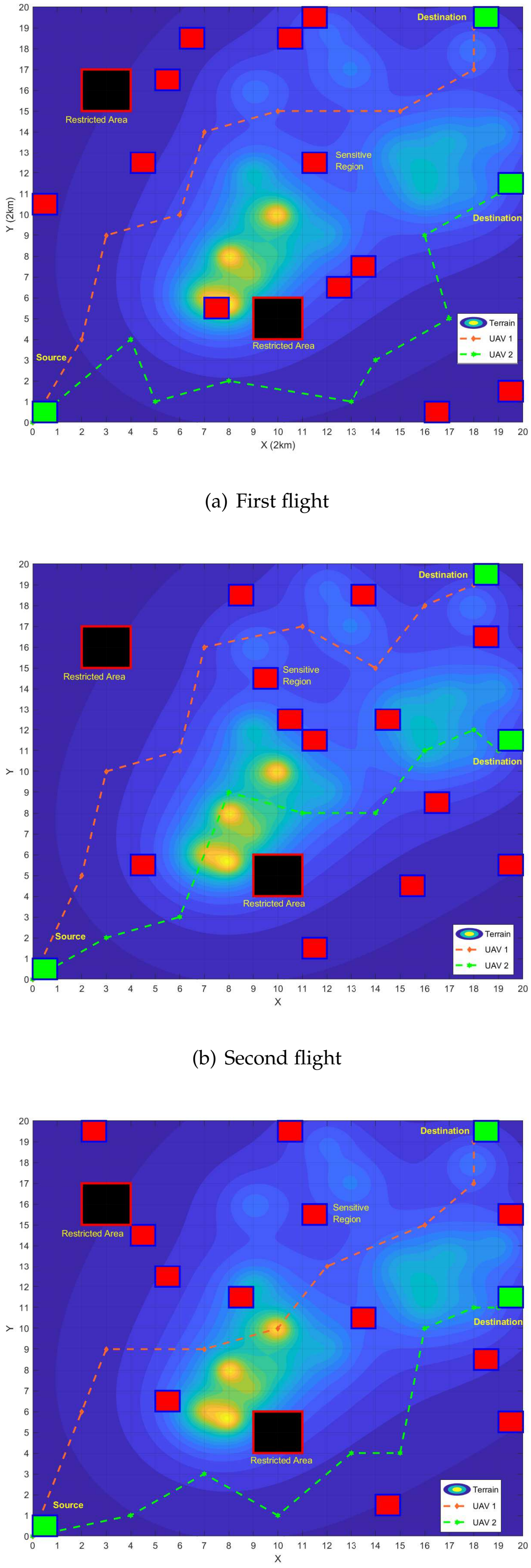
(a)

(b)

Fig. 12. Optimization of the PSO path planner performance in terms of (a) risk estimation and (b) energy consumption

utilized as input to the optimization algorithm for multiple-UAV path planning algorithm. We used different types of test sequences, which can be seen in 5, 6, and 7, in our system to construct an online map of the environment. Figure 8 indicates the features in the consecutive scenes that were matched to simultaneously build an incremental map, which can be seen in Figure 9. The points cloud map contains information on the x, y, z positions and normal at every point. The terrain representations from the points cloud can be seen in Figure 10. We utilized a triangulation algorithm [38] to reconstruct the terrain from the points cloud.

Figure 11 shows the effect of different numbers of particles on the optimal fitness value of the proposed DFF. We can clearly see that the fitness value of the proposed DFF converges to a stable value faster as the number of particles and iterations increases. The optimization performance of the path planner in terms of energy consumption and flight



(a) First flight

(b) Second flight

(c) Third flight

Fig. 13. Optimal trajectories of the UAVs from source to destination using proposed algorithm

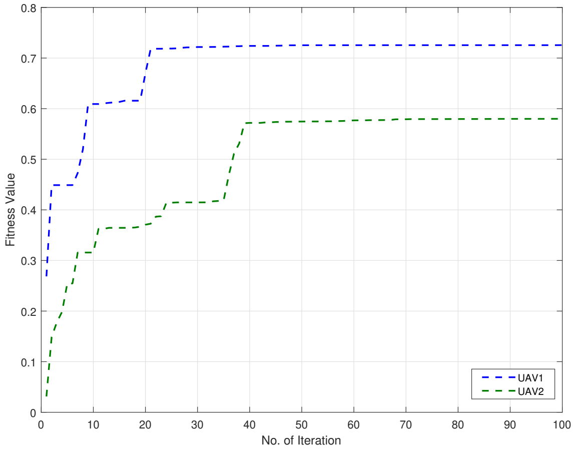


Fig. 14. Optimal fitness values attained

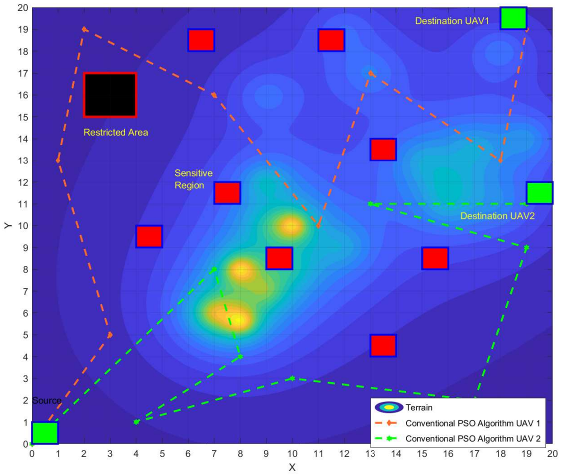
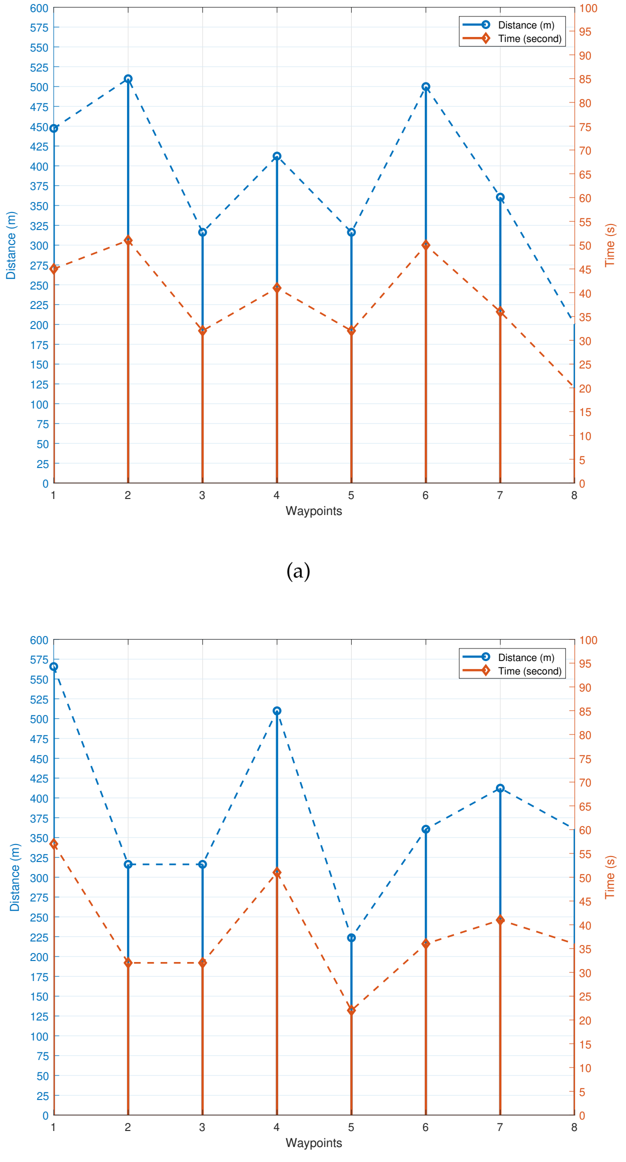


Fig. 15. UAVs Flight using Conventional PSO Algorithm



(a)

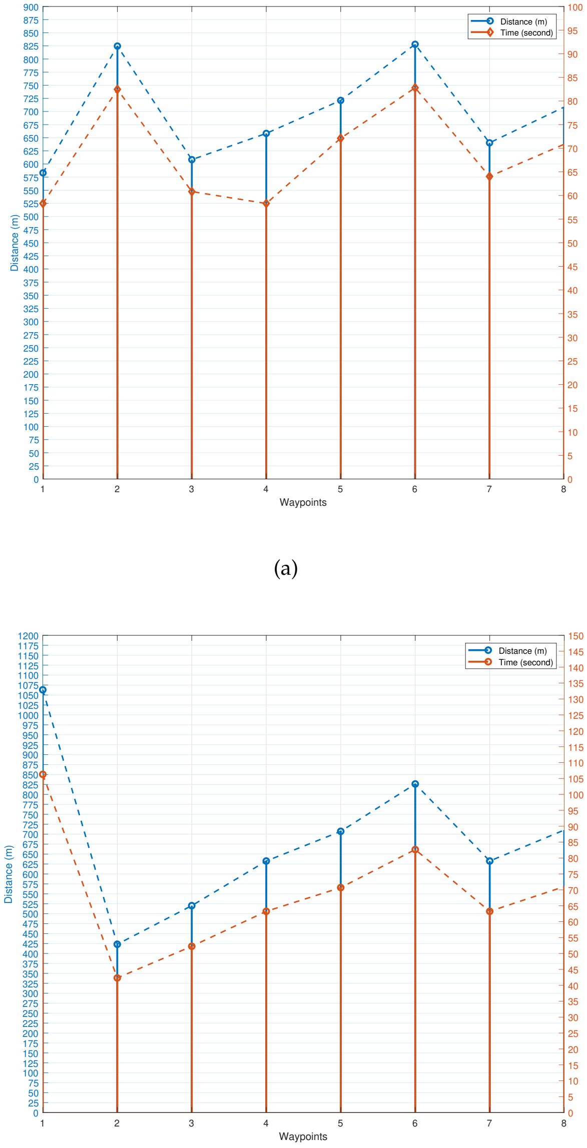
(b)

Fig. 16. UAV flight dynamics in terms of distance and time using Proposed algorithm for (a) UAV 1 and (b) UAV 2

risk estimation can be observed in Figure 12. We utilized 128 particles in our system. As the number of iterations increased, the values of energy consumption and flight risk estimation converged to a stable value. Moreover, the difference between the optimum value of energy consumption, where both UAVs converge, is less than five, and the values of flight risk estimation for both UAVs is similar, which depicts the effectiveness of the proposed path planner by ensuring fairness between the generated paths for both UAVs.

Figure 13 shows the optimal paths followed by UAV 1 and UAV 2 from source to destination while avoiding sensitive regions and restricted areas, respectively, for the first three flights. The small red 1x1 rectangles have a sensitivity greater than the threshold, while the black 2x2 rectangles indicate restricted areas where UAVs are not allowed to fly. The sensitive regions generate randomly, indicating a hazardous event, so the proposed planner optimizes the path until hazardous-free paths to the destinations are determined. We can observe that the trajectories generated for each flight time avoids all the sensitive regions and reach the destination safely. The proposed algorithm also ensures that the multiple UAVs do not collide with each other. The green 1x1 rectangles represent the source and destination. In Figure 13, the yellow highlighted areas are high elevations. We can also see that the trajectory waypoints generated do not overlap, and a UAV reaches the destination by following the shortest path, which indicates the high efficiency of the proposed planner. Therefore, Figure 14 indicates the high fitness value attained by each UAV driven by the proposed path planner.

Figure 15 indicates the trajectories generated by the conventional PSO. blue As the defined environment is dynamically complex due to which conventional PSO is incompatible with adapting the situation; therefore, it takes



(a)

(b)

Fig. 17. UAV flight dynamics in terms of distance and time using conventional PSO algorithm for (a) UAV 1 and (b) UAV 2

very high computational time to converge. Considering the incompatibility of the conventional PSO in our environment, we choose to make the environment less complicated and convenient to converge. The computational time for the conventional PSO for the simple environment is higher than our proposed algorithm in the dynamic and complex environment. The conventional PSO takes 1,767 seconds while our proposed algorithm takes 739.8 seconds. The same computer is used to run the both algorithms. The table 2 indicates the flight statistics of both algorithms for the first flight. The conventional PSO algorithm for both UAVs reaches the destination following the long path. It takes more travel time while our proposed algorithm reaches the destination for both UAVs following the shortest path and in optimal travel time in a highly complex environment.

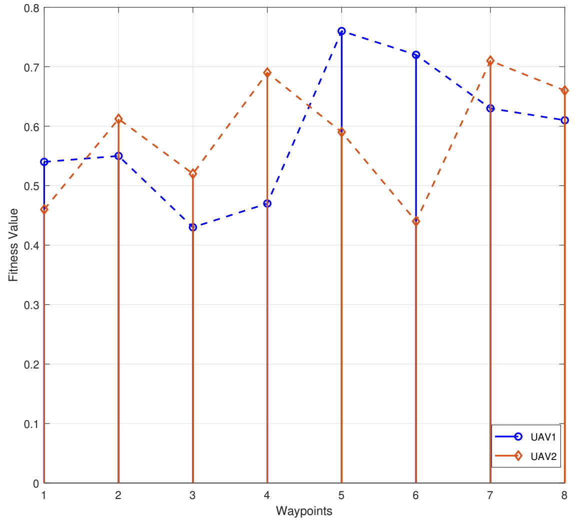


Fig. 18. Fitness values attained at each waypoint during the flight

The distance covered from one waypoint to another and the corresponding flight times for both UAVs can be seen in Figure 16. The total distances from the source to destination covered by UAVs during the first flight were 3,062.4369 m and 3,065.0706 m. Likewise, the times taken to reach the destinations for both UAVs were almost the same i.e., 307 sec. Similarly, Figure 17 indicates the distance covered and corresponding flight time for both UAVs from one waypoint to another using the conventional PSO algorithm. We can observe that the distance and time taken at each waypoint is greater than the proposed algorithm. The total distance covered by the UAVs for the first flight is 5,571.9591 (m) and 6,065.0706 (m). Similarly, the total time consumed by UAV1 and UAVs is 551.6796 (sec) and 549.7633 (sec), respectively.

In Figure 18, we show the fitness values attained at each waypoint by both UAVs during their flights. We sum up the optimal fitness values of all waypoints for UAV 1 and UAV 2. The total optimal fitness for all the wayponts of UAV 1 and UAV 2 were 5.52 and 5.51, respectively, which are virtually the same and which depict the fairness of our proposed two-step path planner.

7 Conclusions

In this paper, we designed a two-step, centralized system to construct a map using state-of-the-art visual-SLAM. We introduce corner-edge points matching mechanism to stabilize the system with the least extracted map points. The proposed algorithm effectively detects the keypoints in different environments and successfully registered the features. The constructed map is processed as an input mean for the particle swarm optimization algorithm to plan UAVs' optimum path. We proposed a dynamic fitness function considering different optimization objectives and constraints in terms of UAV flight risk estimation, energy consumption, and maneuverability for the operational time. We also proposed a path-updating mechanism based on region sensitivity to avoid sensitive regions if any hazardous and unexpected event detects in UAVs' paths. The system effectively avoids the sensitive regions and returns collision-free paths to reach UAV to the destinations safely. The simulation results validate the effectiveness of our proposed PSO-VSLAM system.

We currently consider two UAVs over different flight times to evaluate our proposed PSO-VSLAM system's performance, and it successfully outputs the collision-free trajectories and proves high adaptability towards the complex dynamic environment. Therefore, we plan to consider more than two UAVs in our future work and implement machine learning algorithms because our proposed system effectively achieves the collision-free trajectories for two UAVs while adapting to the highly dynamic and complex environment.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) under Grant NRF-2019R1F1A1061696

References

[1] Cadena, Cesar, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J. Leonard. ”Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age.” *IEEE Transactions on robotics* 32, no. 6 (2016): 1309-1332.

[2] Trujillo, Juan-Carlos, Rodrigo Munguia, Edmundo Guerra, and Antoni Grau. ”Cooperative monocular-based SLAM for multi-UAV systems in GPS-denied environments.” *Sensors* 18, no. 5 (2018): 1351.

[3] Du, Hao, Wei Wang, Chaowen Xu, Ran Xiao, and Changyin Sun. ”Real-Time Onboard 3D State Estimation of an Unmanned Aerial Vehicle in Multi-Environments Using Multi-Sensor Data Fusion.” *Sensors* 20, no. 3 (2020): 919.

[4] Ramezani, Milad, Georgi Tinchev, Egor Iuganov, and Maurice Fallon. ”Online LiDAR-SLAM for Legged Robots with Robust Registration and Deep-Learned Loop Closure.” *arXiv preprint arXiv*:2001.10249 (2020).

[5] Montemerlo, M. ”A Factored Solution to the Simultaneous Localization and Mapping Problem with Unknown Data Association.” *Ph. D. thesis*, Carnegie Mellon University (2003).

[6] Loo, Shing Yan, Syamsiah Mashohor, Sai Hong Tang, and Hong Zhang. ”DeepRelativeFusion: Dense Monocular SLAM using Single-Image Relative Depth Prediction.” *arXiv preprint arXiv*:2006.04047 (2020).

[7] Shakhatreh, Hazim, Ahmad H. Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah, and Mohsen Guizani. ”Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges.” *Ieee Access* 7 (2019): 48572-48634.

[8] Mughal, Umair Ahmad, Jiao Xiao, Ishtiaq Ahmad, and KyungHi Chang. ”Cooperative resource management for C-V2I communications in a dense urban environment.” *Vehicular Communications* 26 (2020): 100282.

[9] Mughal, Umair Ahmad, Ishtiaq Ahmad, and KyungHi Chang. ”Virtual cells operation for 5G V2X communications.” *In Proceedings of KICS* (2019): 1486-1487.

[10] Shakoor, Shanza, Zeeshan Kaleem, Muhammad Iram Baig, Omer Chughtai, Trung Q. Duong, and Long D. Nguyen. ”Role of UAVs in public safety communications: Energy efficiency perspective.” *IEEE Access* 7 (2019): 140665-140679.

[11] Wen, Shuhuan, Yanfang Zhao, Xiao Yuan, Zongtao Wang, Dan Zhang, and Luigi Manfredi. ”Path planning for active SLAM based on deep reinforcement learning under unknown environments.” *Intelligent Service Robotics*, pp. 1-10, (2020).

[12] Kalogeiton, Vicky S., Konstantinos Ioannidis, G. Ch Sirakoulis, and Elias B. Kosmatopoulos. ”Real-time active SLAM and obstacle avoidance for an autonomous robot based on stereo vision.” *Cybernetics and Systems*, vol. 50, no. 3, pp. 239-260, (2019).

[13] Doitsidis, Lefteris, Stephan Weiss, Alessandro Renzaglia, Markus W. Achtelik, Elias Kosmatopoulos, Roland Siegwart, and Davide Scaramuzza. ”Optimal surveillance coverage for teams of micro aerial vehicles in GPS-denied environments using onboard vision.” *Autonomous Robots*, vol. 33, no. 1-2, pp. 173-188, (2012) .

[14] Alzugaray, Ignacio, Lucas Teixeira, and Margarita Chli. ”Short-term UAV path-planning with monocular-inertial SLAM in the loop.” *In 2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2739-2746. IEEE, 2017.

[15] Sánchez-García, Jesús, D. G. Reina, and S. L. Toral. ”A distributed PSO-based exploration algorithm for a UAV network assisting a disaster scenario.” *Future Generation Computer Systems* 90 (2019): 129-148.

[16] Shi, Weisen, Junling Li, Wenchao Xu, Haibo Zhou, Ning Zhang, Shan Zhang, and Xuemin Shen. ”Multiple drone-cell deployment analyses and optimization in drone assisted radio access networks.” *IEEE Access* 6 (2018): 12518-12529.

[17] Ghamry, Khaled A., Mohamed A. Kamel, and Youmin Zhang. ”Multiple UAVs in forest fire fighting mission using particle swarm optimization.” In 2017 *International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 1404-1409. IEEE, 2017.

[18] Cheng, Ze, Ergang Wang, Yixin Tang, and Yucui Wang. ”Real-time Path Planning Strategy for UAV Based on Improved Particle Swarm Optimization.” *JCP 9*, no. 1 (2014): 209-214.

[19] Bircher, Andreas, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. ”Receding horizon” next-best-view” planner for 3d exploration.” *In 2016 IEEE international conference on robotics and automation (ICRA)*, pp. 1462-1468. IEEE, 2016.

[20] Hu, Teng, Ishtiaq Ahmad, M. S. M. Alamgir, and KyungHi Chang. ”3D Optimal Surveillance Trajectory Planning for Multiple UAVs by Using Particle Swarm Optimization with Surveillance Area Priority.” *IEEE Access* (2020).

[21] Pattanayak, Suvranshu, and Bibhuti Bhusan Choudhury. ”Modified crash-minimization path designing approach for autonomous material handling robot.” Evolutionary Intelligence (2019): 1-14.

[22] Yu, Xiaobing, Chenliang Li, and JiaFang Zhou. ”A constrained differential evolution algorithm to solve UAV path planning in disaster scenarios.” Knowledge-Based Systems 204 (2020): 106209.

[23] Dasdemir, Erdi, Murat Köksalan, and Diclehan Tezcaner Öztürk. ”A flexible reference point-based multi-objective evolutionary algorithm: An application to the UAV route planning problem.” Computers and Operations Research 114 (2020): 104811.

[24] Qu, Chengzhi, Wendong Gai, Jing Zhang, and Maiying Zhong. ”A novel hybrid grey wolf optimizer algorithm for unmanned aerial vehicle (UAV) path planning.” Knowledge-Based Systems (2020): 105530.

[25] Atencia, Cristian Ramirez, Javier Del Ser, and David Camacho. ”Weighted strategies to guide a multi-objective evolutionary algorithm for multi-UAV mission planning.” Swarm and evolutionary computation 44 (2019): 480-495.

[26] Zhou, Haoyin, Tao Zhang, and Jayender Jagadeesan. ”Re-weighting and 1-Point RANSAC-Based P n n P Solution to Handle Outliers.” *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 12, pp. 3022-3033, (2018).

[27] Kneip, Laurent, Davide Scaramuzza, and Roland Siegwart. ”A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation.” *In CVPR 2011*, pp. 2969-2976. IEEE, 2011.

[28] Chum, Ondrej, and Jiri Matas. ”Matching with PROSAC-progressive sample consensus.” *In 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1, pp. 220-226. *IEEE*, 2005.

[29] Bellavia, F., D. Tegolo, and Cf Valenti. ”Improving Harris corner selection strategy.” *IET Computer Vision*, vol. 5, no. 2, pp. 87-96, (2011).

[30] Canny, John. ”A computational approach to edge detection.” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 6, pp. 679-698, (1986).

[31] Karami, Ebrahim, Siva Prasad, and Mohamed Shehata. ”Image matching using SIFT, SURF, BRIEF and ORB: performance comparison for distorted images.” *arXiv preprint arXiv*:1710.02726 (2017).

[32] Ohta, Yuichi, and Takeo Kanade. ”Stereo by intra-and inter-scanline search using dynamic programming.” *IEEE Transactions on pattern analysis and machine intelligence* , vol. 2, pp. 139-154 (1985).

[33] Lowe, David G. ”Fitting parameterized three-dimensional models to images.” *IEEE transactions on pattern analysis and machine intelligence*, vol. 13, no. 5, pp. 441-450, (1991).

[34] Zheng, Changwen, Lei Li, Fanjiang Xu, Fuchun Sun, and Mingyue Ding. ”Evolutionary route planner for unmanned air vehicles.” *IEEE Transactions on robotics*, vol. 21, no. 4, pp. 609-620 ,(2005).

[35] Kennedy, James, and Russell Eberhart. ”Particle swarm optimization.” *In Proceedings of ICNN’95-International Conference on Neural Networks*, vol. 4, pp. 1942-1948. IEEE, 1995.

[36] Roberge, Vincent, Mohammed Tarbouchi, and Gilles Labonté. ”Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning.” *IEEE Transactions on industrial informatics*, vol. 9, no. 1, pp. 132-141,(2012).

[37] Fonder, Michaël, and Marc Van Droogenbroeck. ”Mid-Air: A multi-modal dataset for extremely low altitude drone flights.” *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0-0. 2019.

[38] Guo, Xiangkun, Sihuan Chen, Hu Lin, Hongliang Wang, and Shuai Wang. ”A 3D terrain meshing method based on discrete point cloud.” *In 2017 IEEE International Conference on Information and Automation (ICIA)*, pp. 12-17. IEEE, 2017.

Umair Ahmad Mughal received the B.S. in electrical engineering from the University of Engineering and Technology, Peshawar, Pakistan, in 2015, and the Master degree in electrical and computer engineering from INHA University, Korea. From 2020 to 2021, he was with the Oceanic IT Convergence Technology Research Institute, Hoseo University, Korea.

His research interests include Cellular Vehicle-to- Everything (C-V2X) with 5G communications, interference management, UAVs path planning, underwater acoustic communication, machine learning, and AI applications with SLAM technology. He is a recipient of the Jungseok International Scholarship to pursue his M.S. studies at Inha University due to his excellent academic career.

ISHTIAQ AHMAD received the B.S. degree in Electrical Engineering from the University of Engineering and Technology (UET), Peshawar, Pakistan, in 2007, and the M.S. and Ph.D. degree in Electronic Engineering from Inha University, Korea, in 2014 and 2019, respectively.

From 2007 to 2008, he was a BSS Engineer with the O&M Department, Zong Pakistan. Since 2009, he has been a Lecturer with the Faculty of Engineering and Technology (FET), Gomal University, Pakistan. He has authored several International journals and IEEE conference papers and also holds U.S. and Korean patents.

His research interests include interference management in 3GPP LTE-A & 5G systems, public safety and mobile ad-hoc networks (especially for UAV), cellular-V2X technology, and maritime & underwater communications, and applications of Artificial Intelligence technologies. He was a recipient of the Jungseok International Scholarship to pursue his M.S. and Ph.D. degrees at Inha University, due to his excellent academic career. He received the Outstanding Research Award and Excellent Student Award from the Inha University, in 2019, for his excellence of journal publication and outstanding research achievements. He is currently working at Electrical Engineering Department, Faculty of Engineering and Technology, Gomal University, Pakistan.

KYUNGHI CHANG (SM’98) received the B.S. and M.S. degrees in electronics engineering from Yonsei University, Seoul, South Korea, in 1985 and 1987, respectively, and the Ph.D. degree in electrical engineering from Texas A&M University, College Station, TX, USA, in 1992. From 1989 to 1990, he was with the Samsung Advanced Institute of Technology (SAIT) as a member of the research staff and was involved in digital signal processing system design. From 1992 to 2003, he was with the Electronics and Telecommunications Research Institute (ETRI) as a Principal Member of the technical staff, where he led the design teams involved in the WCDMA UE modem and 4G radio transmission technology (RTT). He is currently with the Electrical and Computer Engineering Department, Inha University.

His research interests include radio transmission technology in 3GPP LTE & 5G systems, public safety and mobile ad-hoc networks (especially for UAV), cellular-V2X technology, maritime & underwater communications, applications of AI technologies, and NTN(Non-Terrestrial Network) & network intelligence for 6G. He was a recipient of the LG Academic Awards, in 2006, the Haedong Best Paper Awards, in 2007, the IEEE ComSoc Best Paper Awards, in 2008, the Haedong Academic Awards, in 2010, the SKT SafeNet Best Idea Awards, in 2015, the KICS Outstanding Academic Society Awards, in 2018 and in 2019, and MSIT Minister’s Commendation and KICS Fellow in 2020. He is currently a Chairman of the Expert Committee in SafeNet Forum, and Mobile and Automotive Convergence Committee in 5G Forum. He has served as an Editor-in-Chief and an Executive Director for the Journal of Korean Institute of Communications and Information Sciences (KICS), from 2010 to 2012 and in 2013, respectively, and as a Vice President at the KICS from 2017 to 2018. He has also served as an Editor of ITU-R TG8/1 IMT.MOD.