2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

A Markov Decision Process Approach for Decentralized UAV Formation Path Planning

Francesco Trotti[1], Alessandro Farinelli[2] and Riccardo Muradore[1]

Abstract— **Fleet coordination and formation flight for Unmanned Aerial Vehicles (UAVs) are challenging and important problems that have received significant attention in recent years. In this paper, we propose a decentralized approach based on a Markov Decision Process (MDP) to ensure the control and formation flight of UAVs. We present a methodology for planning trajectories online that enables UAVs to maintain formation geometry while avoiding no-fly zones and reaching a desired goal area. By leveraging the dynamic model of fixed-wing UAV within the MDP formalization, we can provide optimal reference values for the UAV low-level controller. Furthermore, by harnessing the capabilities of MDPs to handle uncertainty, we can consider the behavior of nearby UAVs taking advantage of the predicted state vectors shared among them. Therefore, by exploiting the UAV dynamic model, and the estimation of the possible trajectories of the other UAVs, we can ensure collision-free and feasible actions for the UAV in a decentralized way. This approach has been validated and tested through simulations involving various scenarios.**

I. INTRODUCTION

Fleet coordination and formation flight for Unmanned Aerial Vehicles (UAVs) are well-known problems that have received significant attention in recent years, especially in response to the growing demand for autonomous system capabilities. Achieving autonomy in formation flight and fleet coordination requires control systems capable of handling different levels of uncertainty, arising from the unexpected behavior of agents within the fleet, due to the limited or absent knowledge of the states of the other UAVs, or environmental conditions. Considering these aspects, the planning problem for a fleet becomes more challenging and complex.

Several hierarchical approaches, as proposed in [1], aim to address formation control, managing issues like leader-following, behavioral, and virtual structure problems through control-theoretic methods. Additionally, leveraging control-theoretic techniques, as discussed in [2], helps coordinate a fleet of mobile robots using a feedback control law and a reactive control framework. However, these approaches often require a centralized component, limiting scalability to a large number of robots. A significant number of approaches in the literature are based on the consensus theorem, as noted in [3], [4], and [5]. Some collision avoidance control algorithms for multi-UAV systems designed around consensus-based algorithms and leader-follower control strategies are presented in [6] and [7]. However, these approaches primarily focus on the leader-following consensus problem, assuming that only the parameters of followers are uncertain. In contrast, [8], also utilizing the consensus theorem, addressed the issue of parametric uncertainties and unknown external disturbances for both leaders and followers by employing a multivariable model reference adaptive control (MRAC).

Other approaches, like [9], formalize the problem as a decentralized receding horizon controller, based on the optimization problem formalized as mixed-integer linear programs (MILP). Approaches based on Nonlinear Model Predictive Control (NMPC), as discussed in [10], leverage a polytopic description of each robot's shape and formulate collision avoidance as a dual optimization problem. Mixed approaches, as presented in [11], combine decentralized Model Predictive Control (MPC) formalization with consensus control strategies to address cooperative formation control with collision avoidance. Nevertheless, these approaches often fail to account for various types of uncertainty, such as model errors, sensor measurement errors, or environmental dynamics. Therefore, different approaches formalize the problem of formation flight as a decision-making process, to consider different uncertainties due to the environment [12].

In this paper, we propose a decentralized controller for the formation flight of a fleet of UAVs. Specifically, we formalize the problem as a decision-making process introducing an online path planner (high-level controller) for each UAV, formalized as a Markov Decision Process (MDP). The high-level controller provides the best local reference values to the UAV low-level controller (assumed to be given), exploiting a nonlinear dynamic model of the UAV. The MDP is formalized to consider constraints related to formation flight (e.g., maintaining the correct position within the formation) and constraints related to path planning (e.g., reaching the target area and avoiding designated no-fly zones).

Furthermore, each UAV shares its predicted poses, obtained during the simulation horizon, with the other UAVs to avoid possible collisions. Along the shared predicted poses a probability distribution is modeled to design the uncertain position of the other UAVs in the time. Since only the action at the next sample time is effectively applied to the UAV low-level controller the other predicted states may not be accurate. Therefore, an exponential distribution is modeled so that a higher probability is assigned to states closer in time than to states farther away (since they will be less explored by the algorithm). We propose an online path planner for each UAV utilizing the MDP online solver Monte Carlo Tree Search (MCTS) [13].

[1] Francesco Trotti and Riccardo Muradore are with the Department of Engineering for Innovation Medicine, University of Verona, 37134, Italy francesco.trotti@univr.it, riccardo.muradore@univr.it

[2] Alessandro Farinelli is with the Department of Computer Science, University of Verona, 37134, Italy alessandro.farinelli@univr.it

The main contributions of this paper are:

- The online path planner is formalized as an MDP, which leverages the UAV internal nonlinear dynamic model to compute the best local reference values, taking into account formation and environmental constraints.
- The distributed approach relies on the shared predicted poses to estimate the position of the UAVs over time with a certain probability.

The paper follows this structure: Section II covers MDP and MCTS, revisiting the UAV dynamic model and low-level controller. Section III defines the problem, while Section IV details system operation. Simulation results are presented in Section V. Finally, Section VI concludes and outlines future research directions.

II. BACKGROUND

A. MDP and MCTS

Markow Decision Processes (MDPs) framework allows to generate a policy for an agent in a fully observable environment. A MDP is defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$, where:

- $\mathcal{S}$ is the state space with $s \in \mathcal{S}$
- $\mathcal{A}$ is the action space with $a \in \mathcal{A}$
- $\mathcal{T}$ is the transition model $\mathcal{T}^a_{s,s'} = \mathbb{P}(s_{t+1} = s' | s_t = s, a_t = a)$ that defines the probability to evolve from the current state s at the time t to the future state s' at the time $t+1$ by taking the action a
- $\mathcal{R}(s,a)$ is the reward function from the current state $s \in \mathcal{S}$ taking the action $a \in \mathcal{A}$
- ϵ is the discount factor ($\epsilon \in [0, \dots, 1]$)

Consequently, the MDP policy is a mapping from state to action, determining which action will be chosen from each state at any distract-time t.

The online solvers for the MDP are widely used since they manage to lower computational time and memory usage compared to offline solvers, admitting an approximation of the solutions. A popular online MDP solver is Monte Carlo Tree Search (MCTS), which alternates between planning and execution phases to provide online actions for the agents. MCTS operates through four phases:

1) Selection: The algorithm selects the node to reach the leaf of the tree using Upper Confidence bounds applied to Trees (UCT) [14].
2) Expansion: New nodes are added to the leaf node
3) Simulation: A simulator denoted by $\mathcal{G}$ is used to emulate the possible agent behaviors in the future. Given a state and an action, the simulator provides the state at time $t+1$ (s_{t+1}) and a reward value r_{t+1}. The simulator can be defined as follows: $(s_{t+1}, r_{t+1}) \sim \mathcal{G}$.
4) Back-propagation: The score obtained during the simulation is back-propagated to the root.

B. Aircraft dynamic model

The non-linear dynamic UAV model used in our formulation is taken from [15][16], and [17]. The aircraft state vector is

$$\boldsymbol{x} = [x \;\; y \;\; h \;\; v \;\; \psi \;\;]^T. \tag{1}$$

where x, y, h are the position and altitude of the UAV, v is the UAV airspeed and ψ is the UAV yaw angle. The dynamic model is

$$\begin{aligned} \dot{x} &= v\cos(\psi)\cos(\gamma) \\ \dot{y} &= v\sin(\psi)\cos(\gamma) \\ \dot{h} &= v\sin(\gamma) \\ \dot{v} &= \frac{T-D}{m} - g\sin(\gamma) \\ \dot{\psi} &= \frac{g\tan(\phi)}{v\cos(\gamma)} \end{aligned} \tag{2}$$

The UAV is controlled using an inverse dynamics controller [18], [19] to control the thrust T and the roll and flight path angles ϕ and γ of the UAV defining the desired airspeed value v_d, the desired altitude h_d and the desired yaw angle ψ_d. Therefore the thrust is

$$T = m(K_v(v_d - v) + g\sin\gamma) + D \tag{3}$$

where K_v is the positive speed control gain. The flight path angle is controlled to reach the desired altitude

$$\gamma = \left(\arcsin\left(\frac{K_h(h_d - h)}{v}\right)\right. \tag{4}$$

where K_h is a positive gain, while the roll angle is controlled to reach the desired yaw angle considering the flight path angle,

$$\phi = \arctan\left(\frac{K_\psi(\psi_d - \psi)v\cos(\gamma)}{g}\right) \tag{5}$$

where K_ψ is the positive gain and g is the gravity. The controllers modeled in the MDP are the discrete-time version of the continuous-time control laws (3)-(5).

III. PROBLEM STATEMENT

The fleet is composed of n UAVs that have to keep a desired formation geometry. The fleet has to reach the target area avoiding some no-fly zones in a 3D scenario maintaining the desired formation geometry. A nonlinear dynamic model is used to describe the behavior of the UAVs and each aircraft has an inverse dynamics controller to reach the provided reference values. For each UAV an MDP is formalized to guarantee that

- the UAV maintains the formation geometry
- the UAV avoids the no-fly zones
- the UAV does not collide with the other UAVs

Given these constraints, the MDP has to provide the best reference values (v_d, h_d, ψ_d) for the UAV low-level controller that minimize the cost function, as shown in [20]. Some assumptions commonly used in the literature are needed to understand and justify some behavior of our approach.

Assumption 1: The UAVs in the fleet have the same dynamic model.

Assumption 2: The geometry of the formation is known by all UAVs and it is expressed with respect to the frame of the lead UAV.

A. MDP Formalization

Our approach is based on discrete actions and states; therefore, we discretize the state space $\mathcal{S}$ and the action space $\mathcal{A}$. Since our approach is decentralized, we first present a generic MDP formalization, which is applied to each UAV.

States: The MDP state represents how the system evolves over time. It consists of the UAV state vector and future predictions of the other UAVs

$$s = [\boldsymbol{x}, \hat{\boldsymbol{h}}] \tag{6}$$

where, $\boldsymbol{x}$ represents the UAV state vector (1), and $\hat{\boldsymbol{h}}$ contains the predictions of the future state vectors of the other UAVs. In particular, $\hat{\boldsymbol{h}}$ is defined as:

$$\hat{\boldsymbol{h}} = \begin{bmatrix} \boldsymbol{x}^i_{t+1} & \dots & \boldsymbol{x}^i_{t+N} \\ \vdots & \ddots & \vdots \\ \boldsymbol{x}^n_{t+1} & \dots & \boldsymbol{x}^n_{t+N} \end{bmatrix} \tag{7}$$

where i ranges from 0 to n (number of UAVs in the fleet), $t = NT_s$ with T_s the sampling time, and N is a parameter that defines the length of the horizon.

Actions: The actions are the reference values for the UAV low-level controller (3)-(5)

$$a = [v_d, h_d, \psi_d]^T \tag{8}$$

where v_d, h_d and ψ_d are the desired velocity, altitude and yaw angle. The controller will handle thrust and the tack and bank angles based on these reference values. The action set is defined as a percentage increment or decrement of the reference value.

Transition model: The transition model defines the next state given the current state and an action. In our case, we use the aircraft dynamic model with the control loop explained in subsection II-B. This allows to simulate different reference values (actions) from the current state by evaluating the system evolutions through the cost function. During the simulation, an exponential probability, commonly used in literature to represent the probability of being in that position at specific time, is considered in the predicted poses matrix ($\hat{\boldsymbol{h}}$) to stochastically model the poses of the other UAVs along the horizon. Therefore, the exponential probability is attributed in the following way

$$\mathbb{P}(\boldsymbol{x}^i|t) = e^{-\lambda t} \tag{9}$$

where i is the i-th row in the predicted poses matrix ($\hat{\boldsymbol{h}}$), t is the sampled time in the i-th row ($t = [t+1, \dots, t+N]$) and $-\lambda$ is the decay rate, which determines how fast the probabilities decrease, and finally, the values are normalized to guarantee sum one. The uncertainty arises from the fact that the MDP, solved by MCTS, uses only the action at the time $t+1$. Thereby, the other estimates represent only the best sequence of actions at the time t, but with the next computation the sequence of actions could be different. Additionally, the exponential probability is used to represent the uncertainty in order to give a higher likelihood to the states closer to the current time t and less to the states in the far future.

Cost function: The cost function evaluates the goodness of an action (the cost function is the reward function explained in Section II). In our approach, the cost function aims to minimize a combination of metrics, including reaching the target area, maintaining the correct position in the formation, and avoiding no-fly zones.

The component of the cost function responsible for reaching the target position is defined as the Euclidean distance between the UAV $X_{\text{UAV}} = (x, y, h)$ and the target area $X_T = (x_T, y_T, h_T)$

$$r_D = \|X_T - X_{\text{UAV}}\| \tag{10}$$

The component that keeps the UAV in the right position within the formation is determined by the displacement between the desired pose (position and orientation) in the formation and the current UAV pose. The desired pose is expressed in the UAV frame placed in the point of symmetry of the formations (Γ_M). Therefore, a transformation matrix is needed to express the desired pose (Γ_d) in the current UAV frame (Γ_U). This transformation matrix is obtained by combining the position vector error and the yaw angle error ($\psi_e = \psi_M - \psi_U$) between the two poses. The transformation matrix is defined as

$$T_{M,U} = \left[\begin{array}{ccc:c} \cos(\psi_e) & -\sin(\psi_e) & 0 & (x_M - x_U) \\ \sin(\psi_e) & \cos(\psi_e) & 0 & (y_M - y_U) \\ 0 & 0 & 1 & (h_M - h_U) \\ \hdashline 0 & 0 & 0 & 1 \end{array}\right] \tag{11}$$

where the upper left 3×3 matrix is the rotational matrix, and the vector 4×1 contains distance between the origins of Γ_M in Γ_U.

The desired formation pose expressed in the current UAV frame (Γ_U) is defined as $\Gamma_d = \Gamma_M T_{M,U}$. Consequently, the cost value is

$$r_F = (R_{\Gamma_U} - R_{\Gamma_d}) + \sum_{i=1}^{3} P^i_{\Gamma_d} \tag{12}$$

where $P^i_{\Gamma_d}$ represents the i-th position value of the Γ_d matrix, while R_{Γ_U} and R_{Γ_d} are respectively the current UAV rotation and the desired rotation.

The function to avoid no-fly zones is modeled as a repulsive force [21]

$$r_{Z_i} = \begin{cases} \nu \left(\left(\frac{1}{d_{Z_i}} - \frac{1}{d_0} \right)^2 \right), & \text{if } d_{Z_i} < d_0 \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

where, d_{Z_i} is the distance between the UAV and the i-th no-fly zone (Z is the no-fly zones set), d_0 is the repulsive threshold, and $\nu > 0$ is the repulsive coefficient.

Additionally, a repulsive force generated by the position of the other UAVs in the predicted poses matrix ($\hat{\boldsymbol{h}}$) is added to avoid collisions between the aircrafts. In particular, the repulsive force is scaled by the probability of the state in the predicted poses matrix. In this way, the states with lower probability (states more in the future) have less impact on the total force compared to the states closer to the current

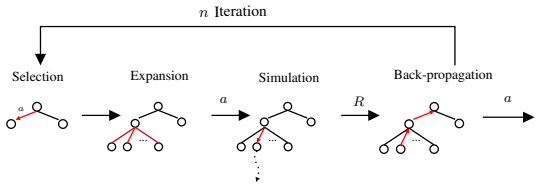


Fig. 1. MCTS algorithm

time (higher probability). This case could happen when the other constraints (e.g. no-fly zones) force the aircraft to break the formation and then they could collide with each other. Therefore, the UAV repulsive force is

$$r_{U_i} = \begin{cases} \nu^{\sigma} \left(\left(\frac{1}{d_{U_i}} - \frac{1}{d_0} \right)^2 \right), & \text{if } d_{U_i} < d_0 \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

where d_{U_i} is the distance vector between the current UAV horizon and the i-th UAV in the predicted poses matrix, d_0 is calculated as the distance between the i-th UAV in the predicted poses matrix and the current formation position for the current UAV; while ν^{σ} is the repulsive coefficient scaled for the probability (according to (9)) along the i-th state vector of the predicted matrix. Therefore, σ is the probability distributed function (PDF) given the state in the predicted poses matrix and the time. The reward value concerning the no-fly zones and the UAVs is:

$$r_R = \sum_{i=1}^{Z} r_{Z_i} + \sum_{i=1}^{n} r_{U_i} \tag{15}$$

The overall cost value is the sum of all these components:

$$r = -r_D - r_F - r_R \tag{16}$$

Note that all the components are negative because we minimize the cost function (the closer is to zero, the better).

IV. FORMATION PLANNING

In this section, we present the evolution of the proposed approach over time. Specifically, the MDP computes the best reference values for the UAV low-level controllers to minimize the cost function considering various constraints. At each time step, each MDP provides the best action to the low-level controller, and after shares its predicted poses of N steps to the other UAVs.

A. MCTS

The MCTS algorithm serves as an online solver for the MDP problems. It's based on tree exploration and expansion and consists of four main phases, performed within a fixed time interval: selection, expansion, simulation, and back-propagation, as illustrated in Figure 1. MCTS is an on-line solver that alternates between simulation and execution phases until the goal is achieved.

Selection phase: In the selection phase, the algorithm chooses the best local action to reach a new state within the tree. This selection is made using the Upper Confidence Bounds applied to Trees (UCT) [14], [13], which is:

$$a^* = \text{argmax}_{a \in \mathcal{A}(s)} \left(\frac{Q(s,a)}{N(s,a)} + C \sqrt{\frac{\log N(s)}{N(s,a)}} \right) \tag{17}$$

where, $\mathcal{A}(s)$ is the action set, $Q(s,a)$ is the value obtained by executing action a in state s, C is the exploration-exploitation constant, $N(s)$ is the number of visits to the node with state s, and $N(s,a)$ is the number of visits to the child node of the node with state s when applying action a. By utilizing the UCT strategy, we can balance the depth and breadth of tree exploration, and it also allows to cross the tree and reach unexplored nodes by evaluating node scores obtained in previous simulations. The depth of the MCTS tree represents how many time steps into the future the algorithm analyzes, while the breadth of each layer of the tree represents how many alternatives the algorithm explores at each time step.

Expansion phase: The selection phase concludes when a leaf node is reached, and from this node, several new nodes (states) equal to the size of the action space $\mathcal{A}$ are created.

Simulation phase: The simulation phase explores more in-depth the possible evolutions in the future from the current state, trying different actions. The simulation phase is based on the use of a simulator $\mathcal{G}$ that, given a state and an action, provides the future state and the reward values, i.e. $(s_{t+1}, r_{t+1}) \sim \mathcal{G}(s_k, a_k)$.

To obtain the next state and reward value, the simulator relies on the transition model and cost function defined in Section III-A. After expanding a node, a new random node is selected, initiating the rollout method. This method tries different random actions from the current state until a specified maximum depth (rollout horizon) is reached. By doing so, the algorithm explores the future of the current state more profoundly by employing the simulator $\mathcal{G}$ with various random actions. Upon reaching the maximum depth for the rollout, a reward value for all simulations is computed utilizing the cost function and discount factor

$$R = \sum_{i}^{H} \epsilon^i r_i \tag{18}$$

where, r_i represents the partial reward value obtained at each step during the rollout, and H stands for the rollout horizon (maximum depth). The discount factor ϵ, or forgetting factor, determines how rewards diminish over time. Values obtained closer to the current state have larger weights than those from more distant time steps. Following the rollout phase, the explored nodes are removed, and only the cumulative reward value R is retained.

Back-propagation phase: The back-propagation phase updates the parameters of the nodes by crossing the tree from leaf to root along the path defined by UCT. In particular, given R the reward obtained during the simulation phase, the $Q(s,a)$ value of each visited node is updated following

$$Q(s,a) = Q(s,a) + \frac{R - Q(s,a)}{N(s)} \tag{19}$$

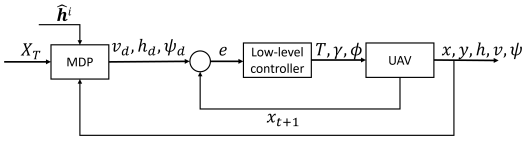


Fig. 2. UAV control schema

	Scenario 1	Scenario 2
# Sim	600	1000
H	2	4
ν	100	100
k	3	3
n UAV	3	3

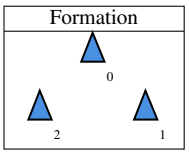


TABLE I

PARAMETERS AND FORMATION

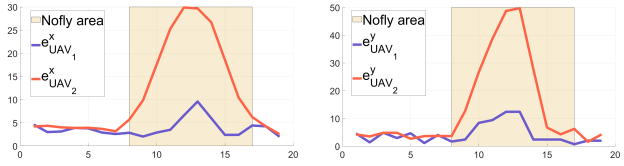
B. System evolution

These four phases of the MCTS are executed in a loop until the time constraints are met. When this occurs, the UCT strategy is applied at the root node to determine the best action at time $t+1$, which is then applied to the low-level UAV controller. Following the execution of this action, the new state vector of the UAV becomes the new root for the next iteration of the MCTS algorithm. To guarantee the correct length of the vector of predicted poses, the number of the MCTS simulations has to be set in order to create a tree with a depth greater than or equal to N. In particular, the vector of predicted poses is formed by applying the UCT strategy to the MCTS tree N times to extract the best estimated states from the current state. It's important to note that only the $t+1$ action is employed in the UAV low-level controller, and the predicted poses are used solely to share a likely trajectory of the other UAVs.

By leveraging this concept, the system remains resilient to communication issues between UAVs when disturbances last for a duration shorter than the vector of predicted poses length. This is because each UAV knows the actions that other UAVs will take in the next N steps. Each UAV independently solves its own MCTS algorithm, generating the best local reference values for its low-level controller while considering the other UAVs. Figure 2 illustrates the loop system for one UAV ($\hat{\boldsymbol{h}}^i$ represents the vector of predicted poses of the i-th UAV). Consequently, the fleet evolves in a decentralized manner over time, and, after the MCTS iterations, each UAV shares its vector of predicted poses with the other UAVs.

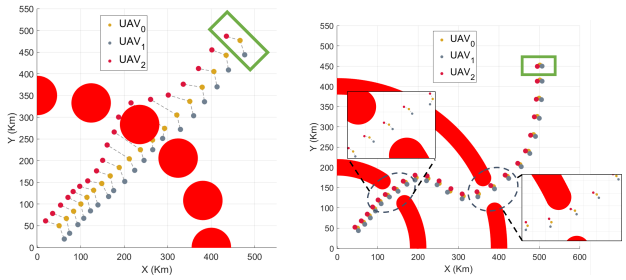
V. SIMULATION RESULT

The proposed technique has been tested in two different simulated scenarios. In both cases, the fleet must reach a desired area while avoiding no-fly zones and attempting to maintain formation whenever possible. In the first scenario, various no-fly zones are taken into consideration, whereas in the second scenario, a no-fly zone barrier with a breach is considered. Table I lists the most important hyper-parameters used during the simulations and also shows the formation



(a) UAV1 and UAV2 absolute error along x axis (b) UAV1 and UAV2 absolute error along y axis

Fig. 3. Absolute error along x and y axis



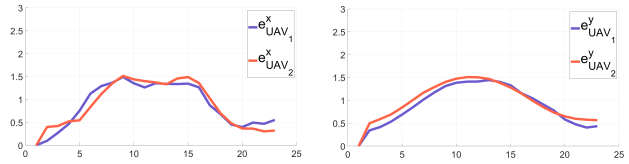
(a) First scenario (b) Second scenario

Fig. 4. Fleet paths. The green rectangle is the target area and the red areas are the no-fly zones

geometry. In particular, the symmetry point of the formation is located at UAV0. As a result, all the other formation points are expressed relative to the reference frame of the UAV0. In both scenarios, the initial conditions for UAV0 are set to $x_0 = [50, 50, 1.5, 200, 0.78]$, while for the other UAVs, the initial conditions are the same, with their poses adjusted to fit the correct formation position. In the experiments the altitude is not shown since the target altitude is equal to the initial condition and the UAVs maintain constant the altitude during the simulation. The proposed results are tested on the feet of three UAVs, but the approach is easily extendable to a fleet with more agents since it is local for each UAV, and the complexity is not directly related to the number of agents.

A. First scenario: Multi No-fly Zones

In the first scenario, the fleet has to reach the target area avoiding some no-fly zones. Figure 4(a) shows the paths generated by the proposed methodology. It is possible to see that the UAVs maintain the formation until the no-fly zones are reached. In particular, the UAV0 avoids the no-fly zone by passing to the right. Since UAV0 defines the formation geometry, the desired formation position for UAV1 also shifts to the right. However, on the right side of UAV1, another no-fly zone is present. Consequently, the optimal local action for UAV1 is to move closer to UAV0. This adjustment occurs because, in the cost function of UAV1, the repulsive no-fly zone component becomes more significant than the formation constraint. As a result, UAV1 moves closer to UAV0. On the other hand, in the presence of the no-fly zone, the UAV2 changes completely the path. This behavior is due to reward values obtained in the simulations. In particular, if the UAV2 had passed to the right side of the no-fly zone a penalty regarding the wrong position in the formation would have been taken into account and also a



(a) UAV1 and UAV2 absolute error along x axis (b) UAV1 and UAV2 absolute error along y axis

Fig. 5. Absolute error along x and y axis

highly repulsive component due to the closeness to the UAV0 and UAV1 would have been added. Whereas, passing on the left side of the no-fly zone, the UAV2 obtains only a penalty from the formation position component and avoids possible collision with the other UAVs. These considerations are also highlighted in Figure 3, where the absolute error along the x (Figure 3(a)) and y (Figure 3(b)) axes are shown for each UAV. It is possible to see the error increases near the no-fly zone (yellow area in the plots) while in the other parts, the error is small.

B. Second scenario: Breach Crossing

In the second case, a scenario where the fleet has to cross different breaches and maintain the formation is considered. As in the previous case the fleet has to avoid the no-fly zones and reach the target area while keeping the formation. Figure 4(b) shows the trajectories executed by the UAVs with the zoom on the most important part of the path. In this case, the formation is maintained throughout the path by all UAVs. In particular, exploiting the greater number of simulations and the rollout depth, more future evolutions of each UAV are analyzed. Additionally, since the no-fly zones are modeled as a barrier, the simulations of all UAVs are quite similar unless the formation position displacement. Figure 5 shows the absolute error along the x (Figure 5(a)) and y (Figure 5(b)) axes for each UAV, the error is small but increases a little in the area between the two no-fly zones. This behavior is due to the target attractive component, leading to a bad reward as the UAVs move away from the target. However, the formation attractive components and the high rollout depth (the future is analyzed more in depth) allow to compensate for the target attractive penalty by finding the best trajectories.

VI. CONCLUSIONS

In this paper, we introduce a decentralized methodology for controlling and ensuring the formation flight of multiple UAVs. Each UAV is modeled as an MDP to determine optimal local reference values for low-level controllers, factoring in planning and formation constraints such as reaching targets, avoiding no-fly zones, maintaining formation positions, and preventing collisions. Leveraging a shared predicted poses matrix, our approach guarantees collision-free paths between UAVs. Validation in two scenarios demonstrates fleet behavior under various constraints and parameters. Future work involves comparing our method with current techniques, incorporating a manned lead aircraft into the fleet model, and refining position estimation filters within the MDP framework.

REFERENCES

[1] R. W. Beard, J. Lawton, and F. Y. Hadaegh, "A coordination architecture for spacecraft formation control," *IEEE Transactions on control systems technology*, vol. 9, no. 6, pp. 777–790, 2001.

[2] H. Yamaguchi, "A cooperative hunting behavior by mobile robot troops," in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, vol. 4. IEEE, 1998, pp. 3204–3209.

[3] W. Ren, R. W. Beard, and E. M. Atkins, "A survey of consensus problems in multi-agent coordination," in *Proceedings of the 2005, American Control Conference, 2005.* IEEE, 2005, pp. 1859–1864.

[4] Q. Wang, H. Gao, F. Alsaadi, and T. Hayat, "An overview of consensus problems in constrained multi-agent coordination," *Systems Science & Control Engineering: An Open Access Journal*, vol. 2, no. 1, pp. 275–284, 2014.

[5] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial informatics*, vol. 9, no. 1, pp. 427–438, 2012.

[6] Y. Kuriki and T. Namerikawa, "Consensus-based cooperative formation control with collision avoidance for a multi-uav system," in *2014 American Control Conference*. IEee, 2014, pp. 2077–2082.

[7] J. Zhang, J. Yan, P. Zhang, and X. Kong, "Collision avoidance in fixed-wing uav formation flight based on a consensus control algorithm," *IEEE Access*, vol. 6, pp. 43 672–43 682, 2018.

[8] Z. Zhen, G. Tao, Y. Xu, and G. Song, "Multivariable adaptive control based consensus flight control system for uavs formation," *Aerospace Science and Technology*, vol. 93, p. 105336, 2019.

[9] T. Keviczky, F. Borrelli, K. Fregene, D. Godbole, and G. J. Balas, "Decentralized receding horizon control and coordination of autonomous vehicle formations," *IEEE Transactions on control systems technology*, vol. 16, no. 1, pp. 19–33, 2007.

[10] R. Firoozi, L. Ferranti, X. Zhang, S. Nejadnik, and F. Borrelli, "A distributed multi-robot coordination algorithm for navigation in tight environments," *arXiv preprint arXiv:2006.11492*, 2020.

[11] Y. Kuriki and T. Namerikawa, "Formation control with collision avoidance for a multi-uav system using decentralized mpc and consensus-based control," *SICE Journal of Control, Measurement, and System Integration*, vol. 8, no. 4, pp. 285–294, 2015.

[12] B. Floriano, G. A. Borges, and H. Ferreira, "Planning for decentralized formation flight of uav fleets in uncertain environments with dec-pomdp," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 563–568.

[13] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in games*, vol. 4, no. 1, pp. 1–43, 2012.

[14] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, no. 2, pp. 235–256, 2002.

[15] J. Sun, J. M. Hoekstra, and J. Ellerbroek, "Openap: An open-source aircraft performance model for air transportation studies and simulations," *Aerospace*, vol. 7, no. 8, p. 104, 2020.

[16] R. W. Beard and T. W. McLain, *Small unmanned aircraft: Theory and practice*. Princeton university press, 2012.

[17] B. L. Stevens, F. L. Lewis, and E. N. Johnson, *Aircraft control and simulation: dynamics, controls design, and autonomous systems*. John Wiley & Sons, 2015.

[18] A. Isidori, *Nonlinear control systems II*. Springer, 2013.

[19] H. Khalil, *Nonlinear Systems*. Prentice Hall, 2002.

[20] F. Trotti, A. Farinelli, and R. Muradore, "An online path planner based on pomdp for uavs," in *2023 European Control Conference (ECC)*. IEEE, 2023, pp. 1–6.

[21] C. W. Warren, "Global path planning using artificial potential fields," in *1989 IEEE International Conference on Robotics and Automation*. IEEE Computer Society, 1989, pp. 316–317.