

Human-Drone Swarm Interaction System for Persistent Monitoring of Large Disperse Area

Petri Kosonen¹, Yeganeh Fadaeian², Reeta-Mari Eura², Jussi Sulameri²,
Muhamad Rausyan Fikri², and Azwirman Gusrialdi²

Abstract—This paper presents a human-drone swarm interaction system that enables adaptive and prioritized monitoring through an ergodic coverage control algorithm. The system is designed to ensure that drone swarms effectively cover a user-defined probability density function while maintaining both safety and persistent monitoring. A robust set of safety features prevents collisions between neighboring drones and prevents the drones from exiting the monitoring area. In addition, a fault tolerance algorithm is implemented to ensure continuous monitoring even if individual drones fail. The proposed algorithm is validated through real-world experiments using Crazyflie 2.1 from Bitcraze. The experimental results demonstrate that the algorithm successfully optimizes coverage as measured by the ergodic metric while maintaining safe operation.

I. INTRODUCTION

Persistent monitoring is a continuous observation of a target area over time. Although applications such as wildfire detection and agricultural maintenance still depend on a single drone [1], its limited flight time and vulnerability to failure make a swarm-based approach preferable. Swarms provide greater efficiency, resilience, and uninterrupted coverage. As a result, drone swarms are increasingly being used for environmental monitoring [2] and wildfire detection [3].

Human-swarm interaction introduces a more flexible system and effectiveness. The superior cognitive ability of humans can provide flexible guidance, allowing real-time decision making and adaptability to evolving monitoring needs [4]. In contrast to fully autonomous swarms, incorporating a human in the loop allows adaptive mission planning [5]. For example, a human operator can point the swarm toward a new area of interest, which improves energy consumption and increases the mission's overall effectiveness. Furthermore, a human operator can add or remove areas of interest, ensuring that the swarm can respond to dynamic environmental changes.

Human-swarm interaction for monitoring has been widely studied in the literature, with most coverage algorithms using a Voronoi-based approach. For example, in [6], a coverage control algorithm based on a user-defined density was proposed for swarm robots. The authors in [7] extended

this concept by developing a human swarm based on eye movement. Although both approaches allow the operator to specify areas of interest, they do not guarantee monitoring all target points, since it was originally designed for (sub)-optimal static deployment of mobile sensors and lack explicit collision avoidance among robots. An alternative approach based on ergodic control method was presented in [5], ensuring persistent region monitoring and obstacle avoidance through local planning. However, the work does not incorporate the failure state condition handler and the drone's velocity limit.

In this paper, we develop a human-drone swarm interaction system for persistent monitoring. Specifically, the system has the following novel features: 1) A human operator can flexibly designate target areas. Aggregating user input target points into clusters enables simultaneous monitoring of multiple areas. 2) To ensure monitoring of all the target areas, an ergodic controller is adopted as in [5]. However, in contrast to [5], we integrate the controller into quadratic programming (QP), enforcing control barrier functions (CBFs) [8], [9] for inter-drone collision avoidance. Furthermore, the CBF-QP also facilitates the computation of the velocity for each drone to ensure that its value remains within the limit and that the drones move within the designated area. 3) We include a fault tolerance mechanism to maintain continuous monitoring, which ignores failed individual drones and removes them from the threat potential. 4) The integration of real-time robot system visualization - RViz enhances monitoring capabilities by allowing operators to observe drone trajectories, assess coverage efficiency, and make informed decisions during operation.

The rest of this paper is structured as follows: Section II defines the problem of continuous monitoring using a drone swarm and outlines key requirements for effective coordination and human interaction. Section III presents the human-drone swarm interaction system, including the user interface for target selection, control algorithm for monitoring, and real-time visualization tools. Section IV demonstrates the implementation and experiments in the real world, validating our approach. Section V concludes the paper.

II. PROBLEM DEFINITION

The goal of our system is to achieve safe persistent monitoring in a two-dimensional field $\mathcal{F} = [0, L_1] \times [0, L_2]$ using a drone swarm integrated with human interaction capability. To realize this goal, the system must satisfy the following requirements:

¹Petri Kosonen is with The Faculty of Information Technology and Communication Sciences (ITC), information technology (computing sciences), Tampere University, Kalevantie 4, 33100 Tampere, Finland. petri.kosonen@tuni.fi

²Reeta-Mari Eura, Jussi Sulameri, Yeganeh Fadaeian, Muhamad Rausyan Fikri, and Azwirman Gusrialdi are with the Automation Technology and Mechanical Engineering, Faculty of Engineering and Natural Sciences, Tampere University. reeta-mari.eura, jussi.sulameri, yeganeh.fadaeian, muhamad.fikri, azwirman.gusrialdi[at]tuni.fi

- 1) **Continuous Monitoring:** The drones must cover the entire area of interest in the field \mathcal{F} , whose coverage is proportional to its importance generated by a human operator.
- 2) **Safe and Fault Free Operation:** Drones operate safely to ensure that there is no collision with other drones. This requirement also includes that the system can prevent drones from moving beyond the field of \mathcal{F} . To ensure continuous monitoring, the fault tolerance mechanism must be incorporated to handle drone failure.
- 3) **Human-Drone Swarm Interaction:** A user interface (UI) developed to specify the targets that define the importance map (represented by the probability density function (PDF)), and visualization tools (RViz) to visualize the persistent monitoring and trajectory of the drones.

The detailed description of the system is provided in the next section.

III. HUMAN-DRONE SWARM INTERACTION SYSTEM

We design a human-drone swarm interaction system as illustrated in Fig. 1. Specifically, the system is divided into three parts: (1) a User Interface, in which the operator inputs targeted areas that the drone swarm will monitor, (2) a safety-ensured control algorithm that controls the movement of the drones to monitor the targeted areas. Finally, (3) a system that visualizes the results, such as the swarm trajectory and the targeted areas.

A. User Interface

The monitoring area is defined as a two-dimensional field $\mathcal{F} = [0, L_1] \times [0, L_2]$, where L_1 and L_2 are the lengths on the horizontal and vertical axes, respectively. A human operator provides a cluster of targeted points that drones will monitor, as shown in Fig. 1. These targeted points are shifted to within \mathcal{F} . Let c be the total number of clusters, for the i -th cluster ($i = 1, \dots, c$), denote the set of target points by $\{q_{ij}\}_{j=1}^{n_i}$ with each $q_{ij} \in \mathcal{F}$ being the j th target point in the i th cluster, n_i and the number of points in the cluster. Given these target points, each cluster is approximated by a probability density function (PDF) with a mean vector μ_i and the covariance matrix Σ_i . Specifically, the PDF of the i -th cluster is defined as

$$z_i = \sqrt{(2\pi)^d \det(\Sigma_i)},$$

$$\phi_i(q) = \frac{1}{z_i} \exp\left(-\frac{1}{2}(q - \mu_i)^T \Sigma_i^{-1} (q - \mu_i)\right), \quad (1)$$

where $q \in \mathcal{F}$. To combine all clusters into the overall target distribution, a mixture model is used. The overall PDF $\phi(q)$ and weight w_i for the i th cluster by its relative number of targeted points are respectively defined as:

$$\phi(q) = \sum_{i=1}^c w_i \phi_i(q), \quad w_i = \frac{n_i}{\sum_{k=1}^c n_k} \quad (2)$$

This overall PDF is computed and then published to the target distribution in RViz, while the drones incorporate the information into their control algorithm.

B. Ergodic Controller for Monitoring

We adopt the ergodic control algorithm from [10] to realize persistent drone swarm monitoring. The key idea is to design control inputs that steer the drone swarm so that their time-averaged spatial distribution matches the desired PDF $\phi(q)$ defined over two-dimensional monitoring area $\mathcal{F} = [0, L_1] \times [0, L_2] \subset \mathbb{R}^2$. By minimizing the ergodic metric, which quantifies the discrepancy between the drones' spatial coverage and the target PDF, the drones spend more time in regions with higher importance, leading to an efficient monitoring strategy.

A detailed discussion of the ergodic control in [10] is provided in the following. Let n denote the total number of drones and $\mathcal{I} = \{1, \dots, n\}$ be the set of drone identifiers. The position of drone $i \in \mathcal{I}$ is denoted by $p_i(t) = [x_i(t) \ y_i(t)]^T \in \mathbb{R}^2$, with $p_i(0) \in \mathcal{F}$. Here, we assume that the altitudes of the drones are constant; thus, the motion of the drones is constrained in 2D space. Moreover, the drones' dynamics is given by a single integrator model:

$$\dot{p}_i = u_i, \quad u_i(t) \in \mathbb{R}^2 \quad (3)$$

where $u_i \in \mathbb{R}^2$ is the control input for drone i . It is assumed that the drones can communicate with each other.

Next, we define the ergodic metric that measures the discrepancy between the spatial coverage distribution and the targeted PDF in (1). Let $\mathcal{K} = \{k = [k_1, k_2] \mid k_i \in \{0, 1, \dots, K\}, i = 1, 2\}$ denote the set of Fourier indices, with maximum frequency index K . The ergodic metric is then defined by

$$\varepsilon(t) = \sum_{k \in \mathcal{K}} \lambda_k [c_k(t) - \varphi_k]^2, \quad (4)$$

where $c_k(t)$ is the time-averaged Fourier coefficients representing the spatial distribution of the drones' trajectories, computed as $c_k(t) = \frac{1}{nt} \sum_{i=1}^n \int_0^t f_k(p_i(\tau)) d\tau$. Here φ_k is the Fourier coefficient of the target PDF $\phi(q)$ obtained from the calculating $\varphi_k = \int_{\mathcal{F}} f_k(q) \phi(q) dq$, and λ_k is the weighting factor defined by $\lambda_k = \frac{1}{(1+||k||^2)^s}$, $s > 0$. For any point $q = [q_1, q_2] \in \mathcal{F}$, the normalized Fourier basis function is defined by

$$f_k(q) = \frac{1}{h_k} \prod_{i=1}^2 \cos\left(\frac{k_i \pi}{L_i} q_i\right), \quad (5)$$

with normalization constant

$$h_k = \left(\int_{\mathcal{F}} \prod_{i=1}^2 \cos^2\left(\frac{k_i \pi}{L_i} q_i\right) dq \right)^{\frac{1}{2}}, \quad (6)$$

ensuring that $\int_{\mathcal{F}} f_k(q)^2 dq = 1$.

The control input is derived from the gradient of the discrepancy between the current coefficients and the target Fourier coefficients to drive the system toward ergodicity. In particular, for each drone, the control direction is computed

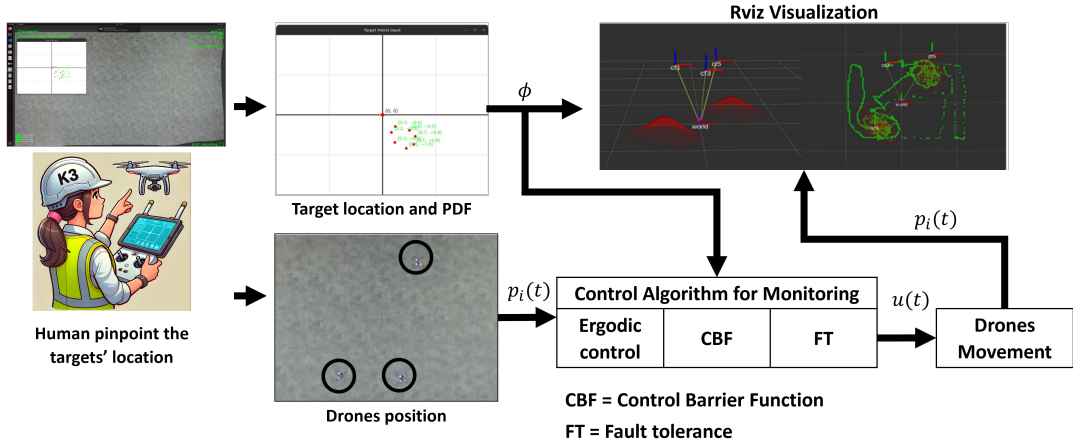


Fig. 1. The persistent monitoring scenario begins with human operators pinpointing the target location, which is then converted into a PDF. The PDF and the drone trajectories are visualized in RViz.

by aggregating the gradients of the Fourier basis functions evaluated at the drone's current position, which is given by

$$b_i(t) = \sum_{k \in \mathcal{K}} \lambda_k (c_k(t) - \varphi_k) \nabla f_k(p_i(t)), \quad (7)$$

where for two components of $\nabla f_k(p_i(t))$ is

$$\nabla f_k(p_i(t)) = \frac{1}{h_k} \begin{bmatrix} -\frac{k_1 \pi}{L_1} \sin\left(\frac{k_1 \pi}{L_1} x_i(t)\right) \cos\left(\frac{k_2 \pi}{L_2} y_i(t)\right) \\ -\frac{k_2 \pi}{L_1} \cos\left(\frac{k_1 \pi}{L_1} x_i(t)\right) \sin\left(\frac{k_2 \pi}{L_2} y_i(t)\right) \end{bmatrix} \quad (8)$$

The ergodic controller for each drone is then given by

$$u_i^{erg}(t) = -u_{\max} \frac{b_i(t)}{\|b_i(t)\|}, \quad (9)$$

where $u_{\max} > 0$ is the maximum limit control magnitude. The controller (9) ensures that each drone adjusts its trajectory in a direction that minimizes the ergodic metric, aligning the time-average coverage with the targeted PDF.

Remark 1: To calculate control input (9), each drone needs to communicate with all the other drones to calculate time-averaged Fourier coefficients $c_k(t)$. The requirement of all-to-all communication can be relaxed to only having the graph be connected (i.e., information from any drone can be passed to any other drone via a sequence of links) using an average consensus algorithm [11], [12] to calculate c_k . This algorithm also enhances the scalability of ergodic control computation as the number of drones increases.

C. Safety Ensured Ergodic Control

To ensure safety during operation, the controller integrates control barrier functions (CBFs) into quadratic programming (QP) framework that projects the nominal ergodic control u_i^{erg} onto a set of safe controls in the monitored area $\mathcal{F} = [0, L_1] \times [0, L_2]$. This guarantees both inter-drone collision avoidance and prevents the drone from exploring beyond the monitored area. The safe control input for drone i is

computed by solving the following QP:

$$\begin{aligned} u_i &= \underset{u_i}{\operatorname{argmin}} \|u_i^{erg} - u_i\|^2 \\ \text{s.t.} \quad & \left(\frac{\partial h_o(p_i, p_j)}{\partial p_i} \right)^T u_i \geq -\gamma_1(h_o(p_i, p_j)), \\ & \forall j \in \mathcal{N}_i^s, \\ & \left(\frac{\partial h_w(p_i)}{\partial p_i} \right)^T u_i \geq -\gamma_2(h_w(p_i)), \\ & \forall w \in \mathcal{N}_w^b, \\ & \|u_i\| \leq u_{\max} \end{aligned} \quad (10)$$

where $h_o(p_i, p_j) = \|p_i - p_j\|^2 - R_s^2$ is the collision avoidance barrier function with R_s denoting the safety distance of a drone, which depends on the drone's size. Furthermore, $\mathcal{N}_i^s = \{j \neq i \mid \|p_i - p_j\| \leq (2R_s)\}$ is the set of neighboring drones within the distance $2R_s$ considered for the collision avoidance. The functions γ_1, γ_2 are extended class \mathcal{K}_∞ functions. The first constraint in (10) ensures collision avoidance between the drones [13]. In addition, we also introduce a CBF to prevent the drones' exploration beyond the boundaries of \mathcal{F} . Let q_w denote the perpendicular projection of the drones' position p_i onto the boundaries considered as obstacles. The corresponding CBF is given by

$$h_w(p_i) = \|p_i - q_w\|^2 - R_{wall}^2, \quad \forall w \in \mathcal{N}_w^b \quad (11)$$

where $w = 1, 2, 3, 4$ denote respectively the left, right, top, and bottom boundaries, and R_{wall} is the minimum safe distance from the boundary. Moreover, the set $\mathcal{N}_w^b = \{w \mid \|p_i - q_w\| \leq (R_{threat})\}$ is the set of boundaries within the distance R_{threat} considered for the collision avoidance. In essence, if a boundary is close to the drone (the distance between the boundary and the drone is less than R_{threat}), the perpendicular projection of the wall near the drone is considered a small stationary circular obstacle. As a result, this adds at most four constraints to the QP related to h_w , but in practice, there will be at maximum two constraints added to the QP when a drone is getting close to a corner. As the drone moves, the obstacle with center q_w projected

on the boundary also moves with it, creating the illusion of walls in the area. Finally, the last constraint in (10) ensures that the drone’s velocity is within its limit.

D. Fault Tolerance Mechanism

In addition to safety, the system is designed to be fault-tolerant. To this end, we introduce an indicator function $\chi_i(t)$ for each drone $i \in \mathcal{I}$ defined as

$$\chi_i(t) = \begin{cases} 1, & \text{if drone } i \text{ is fault-free,} \\ 0, & \text{otherwise.} \end{cases} \quad (12)$$

Here, a fault-free drone is defined as a drone that operates normally (i.e., without any user override, with sufficient battery, and free of failures). The set of active (fault-free) drones at time t is then given by

$$\mathcal{I}_a(t) = \{i \in \mathcal{I} \mid \chi_i(t) = 1\}. \quad (13)$$

While the system continuously monitors all drones, control computations (including collision avoidance and ergodic control) are performed only for those in $\mathcal{I}_a(t)$. Specifically, if some drones become faulty at time t_1 , they are then excluded from the active drones, and the Fourier coefficient for time $t > t_1$ is calculated as

$$c_k(t) = \frac{1}{|\mathcal{I}_a(0)| t_1} \sum_{i \in \mathcal{I}_a(0)} \int_0^{t_1} f_k(p_i(\tau)) d\tau + \frac{1}{|\mathcal{I}_a(t_1)| (t - t_1)} \sum_{i \in \mathcal{I}_a(t_1)} \int_{t_1}^t f_k(p_i(\tau)) d\tau. \quad (14)$$

E. Visualization

The targeted points from the UI are transformed into a PDF, which is then visualized in RViz alongside the drone swarm’s trajectories. The PDF and trajectory points are first converted into a grid format, which is set to span the test area. These grid points are saved as a list that is transmitted to RViz with a ROS publisher-subscriber structure. The transparency of the grid points varies according to their calculated PDF value. Targeted points with low transparency indicate less interest in that area than in areas with a brighter red color. The covered areas are marked with green, indicating the drones’ trajectories. Each drone trajectory has a slightly different shade of green to distinguish it from the other. In this way, the human operator can easily evaluate the drone swarm’s monitoring performance. An example of visualizing drones’ trajectories in RViz is shown in Fig. 2.

IV. DEMONSTRATION

In this section, we demonstrate the human-drone swarm interaction system via an experiment. The experiment also allows us to evaluate the performance of the proposed safety-ensuring ergodic control under uncertainties such as the modeling error of the drone’s dynamics and external disturbances.

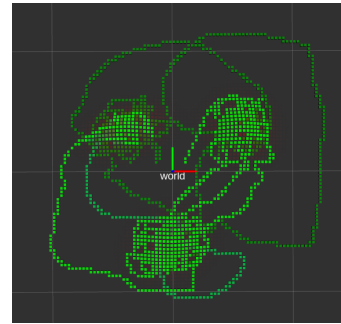


Fig. 2. RViz result shows the trajectory of the drones (green lines) when continuously monitoring the targeted points (red area).

A. Experiment Setup and Scenario

To validate our human-drone swarm interaction system, we conduct experiments using the following experimental setup: Three BitCraze Crazyflie drones, whose dimensions are 92x92x29 mm. The experimental area is designed in $\mathcal{F} = [0, 3\text{m}] \times [0, 3\text{m}]$. The drones’ positions are captured in real-time using six Vicon Vero motion capture cameras. Our setup relies on two dedicated computers: one featuring a 12th Gen Intel(R) i7-12700 processor with 32GB RAM to handle the motion capture camera, and another featuring an Intel(R) Xeon, NVIDIA Quadro P1000, 48GB RAM, Ubuntu 22.04.5 LTS, the Robot Operating System (ROS), Vicon bridge for real-time positioning, and the human-drone swarm interaction user interface. The operator’s interface is fully developed in Python and integrated with ROS.

The following information highlights technical parameters used in the experiment: The safety distance to the boundary is set as $R_{wall} = 0.17$ m, the safety distance between drones as $R_s = 0.2$ m, and the maximum velocity as $u_{\max} = 0.3$ m/s. The functions $\gamma_1(h_o) = \gamma_1 h_o^3$ and $\gamma_2(h_w) = \gamma_2 h_w^3$ where $\gamma_1 = \gamma_2 = 10$. The threat radius R_{threat} , where a drone detects possible obstacles, is set to 0.6 m.

In our experimental scenario, a human operator defines three clusters of importance, each cluster comprising five targeted points that represent regions with high importance. These target points are visualized on the user interface as a PDF, as illustrated in Fig. 3 (see Section III A for further details). The experiment runs for 120 seconds with a sampling interval of 20 ms to execute the ergodic monitoring algorithm explained in Sections III B to III D.

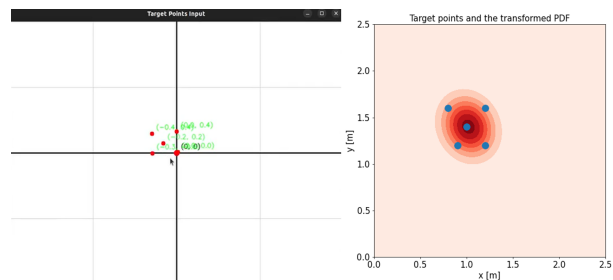


Fig. 3. (Left) The UI. (Right) Example of target points and the resulting PDF. Note that the user input points are also moved from negative space to nonnegative when transformed.

During the 120-second experiment, the drone swarm performs persistent monitoring on the defined search space. We set a landing condition in which at $t = 23$ seconds, one drone is intentionally landed to indicate a failure. This scenario aims to validate the fault-tolerance mechanism and the robustness of the ergodic control strategy, ensuring continuous monitoring despite drone failure.

B. Experimental Results

In this subsection, we highlight the performance of the drone swarm experiment. We present the drone swarm coverage capability performance in Fig. 4, where the percentage of the ergodic metric converges close to zero as the drones persistently monitor the targeted area. As the ergodic metric converges close to zero, it indicates that the drones are efficiently monitoring the targeted PDF.

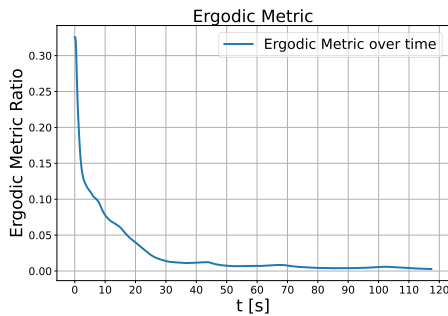


Fig. 4. The percentage of ergodic metric during the experiment.

Fig. 5 shows an empirical distribution of the swarm trajectory in blue (Right) compared to the user-defined PDF in red (Left). The drone swarm trajectory is also visualized. This result confirms the effectiveness of our ergodic coverage algorithm, since the ergodic metric compares the PDF to the traveled trajectory: if the ergodic metric were completely 0, the empirical distribution would be an exact copy of the PDF.

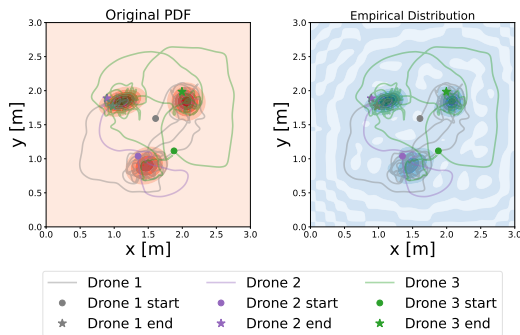


Fig. 5. (Left) Shows the PDF overlaid with the swarm trajectory. (Right) The figure shows an empirical distribution of the swarm trajectory.

Fig. 6 shows the control input behavior of the active drones. At $t = 23$ s, a drone is commanded to emergency land to simulate the fault. It can be observed that the control inputs of the drones are within the velocity

limit. Furthermore, the result in Fig. 6 also shows our fault tolerance mechanism is capable of ensuring the monitoring is running continuously, while still enabling satisfying coverage as measured by the ergodic metric.

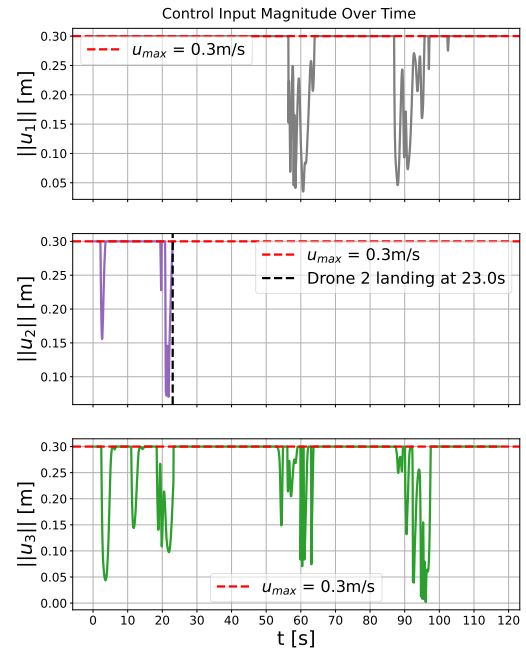


Fig. 6. Control input behavior of drones 1 - 3 during the experiment.

Fig. 7 displays snapshots of the drone trajectories during the first 60 seconds of the experiment. At $t = 23$ s, drone 2 is commanded to emergency land to simulate a fault, and the rest of the swarm continues the monitoring despite one failed drone. At this point, the drone is not considered as an obstacle by the rest of the swarm. As the experiment progresses, the drone swarm converges toward the high-density targeted areas while avoiding the boundaries and other drones. See video here: <https://www.youtube.com/watch?v=iXuJmhCI0rQ>.

Fig. 8 shows the effectiveness of the CBF through distance measurements between drones. In the middle plot, the active drones consistently maintain distance above the safety threshold (indicated by the red dashed line). In contrast, the upper and lower plots show distance below the threshold, which occurs when drone 2 has failed. In this case, the fault-free drones ignore the safety constraint of the failed drone, allowing the fault-free drones to continue the monitoring task in the safety area of the landed drone.

V. CONCLUSION & FUTURE WORK

In this study, we successfully implemented the human-drone swarm interaction system for persistent monitoring, validated through real drone experiments. The system is developed using Python and ROS, allowing smooth interactions between a human operator and drones. The human operator specifies the targeted areas, and the ergodic control processes the information to dictate that drones regularly visit and

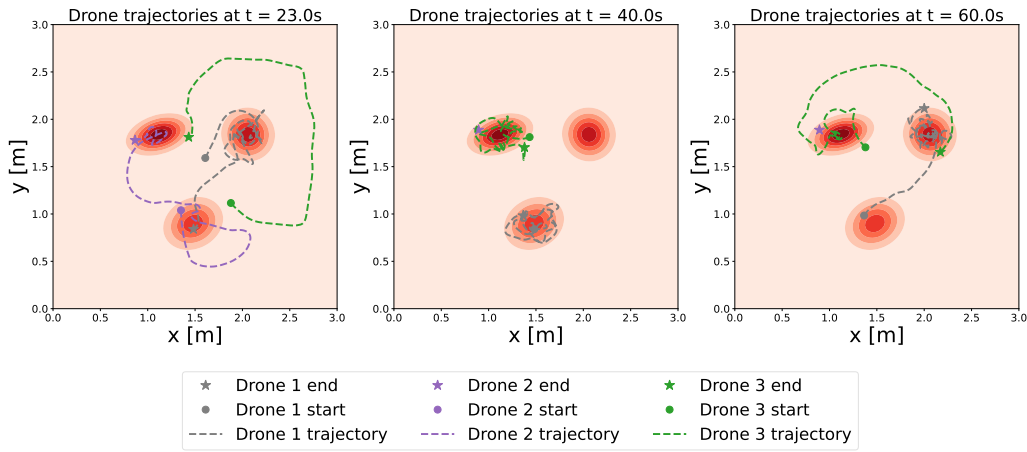


Fig. 7. Three different snapshots of drone trajectories at $t = 23$ s, $t = 40$ s, and $t = 60$ s. Video of the experiment can be viewed in <https://youtu.be/iXuJmhCl0rQ>.

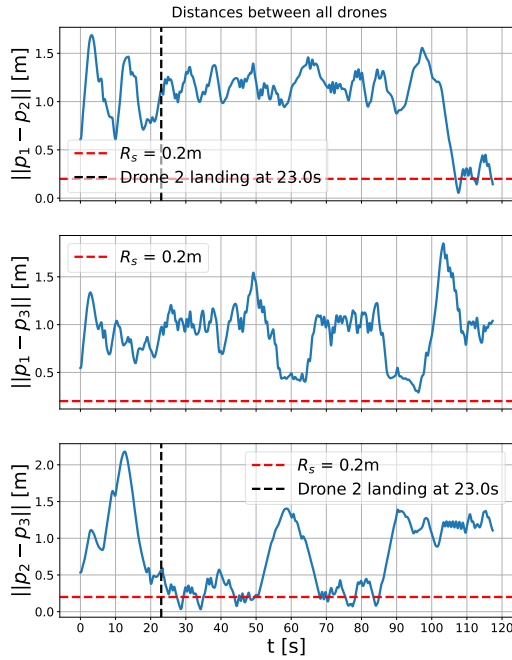


Fig. 8. Distances between the drones during the experiment.

monitor the regions. Safety is maintained through a CBF, which ensures that drones consistently avoid collisions with other drones and remain in the designated area. Additionally, a fault tolerance algorithm is incorporated to prevent mission interruption when individual drones fail. Future work will focus on distributing the system, integrating drone recharging and autonomous swarm reconfiguration under intermittent communication with advanced fault detection and enabling collaborative multi-user interfaces.

REFERENCES

[1] X. Zhan, Y. Chen, X. Chen, and W. Zhang, “Balanced multi-uav path planning for persistent monitoring,” *Robotica*, pp. 1–18, 2024.

[2] M. R. Fikri, M. W. S. Atman, Y. Nikulin, and A. Gusrialdi, “Efficient multi-robot task allocation with nonsmooth objective functions for persistent monitoring in large dispersed areas,” in *IEEE 20th International Conference on Automation Science and Engineering*, Aug. 2024, pp. 573–578.

[3] A. Ollero, J. Martínez-de-Dios, and L. Merino, “Unmanned aerial vehicles as tools for forest-fire fighting,” *Forest Ecology and Management*, vol. 234, S263, 2006.

[4] Z. Jia *et al.*, “Cooperative cognitive dynamic system in uav swarms: Reconfigurable mechanism and framework,” *IEEE Vehicular Technology Magazine*, 2024.

[5] A. Prabhakar, I. Abraham, A. Taylor, M. Schlafly, *et al.*, “Ergodic specifications for flexible swarm control: From user commands to persistent adaptation,” in *Conference on Robotics: Science and Systems (RSS)*, 2020, p. 9.

[6] Y. Diaz-Mercado, S. G. Lee, and M. Egerstedt, “Distributed dynamic density coverage for human-swarm interactions,” in *2015 American Control Conference (ACC)*, IEEE, 2015.

[7] Z. Fang *et al.*, “Eye movement-based human—swarm interaction for coverage control of mobile robots with constraints,” *IEEE Transactions on Industrial Electronics*, 2024.

[8] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control barrier functions: Theory and applications,” in *18th European control conference (ECC)*, 2019, pp. 3420–3431.

[9] C. Lerch, D. Dong, and I. Abraham, “Safety-critical ergodic exploration in cluttered environments via control barrier functions,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2023.

[10] G. Mathew and I. Mezić, “Metrics for ergodicity and design of ergodic dynamics for multi-agent systems,” *Physica D: Nonlinear Phenomena*, vol. 240, no. 4–5, pp. 432–442, 2011.

[11] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[12] A. Gusrialdi, Z. Qu, and M. A. Simaan, “Distributed scheduling and cooperative control for charging of electric vehicles at highway service stations,” *IEEE Trans. on Intelligent Transportation Systems*, vol. 18, no. 10, pp. 2713–2727, 2017.

[13] M. W. S. Atman, M. R. Fikri, K. Priandana, and A. Gusrialdi, “A two-layer control framework for persistent monitoring of a large area with a robotic sensor network,” *IEEE Access*, vol. 12, pp. 4153–4165, 2024.