Hierarchical Control of Smart Particle Swarms

Vivek Shankar Varadharajan, Sepand Dyanatkar and Giovanni Beltrame

Abstract—We present a method for the control of robot swarms using two subsets of robots: a larger group of simple, oblivious robots (which we call the workers) that is governed by simple local attraction forces, and a smaller group (the guides) with sufficient mission knowledge to create and displace a desired worker formation by operating on the local forces of the workers. The guides coordinate to shape the workers like smart particles by changing their interaction parameters. We study the approach with a large scale experiment in a physics based simulator with up to 5000 robots forming three different patterns. Our experiments reveal that the approach scales well with increasing robot numbers, and presents little pattern distortion. We evaluate the approach on a physical swarm of robots that use visual inertial odometry to compute their relative positions and obtain results that are comparable with simulation. This work lays the foundation for designing and coordinating configurable smart particles, with applications in smart materials and nanomedicine.

I. INTRODUCTION

Decentralized robot swarms have many desirable properties like scalability, robustness, and flexibility [1]. Robot swarms are generally composed of a team of robots that are simple and inexpensive, but when working together, they are capable of performing tasks reserved to more complex and expensive robots. Some futuristic views on swarm robotics [2] suggest that engineering methods for heterogeneous, hierarchical self-organization is required in robot swarms to transition from laboratories to real-world applications. This work is a step in this direction that leverages self-organization through local interactions in a two-level robot hierarchy. We posit that hierarchies in robot swarms can be of two types: information hierarchies and intelligence hierarchies. Information hierarchies consider identical robots with varying level of mission-related information: the robots with more information can make better mission decisions and are placed at the top of the hierarchy. Intelligence hierarchies consider heterogeneous swarms where some robots have capabilities that can lead to enhanced intelligence and decision-making, thus placing them at top of the hierarchy.

Hierarchies allow for fewer complex robots with a global view of the mission, while larger numbers of simpler robots perform the bulk of the work. As an example, guides can lead nanorobots with limited sensing to deliver drugs directly inside a cancer, where they would not be able to enter due to their size. We posit that hierarchies in robots enable building complex behaviors in a more traditional way while retaining the properties of scalability, fault tolerance, and cost-effectiveness of robot swarms.

Hierarchical approaches to swarm intelligence have been investigated in the context of particle swarm optimization [3], [4], but they have received very little attention in robotics [5]. The shepherding problem [6], [7], [8] considers a hierarchy with sheep that need to be herded by robot dogs. The problem considered in this work is comparable with shepherding, with the main difference being that our guides (or dogs in the shepherding analogy) shape and maintain the pattern made by workers (a.k.a. the sheep) by operating on the rules that govern the workers swarm. Our approach also resembles smart particle design [9] and finds potential applications in smart materials and targeted drug delivery. Just like smart particles can be designed to be dynamically programmed to change their attraction forces, a few intelligent robot guides can order the workers to a desired shape and move them to a target location by operating on the attractive and repulsive forces between the workers. We open-source the code [10] used in this work.

II. RELATED WORK

Patterns in robot swarms are generally achieved by designing collective behaviors that emerge as a result of local interactions with neighbors [11]. Some examples of pattern formation are morphogenesis [11] and self-assembly [12]. These methods create stable structures using only local interactions.

Flocking. In some applications, robots are required to move while maintaining a pattern or formation. This be-
behavior is usually referred to as flocking [13], [14], taking inspiration from the behavior of birds. Flocking is extensively studied in literature [15], starting from the basic microscopic model [16] of attraction, repulsion and cohesion.

**Leader-follower methods.** Some approaches use a leader-follower strategy for flocking, where a few robots act as leaders [17] that share metrics and positions with the other robots, which in turn use this information to coordinate their motion. The leader-follower problem is widely studied [18], [19]. Centralized approaches use techniques like algebraic graph theory [18], Hilbert space projection, and dynamic programming [18], [20]. Decentralized versions employ techniques like tractor-trail [19] and virtual springs [21]. Most of these methods are computationally intensive, with rare exceptions [21] that rely on local information and do not suffer from scalability issues.

**Shepherding.** The leader-follower approach has its natural extension to the shepherding problem: simple robots (the ‘sheep’) that are herded by one [22] or a group of robots (the ‘shepherds’) [23]. One common approach to shepherding is to approximate a cluster of sheep as a circle or ellipse that needs to be maintained over motion [23], [7], [8] by placing the shepherds behind or around the sheep herd. Other comparable approaches use Hilbert space filling and path planning [24], repulsive forces [25] and connected-component labeling [7]. Many approaches require a reliable communication between the shepherd and sheep, besides a few that rely on local measurements [8]. Similarly to flocking, most shepherding approaches do not focus on maintaining a pattern during the translation to a goal.

We consider a variation to the shepherding problem with two main differences: first, we maintain the pattern during movement to the goal. Maintaining the pattern is important for applications like nanomedicine or smart materials, as the relative position of the particles might have important functional implications; second, the shepherds (the guides in our parlance) have the ability to change the interaction forces between the sheep (i.e., the workers), which truly establishes a hierarchical control system on the swarm.

III. Model and System

Consider a group of \( n \) robots with each robot \( i \) having the following dynamics

\[
\dot{x}_i = v_i, \quad v_i = u_i, \quad i \in R, \quad R = \{r_1, ..., r_n\}
\]  

with \( x_i, \quad v_i \) and \( u_i \in \mathbb{R}^2 \) being the position, velocity and control input respectively. The relative position (\( x_{ij} \)) and velocity (\( v_{ij} \)) of two robots \( i \) and \( j \) are \( x_{ij} = x_i - x_j \) and \( v_{ij} = v_i - v_j \). We define the set of relative inter-agent distances \( D = \{d_{ij} = ||x_i - x_j|| \in \mathbb{R}^2 : i, j = 1, ..., N, i \neq j \} \) and the communication graph formed between the robots as \( G = (R, E, W) \) with \( R = \{r_1, ..., r_n\} \) the set of vertices, \( E = \{e_{ij} | i, j \in N, i \neq j \} \) the edge set and \( W = \{w_{ij} \in R_+ | e_{ij} \in E \} \) are weights assigned to the edges. Each robot \( i \in R \) forms a neighbors set \( N_i = \{j \in R | ||x_{ij}|| \leq d_{com} \} \) as dictated by \( E \), with \( d_{com} \) representing the communication range of the robots.

Guides and workers further divide the graph vertex set \( R \) into \( R^G \) (vertices corresponding to guides) and \( R^W \) (vertices corresponding to workers), with \( R^G \cap R^W = \emptyset \). The guide robots are assumed to be equipped with sophisticated sensors capable of performing localization, path planning, traversability analysis, etc. The worker robots are simple and oblivious, being capable of only communicating and measuring the relative position \( x_{ij} \) of their neighbors. The enhanced capabilities of the guide robots place them in a leadership role: they perceive mission goals and provide directives to the worker robots both indirectly (by moving to a different position) and directly (by ordering a change to the attractive/repulsive forces among the workers).

The neighbor set \( N_i \) of robot \( i \) can be divided into guide neighbors \( N_i^G \) and worker neighbors \( N_i^W = N_i \cap R^W \). The center of mass of the worker robots is defined as \( \bar{x}^i = \frac{1}{|R^W|} \sum_{j \in R^W} x_{ij} \), and assumes a circular approximation of the worker cluster. The robots are only capable of measuring relative positions \( x_{ij} \) and hence the center of mass perceived by robot \( i \) is \( \bar{x}^i = \frac{1}{N_i} \sum_{j \in N_i} x_{ij} \).

Fig. 1 illustrates the different states of the guide robots. To shape a cluster of workers, the guides are given \( k \) shaping targets, each specified by an angle (\( \theta_k^p \)) and a distance (\( d_k^p \)) with respect to the worker center of mass (\( \bar{x}^c \)). Consider the shape angle set \( \Theta^s = \{\theta_1^s, ..., \theta_k^s\} \) and shape distance set \( D^s = \{d_1^s, ..., d_k^s\} \). A guide robot \( i \in R^G \) negotiates the lowest cost shaping target \( \theta_i \in \Theta^s \) (cost determined using distance to target) through a decentralized task allocation mechanism from our previous work [26]. To reach its target angle \( \angle(x_i - \bar{x}^c) - \theta_i^p < \delta \), with \( \delta \) being a small tolerance, a guide \( i \) follows the edge of the worker cluster. Upon reaching \( \theta_i^p \), the robots translate to ensure a distance \( d_i^p \) from the center of mass of the workers cluster, with a small tolerance \( d_i^s \): \( d_i^s = ||x_{ij}|| < d_i^s \). Once a shaping target is reached (both angle and distance), the guides wait for all the other guides to reach their respective shape targets using an agreement mechanism called a barrier [27].

Once at their shaping targets, the guides translate to a target distance \( d_{wp} \) from the center of mass of the worker
cluster, thus forcing the workers into the desired shape. We use a barrier to ensure all the guides have reached their negotiated locations, upon which the guides start moving to a sequence of targets \( T = \{x_1, \ldots, x_n\}, \forall x_i \in \mathbb{R}^2 \) maintaining the worker formation. The guides interact with the workers through simple potentials by changing their position, resulting in repulsive action on the workers.

The guides can also manipulate the parameters of the potential fields that control the workers. These parameters can be propagated across the swarm and change the workers reactions to the actions of guides. Fig. 3 summarizes the parameters: Density \( \rho \) defines the distance to maintain for robots in set \( N^W_i \) for any robot \( i \) within the field of view (FoV), \( \rho \) and FoV influence the strength of the robots in formation. The mode \( M \in \{ \text{Loose, Shape, Rigid} \} \) defines the worker’s behavior to maintain distance between neighbors. In \textit{Loose}, the workers maintain a distance \( \rho \) between all robots in \( N^W_i \) with a distance less than FoV. Workers store a tuple set \( dM^W_i = \{j, d_{ij}\} | j \in N^W_i \) to keep a formation every time a mode switch to \textit{Shape} or \textit{Rigid} happens. In \textit{Shape}, removal and addition of new robots to \( dM^W_i \) are allowed as they appear in the neighbors list \( N^W_i \), in contrast to \textit{Rigid} where \( dM^W_i \) is frozen. In both \textit{Shape} and \textit{Rigid}, the robots maintain the distances in \( dM^W_i \) to preserve the formation. The reaction distance RFoV is the distance at which the workers start responding to a guide robot.

IV. METHODOLOGY

A. Worker Swarm

The four Control Parameters (CP) of the worker robots are propagated by the guide robots using virtual stigmergy [28], a conflict-free replicated data structure that acts as a global distributed shared memory for the swarm. Virtual stigmergy uses gossip-based communication that propagates through the network updating the local copies of information on the robots, and was shown to converge if a communication path exists (connected graph). Fig. 5 shows the convergence time of propagating a change in CP among the worker robots with various number of guides. The propagation of a change in CP takes about 1.5s with 4 guides for a 5k worker swarm and increases to under 3.5s with a single guide.

A common method to design a control law for the sheep in shepherding [6] or flocking [29] is to use artificial potential functions that produce attraction and repulsion between the robots. Unlike the shepherding problem, we need to maintain the desired formation during its translation. Potential functions used in modeling the bonding structure of atomic molecules [30] are some interesting candidates for enforcing a bond between two robots. We considered various potentials like Morse (used in modeling diatomic molecules), Lennard-Jones (used in modeling rare gases), Buckingham (used in modeling ceramic materials) and the harmonic approximation (used in modeling crystalline lattice structures) [30]. These potentials produce uneven attraction and repulsion forces when applied on robots, for instance produce stronger repulsion and weaker attraction becomes a problem when these parameters are dynamically configured by the guide robots. We therefore choose the harmonic approximation to define the bonding force between the worker robots. A standardized Harmonic potential is of the following form

\[
\phi(d) = a_0 + \frac{kd^3}{2|d|} \tag{2}
\]

with \( a_0 \) and \( k \) being design constants. The feedback control contribution for maintaining the formation between other worker robots within the field of view (determined based on the current mode) is \( w_i \) and the control contribution for producing a repulsive reaction from the guide robots within
the reactive field of view $R FoV$ is $g_i$

\[ u_i = \alpha w_i + \beta g_i \]  

(3)

\[ w_i = \frac{1}{|N_i^{FoV}|} \sum_{j \in N_i^{FoV}} \phi(d_{ij} - d_{\rho}) \frac{x_{ij}}{\|x_{ij}\|} \]  

(4)

\[ g_i = \frac{1}{|N_i^{RFoV}|} \sum_{j \in N_i^{RFoV}} \phi^+(d_{ij} - d_{\rho}) \frac{x_{ij}}{\|x_{ij}\|} \]  

(5)

The potential function $\phi(d_{ij} - d_{\rho})$ provides the magnitude and $\frac{x_{ij}}{\|x_{ij}\|}$ determines the direction of each $w_{ij}$ and $g_{ij}$ contribution. $\alpha$ and $\beta$ are design parameters that define the relative importance of maintaining formation with respect to reacting to the guides in proximity. The function $\phi^+(d) \rightarrow \mathbb{R}^+$ produces only repulsion mapping to positive real values if and only if the input domain is $d \in [0, \infty)$. Similarly, the function $\phi(d) \rightarrow \mathbb{R}$ produces real values when $d \in [-\infty, \infty]$. Both functions $\phi^+(\cdot)$ and $\phi(\cdot)$ are Lipschitz continuous and belong to class function $\phi^+(\cdot) \in S^+$ and $\phi(\cdot) \in S$. The class functions are defined as $S^+ = \{ f : [0, \infty) \rightarrow \mathbb{R} | f(d) = 0, \forall d \leq 0, f(d) > 0, \forall d > 0 \}$ and $S = \{ f : (-\infty, \infty) \rightarrow \mathbb{R} | f(d) < 0, \forall d < 0, f(d) > 0, \forall d > 0 \}$. From the class definition and equ. 3, an attraction force exists between all the robots with distance $d_{ij} > d_{\rho}$, $\forall i, j \in E$ and a repulsive force acts on the robots with distance $d_{ij} < d_{\rho}$. The equilibrium point with $u_i = 0$ is when all the neighboring robots of robot $i$ are at distance $d_{\rho}$ and with no guide robots within $d_{RFoV}$.

**B. Guide Swarm**

The guide swarm is the brains of the whole robot group, controlling the worker swarm, which is analogous to the muscles that do the work. Guides have the following high-level states: 1. Task Allocation, 2. Shaping setup, 3. Shaping, and 4. Movement. The high-level state machine of the guides is shown in fig. 3. Our depiction continues to assume a single shape set of shaping parameters. However, we can do multiple shapes in a single execution by activating dotted connections. All the state transitions between the four high-level states occur only with an agreement with all the guides. We use a barrier [27], a decentralized method that holds the state of the robots in a standby state until all the robots satisfy the conditions for the state transition.

**Task Allocation** The multi-robot task allocation problem [31] is a combinatorial optimization problem requiring heuristics to obtain an approximate solution in polynomial time. In our case, the Single Assignment (SA) problem and can be solved using auction or consensus. The SA problem used by the guide robots to assign $t \in \Theta_s$ to $|R^G|$ robots has the form:

\[ \min \sum_{i \in R^G} \sum_{j \in \Theta_s} c_{ij}(x_i)a_{ij} \]  

(6)

\[ \text{st. } \sum_{j \in \Theta_s} \leq 1, \forall i \in R^G \text{ and } \sum_{i \in R^G} \leq 1, \forall j \in \Theta_s \]

with $c_{ij} = \|x_i - x_j\|$ being the cost function of assigning task $j$ to robot $i$ and $a_{ij} \in \{0, 1\}$ being the binary assignment variable. We solve the single worker control problem of equ. 6 using the decentralized consensus described in [26].

**Shaping Setup and Shaping** The guides edge-follow the worker swarm until they reach their assigned shape angle. We refer the reader to the supplementary material in our code repository [10] for the details. The algorithm is partly inspired from [12] and adds a recently-followed neighbor list to escape concave edge areas (otherwise causing robots to be stuck on approach as in [12]). During edge-following, the robots maintain a list of ten recent neighbors to avoid committing and following a previously followed neighbor. When a guide comes in communication range with other guides, the robots coordinate to enable simultaneous edge-following. In Shaping, the guide robots move into the worker swarm at a given speed to attain the required shape.

**Movement** The task of translating in formation is defined as formation shepherding, requiring the robots to preserve the edges $E$ in its communication graph $G$ (edge-consistent motion). This part of our solution is the one mostly resembling the shepherding problem from literature. The guide swarm uses this control law to translate towards the goal while keeping the worker formation:

\[ u_i = \alpha_g g_i + \beta_g c_i + \gamma_g a_i \]  

(7)

\[ a_i = p(\theta_i^{ec}) \left( \frac{x_i^c}{\|x_i^c\|} \right) \]  

(8)

\[ p(\theta) = \begin{cases} -1, & \theta < \pi \text{ and } \theta < (\theta_t + \pi) \text{ and not}(\theta < \theta_t) \\ 1, & (\theta < \pi \text{ and } \theta < (\theta_t + \pi)) \text{ or } \theta > \theta_t \\ -1, & \text{otherwise} \end{cases} \]  

(9)

with $\theta_i^{ec} = \angle(x_i^c - x_t)$. The parameters $\alpha_g$, $\beta_g$ and $\gamma_g$ are design parameters respectively that define the importance of moving towards the target ($g_i$), conserving the desired center of mass distance to the workers ($c_i$) and maintaining the orientation between the target and worker center of mass ($a_i$).
Both $g_i$ and $c_i$ use the harmonic function potential defined in equ. 2 as in equ. 4. Whereas the contribution to maintain the orientation is performed using equ. 8.

V. EXPERIMENTS

We evaluate our approach both in simulation and on robots. All the code used in the experiments is available as open source [10]. We implemented our approach using Buzz [32] an extensible domain-specific programming language for robot swarms. We used the Khepera IV robot model in simulation and deployed customized Khepera IV robots equipped with an Intel RealSense T265 [33] and an Nvidia Jetson TX1 [34] (some robots in the worker class were using a Raspberry Pi 4) as their on-board computer. Robots in simulation use a range and bearing sensor (from our simulator, ARGoS [35]) to communicate, while the real robots use ad hoc networking and local broadcast. We implemented range and bearing in the real robots by creating data packets pairs with [position, data] using the positional information computed from their T265 camera.

A. Simulation

Simulation experiments were performed using a physics-based simulator, ARGoS [35]. Simulation configuration used during our experiments were: number of worker robots $N \in \{50, 100, 300\}$ and three different shapes $Shape \in \{\text{Clover}, \text{Dumbbell}, \text{Circle}\}$ with 4 to 16 guides, 30 times per configuration with random initialization. We also run experiments with 1000 robots and 32 guides for the clover shape. All parameters used in the experiments can be found in our code repository [10]. Fig. 3 shows the configurations, and Fig. 4 the trajectories of the robots moving the shapes 10m from the initial location.

Our primary metric for performance is the distortion $\Upsilon$. Here we see how the dumbbell is particularly subject to distortion with smaller worker and guide swarms, although distortion decreases with larger worker swarms. On the other hand, the circle and clover shapes seem to have greater distortion with increasing worker swarm size.

Fig. 6. The different trajectories undertaken by robots in each shape: clover on the left, dumbbell in the center and circle on the right. Here we can already see the motion of the circle is more direct than the others, with the dumbbell showing noticeable vertical and horizontal distortion en route to its destination.

Fig. 7. Time taken by the worker robots to transition from one state to another. We can see consistently longer and more varying times in the movement of clover and dumbbell shapes. The task allocation, shaping setup, and shaping of clovers and dumbbells also appear to be affected by increasing the swarm size.

Fig. 8. Distortion incurred by the swarm in different states computed using $\Upsilon$. Here we see how the dumbbell is particularly subject to distortion with smaller worker and guide swarms, although distortion decreases with larger worker swarms. On the other hand, the circle and clover shapes seem to have greater distortion with increasing worker swarm size.
positional relationship of a robot with respect to the worst case encountered in the experimental configuration.

Fig. 7 breaks down the robot activities as described in Section IV-B: Task Allocation for the guide robots to select shaping targets, Shaping Setup to locate the worker swarm and edge-follow to enclose it, Shaping to create a desired formation with the worker, and Movement takes the desired formation to a target. Task Allocation has a monotonic increase in time with number of robots and does not vary with configurations since it is a function of number of robots and available tasks. Shaping Setup and Shaping exhibit a similar pattern that depends on the configuration: this is due to the fact that both require reaching a given alignment. Movement exhibits different dynamics that depend on each shape and number of workers. It is worth noting that the circle shape took the least amount of time because the robots require fewer corrective moves to maintain formation and spend more time moving towards the target.

Fig. 8 compares the distortion from motion for various shapes. For Clover and Circle, the distortion increases with the number of robots, but decreases for Dumbbell, although it is generally higher for the latter. We believe that the higher number of guides mitigate the distortion for the more complex shape of the Dumbbell.

B. Physical robots

We used 6 Khepera IV robots with 4 as workers and 2 as guides. We used a simple square like shape to be formed by the robots. The robots were able to task allocate, preform the shaping setup to reach the provided shaping targets and move to a target of about 1m in formation. Fig. 9 top shows the trajectory of the robots for a similar shape in simulation, middle, the trajectory of the real robots and the bottom, shows the time taken to reach the target in formation. The experiments reported 240s on average with real robots and 247s in simulation, matching expectations.

VI. CONCLUSIONS

We propose a hierarchical approach to control a swarm of robots that form shapes and translate in formation with two types of robots: a large group of simple robots (workers) and a small swarm of intelligent robots (guides). The process of decision-making for formation and translation to targets is performed by the guides, and workers act as the muscle in forming shapes with no knowledge of the overall task. The interactions of the swarms are designed to be simple and local through virtual potential fields that depend on the relative position of the robots. This approach relies on the knowledge of the guide swarm, which coordinates to obtain the desired shape of the worker swarm, like smart particles. The interaction properties of the worker swarm are dynamically adapted by the guide swarm, leveraging its additional sensing and knowledge. The prototype framework implemented in this work lays the foundation for hierarchically self-organizing swarms. Potential uses of the approach could be targeted drug delivery, cancer treatment, and smart materials.
REFERENCES

[1] M. Dorigo, G. Theraulaz, and V. Trianni, “Swarm robotics: Past, present, and future,” Proceedings of the IEEE, vol. 109, no. 7, pp. 1152–1165, 2021.

[2] Y. Zhang, “Reflections on the future of swarm robotics,” Science Robotics, vol. 5, no. 49, pp. 9–12, 2020.

[3] M. Scheutz, “Real-time hierarchical swarms for rapid adaptive multi-level pattern detection and tracking,” in 2007 IEEE Swarm Intelligence Symposium. IEEE, 2007, pp. 234–241.

[4] H. Chen, Y. Zhu, K. Hu, and X. He, “Hierarchical swarm model: a new approach to optimization,” Discrete Dynamics in Nature and Society, vol. 2010, 2010.

[5] Y. Zhang, S. Ogtuz, S. Wang, E. Garone, X. Wang, M. Dorigo, and M. K. Heinrich, “Self-reconfigurable hierarchical frameworks for formation control of robot swarms,” IEEE Transactions on Cybernetics, pp. 1–14, 2023.

[6] J. Hu, A. E. Turgut, T. Krajnik, B. Lennox, and F. Arvin, “Oclusion-based coordination protocol design for autonomous robotic shepherdering tasks,” IEEE Transactions on Cognitive and Developmental Systems, 2020.

[7] S. Razali, N. F. Shamsudin, M. Osman, Q. Meng, and S.-H. Yang, “Flock identification using connected components labeling for multi-robot shepherdering,” in 2013 International Conference on Soft Computing and Pattern Recognition (SoCPAR). IEEE, 2013, pp. 298–303.

[8] A. Özdemir, M. Gauci, and R. Groß, “Shepherding with robots that do not compute,” in ECAL 2017, the Fourteenth European Conference on Artificial Life. MIT Press, 2017, pp. 332–339.

[9] C. Huang, G. Yang, Q. Ha, J. Meng, and S. Wang, “Multifunctional “smart” particles engineered from live immunocytes: toward capture and release of cancer cells,” Advanced Materials, vol. 27, no. 2, pp. 310–313, 2015.

[10] Particle swarm code repository,” https://github.com/MISTLab/Particle_swarm.git.

[11] I. Slavkov, D. Carrillo-Zapata, N. Carranza, X. Diego, F. Jansson, J. A. Kaandorp, S. Hauert, and J. Sharpe, “Morphogenesis in Robot Swarms,” Accepted for publication on Science Robotics. Available on Science Robotics, vol. 19, Dec 2018, no. 24, pp. 1–17, 2018.

[12] M. Rubenstein, A. Cornejo, and R. Nagpal, “Programmable self-assembly in a thousand-robot swarm,” Science, vol. 345, no. 6198, pp. 795–799, 2014.

[13] G. Vásárhelyi, C. Virág, G. Somorjai, T. Nepusz, A. E. Eiben, and R. P. Ramos, S. M. Oliveira, S. M. Vieira, and A. L. Christensen, “Leader–follower formations in obstacle environments,” Autonomous Robots, vol. 35, no. 2, pp. 195–219, 2013.

[14] C. W. Reynolds, “Flocks, herds and schools: A distributed behavioral model,” in Proceedings of the 14th annual conference on Computer graphics and interactive techniques, 1987, pp. 25–34.

[15] V. Dimarogonas, M. Egerstedt, and K. J. Kyriakopoulos, “A leader-based containment control strategy for multiple unicycles,” in Proceedings of the 45th IEEE Conference on Decision and Control. IEEE, 2006, pp. 5968–5973.

[16] M. Ji, A. Muhammad, and M. Egerstedt, “Leader-based multi-agent coordination: Controllability and optimal control,” in 2006 American Control Conference. IEEE, 2006, pp. 6–pp.

[17] D. Panagou and V. Kumar, “Cooperative visibility maintenance for leader–follower formations in obstacle environments,” IEEE Transactions on Robotics, vol. 30, no. 4, pp. 831–844, 2014.

[18] S. Björkenstam, M. Ji, M. B. Egerstedt, and C. F. Martin, “Leader-based multi-agent coordination through hybrid optimal control,” Georgia Institute of Technology, 2006.

[19] J. Wiech, V. A. Ereneyev, and I. Giorgio, “Virtual spring damper method for nonholonomic robotic swarm self-organization and leader following,” 2018.

[20] H. Hoshi, I. Imura, S. Nakayama, Y. Moriyama, and K. Ishibashi, “Robustness of herding algorithm with a single shepherd regarding agents’ moving speeds,” Journal of Signal Processing, vol. 22, no. 6, pp. 327–335, 2018.

[21] M. Kuro, M. Tashiro, H. Sato, and A. Yamaguchi, “Herd guidance by multiple sheepdog agents with repulsive force,” Artificial Life and Robotics, pp. 1–12, 2022.

[22] V. S. Varadarajan, D. St-Onge, B. Adams, and G. Beltrame, “Swarm relays: Distributed self-healing ground-and-air connectivity chains,” IEEE Robotics and Automation Letters, vol. 5, no. 4, pp. 5347–5354, 2020.

[23] M. Ji, A. Label, G. Beltrame, “A temporal programming language for heterogeneous swarm robotics,” in Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), 2016, pp. 287–294.

[24] H. T. Zhang, C. Zhai, and Z. Chen, “A general alignment repulsion algorithm for flocking of multi-agent systems,” IEEE Transactions on Automatic Control, vol. 56, no. 2, pp. 430–435, 2011.

[25] S. J. Stuart, A. B. Tutein, and J. A. Harrison, “A reactive potential field for hydrocarbons with intermolecular interactions,” The Journal of chemical physics, vol. 112, no. 14, pp. 6472–6486, 2000.

[26] G. A. Korsah, A. Stentz, and M. B. Dias, “A comprehensive taxonomy for multi-robot task allocation,” The International Journal of Robotics Research, vol. 32, no. 12, pp. 1495–1512, 2013.

[27] C. Pinciroli and G. Beltrame, “Buzz: An extensible programming language for heterogeneous swarm robotics,” in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2016, pp. 3794–3800.

[28] Intel Realsense Tracking Camera,” https://developer.intelrealsense.com/tracking-camera-265/.

[29] NVIDIA TX1 product website,” https://developer.nvidia.com/embedded/jetson-tx1.

[30] C. Pinciroli, V. Trianni, R. O’Grady, G. Pini, A. Brutschi, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, F. Fucatelie, et al., “Argos: a modular, parallel, multi-engine simulator for multi-robot systems,” Swarm intelligence, vol. 6, no. 4, pp. 271–295, 2012.