Evolutionary Online Swarm Path Planning for UAV Formation Transition in Dynamic Environments

Jyun-Siang Huang
Department of Computer Science
National Tsing Hua University
Hsinchu 300044, Taiwan
s111033105@m111.nthu.edu.tw

Jia-He Tee
Department of Computer Science
National Tsing Hua University
Hsinchu 300044, Taiwan
s112062806@m112.nthu.edu.tw

Chuan-Kang Ting
Department of Computer Science
National Tsing Hua University
Hsinchu 300044, Taiwan
ckting@cs.nthu.edu.tw

***Abstract*—Unmanned aerial vehicle (UAV) swarms have attracted increasing attention for cooperative missions such as surveillance, environmental monitoring, and formation flight. In formation transition tasks, a swarm must move from an initial formation to a prescribed target formation while respecting kinematic limits, maintaining safe inter-UAV separation, and avoiding environmental hazards. The transition tasks become more challenging in dynamic environments, where observable obstacles must be avoided proactively, and unobserved disturbances can perturb executed motion and degrade formation accuracy unless timely path planning is performed. This study proposes the evolutionary online swarm path planning (EOSPP) framework, which formulates swarm navigation as a sequence of time-bounded constrained optimization problems solved along the flight. The EOSPP employs evolution strategies (ES) as the backbone optimizer to generate control vectors for each UAV within a fixed computation budget, explicitly enforcing speed limits, minimum inter-UAV separation, and obstacle clearance during optimization. Unobserved wind disturbances are modeled as execution-time perturbations and are compensated through repeated online path planning. Experiments in a bounded two-dimensional workspace show that EOSPP achieves collision-free convergence to the target formation in the presence of uniformly sampled obstacles and maintains convergence under unobserved wind gusts, demonstrating the practicality of evolutionary online path planning for safety-critical swarm formation transitions.**

***Index Terms*—UAV swarm, formation transition, online path planning, constrained optimization, evolution strategies, partially observable dynamic environment.**

I. Introduction

Unmanned aerial vehicle (UAV) swarms have attracted increasing attention for cooperative missions such as surveillance, environmental monitoring, search-and-rescue, and formation flight [1]. In formation transition tasks, a swarm is required to move from an initial formation to a prescribed target formation while respecting kinematic limits, maintaining safe inter-UAV separation, and avoiding environmental hazards such as obstacles. These requirements become more challenging when the environment is dynamic. Observable hazards such as obstacles must be avoided proactively, while unobserved disturbances such as wind can perturb the executed motion and gradually degrade formation accuracy unless corrective replanning is performed.

A variety of path planning approaches have been studied for UAV navigation and coordination. Classical graph-based planners such as A* [2], D* [3], and D* Lite [4] formulate navigation as shortest-path search over discretized spaces and provide strong theoretical properties. However, in dynamic and multi-agent settings, frequent replanning can incur substantial computational cost, and the discretization required for tractability can limit real-time responsiveness. Sampling-based planners, including rapidly-exploring random tree (RRT) [5] and RRT* [6], explore continuous spaces via randomized sampling and offer probabilistic completeness or asymptotic optimality. Nevertheless, they typically produce geometric paths that require replanning or tree repair when conditions change, and extending them to tightly coupled swarms is nontrivial because inter-agent constraints significantly increase the complexity of feasible planning.

Model predictive control (MPC) provides an online alternative by repeatedly solving constrained optimization problems over a receding horizon [7]. Variants such as nonlinear MPC (NMPC) [8], stochastic MPC (SMPC) [9], and distributed MPC (DMPC) [10] have been applied to UAV navigation and coordination. Solving nonlinear constrained optimization problems at each control cycle, however, can become computationally demanding as the swarm size increases and coupling constraints intensify.

Evolutionary algorithms (EAs) are powerful optimization techniques inspired by Darwinian evolution. Owing to their gradient-free search capability and robustness under complex, nonconvex, and constrained landscapes, EAs have been widely adopted in robotics, control, and autonomous systems [11], [12]. In particular, EC-based approaches have been explored for UAV path planning because they can naturally accommodate nonlinear objectives and constraints. Early studies typically evolved waypoint-based trajectories offline [13], and later works extended evolutionary planning to multi-UAV settings [14]–[16]. Despite their global search capability, many existing evolutionary approaches remain offline, which limits adaptability under disturbances and hazards that occur during execution.

To address the need for real-time responsiveness in UAV swarm formation transition, this study proposes an evolutionary online swarm path planning (EOSPP) framework for a UAV swarm operating in a partially observable dynamic environment. The EOSPP framework formulates formation transition as a constrained online optimization problem solved at each planning time step along the flight to compute swarm-level control vectors, where each control vector consists of the speed and heading angle of a UAV. In this study, evolution strategies (ES) [17] is adopted as the backbone optimizer to perform optimization within a fixed computational budget at every planning step. Observable obstacle information is incorporated into constraint checking and fitness evaluation to enable anticipative collision avoidance, while unobserved wind disturbance perturbs the executed motion and is compensated through repeated replanning based on the updated swarm state. A series of experiments were conducted to validate the effectiveness and efficiency of the EOSPP framework. Overall, the EOSPP framework provides a practical approach for real-time formation transition with explicit constraint handling and online correction under disturbances.

The remainder of this paper is organized as follows: Section II reviews the related work. Section III elaborates the design of the EOSPP framework. Section IV presents the experimental settings and results. Finally, a concluding remark is given in Section V.

II. Related Work

UAV swarm formation transition and navigation have been studied from multiple perspectives, including graph-based and sampling-based motion planning, optimal control, deterministic assignment, and evolutionary optimization. Graph-based planners formulate navigation as a shortest-path search over a discretized state space. A* [2] uses heuristic guidance to expand nodes efficiently and guarantees optimality under admissible heuristics, while D* [3] and D* Lite [4] improve practicality in changing environments through incremental replanning. Despite these advances, repeated graph exploration remains a core bottleneck. For swarm navigation, joint-state representations and inter-agent collision constraints substantially enlarge the search space, and the replanning cost can grow rapidly, making strict real-time execution difficult without aggressive simplifications.

Sampling-based planners address continuous configuration spaces by constructing trees through randomized sampling. RRT [5] emphasizes exploration, and RRT* [6] incorporates rewiring to achieve asymptotic optimality. Extensions such as RRTX [18] further enable replanning by maintaining and repairing trees when costs or obstacles change. However, maintaining large trees and repairing them online can still incur nontrivial overhead. In addition, these methods typically output geometric paths and rely on separate tracking controllers. When disturbances occur during execution or when multi-agent coupling is strong, replanning latency and constraint handling can become critical concerns.

MPC provides an online planning paradigm by repeatedly solving constrained finite-horizon optimization problems in a receding-horizon manner [7]. Variants including NMPC [8], SMPC [9], and distributed MPC (DMPC) [10] have been applied to UAV navigation and coordination. These approaches offer principled constraint handling and closed-loop feedback. Nevertheless, solving nonlinear constrained optimization problems at every control cycle can be computationally expensive, and the burden increases with swarm size because coupling constraints typically increase both decision dimensionality and solver complexity. As a result, achieving strict real-time feasibility often requires simplifying the dynamics, reducing the horizon length, or relaxing the coupling, which may compromise adaptability or safety margins.

On the other hand, a standard engineering baseline for formation transition decomposes the problem into target assignment and trajectory execution. The Hungarian algorithm [19] is widely used to compute an optimal one-to-one assignment that minimizes the total travel distance between the current UAV positions and the target formation points. This assignment step effectively reduces unnecessary travel and is frequently paired with simple motion policies, such as directing each UAV to its assigned target. However, when assignment and control are computed only once before execution, the resulting plan can be brittle in the face of hazards encountered along the path or disturbances that cause drift.

EAs have also been extensively applied to UAV path planning due to their ability to optimize nonlinear objectives under complex constraints without requiring gradients. Early studies evolved waypoint-based trajectories offline using EAs [13]. To improve scalability, cooperative coevolutionary algorithms decompose high-dimensional decision vectors into interacting components and evolve subpopulations cooperatively [14], [15]. While these approaches alleviate dimensionality growth and can achieve high-quality solutions, they are commonly designed for offline optimization or for long planning horizons and do not explicitly target time-bounded decision-making in online replanning settings. Consequently, the ability to produce feasible control decisions reliably within a small computation budget at every replanning step remains underexplored. Overall, these studies motivate a time-bounded online optimization framework that recomputes swarm control decisions at every replanning step, incorporates observable hazards directly into feasibility checks and evaluation, and compensates for unobserved disturbances through repeated replanning based on updated swarm states.

III. Methodology

This study proposes the EOSPP framework for the UAV swarm operating in a partially observable dynamic environment. ES is adopted as the backbone optimizer to enable efficient online planning toward the target formation while satisfying safety constraints. The following describes the EOSPP framework in detail.

A. Problem Formulation

In this study, the considered scenario involves a swarm of N autonomous UAVs required to move from an initial formation C_{start} to a target formation C_{target} in a two-dimensional Euclidean space. During the flight, environmental conditions may vary over time, such as the appearance of obstacles along the path or the occurrence of wind gusts, which necessitate

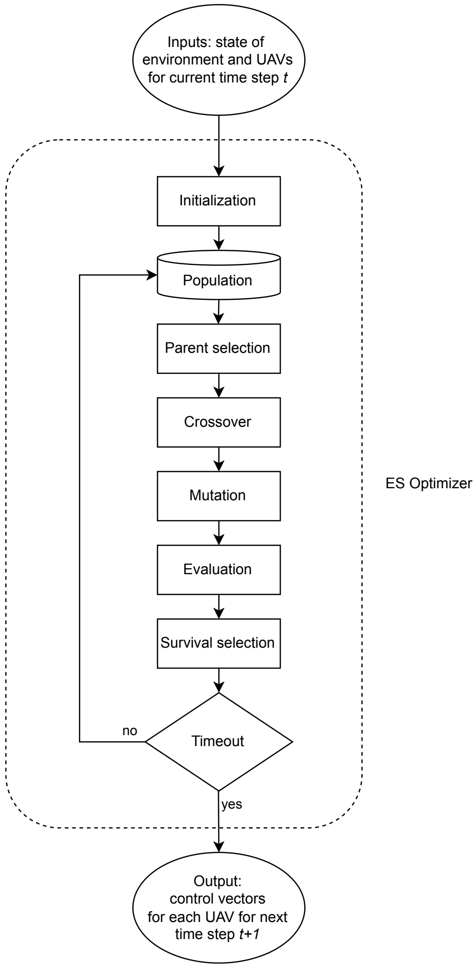


Fig. 1: Workflow of the EOSPP framework.

online adjustments of UAV trajectories. Accordingly, the path planning task is formulated as a real-time constrained online optimization problem in a partially observable dynamic environment, where an optimization procedure is executed at every discrete time step to compute control vectors for the swarm. The flight from C_{start} to C_{target} is divided into uniform planning steps indexed by $t \in \{0, 1, 2, \ldots\}$, in which each step corresponds to a fixed planning interval of duration τ, such that the transition from step t to $t+1$ represents the swarm motion executed over a time span of τ.

Mathematically, the UAV swarm can be modeled as a set of N particles. For each UAV $i \in \{1, \ldots, N\}$, the position at time step t is denoted by a position vector $[x_i(t),\, y_i(t)]^T$. The control action of UAV i at time step t is represented by a control vector $\boldsymbol{u}_i(t) = \big[s_i(t),\, \alpha_i(t)\big]^T$, where $s_i(t)$ is the speed and $\alpha_i(t)$ is the heading angle. For notational convenience, the corresponding commanded translational velocity vector is defined as $\boldsymbol{v}_i(t) = s_i(t)\big[\cos\alpha_i(t),\, \sin\alpha_i(t)\big]^T$.

To model dynamic environmental disturbances, a time-varying wind gust is incorporated into the executed motion as an additive velocity term. The wind disturbance is assumed to be spatially uniform across the operating region and identical for all UAVs at a given time step. Let $\boldsymbol{v}_g(t) = [v_g^x(t),\, v_g^y(t)]^T$ denote the wind-induced velocity vector at time step t. Since $\boldsymbol{v}_g(t)$ is not observable during online optimization, candidate control vectors are evaluated under a kinematic model without wind disturbance. Accordingly, two state transitions are distinguished. Given a fixed planning interval τ, the predicted next position used for optimization is defined as

$$\hat{\boldsymbol{p}}_i(t+1) = \boldsymbol{p}_i(t) + \tau\, \boldsymbol{v}_i(t), \tag{1}$$

while the executed next position is affected by the unobserved wind disturbance and follows

$$\boldsymbol{p}_i(t+1) = \boldsymbol{p}_i(t) + \tau\big(\boldsymbol{v}_i(t) + \boldsymbol{v}_g(t)\big). \tag{2}$$

At each time step t, the wind disturbance $\boldsymbol{v}_g(t)$ is unobserved during optimization and is reflected only in the realized state transition. As a result, the online planning process serves as a feedback correction mechanism, where deviations from the nominal trajectory induced by wind are compensated in subsequent time steps based on the updated swarm state.

In the target formation C_{target}, a set of N target points is specified. Let $\boldsymbol{q}_i$ denotes the target point assigned to UAV i. The assignment of the target points is determined using the Hungarian algorithm [19] to minimize the total assignment cost, defined by the sum of Euclidean distances between the current UAV positions and the candidate target points. This assignment step ensures that the swarm transition toward C_{target} starts from a distance-efficient matching, thereby reducing unnecessary travel.

At each time step t, an online optimization problem is solved within a finite time budget corresponding to one planning interval τ. The decision variables are the control vectors of all UAVs at time step t, equivalently the speed and heading angle $\{s_i(t), \alpha_i(t)\}_{i=1}^N$. The objective is to minimize the aggregate distance between the predicted next position $\hat{\boldsymbol{p}}_i(t+1)$ and the assigned target point $\boldsymbol{q}_i$ of each UAV i, which leads to the following constrained optimization problem. Let $\mathcal{N} = \{1, \ldots, N\}$ denote the set of UAV indices, and $\mathcal{P} = \{(i,j) \in \mathcal{N} \times \mathcal{N} \mid i < j\}$ denote the set of UAV pairs used in the inter-UAV safety constraint. The constraints are enforced for all $i \in \mathcal{N}$, all $(i,j) \in \mathcal{P}$, and all $o \in \mathcal{O}(t)$.

$$\begin{aligned}
\min_{\{s_i(t),\, \alpha_i(t)\}_{i=1}^N} \quad & \sum_{i=1}^{N} \|\boldsymbol{q}_i - \hat{\boldsymbol{p}}_i(t+1)\| \\
\text{s.t.} \quad & \hat{\boldsymbol{p}}_i(t+1) = \boldsymbol{p}_i(t) + \tau\, \boldsymbol{v}_i(t) \\
& \boldsymbol{v}_i(t) = s_i(t) \begin{bmatrix} \cos\alpha_i(t) \\ \sin\alpha_i(t) \end{bmatrix} \\
& 0 \le s_i(t) \le s_{\max} \\
& 0 \le \alpha_i(t) < 2\pi \\
& \|\hat{\boldsymbol{p}}_i(t+1) - \hat{\boldsymbol{p}}_j(t+1)\| \ge d_{\text{uav}} \\
& \|\hat{\boldsymbol{p}}_i(t+1) - \boldsymbol{p}_o\| \ge r_o + d_{\text{obs}}
\end{aligned}$$

The above constraints ensure feasibility and safety requirements for real-time swarm navigation. The speed constraint bounds the speed of each UAV by a maximum speed $s_{\max}$. The inter-UAV safety constraint enforces a minimum safety separation d_{uav} between any pair of UAVs, thereby preventing inter-UAV collisions during formation transitions. Furthermore, obstacle avoidance is imposed with respect to a set of obstacles $\mathcal{O}(t)$, where each obstacle $o \in \mathcal{O}(t)$ is described by a center position $\boldsymbol{p}_o = [x_o,\, y_o]^T$ and a radius r_o. A clearance margin d_{obs} is introduced so that each UAV maintains a safe distance from obstacle boundaries. The obstacle avoidance constraint is enforced with respect to the predicted next positions $\hat{\boldsymbol{p}}_i(t+1)$ using the observed obstacle set $\mathcal{O}(t)$, while wind-induced deviations are corrected through repeated planning across time steps. Under this formulation, the control vector of each UAV at each time step is obtained by solving the above constrained optimization problem, enabling the swarm to approach C_{target} progressively while reacting online to dynamic environmental changes.

B. EOSPP Framework

The workflow of the EOSPP framework is shown in Figure 1. At each discrete planning step t, the current swarm state and observable environmental information are collected, including the UAV positions $\{\boldsymbol{p}_i(t)\}_{i\in\mathcal{N}}$ and the obstacle set $\mathcal{O}(t)$. These observations are provided to an ES optimizer, which solves the constrained online optimization problem in Section III-A to determine the control vectors $\{\boldsymbol{u}_i(t)\}_{i\in\mathcal{N}}$.

Given the current state at time step t, the ES-based optimizer initializes a candidate population of control vectors and iteratively refines it within a fixed computation budget determined by the planning interval τ. Specifically, the population is updated through repeated parent selection, crossover, mutation, and fitness evaluation, and the survival selection stage preserves higher-quality candidates for subsequent iterations. The optimization loop terminates when the time budget is exhausted, and the best-so-far control vectors are returned as the output for time step t.

After the control vectors are applied, the swarm state is advanced to the next step. During optimization and fitness evaluation, candidate solutions are propagated using the kinematic model in (1) without wind to obtain the predicted next positions $\{\hat{\boldsymbol{p}}_i(t+1)\}_{i\in\mathcal{N}}$. In execution, the realized next positions $\{\boldsymbol{p}_i(t+1)\}_{i\in\mathcal{N}}$ follow the wind-disturbed kinematic model in (2), where the unobserved wind-induced velocity $\boldsymbol{v}_g(t)$ perturbs the commanded motion. The realized positions $\{\boldsymbol{p}_i(t+1)\}_{i\in\mathcal{N}}$ then serve as the initial swarm state for planning optimization at time step $t+1$, enabling continuous online correction under unobserved disturbances and anticipative avoidance of observed obstacles throughout the flight toward the target formation C_{target}.

C. ES Optimizer

1) Representation: For the ES optimizer employed in the EOSPP framework, each chromosome encodes a complete candidate control solution for the UAV swarm at a given time step. A chromosome $\boldsymbol{c}$ is constructed by concatenating the control variables and the associated strategy parameters of all N UAVs, and is defined as

$$\boldsymbol{c} = (\underbrace{v_1, \alpha_1, \zeta_1, \sigma_1}_{\text{UAV } 1},\ \dots,\underbrace{v_N, \alpha_N, \zeta_N, \sigma_N}_{\text{UAV } N}).$$

For UAV $i \in \{1, \dots, N\}$, the corresponding four genes $(v_i, \alpha_i, \zeta_i, \sigma_i)$ are interpreted as follows.

- Speed level (v_i): $v_i \in \{0, 1, 2, 3\} \subset \mathbb{N}$ is an integer-encoded speed level, which is mapped to a physical speed from the set $\{0,\ 0.75,\ 1.5,\ s_{\max}\}$.
- Heading angle (α_i): $\alpha_i \in [0, 2\pi) \subset \mathbb{R}$ denotes the forward heading angle of UAV i, which determines the direction of motion at the current time step.
- Integer-step mutation parameter (ζ_i): $\zeta_i \in \mathbb{R}$ controls the mutation step size associated with the integer-encoded variable v_i.
- Real-step mutation parameter (σ_i): $\sigma_i \in \mathbb{R}$ controls the mutation step size associated with the real-valued variable α_i.

2) Crossover: Crossover is applied to generate offspring by recombining information from two selected parent chromosomes. Since chromosome $\boldsymbol{c}$ contains both decision variables $\{v_i, \alpha_i\}$ and strategy parameters $\{\zeta_i, \sigma_i\}$, global crossover with different operators is adopted:

- Uniform crossover for control variables (v_i, α_i): For each gene, the offspring inherits the corresponding value from either parent with equal probability 0.5.
- Whole arithmetic crossover for strategy parameters (ζ_i, σ_i): With $\alpha = 0.2$, the offspring parameters are computed by

$$\sigma_i' = (1-\alpha)\sigma_i^{(1)} + \alpha\sigma_i^{(2)} \ \text{ and } \ \zeta_i' = (1-\alpha)\zeta_i^{(1)} + \alpha\zeta_i^{(2)}.$$

3) Mutation: To handle the mixed discrete continuous decision space arising from the speed level v_i and the heading angle α_i, the mixed integer evolution strategies (MIES) [20] is adopted. MIES applies an uncorrelated mutation scheme with self adaptation, enabling integer and real variables to be evolved jointly. In particular, a geometric-based mutation operator is used for the integer variable v_i, while a Gaussian-based mutation operator is used for the real valued variable α_i. The mutation strengths are controlled by the strategy parameters ζ_i and σ_i. The self-adaptation of these strategy parameters is driven by a global learning rate γ and a local learning rate γ'. These rates are set as $\gamma = 1/\sqrt{2n}$ and $\gamma' = 1/\sqrt{2\sqrt{n}}$, where n denotes the dimension of the respective decision variable space. The global learning rate γ controls the overall mutation scale for an individual, whereas the local learning rate γ' allows for coordinate-specific fine-tuning of the step sizes.

- Gaussian mutation for α_i: The heading angle $\alpha_i \in [0, 2\pi)$ is mutated by adding a Gaussian perturbation with a self adapted step size. Specifically, the step size σ_i is updated first as $\sigma_i' = \sigma_i \cdot \exp(\gamma\mathcal{N}(0,1) + \gamma'\mathcal{N}_i(0,1))$, and the decision variable is then mutated by

$$\alpha_i' = \alpha_i + \sigma_i' \cdot \mathcal{N}_i(0,1),$$

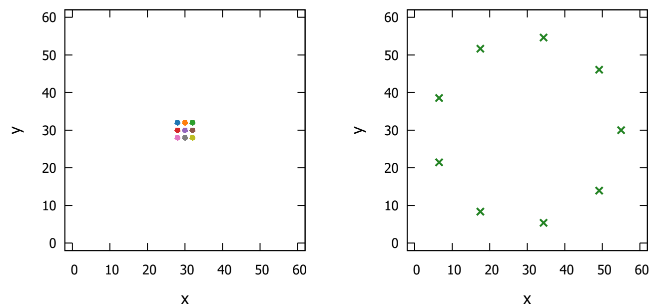
where $\mathcal{N}(0,1)$ denotes a Gaussian random variable shared by all dimensions of the offspring, and $\mathcal{N}_i(0,1)$ denotes an independent Gaussian random variable associated with UAV i.

- Integer mutation for v_i: The speed level $v_i \in \{0,1,2,3\}$ is mutated using the geometric-based operator in MIES. Specifically, the step size ζ_i is updated first as $\zeta_i' = \zeta_i \cdot \exp(\gamma\,\mathcal{N}(0,1) + \gamma'\,\mathcal{N}_i(0,1))$. Then, the decision variable is mutated as
$$v_i' = v_i + (G_1 - G_2),$$
where G_1 and G_2 are independent geometrically distributed random variables whose distribution width is controlled by the step size ζ_i. The difference $(G_1 - G_2)$ produces a symmetric heavy tailed perturbation, which supports occasional larger jumps while maintaining local search capability.

4) Fitness Evaluation: During fitness evaluation, chromosomes that violate any hard constraint are discarded. The remaining feasible chromosomes are evaluated by a distance-based cost with an exponential transformation to promote rapid convergence, augmented with a penalty that discourages near-collision behaviors.

Let $\boldsymbol{q}_i$ denote the target point assigned to UAV i, and the distance of UAV i to the assigned target point at time step $t+1$ is defined as $d_i = \|\boldsymbol{q}_i - \hat{\boldsymbol{p}}_i(t+1)\|$. Let $\mathcal{N} = \{1,\ldots,N\}$ denote the set of UAV indices, and $\mathcal{P} = \{(i,j) \in \mathcal{N}\times\mathcal{N} \mid i < j\}$ denote the set of UAV pairs. The fitness of a chromosome $\boldsymbol{c}$ is defined as
$$f(\boldsymbol{c}) = J(\boldsymbol{c}) + w_{\mathrm{p}} P(\boldsymbol{c}),$$
where $w_{\mathrm{p}} > 0$ is a penalty weight. The objective term $J(\boldsymbol{c})$ is formulated by
$$J(\boldsymbol{c}) = \exp\left(\frac{1}{N}\sum_{i\in\mathcal{N}} d_i\right),$$
and the penalty term $P(\boldsymbol{c})$ aggregates proximity penalties for both UAV-to-UAV separation and UAV-to-obstacle clearance, such that
$$P(\boldsymbol{c}) = \sum_{(i,j)\in\mathcal{P}} \phi_{\mathrm{uav}}(d_{i,j}) + \sum_{i\in\mathcal{N}}\sum_{o\in\mathcal{O}(t)} \phi_{\mathrm{obs}}(d_{i,o}),$$
where $d_{i,j}$ denotes the minimum Euclidean distance between UAV i and UAV j during the next planning interval, determined along their predicted one-step trajectories from $\boldsymbol{p}_i(t)$ to $\hat{\boldsymbol{p}}_i(t+1)$ and from $\boldsymbol{p}_j(t)$ to $\hat{\boldsymbol{p}}_j(t+1)$, respectively. Similarly, $d_{i,o}$ denotes the minimum Euclidean distance between UAV i and the boundary of obstacle $o \in \mathcal{O}(t)$ during the next planning interval.

In $P(\boldsymbol{c})$, a buffer margin $\delta = 1$ is introduced so that the penalty is applied only when the predicted separation falls within δ of the corresponding safety threshold. Specifically, the UAV-to-UAV warning penalty is defined using the minimum safety separation d_{uav} as
$$\phi_{\mathrm{uav}}(d_{i,j}) = \begin{cases} \left(d_{\mathrm{uav}} + \delta - d_{i,j}\right)^2, & d_{\mathrm{uav}} < d_{i,j} < d_{\mathrm{uav}} + \delta, \\ 0, & d_{i,j} \geq d_{\mathrm{uav}} + \delta. \end{cases}$$



(a) Initial formation (b) Target formation

Fig. 2: Initial and target formation of the UAV swarm.

On the other hand, UAV-to-obstacle warning penalty is defined using the clearance margin d_{obs} as
$$\phi_{\mathrm{obs}}(d_{i,o}) = \begin{cases} \left(d_{\mathrm{obs}} + \delta - d_{i,o}\right)^2, & d_{\mathrm{obs}} < d_{i,o} < d_{\mathrm{obs}} + \delta, \\ 0, & d_{i,o} \geq d_{\mathrm{obs}} + \delta. \end{cases}$$
Under this design, selection favors chromosomes that reduce the predicted distance to the target formation quickly while maintaining conservative separation margins from other UAVs and obstacles.

IV. Experimental Results

This study conducts several experiments to validate the effectiveness of the EOSPP framework in a partially observable dynamic environment, integrated with ES as the backbone optimizer.

A. Experimental Settings

The experiments are conducted in a two-dimensional Euclidean space bounded by $[0,60] \times [0,60]$. A swarm of $N = 9$ UAVs is considered. The flight is discretized into uniform path planning time steps indexed by $t \in \{0,1,2,\ldots\}$, where each step corresponds to a planning interval of duration $\tau = 0.2s$. At each time step t, the ES-based optimizer optimizes control vectors $\{\boldsymbol{u}_i(t)\}_{i\in\mathcal{N}}$, where $\boldsymbol{u}_i(t) = [s_i(t),\, \alpha_i(t)]^T$, and the commanded translational velocity is $\boldsymbol{v}_i(t) = s_i(t)[\cos\alpha_i(t),\, \sin\alpha_i(t)]^T$.

The maximum speed is set to $s_{\max} = 2.0$. The minimum inter-UAV separation and obstacle clearance parameters are set to $d_{\mathrm{uav}} = 1.5$ and $d_{\mathrm{obs}} = 1.0$, respectively. The swarm is initialized as a 3×3 grid formation with edge length 2, centered at $(30,30)$, as shown in Figure 2a. The pattern of the target formation is defined as a circle centered at $(30,30)$ with radius 25, where the nine target positions are uniformly distributed along the circumference, as shown in Figure 2b. A one-to-one assignment between UAVs and target points is computed using the Hungarian algorithm [19] whenever a target formation is specified.

To demonstrate the necessity of online path planning, EOSPP is compared with a baseline strategy without online path planning. In the baseline, the Hungarian algorithm [19] is first used to determine the target assignment. Then, only a

TABLE I: Parameter settings.

Parameters	Value
Population size μ	30
Offspring subpopulation size λ	2μ
Parent selection	6-tournament
Mutation rate p_m	1.0
Crossover rate p_c	1.0
Survival selection	(μ, λ)
Termination	0.2s

single optimization is performed at the beginning of the flight to determine the control vectors. No real-time optimization or feedback-based adjustment is performed in subsequent time steps. In contrast, EOSPP performs time-bounded evolutionary optimization at every planning step. At each step t, the current swarm state $\{\boldsymbol{p}_i(t)\}_{i\in\mathcal{N}}$ and the observable obstacle set $\mathcal{O}(t)$ are provided to the ES optimizer. The ES optimizer searches for control vectors within the computation time budget associated with each planning interval τ =0.2s, and the best-so-far solution is immediately applied for execution. Due to the stochastic nature of EAs, each experimental scenario is repeated 30 times. Table I lists the ES optimizer settings.

In this study, two scenarios are designed to test different types of online adaptability: 1) obstacle scenario, and 2) wind scenario. In the obstacle scenario, a static circular obstacle of radius $r_o = 2.5$ is uniformly sampled at the two-dimensional Euclidean space bounded by $[0, 60] \times [0, 60]$. The obstacle is assumed observable and is included in constraint checking and fitness evaluation through $\mathcal{O}(t)$. In the wind scenario, a wind gust is injected as an additive velocity disturbance during execution. The wind is assumed spatially uniform and identical for all UAVs at a given time step. In this scenario, a constant wind $\boldsymbol{v}_g(t) = [0.25, 0.5]^T$ is applied starting from a random time step and remains active thereafter.

B. Results and Discussion

First, the effectiveness of the EOSPP framework is evaluated in the presence of a static obstacle, with the goal of verifying whether real-time path planning can preserve safety while guiding the UAV swarm toward the target formation under obstacle-induced constraints. Figures 3 shows the anytime behavior of the UAV swarm in the obstacle scenario, in comparison with that in the wind scenario (Figure 4). The performance is measured by the sum of Euclidean distances from all UAVs to their assigned target points at each time step. With online planning (blue curve), the aggregate distance decreases steadily and ultimately approaches zero, indicating that the swarm completes the formation transition while satisfying the safety constraints. In contrast, the baseline without online planning (red curve) only reduces the distance in the early stage and then terminates prematurely because at least one UAV collides with the obstacle.

Figures 5 and 6 visualize the swarm evolution at representative time steps with online path planning using EOSPP, and without online path planning, respectively. Under online path planning, the swarm expands outward from the compact initial

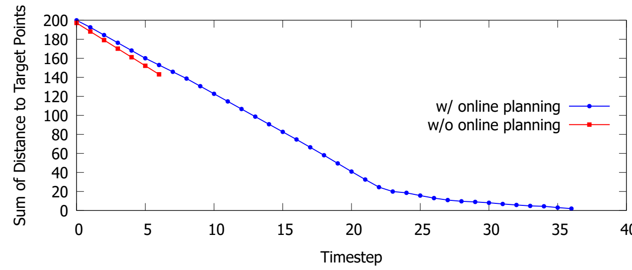


Fig. 3: Anytime behavior of the UAV swarm in the obstacle scenario.

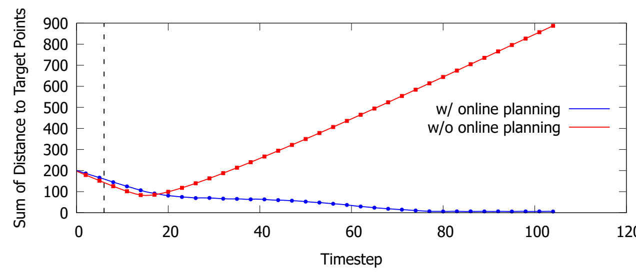
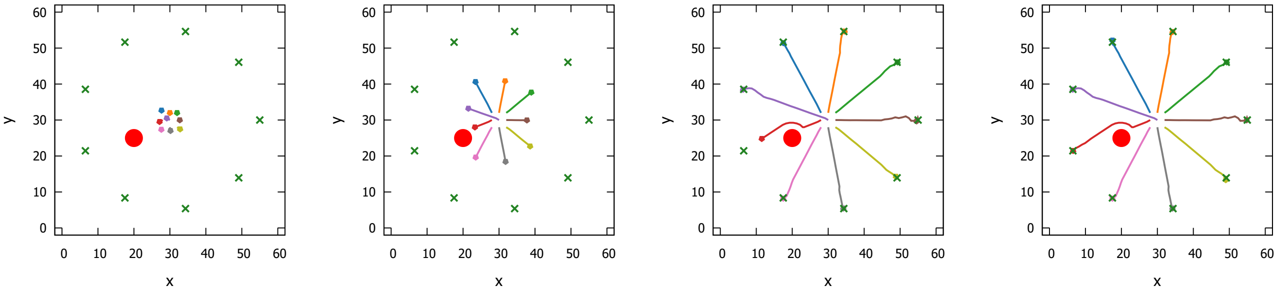


Fig. 4: Anytime behavior of the UAV swarm in the wind scenario.

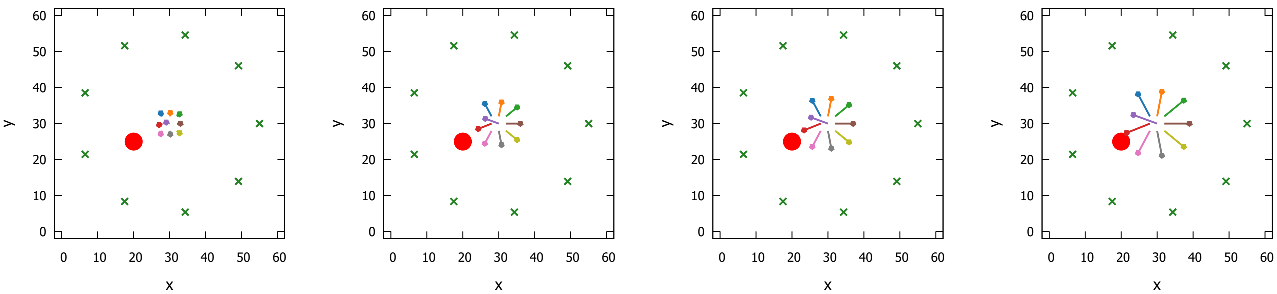
grid, locally adjusts headings to detour around the obstacle, and then resumes directed motion toward the assigned targets once the obstacle is cleared. The resulting trajectories are smooth and collision-free, indicating that EOSPP with ES optimizer can generate control vectors that anticipate the obstacle before contact based on the observed obstacle set. Overall, these results confirm the necessity of online planning for safety-critical formation transitions, where feasibility cannot be ensured by an initial plan alone, while iterative optimization with updated swarm states enables robust convergence to the target formation.

Next, the effectiveness of the EOSPP framework is evaluated in the presence of an unobserved wind disturbance that perturbs the executed motion of the UAV swarm. The goal is to examine whether continual planning can compensate for wind-induced drift and still achieve convergence to the target formation. Figure 4 shows the anytime behavior of the swarm in the wind scenario, where the dashed vertical line indicates the time step at which a constant wind is activated and remains present thereafter. Before the wind is applied, both methods reduce the sum of distances to the target points, reflecting that an initially optimized plan can drive the swarm toward the targets in a disturbance-free setting. After the wind onset, the baseline without online planning (red curve) exhibits a persistent increase in the distance, indicating that the fixed control strategy cannot counteract the accumulating drift and thus moves the swarm away from the intended formation. In contrast, EOSPP with online planning (blue curve) continues to decrease the distance over time and ultimately converges, demonstrating that the online path planning strategy provides an effective feedback correction mechanism under partially



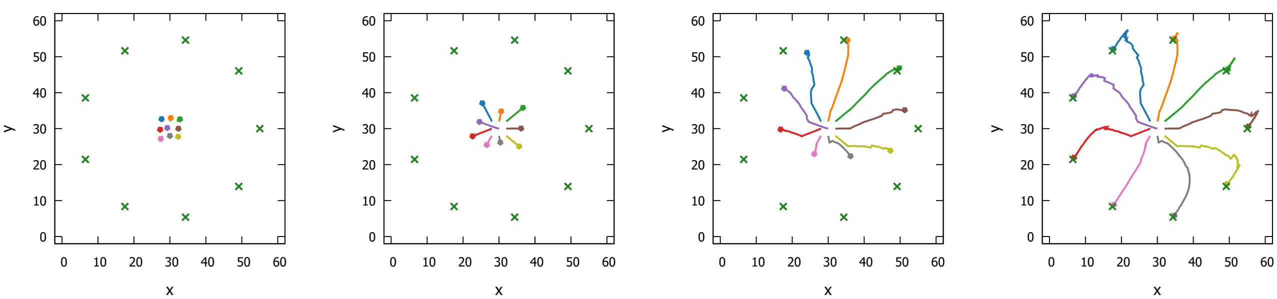
(a) $t = 0$ (b) $t = 9$ (c) $t = 30$ (d) $t = 36$

Fig. 5: Obstacle scenario: trajectories of the UAV swarm with online path planning



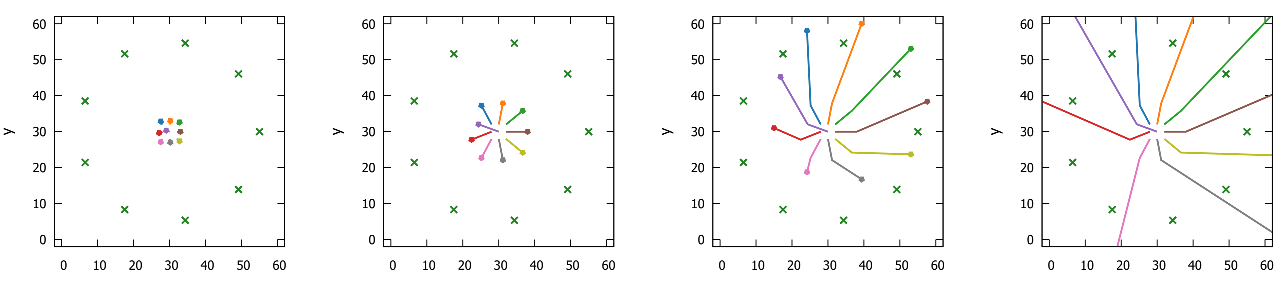
(a) $t = 0$ (b) $t = 3$ (c) $t = 4$ (d) $t = 6$

Fig. 6: Obstacle scenario: trajectories of the UAV swarm without online path planning



(a) $t = 0$ (b) $t = 5$ (c) $t = 19$ (d) $t = 104$

Fig. 7: Wind scenario: trajectories of the UAV swarm with online path planning



(a) $t = 0$ (b) $t = 5$ (c) $t = 19$ (d) $t = 104$

Fig. 8: Wind scenario: trajectories of the UAV swarm without online path planning

TABLE II: Performance of EOSPP over 30 independent runs per scenario.

Scenario	Success Rate	Path Length	#Timesteps
Obstacle	1.0	232.3 ± 8.2	38.3 ± 1.6
Wind	1.0	384.7 ± 19.4	104.0 ± 5.1

observable disturbances.

Figures 7 and 8 further visualize how the two strategies respond to wind. With online planning using EOSPP, the trajectories progressively adapt after the wind onset by adjusting headings and speeds so that the realized positions remain aligned with the target-directed motion despite the unmodeled disturbance. In comparison, the baseline trajectories deviate increasingly from the intended directions after the wind is applied, and the resulting drift accumulates because no corrective action is taken beyond the initial optimization. Overall, these results confirm that online path planning is necessary for maintaining formation-transition performance under unobserved environmental disturbances.

Table II summarizes the reliability of the EOSPP framework over 30 independent runs per scenario. A run is counted as a success if the swarm reaches the target formation within the predefined time-step limit under either randomly placed static obstacles or a uniform wind gust. EOSPP achieves a success rate of 1.0 in both scenarios, indicating consistent convergence across repeated trials. The small standard deviations in total path length and the number of required time steps further supporting the reliability of the EOSPP framework.

V. Conclusions

This study presents the EOSPP framework for UAV swarm path planning in partially observable and dynamic environments. In the EOSPP, swarm navigation is formulated as a sequence of time-bounded constrained optimization problems solved at each planning step along the flight using the current swarm state and observable obstacle information. An ES optimizer is utilized within a fixed computation budget to generate swarm-level control vectors that satisfy kinematic feasibility, inter-UAV separation, and obstacle clearance constraints. By repeatedly planning under a time limit, the EOSPP framework enables continuous correction of executed motion under unobserved wind disturbances while preserving real-time feasibility. Experimental results in obstacle and wind scenarios demonstrate the effectiveness of EOSPP. Compared with a baseline that performs only a single optimization without online path planning prior to flight execution, EOSPP consistently achieves safe convergence to the target formation, whereas the baseline fails due to collision in the obstacle scenario or accumulated drift after wind onset. In addition, the anytime behavior confirms that EOSPP with ES can provide high-quality best-so-far solutions within strict computation budgets, supporting the practicality of evolutionary online path planning for safety-critical swarm formation transitions.

Future research can extend the EOSPP framework in several directions. First, the current one-step formulation can be extended to multi-step predictive planning to improve trajectory smoothness under more complex environments. Second, improving computational efficiency and parallelization may enable scaling to larger swarms under tighter time budgets. Finally, extending the framework to three-dimensional workspaces and heterogeneous UAV dynamics remains an important direction for broadening applicability in real-world swarm missions.

References

[1] N. Cheng, S. Wu, X. Wang, Z. Yin, C. Li, W. Chen, and F. Chen, "AI for UAV-assisted IoT applications: A comprehensive review," *IEEE Internet of Things Journal*, vol. 10, no. 16, pp. 14 438–14 461, 2023.

[2] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[3] A. Stentz *et al.*, "The focussed d* algorithm for real-time replanning," in *Proceedings of the International Joint Conference on Artificial Intelligence*, vol. 95, 1995, pp. 1652–1659.

[4] S. Koenig and M. Likhachev, "D* lite," in *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, 2002, pp. 476–483.

[5] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[6] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[7] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[8] M. Diehl, H. G. Bock, and J. P. Schlöder, "A real-time iteration scheme for nonlinear optimization in optimal feedback control," *SIAM Journal on Control and Optimization*, vol. 43, no. 5, pp. 1714–1736, 2005.

[9] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Systems Magazine*, vol. 36, no. 6, pp. 30–44, 2016.

[10] A. Richards and J. P. How, "Robust distributed model predictive control," *International Journal of Control*, vol. 80, no. 9, pp. 1517–1531, 2007.

[11] T. Back, U. Hammel, and H.-P. Schwefel, "Evolutionary computation: Comments on the history and current state," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 3–17, 1997.

[12] A. E. Eiben and J. E. Smith, *Introduction to evolutionary computing*. Springer, 2015.

[13] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras, "Evolutionary algorithm based offline/online path planner for uav navigation," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 33, no. 6, pp. 898–912, 2003.

[14] J. Zhang, N. Li, D. Zhang, X. Wei, and X. Zhang, "Multi-UAV cooperative route planning based on decision variables and improved genetic algorithm," in *Journal of Physics: Conference Series*, vol. 1941, no. 1. IOP Publishing, 2021, p. 012012.

[15] H. Bai, T. Fan, Y. Niu, and Z. Cui, "Multi-UAV cooperative trajectory planning based on many-objective evolutionary algorithm," *Complex System Modeling and Simulation*, vol. 2, no. 2, pp. 130–141, 2022.

[16] J. Zheng, M. Ding, L. Sun, and H. Liu, "Distributed stochastic algorithm based on enhanced genetic algorithm for path planning of multi-UAV cooperative area search," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 8, pp. 8290–8303, 2023.

[17] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies-a comprehensive introduction," *Natural Computing*, vol. 1, no. 1, pp. 3–52, 2002.

[18] M. Otte and E. Frazzoli, "RRTX: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *The International Journal of Robotics Research*, vol. 35, no. 7, pp. 797–822, 2016.

[19] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1–2, pp. 83–97, 1955.

[20] R. Li, M. T. Emmerich, J. Eggermont, T. Bäck, M. Schütz, J. Dijkstra, and J. H. Reiber, "Mixed integer evolution strategies for parameter optimization," *Evolutionary Computation*, vol. 21, no. 1, pp. 29–64, 2013.