Multi-Base Dynamic Role Tracking of a Fast Target with Swarm Robots

Gabriel Twigg-Ho, Gabriel Barrasso, and Dr. Xiaohua (Jamie) Ge
School of Engineering, Swinburne University of Technology, Melbourne, Vic 3122, Australia
103597673@student.swin.edu.au; 103600160@student.swin.edu.au; xge@swin.edu.au.

***Abstract*—This paper presents a modular, 2D swarm robotics simulator designed to investigate deployment centric strategies for the persistent tracking of a target that is double the speed of any individual agent. The research methodology builds upon a Particle Swarm Optimization (PSO) inspired framework [5], evolving from a baseline implementation that yielded a modest 14.6% line-of-sight (LOS) persistence to a highly effective, hybrid control system. Through systematic optimization and iterative design, a novel, state-dependent behavioural configuration was developed. This adaptive strategy which delegates tasks to specialized interception, pursuit and search algorithms based on the tactical context achieved a final LOS persistence of 94.41%. The results demonstrate that a deployment centric, behaviourally adaptive approach provides a robust and resource efficient solution to the challenging fast target pursuit problem.**

Keywords:

Particle Swarm optimization (PSO)

Line of sight (LOS)

Swarn Robotics

Target Tracking

I. Introduction

Project Context and Semester 1 Foundations

The tracking of a faster non-evasive target by a swarm of slower autonomous agents is a cornerstone challenge in swarm robotics. Unlike centralized systems which depend on a single point of control swarm robotics puts decision making and communication into the local control of each agent [1]. The problem is of significant academic and practical interest with applications in surveillance, environmental monitoring and security [3]. Its complexity is such that it has been compared to NP-Hard [4] problems, as it requires the coordination of multiple agents with limited information to solve a dynamic global objective.

The project focused on establishing the foundational groundwork to address this problem. A modular 2D swarm tracking simulator was developed using Python and the Pygame library creating a flexible testbed for algorithmic exploration. The initial research focus was on implementing and adapting a Particle Swarm Optimization (PSO) (Figure.1, Figure.2) inspired algorithm drawing conceptual inspiration from the work of Kwa et al. on the interplay between social interaction (k-nearest neighbours) and agent memory [5]. PSO, a population-based, bio-inspired optimization algorithm, models the collective movement of a swarm, where each particle adjusts its velocity based on factors like its global and personal best sighting validity.

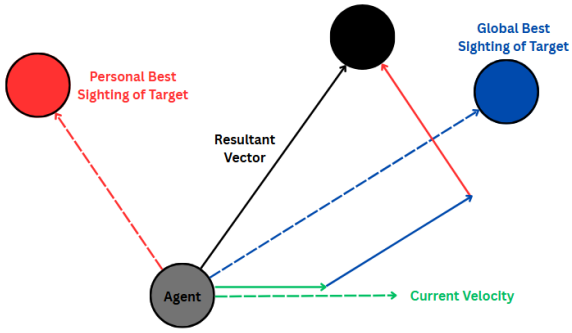

Figure 1 General PSO Vector Illustration

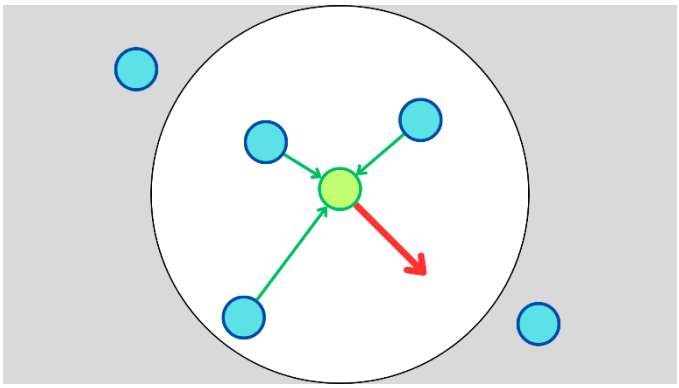

Figure 2 General PSO Separation Illustration

This first phase concentrated on baseline agent behaviours, namely attraction to a target's last known position and repulsion/separation from neighbouring agents. However, preliminary testing of this monolithic behavioural strategy yielded a modest line-of-sight (LOS) maintenance of only 14.6%. This result, while establishing a crucial performance baseline, highlighted the inherent difficulty of the problem and

underscored the inadequacy of a single, non-adaptive behavioural model.

Research Aim

The ultimate aim of this research is to implement and validate an optimal hybrid configuration of swarm agent behaviours that can achieve line of sight (LOS) persistence of over 90% against a faster moving target

This involves moving beyond a single, monolithic algorithm to a more sophisticated adaptive approach. The core of this strategy is a state machine controller, implemented as "HasMemory_DotProductMode" in the codebase, which delegates control to different specialized sub-behaviours based on the agent's real-time tactical situation relative to the target. The research will systematically optimize the parameters of this hybrid system and quantify the performance gains achieved, with the goal of providing a robust solution to the faster moving target pursuit problem.

II. Literature Review/Background

The field of swarm robotics has extensively explored the problem of target tracking, drawing inspiration from natural collective behaviours to design scalable and robust multi agent systems. This review surveys the foundational algorithmic approaches and identifies key methodological trends and highlights the critical gap in the study of deployment strategies for tracking faster moving targets.

Foundational research in swarm robotics has been dominated by bio-inspired, heuristic-based algorithms. Comprehensive surveys provide a thorough overview of these methods, including Particle Swarm Optimization (PSO), Ant Colony Optimization and Artificial Bee Colony [2]. These algorithms which are decentralized in nature promote scalability and reduce error by avoiding single points of failure. The core principle of PSO introduced by Kennedy and Eberhart [8] involve a population of "particles" that navigate a search space influenced by their own best known position and the best known position of the swarm. This logic has shown [8] success for target tracking applications as the attraction to a "global best" position can be mapped to a swarm's attraction to a target's location.

Building on these foundations, a significant body of work has focused on the specific mechanics of tracking dynamic targets. Further research [3][4] has introduced distributed algorithms for the multi robot observation of multiple targets, laying the groundwork for cooperative control. A central theme in this area is the use of force based models, where agent motion is governed by a vector sum of attractive and repulsive forces. The research that serves as the primary inspiration for our work, conducted by Kwa et al. [5], directly investigates the critical challenge of tracking multiple targets that are faster than the individual swarm agents. Their work utilized a modified PSO inspired approach to demonstrate the crucial interplay between inter agent communication (defined by the number of k-nearest neighbours) and agent memory (the duration of which target information is retained). Their findings established that tuning these social and memory parameters is key for balancing exploration (searching for the target) and exploitation (converging on the target), providing a clear argument for the parameter driven, behaviour based methodology employed in our simulator.

More recently, advanced machine learning techniques, particularly Deep Reinforcement Learning (DRL) have been applied to the swarm pursuit problem. Studies [6][7] have demonstrated the use of DRL algorithms like Multi Agent Deep Deterministic Policy Gradient to enable swarms of slower agents to learn collaborative strategies for pursuing targets. While these methods show great promise and can generate highly complex behaviours, they often rely on centralized training paradigms and make simplifying assumptions about the environment, such as full visibility or fixed evader policies. This can limit their applicability in noisy, real-world scenarios and contrasts with the heuristic-based, decentralized execution model that is the focus of our research.

Synthesizing the existing literature reveals a persistent and critical research gap. Across heuristic, force-based and DRL approaches, a common simplifying assumption is the use of a fixed, centralized deployment location. In the vast majority of studies, the swarm is initialized as a single cohort at a predetermined point in the environment. This convention, while simplifying experimental setup, overlooks a crucial aspect of real world applicability, where autonomous systems like drones would likely be deployed from multiple spatially distributed bases (e.g., rooftops, charging stations). The question of how this initial spatial distribution affects the swarm's ability to acquire and maintain contact with a target remains largely unexplored.

Thus as it is unexplored our research direction focuses on how we may maintain a LOS of 90% on a single faster moving target with spatial spaced starting locations

III. Problem Definition

Core Technical Challenge

The problem statement for this research is formally defined as: How can a decentralized swarm of autonomous agents each with a maximum speed half of the target's, maintain a persistent line of sight (LOS) of over 90% on a faster, non-evasive target within a 2D, obstacle free environment?

This requires a system that executes coordinated group behaviour using only locally available information: an agent's own sensor data (LOS), finite memory of past sightings and information shared within a limited communication network of its k-nearest neighbours.

Key Behavioural Sub-Problems

The core challenge decomposes into four distinct behavioural sub-problems:

1. **Target Reacquisition:** Following LOS loss, agents must efficiently search to relocate the target, balancing convergence with exploratory dispersion to maximize collective sensor coverage.

2. **Information Propagation:** Target location data decays rapidly due to the target's superior speed. The system must facilitate rapid propagation of sighting information throughout the swarm via k-nearest neighbour communication.
3. **Spatial Distribution:** Agents must maintain effective separation to prevent clustering around last known positions, which reduces collective sensor footprint and inhibits reacquisition.
4. **Adaptive Pursuit:** The system must dynamically switch between specialized strategies, interception when ahead of the target, aggressive catch-up when behind and search when information is stale all based on the agent's geometric relationship to the target.

IV. Main Results

Simulation Environment and Architecture

All experiments are conducted within the custom built 2D simulator. The architecture is highly modular, with distinct Python modules for core simulation logic (simulation.py), agent behaviour definitions (agent_behaviors.py) and target path scenarios (paths.py). This design enables rapid and controlled experimentation. A powerful testing and optimization framework (testing.py, optimization_strategies.py) orchestrates headless simulations for batch behaviour parameter tuning, utilizing CPU parallel processing to accelerate findings of optimal parameters.

Experimental Methodology

To identify the optimal behaviour configuration a rigorous, multi-stage evaluation was executed. The primary performance metric is LOS% defined as the percentage of simulation ticks where at least one agent has direct LOS to the target, averaged over multiple Monte-Carlo trials. For all tests the target speed was set to be two times the agents's maximum speed with a cohort of 30 agents being used in the simulation.

Central to this investigation is a diverse catalogue of agent behaviours, which were developed and categorized based on their tactical function. These behaviours are broadly divided into two groups:

- **Catch Behaviours** detailed in Table 1, are pursuit-focused algorithms designed to maintain contact or close the distance to a target whose location is known from recent sightings.
- **Search Behaviours** detailed in Table 2, are reacquisition focused algorithms designed for efficient area coverage when the target's location is unknown and memory has expired.

With this catalogue of behaviours, the evaluation pipeline proceeded as follows:

1. **Single Behaviour Screening:** Each "catch" behaviour (Table 1) was tested in isolation using a combination of RandomSearch (100 trials) and SequentialSearch to find its optimal parameters and establish a performance baseline. The top three performing behaviours were promoted. The results of this initial screening which established a clear performance hierarchy among the catch strategies are visualized in Figure 3.
2. **Dot Product Mode Pairing:** The top performing behaviours were then paired within the "HasMemory_DotProductMode" state machine, which assigns different behaviours based on whether the agent is geometrically ahead of ("greater than") or behind ("less than") the target, results shown in Figure 4.
3. **Head-to-Head Validation:** The two best pairings from the previous stage were subjected to a large-scale validation run, involving 4,000 additional random-search trials and an extensive sequential search, to statistically confirm the superior combination.
4. **Tertiary-Behaviour Sweep:** With the optimal "ahead/behind" pair fixed, every "Search" behaviour (Table 2) was then tested as the third component, responsible for the "no-memory" or "lost" state in the state machine with results shown in Figure 5.

In parallel to this systematic testing approach a exploratory design methodology was employed, wherein new behaviours were designed and implemented specifically to address observed weaknesses in the existing catalogue.

Table 1 Catch Behaviours

Catch Behaviour	Description
SeparationBehavior	Maintains spacing while moving toward last known target position using repulsion and hard separation. Goes directly to target when has information
PredictiveBehavior	Estimates the target's future position and moves toward the predicted intercept while maintaining separation.
PredictiveInterceptMultiplierBehavior	Enhances predictive interception with an attraction multiplier when ahead and defaults to separation when no memory is available.
WedgePredictiveBehavior	Uses predictive pursuit combined with lateral spreading to avoid clustering near nearby agents.
FollowSightingsBehavior	Tracks the freshest target information by moving toward agents with the most recent sighting instead of only the last known location.
FrontierGapSeekingBehavior	Seeks gaps between neighbouring agents near the last sighting to improve coverage and minimize redundancy.

Table 2 Search Behaviours

Search Behaviour	Description
SeparationBehavior	Maintains spacing while moving toward last known target position using repulsion and hard separation. Evenly separates from every other agent

ExpandingSpiralSearchBehavior	Performs outward spiral motion from the last sighting or start point to reacquire the target.
LaneSweeperBehavior	Conducts horizontal sweeping motions across the environment based on last sighting or starting location.
LevyBiasedStationSearchBehavior	Returns to the starting position before performing outward-biased Lévy search patterns when memory is lost.

Synopsis of Key Findings

Overall the testing produced several critical insights that have shaped the findings of the project.

- **Hybrid Strategies are Superior:** No single behaviour was found to be optimal across all scenarios. The highest performance was consistently achieved by hybrid configurations that adapt their strategy based on the situation.
- **Systematic Search Identified a Strong Baseline:** The pipeline culminated in identifying a strong three-part strategy consisting of FollowSightings (ahead), Predictive (behind), and LevyBiased search (no-memory), which achieved an LOS score of approximately 72%.
- **Exploratory Design Led to a Breakthrough:** The design of purpose built behaviours to fill specific strategic roles yielded a significant performance breakthrough. This approach led to the development of GetInFrontBehavior (GIF) for interception and InverseSquareRepulsionBehavior (ISR) for efficient searching, which dramatically outperformed their pre-existing counterparts as shown in Figure.5.

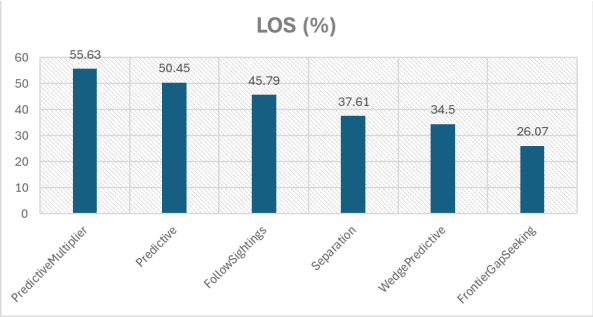

Figure 3 Single Catch Behaviours LOS% Score

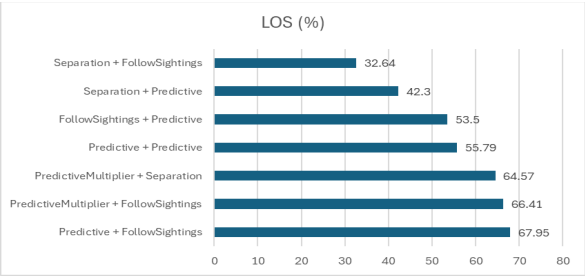

Figure 4 Catch Behaviour Configuration LOS% Score

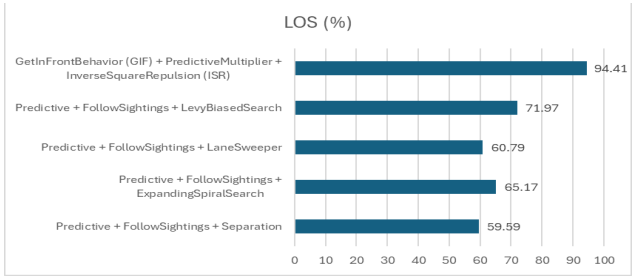

Figure 5 Catch + Search Behaviour Configuration LOS% Score

V. Results and Discussion

Best Behavioural Configuration

The combined results of the systematic and exploratory investigations have identified a definitive champion behavioural configuration, managed by the "HasMemory_DotProductMode" state controller. This three part adaptive strategy directly addresses each of the four sub-problems defined in Section III:

State 1: Has Memory & Ahead of Target (dot product > 0) (Figure.6)

- Behaviour: GetInFrontBehavior (GIF)
- Role: Specialized interception algorithm that positions agents directly in the target's predicted path
- Addresses Sub-Problem 4 (Adaptive Pursuit): Provides optimized ahead of target interception logic

State 2: Has Memory & Behind Target (dot product < 0) (Figure.6)

- Behaviour: "PredictiveInterceptMultiplierBehavior"
- Role: Aggressive predictive pursuit with "ahead bonus" multiplier for flanking agents
- Addresses Sub-Problem 4 (Adaptive Pursuit): Enables rapid catch up when trailing the target

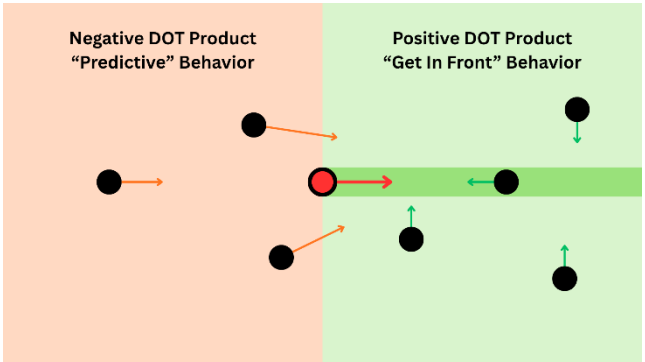

Figure 6 Has Memory Behaviour Illustrations

State 3: No Memory

- Behaviour: "InverseSquareRepulsionBehavior" (ISR) (Figure.7)

- Role: Dispersive search behaviour using inverse square repulsion forces
- Addresses Sub-Problems 1 & 3 (Target Reacquisition and Spatial Distribution): Forces agents to spread from one another preventing clustering and maximizing collective sensor footprint

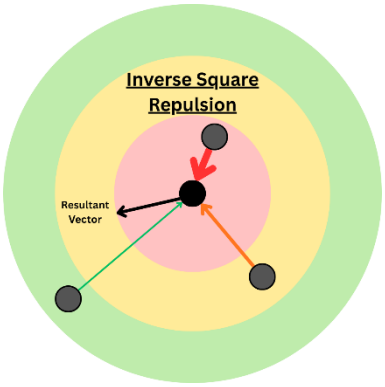

Figure 7 No Memory Behaviour Illustration

All three states leverage the k-nearest neighbour communication network (tuneable parameter) to share sighting information, which addresses Sub-Problem 2 (Information Propagation).

Performance Analysis and Metrics

After extensive parameter optimization using parallel random search and sequential optimization strategies, the configuration achieved a final LOS score of 94.41%. This represents:

- A 6× improvement over the initial baseline (14.6% LOS)
- A 31% relative improvement over the best purely systematic search result (~72% LOS)
- Achievement of the research objective (>90% LOS persistence)

Interpretation of Outcomes

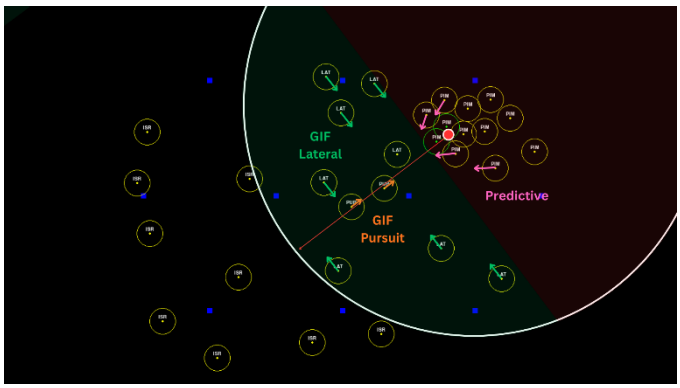

Figure 8 Behaviours in Action

The systematic search pipeline, while effective at parameter tuning, reached a performance ceiling of approximately 72% LOS. Analysis revealed a critical tactical gap: agents positioned ahead of the target would pursue predicted intercept points perpendicular to the target's direction of travel, creating gaps in coverage along the target's actual path.

The "GetInFrontBehavior" (GIF) (Figure.8) algorithm addresses this through a two-phase approach:

1. Lateral Alignment Phase: When the agent's perpendicular offset from the target's velocity vector exceeds a threshold (`GIF_LATERAL_OFFSET`), the agent moves laterally to position itself directly in the target's path

2. Pursuit Phase: Once aligned, the agent moves directly toward the target to maintain forward blocking position

This geometric strategy ensures agents form a "wall" in the target's path rather than converging from the sides, dramatically reducing escape opportunities and enabling the performance gain from 72% to 94.41%.

Real World Context for Simulation Parameters

The parameters chosen for this simulation were explicitly modelled to reflect a realistic deployment scenario within a typical suburban environment. The agents (drones) were initially deployed from locations meant to mirror the spacing of operational bases (police stations) in such areas. Agent and target speeds were calibrated to match typical operational values for drones and cars respectively, establishing a challenging 2:1 pursuit ratio. Furthermore, the agent's limited Line-of-Sight (LOS) range was set to approximate the camera view distances of a surveillance drone operating at typical altitudes. This grounding in realistic parameters ensures that the derived hybrid strategy is a relevant and viable solution for resource-constrained, real-world fast target pursuit.

Future Improvements

1. Robustness and Generalization Testing

The current behavioural configuration achieved its high performance (94.41% LOS) after extensive parameter tuning on a fixed set of four target paths used for both optimization and evaluation. A key next step is to rigorously test for parameter overfitting.

New Validation Paths: The configuration's efficacy must be tested against a new, unseen set of validation paths. This set should include more complex, non-deterministic, and highly varied target movement patterns, such as those with frequent, sharp turns or stop-and-go manoeuvres, to ensure the parameters aren't optimized solely for the original, specific set of scenarios.

Initial Deployment Variation: The current simulation used the same startup locations for the agents. Future work must investigate the impact of changing the agent's initial spatial distribution from uniform to chaotic/random on the strategy's performance. This directly relates to the original research gap identified regarding the impact of deployment strategies.

2. Scalability and Resource Efficiency Analysis

The current performance metrics were established using 30 tracking agents and a fixed target-to-agent speed ratio of 2:1. Future work must analyse how these key parameters affect the system's performance.

Agent Number Scaling: Further tests are required to systematically determine how performance scales with the number of agents. The goal is to establish the minimum number of agents required to consistently maintain the target LOS persistence above the 90% objective. This provides a valuable measure of the solution's resource efficiency for real-world application.

Speed Ratio Impact: The investigation must also explore how changing the target speed ratio (e.g 3:1, 1.5:1). This will help define the effective operational limits of the strategy and its applicability to targets of varying speeds.

3. Increased Environmental and Physical Complexity

To bridge the gap between simulation and real-world deployment, the current 2D, obstacle-free environment needs to be augmented with realistic constraints.

Environmental Constraints: The introduction of obstacles and dynamic no-fly zones will test the agents' ability to maintain coordination and execute their search/pursuit behaviours in a constrained space, forcing them to incorporate path-planning logic.

Communication Noise and Latency: The current system relies on a reliable k-nearest neighbour communication network to propagate sighting information. Introducing communication noise (i.e., data loss) or latency models would test the robustness of the information propagation mechanism.

Agent Energy Models: The simulation should incorporate agent energy models. This will force the system to optimize not just for LOS persistence, but also for energy consumption, which is critical for autonomous systems like drones that have finite battery life.

VI. Conclusion

The purpose of the research was to identify an optimal, adaptive control strategy for persistent swarm tracking of a faster moving target. The experimental setup consists of 30 tracking agents operating in a 1280×720 pixel workspace, tasked with tracking a target moving at two times the agents maximum velocity.

This research makes two primary contributions. First, it provides a concrete, high performing solution to the fast target pursuit problem in the form of the GIF/PredictiveMultiplier/ISR hybrid configuration. Second it contributes a methodological insight that while systematic parameter tuning is crucial for optimization, creative, purpose driven design is a powerful tool for achieving step changes in performance by identifying and filling conceptual gaps in existing strategies.

Looking ahead, several avenues for future work have been identified to validate and extend these findings. The immediate priority is to address the potential for parameter overfitting by testing the champion configuration's robustness against a new unseen suite of validation paths to ensure its generalizability. In line with the deployment centric focus of this research this validation should also include an analysis of how varying agent startup locations affects overall efficacy. Beyond validation, a scalability analysis is required to understand the operational limits of the solution. This would involve determining the minimum number of agents required to maintain the >90% LOS threshold a key measure of resource efficiency and assessing how performance is impacted by increasing the target to agent speed ratio. Finally, a long term extension of this work would be to introduce environmental complexity, such as obstacles, communication noise, or agent energy models to provide a more realistic test of the strategy's robustness and open new avenues for research into fault tolerant swarm coordination.

VII. References

[1] M. M. Shahzad et al., "A review of swarm robotics in a nutshell," Drones, vol. 7, no. 4, Apr. 2023, Art. no. 269, doi: 10.3390/drones7040269.

[2] M. Senanayake, I. Senthooran, J. C. Barca, H. Chung, J. Kamruzzaman, and M. Murshed, "Search and tracking algorithms for swarms of robots: A survey," Robot. Auton. Syst., vol. 75, pp. 422–434, Jan. 2016, doi: 10.1016/j.robot.2015.08.010.

[3] L. E. Parker, "Distributed algorithms for multi-robot observation of multiple moving targets," Auton. Robot., vol. 12, pp. 231–255, May 2002, doi: 10.1023/A:1015256330750.

[4] L. E. Parker and B. A. Emmons, "Cooperative multi-robot observation of multiple moving targets," in Proc. IEEE/RSJ Int. Conf. Grenoble, France, Sep. 1997, pp. 925–932, doi: 10.1109/IROS.1997.655115.

[5] H. L. Kwa, J. L. Kit, and R. Bouffanais, "Tracking multiple fast targets with swarms: Interplay between social interaction and agent memory," in Proc. 2021 Conf. Artif. Life, Jul. 2021, pp. 484–493, doi: 10.1162/isal_a_00376.

[6] G. Singh, D. Lofaro, and D. Sofge, "Pursuit-evasion with decentralized robotic swarm in continuous state space and action space via deep reinforcement learning," in Proc. Int. Conf. Agents Artif. Intell. (ICAART), Valletta, Malta, Feb. 2020, pp. 226–233, doi: 10.5220/0008971502260233.

[7] V. Gabler and D. Wollherr, "Decentralized multi-agent reinforcement learning based on best-response policies," Front. Robot. AI, vol. 11, Apr. 2024, Art. no. 1229026, doi: 10.3389/frobt.2024.1229026.

[8] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Proc. Int. Conf. Neural Netw. (ICNN'95), Perth, WA, Australia, 1995, vol. 4, pp. 1942–1948, doi: 10.1109/ICNN.1995.48896488968.