2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

# Onboard Controller Design for Nano UAV Swarm in Operator-Guided Collective Behaviors*

Tugay Alperen Karagüzel[1], Victor Retamal[1], Eliseo Ferrante[1,2]

*Abstract*— **In this paper, we present a swarm of Crazyflie nano-drones. The swarm can show various collective behaviors: Flocking, gradient following, going to a chosen point, formation, and scattered search of the environment. The methodology behind the behaviors is executed entirely on-board. Crazyflies use a common radio channel to share positions with each other. If desired, an operator can use the same channel and start, end, change or guide the collective behaviors online during the flight. We use the virtual force vectors and modify the way they are combined to achieve different behaviors instead of developing unique algorithms for each. This allows us to develop more collective behavior types with less effort. In the results, we show a detailed analysis of the behaviors and assess the coordination and the safety of the agents in addition to the performance as a collective. We conclude that our swarm of 6 Crazyflies was successful in the desired behaviors.**

## I. INTRODUCTION

Unmanned aerial vehicles (UAVs) present significant advantages in reconnaissance tasks compared to the platforms in other mediums with their flexibility and field of view [1]. Advancements in UAV control, navigation, and communication systems offer new possibilities. However, single UAV applications may still be inadequate or inefficient. In this case, swarms of UAVs show great potential in leveraging the individual capabilities [2]. A swarm is a group of autonomous entities where the desired collective behavior arises from interactions among entities and their environments, resulting in capabilities not present in individuals [3].

Studies on autonomous UAV swarms can be found both indoors and outdoors. Outdoor swarms generally consist of platforms with advanced sensors (GPS, LIDAR, RGB-D cameras, etc.) [4]. Although the results are concordantly impressive, experimenting outdoors can be expensive, time-consuming, and dangerous. Whereas indoors, UAV swarms present a cheaper, safer, and easier to maintain alternative. Still, indoor UAVs need to be smaller and lighter, bringing the constraints on flight time and capabilities of the platforms in sensing and computation. The constraints are overcome mainly by employing very accurate positioning systems and running complex algorithms on a central computer.

Hereby, we see great value in developing an on-board controlled indoor UAV swarm operating with an off the shelf UWB (Ultra wide band) based positioning system. We aim for a real life implementation of a swarm capable of demonstrating well known collective behaviors like flocking in addition to novel behaviors like emergent sensing [5], [6]. Moreover, a UAV swarm must perform real-world tasks such as reaching a goal position as a coordinated swarm, forming shapes, or conducting individual area searches. The significance of this feature was examined in [7], which explored how flocking behaviors can be combined with collective objectives using lightweight algorithms without relying on optimization techniques or filters. In this paper, we center on utilizing a nano drone, namely Crazyflie, and present a comprehensive methodology that is reproducible and modifiable. This approach has the potential to be applied to various tasks, as it involves easy-to-implement lightweight methods that can be generalized. Importantly, these methods run on-board the drone, in a fully distributed fashion, without requiring a central computation unit.

### A. Related Work

The research on outdoor UAV swarms demonstrates solutions for coordinated collective behavior. In [8] a swarm of 30 UAVs flocks in the open air with a considerable speed, ensuring the safety of all in the meanwhile. Another example at [9] shows the capability of a multi-vehicle system to flock in a dense forest which requires a very high spatial awareness. There exists more functional applications like search and rescue with cooperating UAVs [10]. However, the requirements are usually complex and expensive (communication over larger distances or sophisticated sensors), therefore unsuitable for rapid prototyping and testing on more limited platforms.

Other approaches have been proposed to solve the problem of achieving robust collective behaviors with indoor UAVs. We will be specifically focusing on the ones with the Crazyflie, which is one of the most used indoors alternatives. In [11], distributed model predictive control is utilized to guide a swarm of Crazyflies flocking in the desired trajectory while avoiding collisions. A motion capture system supplies position feedback. A formation control with collision avoidance with multiple Crazyflies is presented in [12]. By communicating the full state, each agent calculates the points of the desired trajectory constrained to a specific set of rules. In [13], a model-free control method with unknown system dynamics is utilized to achieve flocking. In [14], a novel method applying mean-field approximation was proposed to prevent unattainable control inputs and achieve reliable real-time collision-avoidance for safe flocking by operating with a motion capture system. While all the methods above produce accurate trajectories, they require more computational power

*This work was supported by Technology Innovation Institute Abu Dhabi.
[1]Vrije Universiteit Amsterdam, Noord-Holland, The Netherlands, t.a.karaguzel@vu.nl, v.retamalguiberteau@student.vu.nl
[2]Technology Innovation Institute Abu Dhabi, United Arab Emirates e.ferrante@vu.nl

than Crazyflie has. Hence, the implementation is always tied to an external computer inducing communication and deployment constraints. As a standing out example, [15] demonstrates formations with on-board controlled Crazyflies trained by deep reinforcement learning. However, agents rely on a motion capture system and the episodic training period.

The work in [16] demonstrates a successful gas source localization with an on-board controlled Crazyflie swarm, using a special sensor. Although a powerful functionality of the swarm is presented, capabilities are designed for this specific task and do not involve flexibility in any collective behavior. In another study [17], a Crazyflie swarm is presented where agents can localize each other by using UWB signals and inertial measurements. Agents can maintain a formation and show leader following behavior. [17] shows the possibility of the on-board control and relative localization simultaneously. Unfortunately, there is no user involvement or flexibility to modify and/or guide collective behavior.

In this paper, we present a swarm of Crazyflies where agents are controlled on-board and use radio communication to share limited information with others. They estimate positions on-board with the help of a UWB positioning system. The work presented in [18] and discussed in Section II lays the foundation for our implementation to demonstrate a range of collective behaviors, such as flocking, gradient following, formation, go-to-point, and area search. Of particular interest to us is the reactive and simplistic nature of the algorithm, with emphasis placed on motion control, that allowed us to adapt the implementation from ground robots to UAVs. An operator can guide these behaviors and dynamically switch on air by a simple parameter update over the radio. If the operator intervention is not desired, the swarm can operate without a central computer. None of the methods depends on the number of peers or identities. Hence, an arbitrary number can be discarded from the swarm during operation. We believe the intuitive and reproducible methodology in our flexible framework will be a proper test bed for the researchers to develop further capabilities.

## II. Methodology

Agents of the swarm are controlled individually and move at a fixed altitude. Agents can translate only in their heading direction. We impose this constraint to be able to apply our methodology, not because the platform structure forces it. The real yaw angle never changes; heading only implies a virtual direction of motion. The heading is changed by the angular speed, and the translation will be according to the linear speed.

Agents exchange positions with peers to estimate the relative distances and bearings. Only in flocking, they also exchange headings. Linear and angular speeds are calculated upon this information by calculating virtual force vectors.

**Flocking** is a type of collective motion where the swarm wanders within the boundaries as a cohesive and ordered group towards an emerging direction rather than a specified one. The method of flocking [18] involves three components: Proximal control vector ($\vec{p}_i$), alignment control vector ($\vec{h}_i$) and boundary avoidance vector ($\vec{r_i^b}$). $\vec{p}_i$ acts as a virtual spring between peers: It produces repulsion if the peer is closer than the desired distance and becomes attractive if the peer is further. $\vec{p}_i$ exists in all behaviors to avoid collisions, and it is modified accordingly if the cohesion of the group is not desired. $\vec{p}_i$ is calculated as follows:

$$\vec{p}_i = \sum_{m \in N} -\epsilon \left[ 2 \frac{\sigma_i^4}{(d_i^m)^5} - \frac{\sigma_i^2}{(d_i^m)^3} \right] e^{j\phi_i^m} \tag{1}$$

Here, $\epsilon$ is the proximal vector strength. The relative distance of the neighbour $m$ (among all $N$) is $d_i^m$ and the relative bearing angle of $m$ is $\phi_i^m$. $\sigma_i$ is the term regulating the desired distance ($d_{des}$) of the focal agent to peers in a way such that: $d_{des} = \sigma_i \sqrt{2}$.

The alignment control vector ($\vec{h}_i$) is maintains similar headings with the peers. The focal agent exchanges heading information with the peers and calculates a unit vector with the angle of the average heading of the peers.

$$\vec{h}_i = \frac{e^{j\theta_0} + \sum_{m \in \mathcal{N}} e^{j\theta_m}}{||e^{j\theta_0} + \sum_{m \in \mathcal{N}} e^{j\theta_m}||} \tag{2}$$

Here, $\theta_0$ is the heading angle of the focal agent, and $\theta_m$ stands for the peer's heading angle, both in a common reference frame.

Finally, the boundary avoidance vector ($\vec{r_i^b}$) [19] ensures a safe distance from boundaries only in cases where the focal agent is closer than the specified distance ($D_b$) to a boundary or multiple boundaries.

$$\vec{r_i^b} = \sum_{b \in B} k_{rep} \left( \frac{1}{L_b} - \frac{1}{L_0} \right) \left( \frac{\vec{p_i^b}}{L_b^3} \right) \tag{3}$$

Effects from all sensed boundaries ($B$) are calculated individually for each boundary ($b$). In above, $k_{rep}$ stands for the strength of $\vec{r_i^b}$, $L_0$ for the relaxation threshold and $L_b$ for the shortest distance to boundary $b$. The unit vector $\vec{p_i^b}$ indicates the direction of the closest point on $b$.

The total force vector is calculated by combining all vectors with corresponding weight coefficients $\alpha$, $\beta$, and $\gamma$:

$$\vec{f_i} = \alpha \vec{p}_i + \beta \vec{h}_i + \gamma \vec{r}_i \tag{4}$$

Next, the translation and angular speeds ($U_i$ and $\omega_i$) will be produced as follows:

$$U_i = K_1 f_x + u_{add}, \quad \omega_i = K_2 f_y \tag{5}$$

In the above, $f_x$ and $f_y$ are the components of $\vec{f_i}$ on agent's local frame. This local frame is defined such that the x-axis of this orthogonal and right-hand sided frame is parallel to the heading of the focal agent. $K_1$ and $K_2$ are respective gains for the speeds. Finally, $u_{add}$ stands for the additional translation speed to increase self propulsion of the agents.

The parameters for flocking is chosen as: $\alpha = 1.0$, $\beta = 2.0$, $\gamma = 1.0$, $\epsilon = 12.0$, $L_0 = 0.5$, $K1 = 0.1$, $K2 = 0.6$, $\sigma_i = 0.6$, $D_b = 0.5$ and $u_{add} = 0.05$. $\epsilon$, $\gamma$, $D_b$, and $L_0$ values are constant for all behaviors. In addition, for all behaviors, $U_i$ is clamped in the range [0, 0.15] and $w_i$ in the range [0, $\pi$/2]. Parameters are selected experimentally and manually adjusted to ensure group cohesion, prevent dangerously close agent-to-agent interactions, and maintain a smoothly moving swarm. In addition, we pay regard to the limits of the experimental platform. In other specific behaviors, we also evaluate whether the swarm achieves the desired collective objective.

**Gradient following** is the behavior where agents calculate a local value in a pre-defined gradient model by using their positions. It resembles the case where agents sense a scalar physical property, such as light intensity, a chemical substance, or even a radioactive particle. Depending on the local value, agents modify only their desired distance coefficients ($\sigma_i$) as follows [5]:

$$\sigma_i = \sigma_L + (G^\circ / G_{max})\sigma_r \tag{6}$$

Here, $\sigma_L$ is the lowest allowed value of $\sigma_i$ and $\sigma_r$ is the variance range. $G^\circ$ is the local value of the gradient (in the range [0, 255]), and $G_{max}$ resembles the maximum local value. In the gradient following, alignment control is not required and used; hence $\beta$ is always set to 0. The total force vector ($\vec{f_i}$) is calculated as in Eq. 4 and speeds are produced as in Eq. 5. The swarm is expected to follow an increasing gradient and wander around the maximum. This is an emergent behavior since there is no information exchange about the local values ($G^\circ$) between agents. The focal agent with a higher local value increases its desired distance. No other peer is informed about this irregularity. Thus, the focal agent is repelled by the neighbor peers to comply with the new desired distance. On the other hand, the neighbors feel attracted to the focal agent because of their shorter desired distance. This chaser-evader situation navigates the swarm toward the direction in which the focal agent has experienced the first high local value. The parameters for gradient are chosen as: $\alpha = 1.0, \beta = 0.0, K1 = 0.3, K2 = 0.1, \sigma_l = 0.5$, $\sigma_r = 0.3$, $G_{max} = 255$ and $u_{add} = 0.05$.

Goal based behaviors work on the basis of goal points, as the name implies. To extend, when a goal point ($P_g$) is assigned by the focal agent, the agent calculates a goal attraction vector ($\vec{g_i}$). Agents do not necessarily have the same location as the goal point. In goal based behaviors, alignment control is not applied; hence $\beta$ is set to 0. The corresponding weight coefficient for $\vec{g_i}$ is $\kappa$, which is chosen as 10 for the goal based behaviors. $\vec{g_i}$ is calculated as follows, where $\phi_i^g$ represents the angle from the agent location to the goal point: $\vec{g_i} = e^{j\phi_i^g}$.

The total force vector ($\vec{f_i}$) for goal based behaviours is as follows:

$$\vec{f_i} = \alpha\vec{p_i} + \gamma\vec{r_i} + \kappa\vec{g_i} \tag{7}$$

The speed commands for goal based behaviors are produced as in in Eq. 5. We introduce three types of goal based behavior: Go to point, circle formation, and scattered search.

**Go To Point** behavior makes the swarm location a controllable entity. When the goal point is updated, each agent calculates an attractive virtual force and moves with its influence. Agents are expected to reach the given goal point. Since all agents are given the same goal point ($P_g$), conflict occurs between them. Yet, this conflict is emergently solved by the virtual force vectors with the cost of not guaranteeing that every agent will be at the same location, which is unrealistic. Instead, the swarm centroid is expected to be located at $P_g$. The parameters for go to point behavior are chosen as: $\alpha = 1.0$, $\beta = 0.0$, $K1 = 0.3$, $K2 = 0.1$, $\sigma_i = 0.6$ and $u_{add} = 0.05$.

**Circle formation** is the behavior in which agents assign a goal point, but in addition, they calculate an offset for the pre-defined formation. Since each agent is designed to have a unique angle around the goal point, they also have distinct goal points ($P_g$). The desired behavior is a circle shape formed by the swarm around a point given by the operator. When a given point is shared with all agents, each agent uses its pre-defined offset around it to calculate its distinct $P_g$. Although the calculation of $\vec{f_i}$ is as in Eq. 7, a modification is needed, which ignores all $\vec{p_i}$ calculated for peers further than $\%70$ of $d_{des}$. The motivation here is to eliminate attraction from peers, which is not needed for the formation since the repulsion is enough to avoid collisions and reach distinct $P_g$s. Lastly, since the increased self-propulsion is not desired for maintaining a formation, $u_{add}$ is set to 0. The parameters are chosen for the formation behavior as: $\alpha = 1.0$, $\beta = 0.0$, $K1 = 0.1$, $K2 = 0.6$, $\sigma_i = 0.6$ and $u_{add} = 0.00$.

**Scattered search** is the behavior in which the agents separate to conduct an individual search around different pre-defined locations. Each agent first assigns the start location of the searching path as $P_g$, which will be the lower left corner of the square-shaped searching path. Later, the x and y components ($P_{gx}, P_{gy}$) of $P_g$ is incremented by 0.5 m to produce the path. Each corner of the square is accepted to be reached if the agent becomes closer than 10 cm. The parameters are chosen for the scattered search as: $\alpha = 1.0$, $\beta = 0.0$, $K1 = 0.3$, $K2 = 0.1$, $\sigma_i = 0.6$ and $u_{add} = 0.05$. Although more sophisticated scanning and coverage algorithms can be employed here to guarantee a full and adaptive coverage, we show this simple approach in our case as a proof of concept.

## III. Implementation and Experimental Setup

Presented algorithms are implemented on-board with the Crazyflie. For implementation, the *App-Layer* functionality is used [20]. This allows users to implement desired behaviors to run autonomously without requiring a central computer. For information exchange between agents, *Peer to Peer API* is employed [21]. This API ($P2P$) makes use of the radio communication capability of Crazyflie. Each agent broadcasts its position (x,y coordinates, and only for flocking, the heading) on the same channel while simultaneously receiving messages from other agents. In theory, this happens only within a pre-defined sensing range yet in our case, due to the limited size of the arena, all agents remain within the sensing range of each other. However, the methodology has been demonstrated to function effectively on simulation even

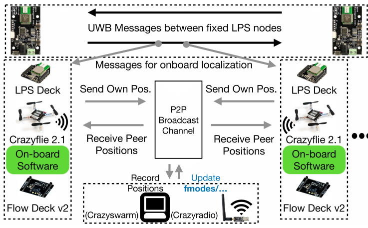



Fig. 1: System architecture

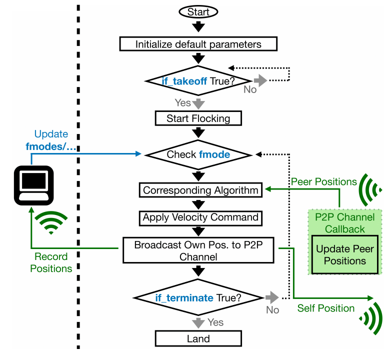



Fig. 2: Flow in the on-board implementation

if this is not the case [5]. The localization is done on-board as well, with a UWB localization system. Crazyflie also has an extension that carries a flow camera and altitude sensor, which maintains a stable flight. Finally, the central computer collects data for logging, and it is used for triggering commands by changing parameters to start the experiment or to change the collective behavior. We use Crazyswarm [22] to exchange information with Crazyflies from the computer side. The source for the computer side can be accessed here [23]. General system architecture and information flow are visualized in Figure 1.

The flow of the $App-Layer$ implementation can be observed in Figure 2. The firmware, including the software we used for the experiments, can be accessed here [24]. When Crazyflie is powered on the ground, it waits in an idle state and checks the parameter **fmodes/if_takeoff**. In the firmware, all parameters under the group **fmodes** are added and used for the behaviors in this paper. If the operator modifies the required parameter for takeoff, the agent takes off and checks the corresponding parameter (**fmodes/fmode**) to decide which behavior to show, which dictates flocking in default. After the chosen algorithm calculated $U_i$ and $w_i$, they are converted to speed commands which are valid in the real world ($V_x$ and $V_y$). This conversion is required because $U_i$ represents the translation only in the heading direction of the agent. This translation is converted to a vector in the real-world frame of reference using the current heading angle. Later, the agent broadcasts its position on $P2P$. The calculations are repeated 20 times a second. Meanwhile, a callback function updates the peer positions in parallel. When one of the goal based behaviors is active, the goal point $P_g$ plays an important role and should be updated. Corresponding parameters for that are **fmodes/g_x** and **fmodes/g_y** for $P_{gx}$ and $P_{gy}$ correspondingly. Finally, the parameter **fmodes/if_terminate** is checked to decide if to land or not.

The size of our flight arena is [6.5 x 4.0] m. The fixed altitude for the agents is chosen as 0.5 m. The maximum allowed flight duration is 200 seconds. If the operator does not terminate the flight, the swarm will land after this duration automatically. The central computer logs the states of the agents at 1 Hz.

## IV. Results and Discussion

We analyze the resulting collective behaviors by presenting agent trajectories and using specific metrics to evaluate swarm performance. Results are discussed after presenting figures for each behavior. Experimental swarm photos are shown in Figure 3. Videos can be found at [24].

Figure 4 depicts the results for flocking, displaying the centroid trajectory, swarm order, and minimum distance to boundaries. Swarm order ($\Psi$) is a metric assessing the heading consensus among all agents, with a maximum value of 1 indicating full agreement and a value close to zero indicating divergence. The order is calculated as follows, where $\theta_i$ stands for the heading of each agent among all ($N_A$):

$$\Psi = \| \sum_{i \in N_A} e^{j\theta_i} \| / N_A \tag{8}$$

When Figure 4 is observed, it can be seen that the swarm traveled with a high order for most of the duration. Since there is also a boundary repulsion effect, the expected behavior is a smooth trajectory of the swarm with occasional bounces from the boundaries. The bounces can be spotted in Figure 4b, where the minimum distance to the boundary line produces pits. When the order and minimum distance lines are observed together, the correlation between them can also be spotted. This correlation is expected since when the swarm becomes closer than $D_b$ to a boundary, agents start to change headings under the influence of the boundary repulsion. Since the control of each agent is on-board and individual, the change of headings is not synchronous among the swarm. This is the reason behind having simultaneous pits in the order and the minimum distance. However, the swarm was capable enough to align quickly until facing another boundary. During the experiment, separation of the swarm and collision between agents are not observed.

Figure 5 shows the results of gradient following, where a unimodal and circular gradient model is used in the

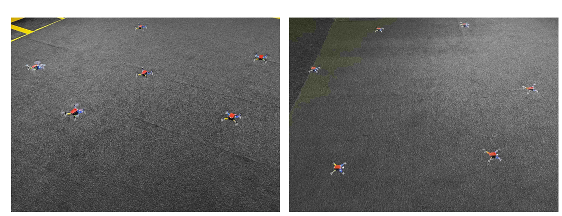
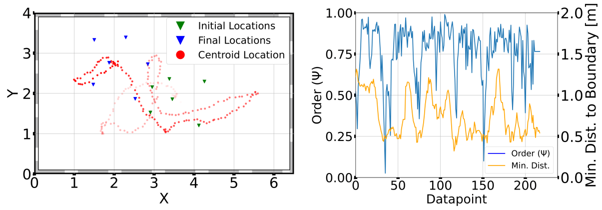
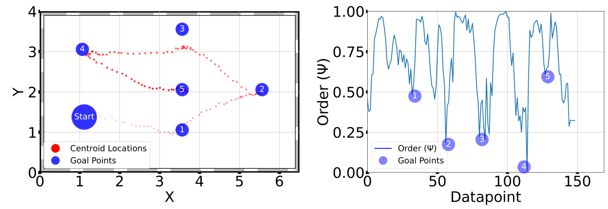


Fig. 3: Crazyflie swarm; flocking (left) and formation (right)




(a) Centroid trajectory (b) Order and min. distance to any boundary

Fig. 4: Flocking




(a) Centroid trajectory (b) Order

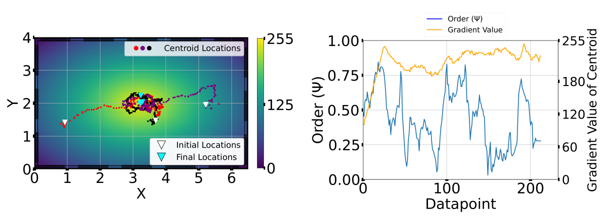
Fig. 6: Go to point

Crazyflies. The swarm is initiated from three different start points, including the left side, right side, and middle, to demonstrate the capability of the swarm to reach and remain close to the maximum value of the gradient, independent of the starting location. Also, Figure 5 shows that the swarm successfully arrived at and remained near the maximum value of the gradient, regardless of the starting point. There were no collisions or group separations. As gradient following does not use alignment control, a lower swarm order, as shown in Figure 5b, is expected. In Figure 5b, the experiment starting from the left is shown, but it does not affect the performance, as evidenced by the gradient value line. The peaks and ascents in the gradient value coincide with the order, indicating that following the increasing gradient improves alignment without using alignment control. This finding is important for gradient following and other approaches without alignment control. If on-board relative localization is achieved in the future, estimating peer positions will be relatively easy compared to estimating their directions of motion. Thus, not needing alignment control increases the real-life portability of the framework. Furthermore, the agents achieve a centroid gradient value near 255 and maintain it without exchanging information on gradient values, which could be similar to an on-board sensor. This emergent behavior can be used to develop a swarm capable of locating a real source with a specialized sensor.

In Figure 6, we present centroid trajectories and the order of the swarm during go to point behavior. The $P_g$s sent to swarm during the experiment can be located in Figure 6a and the timing of the $P_g$ updates can be seen in Figure 6b. In Figure 6a, the swarm centroid trajectories exhibit a successful go-to-point behavior, where the marked $P_g$s are closely approached by the swarm. However, the exception is observed in point (3), which is positioned in close proximity to the upper boundary. As a result, some agents need to approach the boundary closer than $D_b$ to match the centroid location with (3). The desired behavior is observed where the repulsion effect dominates over the goal effect, preventing the swarm from escaping the boundary even when the goal point is close to it. Although go to point behavior does not employ alignment control, we observe high order plateaus followed by pits. By analyzing the timing of ascends to these plateaus, we conclude that an update of the common goal point improves swarm alignment. After reaching the goal point, the order decreases due to the frequently changing relative direction of the goal point to the agents.

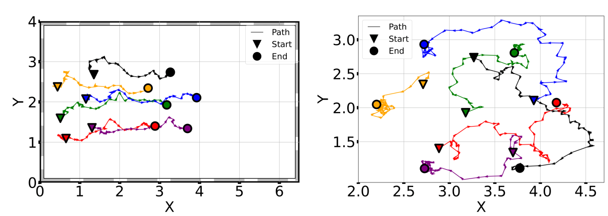
In Figure 7, we present the trajectories of agents during different phases of the circle formation experiment and the minimum distance between all agents. The first phase of the experiment is ”Go to”, where the swarm moves from the initial location to another location where the formation will be produced. Next is ”Forming”, where the formation is realized. The last is ”Reversing” where each agent changes place with the agent on the opposite side. In other words, the agent on the circle with 180° different location.

In Figure 7a, we observe the same behavior with any two consecutive points in Figure 6a. Yet differently, here we can observe individual trajectories. As desired, all agents move in similar directions at a safe distance. In Figure 7b, circle formation starts. We can observe that each agent arrives

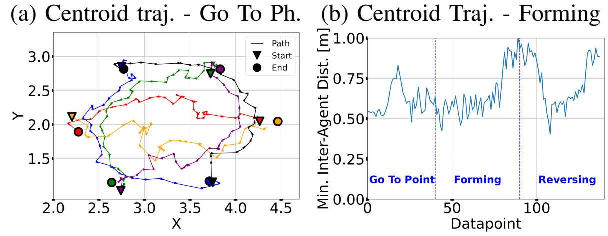



(a) Centroid trajectory (b) Order and the local grad. value of centroid

Fig. 5: Gradient following




(a) Centroid traj. - Go To Ph. (b) Centroid Traj. - Forming



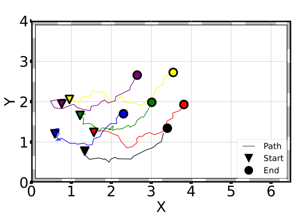

(c) Centroid Traj. - Reversing (d) Min. inter-agent distance

Fig. 7: Circle formation

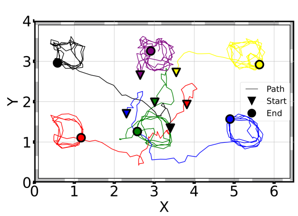
at the point it calculated by adding an offset to the given $P_g$. When Figure 7b and 7d are considered together, we can conclude that during the placement, agents avoided any collision. This successful avoidance within a limited space, dictated by only the proximal control force vector, shows how powerful this simple method is in ensuring safety within a swarm. Figure 7c shows agent trajectories during reversing. This case is the most challenging one. It requires 6 agents to solve a problem involving all of them. They have to do this distributed and without any explicit communication specific to the problem. Nevertheless, we see an emergent solution. First, agents approach the center point. Second, the situation becomes a lock, so they stop approaching each other. The lock can be observed as a pit in Figure 7d. Next, they start to rotate around the center point until each agent arrives at the corresponding goal point. Having this solution emergently produced by a simple effect instead of engineering it highlights the capability of our methodology to produce emergent solutions to different problems.

The results for the scattered search are presented in Figure 8. Scattered search has 3 phases: Swarm goes to an arbitrary point, each agent splits for individually predefined search areas and conducts periodic movements, and finally, the swarm gathers around a chosen point. To analyze the behavior, we presented agent trajectories and minimum distances observed between agents.

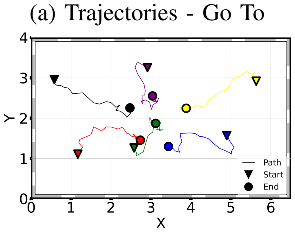
Figure 8a presents the agent trajectories during the go to phase. Our observation concludes that it is a successful collective behavior when the inter-agent distances and final points are considered. Although it is in the initial stage, we can detect a potentially dangerous approach between agents (in Figure 8d, Go To phase). Nevertheless, we see that agents recovered quickly from that possibility. In Figure 8b, we see agents move to their corresponding search areas and start the periodic motion. Before starting, agents are not located such that everyone chooses the point with minimum path conflicts. Nevertheless, the swarm members manage to solve conflicts emergently to proceed to the search areas. Since the criteria to proceed to the next step in the search path is approaching the current one closer than 10 cm, trajectories are not squares as designed but have smoothed corners. When we consider the accuracy of the positioning system (~10 cm) and the limited space given to Crazyflies, we can foresee that these disturbances will be less noticeable and less effective on a larger scale. Finally, in Figure 8c, we observe the agents are gathering around $P_g$ by leaving their corresponding search areas. This capability shows us that the bond between swarm agents can be created and destroyed on demand, without switching algorithms but rather only changing the goal points. When the swarm is formed again, inter-agent distances return to the values we also observed in other behaviors. These values agree with the parameter choices we have made in our methodology. An important note here is that by changing the parameter values, we can obtain variances in the collective states and dynamics. Although this analysis may reveal more possibilities for the swarm, we envision a detailed investigation as a future work, beyond the extent of this paper.



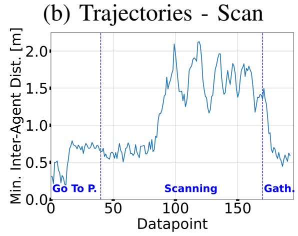

(a) Trajectories - Go To




(b) Trajectories - Scan




(c) Trajectories - Gather




(d) Min. inter-agent distance

Fig. 8: Scattered Search

## V. Conclusions

This paper presents and analyzes collective behaviors that serve as fundamental building blocks for capable and onboard controlled UAV swarms. Flocking demonstrates the real-time coordination and alignment of agents' motions, while gradient following showcases the swarm's emergent sensing capabilities, which can be adapted to different physical variables. This is strong proof of how being a swarm can leverage the individual capabilities of the agents. The essential element of the goal based behaviors, "go to point", establishes a necessary connection between a UAV swarm and the real use cases. The formation and search experiments exemplify applications with "go to point". The formation experiment shows us that a challenging conflict between 6 Crazyflies can be solved emergently and distributed, even within a very limited space for the solution. The solution we see in the formation reverse experiment is a resulting collective behavior, emergent rather than being influenced or fully engineered by us. Finally, scattered search demonstrates the capability of dynamically bonding and breaking coordination between agents, which adds more flexibility and use case for a UAV swarm.

The methodology behind all the collective behaviors proves that a simple yet robust algorithm can be sufficient for a capable swarm. This swarm can fulfill various tasks which require strong coordination or involve individual actions of the agents. Thanks to its simplicity, the methodology is applicable on-board with a nano-drone, Crazyflie. Up to our knowledge, for the first time, an on-board flocking is shown with Crazyflie. In addition, the framework we share constitutes a base for other applications since the behaviors are easy to modify and improve. We believe that with the contributions of others in the future, the framework can be used for developing and testing more collective behaviors. These behaviors can be for indoors applications in addition to being a safe test bed for outdoor swarms.

# REFERENCES


[1] Y. Liu, Z. Luo, Z. Liu, J. Shi, and G. Cheng, "Cooperative routing problem for ground vehicle and unmanned aerial vehicle: The application on intelligence, surveillance, and reconnaissance missions," *IEEE Access*, vol. 7, pp. 63 504–63 518, 2019, https://doi.org/10.1109/ACCESS.2019.2914352.

[2] M. Campion, P. Ranganathan, and S. Faruque, "Uav swarm communication and control architectures: a review," *Journal of Unmanned Vehicle Systems*, vol. 7, no. 2, pp. 93–106, 2019, https://doi.org/10.1139/juvs-2018-0009.

[3] E. Şahin, "Swarm robotics: From sources of inspiration to domains of application," in *International workshop on swarm robotics*. Springer, 2004, pp. 10–20, https://doi.org/10.1007/978-3-540-30552-1_2.

[4] M. Coppola, K. N. McGuire, C. De Wagter, and G. C. De Croon, "A survey on swarming with micro air vehicles: Fundamental challenges and constraints," *Frontiers in Robotics and AI*, vol. 7, p. 18, 2020, https://doi.org/10.3389/frobt.2020.00018.

[5] T. A. Karagüzel, A. E. Turgut, A. Eiben, and E. Ferrante, "Collective gradient perception with a flying robot swarm," *Swarm Intelligence*, pp. 1–30, 2022, https://doi.org/10.1007/s11721-022-00220-1.

[6] A. M. Berdahl, A. B. Kao, A. Flack, P. A. Westley, E. A. Codling, I. D. Couzin, A. I. Dell, and D. Biro, "Collective animal navigation and migratory culture: from theoretical models to empirical evidence," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 373, no. 1746, p. 20170009, 2018, https://doi.org/10.1098/rstb.2017.0009.

[7] "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Transactions on automatic control*, vol. 51, no. 3, pp. 401–420, 2006, https://doi.org/10.1109/TAC.2005.864190.

[8] G. Vásárhelyi, C. Virágh, G. Somorjai, T. Nepusz, A. E. Eiben, and T. Vicsek, "Optimized flocking of autonomous drones in confined environments," *Science Robotics*, vol. 3, no. 20, p. eaat3536, 2018, https://doi.org/10.1126/scirobotics.aat3536.

[9] X. Zhou, X. Wen, Z. Wang, Y. Gao, H. Li, Q. Wang, T. Yang, H. Lu, Y. Cao, C. Xu, and F. Gao, "Swarm of micro flying robots in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm5954, 2022, https://doi.org/10.1126/scirobotics.abm5954.

[10] Y. Cao, F. Qi, Y. Jing, M. Zhu, T. Lei, Z. Li, J. Xia, J. Wang, and G. Lu, "Mission chain driven unmanned aerial vehicle swarms cooperation for the search and rescue of outdoor injured human targets," *Drones*, vol. 6, no. 6, 2022, https://doi.org/10.3390/drones6060138.

[11] Y. Lyu, J. Hu, B. M. Chen, C. Zhao, and Q. Pan, "Multivehicle flocking with collision avoidance via distributed model predictive control," *IEEE Transactions on Cybernetics*, vol. 51, no. 5, pp. 2651–2662, 2021, https://doi.org/10.1109/TCYB.2019.2944892.

[12] A. T. Nguyen, J.-W. Lee, T. B. Nguyen, and S. K. Hong, "Collision-free formation control of multiple nano-quadrotors," 2021, https://doi.org/10.48550/ARXIV.2107.13203.

[13] A. Brandstätter, S. A. Smolka, S. D. Stoller, A. Tiwari, and R. Grosu, "Multi-agent spatial predictive control with application to drone flocking (extended version)," 2022, https://doi.org/10.48550/ARXIV.2203.16960.

[14] M. Fernando, "Online flocking control of uavs with mean-field approximation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 8977–8983, https://doi.org/10.1109/ICRA48506.2021.9560899.

[15] S. Batra, Z. Huang, A. Petrenko, T. Kumar, A. Molchanov, and G. S. Sukhatme, "Decentralized control of quadrotor swarms with end-to-end deep reinforcement learning," in *Proceedings of the 5th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 164. PMLR, 2022, pp. 576–586.

[16] B. P. Duisterhof, S. Li, J. Burgués, V. J. Reddi, and G. C. H. E. de Croon, "Sniffy bug: A fully autonomous swarm of gas-seeking nano quadcopters in cluttered environments," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 9099–9106, https://doi.org/10.1109/IROS51168.2021.9636217.

[17] S. Li, M. Coppola, C. De Wagter, and G. C. H. E. de Croon, "An autonomous swarm of micro flying robots with range-based relative localization," 2020, https://doi.org/10.48550/ARXIV.2003.05853.

[18] E. Ferrante, A. E. Turgut, C. Huepe, A. Stranieri, C. Pinciroli, and M. Dorigo, "Self-organized flocking with a mobile robot swarm: a novel motion control method," *Adaptive Behavior*, vol. 20, no. 6, pp. 460–477, 2012, https://doi.org/10.1177/1059712312462248.

[19] B. Khaldi and F. Cherif, "A virtual viscoelastic based aggregation model for self-organization of swarm robots system," in *Towards Autonomous Robotic Systems*, L. Alboul, D. Damian, and J. M. Aitken, Eds. Springer International Publishing, 2016, pp. 202–213, https://doi.org/10.1007/978-3-319-40379-3_21.

[20] (2022, Sept.) Bitcraze - crazyflie firmware - app layer documentation. [Online]. Available: https://www.bitcraze.io/documentation/repository/crazyflie-firmware/master/userguides/app_layer/

[21] (2022, Sept.) Bitcraze - crazyflie firmware - p2p documentation. [Online]. Available: https://www.bitcraze.io/documentation/repository/crazyflie-firmware/2020.02/p2p_api/

[22] J. A. Preiss*, W. Hönig*, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A large nano-quadcopter swarm," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3299–3304, https://github.com/USC-ACTLab/crazyswarm. [Online]. Available: https://doi.org/10.1109/ICRA.2017.7989376

[23] (2022, Sept.) Crazyswarm framework source used in the experiments. [Online]. Available: https://github.com/RetamalVictor/crazyswarm-VU

[24] (2022, Sept.) Crazyflie firmware source used in the experiments. [Online]. Available: https://github.com/RetamalVictor/crazyflie-firmware-VU