DOI 10.15622/ia.24.1.3

A. HAMMOUD, A. ISKANDAR, B. KOVÁCS

DYNAMIC FORAGING IN SWARM ROBOTICS: A HYBRID APPROACH WITH MODULAR DESIGN AND DEEP REINFORCEMENT LEARNING INTELLIGENCE

Hammoud A., Iskandar A., Kovács B. **Dynamic Foraging in Swarm Robotics: A Hybrid Approach with Modular Design and Deep Reinforcement Learning Intelligence.**

Abstract. This paper proposes a hybrid approach that combines intelligent algorithms and modular design to solve a foraging problem within the context of swarm robotics. Deep reinforcement learning (RL) and particle swarm optimization (PSO) are deployed in the proposed modular architecture. They are utilized to search for many resources that vary in size and exhibit a dynamic nature with unpredictable movements. Additionally, they transport the collected resources to the nest. The swarm comprises 8 E-Puck mobile robots, each equipped with light sensors. The proposed system is built on a 3D environment using the Webots simulator. Through a modular approach, we address complex foraging challenges characterized by a non-static environment and objectives. This architecture enhances manageability, reduces computational demands, and facilitates debugging processes. Our simulations reveal that the RL-based model outperforms PSO in terms of task completion time, efficiency in collecting resources, and adaptability to dynamic environments, including moving targets. Notably, robots equipped with RL demonstrate enhanced individual learning and decision-making abilities, enabling a level of autonomy that fosters collective swarm intelligence. In PSO, the individual behavior of the robots is more heavily influenced by the collective knowledge of the swarm. The findings highlight the effectiveness of a modular design and deep RL for advancing autonomous robotic systems in complex and unpredictable environments.

Keywords: swarm robotics, foraging task, modular design, reinforcement learning, particle swarm optimization.

1. Introduction. Swarm robotics realizes the principle of decentralized control among multiple robots, drawing inspiration from natural phenomena where collective behaviors emerge from simple individual actions, such as in ant colonies or bird flocks. This field leverages the scalability, robustness, and flexibility inherent to swarms, which represent the main features of swarm robots' systems. They allow them to solve complex tasks more efficiently than individuals do [1]. To achieve the goal required of robots in the swarm concept, the robots should act with a high level of autonomy and have local knowledge about their environment. Namely, a microscopic view. While coordination and communication among robots reflect the concept of collective behavior, which arises from the aggregation of simple rules followed by individual robots, leading to emergent behaviors that require no centralized oversight, this represents the macroscopic level [2]. For example, aggregation behavior occurs when robots come together to form clusters or groups while foraging behavior emerges during the coordinated effort of the robots to search, identify, collect, and transport resources to a destination called the nest. Other

examples include collective searching and exploration, pattern formation, and others [3]. In general, collective behavior is obtained by various algorithms that enable individual robots to make decisions based on local perception and the cumulative contribution of all robots in achieving the desired outcomes. This involves communication and collaboration among the robots as well. Most of the methods used in swarm engineering can be divided into two basic categories: Behavioral design methods like the probabilistic finite state machine (PFSM) method, where the robots' behaviors are broken down into a finite number of states and transitions between these states. These transitions are triggered by events or conditions [4]. On the other hand, automatic design methods such as RL and PSO stand out. RL fosters the robot to learn the required behavior at a microscopic level through interaction with the environment. By deploying RL, the robots learn and adapt their actions through a continuous process, gradually refining their policies. Conversely, PSO is a mathematical model that reflects social behavioral patterns and guides individuals within the swarm toward optimal solutions [5]. Each method has its own set of advantages. RL is renowned for its adaptability to the environment's changes, enhancing autonomy and decision-making abilities, while PSO is lauded for its simplicity and straightforward implementation. Nonetheless, RL's computational cost and PSO's potential ineffectiveness in dynamic settings pose significant challenges. Many RL algorithms can potentially be used for generating collective behavior, like Deep-Q networks [6], SARSA, and other value-based methods [7]. In addition to policy-based methods including DDPG [8], PPO, and others. Proximal Policy Optimization (PPO) is particularly favored due to its efficient balance between sample efficiency and implementation simplicity, making it well-suited for the dynamic and uncertain scenarios often encountered in swarm robotics [9].

The swarm architecture and design process, including the collective behavior, depend on several factors, such as the nature of the task required by a swarm, the environment, whether it's dynamic or static, and others. This process emphasizes the importance of understanding both the microscopic (individual agents and interactions) and macroscopic (collective behavior and task achievement) in creating effective swarm systems. For example, one of the challenges in creating an effective foraging swarm is adapting methods for searching and navigation in a continuous environment. Another challenge is ensuring the scalability of learned behaviors for different swarm sizes and configurations. Addressing the challenges requires sophisticated and adaptive algorithms that can effectively deal with the dynamic nature of swarm challenges.

2. Related works. The integration of the RL approach in swarm robotics offers significant benefits such as adaptability. This allows robots to adjust their behavior to dynamic environments through trial and error. It enhances autonomy by allowing robots to make self-decisions until they learn the optimal strategies. Moreover, RL adapts with scalability and flexibility, making the swarm applicable across various tasks [10, 11]. Particularly, RL in the foraging task optimizes resource collection and exploration, contributing to more efficient environmental mapping and more robust mission execution. However, some challenges persist, including managing the complexity of dynamic environments, ensuring the scalability of learned behaviors, overcoming communication limitations, balancing exploration with exploitation, and addressing energy and computational constraints [12]. Most recent studies have been conducted on strategies for deploying RL on generating foraging collective behavior. These challenges are addressed in the Table 1 with proposed solutions encompassing a spectrum of strategies.

Table 1. Challenges of deploying RL in foraging swarms.

Challenge	**Description**
Complexity of Dynamic Environments	Adapting to unpredictable changes, including moving targets and obstacles.
Scalability of Learning	Applying learned behaviors to swarms of varying sizes and compositions.
Communication Limitations	Coordinating actions without overwhelming the network or centralized control.
Balancing Exploration and Exploitation	Optimizing foraging efficiency by finding the right balance.
Energy and Computational Constraints	Managing energy consumption and computational demands for efficient operation.

Many researchers address the "complexity of dynamic environments" challenge in swarm robotics through various approaches. One study developed a macroscopic foraging behavior using deep RL, combined with microscopic behaviors controlled by fuzzy logic for obstacle avoidance and low-level navigation. This hybrid approach simplifies the RL search space, and achieves robust and scalable foraging behavior in swarms, even in scenarios that are not encountered during the training phase [13]. Another paper optimized the foraging strategy for active particles using Multi-Agent Reinforcement Learning (MARL). It addressed the problem by enabling the active particles to locate and efficiently forage from randomly occurring food sources. The paper demonstrated that individual optimization could lead to the emergence of collective behaviors that are beneficial to the swarm's overall foraging efficiency [14]. Additionally, a distinct approach utilizes deep RL with

curriculum learning to sequentially tackle navigation tasks, enhancing learning efficiency and adaptability to the environment. These methods collectively demonstrate the potential of advanced computational strategies to navigate and adapt to the uncertainty of dynamic environments [15].

Studies have addressed the scalability challenge by proposing various strategies. In [16], the authors focused on improving the system performance without increasing the complexity of individual robots or the need for heavy communication among them. They proposed leveraging simple, decentralized interactions among robots to achieve complex tasks. Also, [17] proposed a self-organizing task allocation model that enables swarm robots to dynamically distribute complex foraging tasks. This approach utilizes a response threshold model, which ensures efficient task allocation without the need for centralized control or extensive communication. It guarantees robust performance under varying conditions by effectively managing task distribution through local interactions. This makes it a scalable solution for complex swarm robotics applications. In addressing the challenges of communication limitations within swarm robotics, researchers have highlighted the development of innovative solutions to enhance the robustness and efficiency of systems in constrained communication environments. These strategies include the use of federated learning and deep RL to improve generalization and performance [8], alongside the adoption of biologically inspired communication mechanisms that enable decentralized operations [18]. These approaches significantly enhance the adaptability of swarm systems to dynamic conditions without the need for complex individual robot capabilities. For balancing exploration and exploitation, researchers have explored methods like Mutual-Information Upper Confidence Bound (MI-UCB) [19] and virtual pheromone mechanisms [20]. MI-UCB enhances drone coordination through decentralized Monte Carlo Tree Search. It improves surveillance performance by balancing information gain with reward maximization. Meanwhile, the use of virtual pheromones allows minimalist agents to effectively switch between exploring new resources and exploiting known ones. Finally, [21] conducted a study on the challenge of energy and computational constraints. It proposed a mobile edge computing solution integrated with a mobility-aware deep RL model for computation considerations. This approach reduces the computational cost of the robots, allowing for energy-efficient operation while meeting computation latency requirements. Compared to approaches that individually address challenges using deep RL with fuzzy logic, MARL, MI-UCB, curriculum learning, and others, our research emphasizes the advantages of a modular approach combined with the adaptability and efficiency of the PPO in dynamic environments. It enhances manageability,

adaptability, and efficiency in non-static environments. Specifically, it overcomes the observed limitations by providing an adaptive system that simplifies the debugging and computational demands, showcasing superior performance in terms of task completion and resource collection. Our approach aims to achieve individual learning and decision-making capabilities within a collective swarm intelligence framework.

3. System Description

3.1. Environment setup and system structure. The swarm system was implemented in a 3D robot simulator called Webots [22] where the E-Puck mobile robot was selected to form the swarm. A foraging collective behavior was produced to search for small and big boxes through the environment and then transport them to the nest. This behavior was evaluated through the environment as shown in Figure 1. The dimensions of the workspace were defined as $3 \times 3m^2$, forming a square area surrounded by four walls. The parameters of E-Puck robots were set as follows: linear velocity $V = [0, 0.25]m/s$, angular velocity $W = [-3.14, 3.14]rad/s$, and light sensors' readings LS_0, LS_7 corresponded to the light intensity $[0, 4095]$.

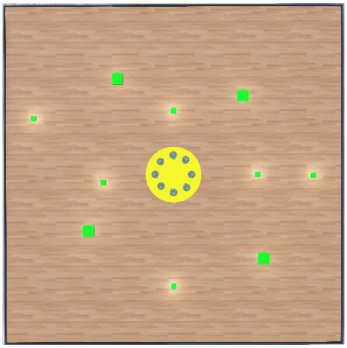


Fig. 1. Foraging environment

3.2. Modular Design. The system's modular design segregates the foraging task into discrete, manageable components, as shown in Figure 2:

– **Searching**: robots use light sensors to locate boxes, which emit light at varying intensities using PPO or PSO.

– **Gripping**: once a box is located, robots catch the box.

– **Waiting**: this state is for big boxes which require cooperative effort.

– **Transporting**: robots navigate back to the nest using PPO.

– **Release**: upon reaching the nest, robots release the box.

– **Return**: robots return to the searching phase, creating a continuous operational loop.

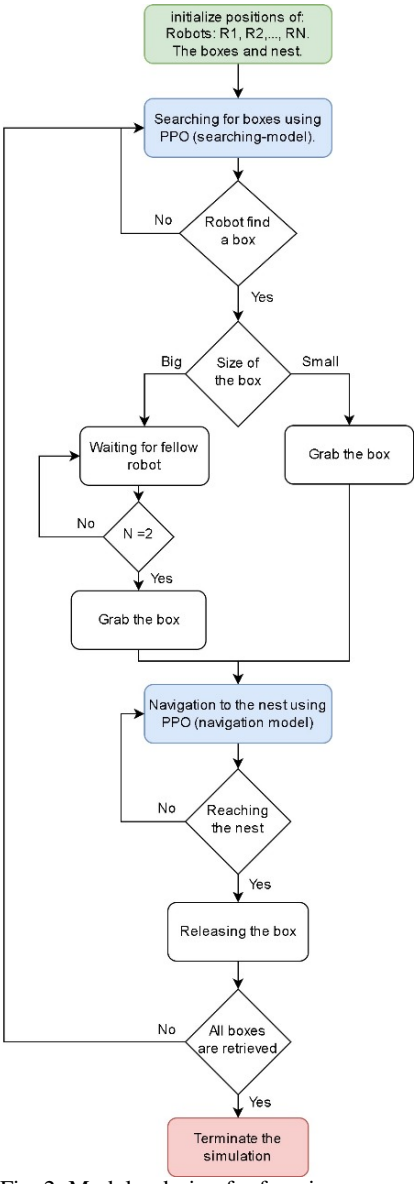


Fig. 2. Modular design for foraging swarm

In Figure 2, at the beginning, the system initializes the positions of all the entities involved: the robots, the foraging boxes (targets), and the nest. After initialization, the robots start the search phase. This first phase uses a PPO model or PSO algorithm specifically tuned for the search task. The algorithm guides the robots towards the targets, based on inputs from light sensors. These sensors assume that the boxes emit light, making them detectable. After finding a box, a decision determines the next steps based on the size of the box. If the box is small, a single robot can retrieve it. If the box is big, the robot waits until another robot arrives to help, ensuring cooperative transportation. This reflects a real-world situation where tasks may require different levels of effort and collaboration. After catching the box, the robots use another PPO model for navigation to return to the nest. This model processes the distance and angle between the robot and the nest to optimize the path. After successful delivery, the robot checks if any boxes are still uncollected. If so, it reverts to the search phase, creating a cycle of the foraging process. Once all targets have been retrieved, the simulation ends.

3.3. PPO architecture. It is used in the searching and transporting phases, chosen for its stability and robust performance in environments with high uncertainty. PPO operates via a policy gradient method that maximizes an objective function by using a clipped surrogate objective to keep updates stable. The main architecture of the PPO neural networks, consisting of an actor and a critic with fully connected layers, are shown in Figure 3.

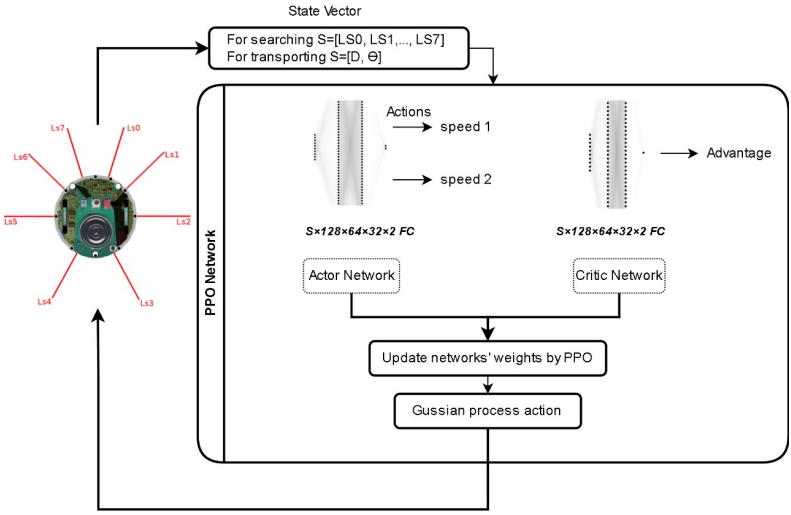


Fig. 3. PPO architecture

The problem is formulated as a MDP represented by the tuple (S, A, T, R, γ). The state space S comprised of light sensor readings during searching phase in addition to the distance between each robot and the nest D, and the angle between the robot and the nest θ in the transport phase. The action space A included the two velocities of the left and right motors. The transition function T describes the dynamics of the system, while the reward function R provides feedback based on the achieved objectives.

3.3.1. PPO Searching. The PPO's actor-critic networks receive input from the light sensors and outputs wheel velocities, adjusting the robot's trajectory towards the light source. A shaping reward is provided based on the change in light intensity, encouraging the robot to move towards brighter areas, i.e., closer to the boxes. The reward function $R(t)$ at time t is given as in Equation 1, 2:

$$R(t) = \frac{(LS_0^{(t)} - LS_0^{(t-1)}) + (LS_7^{(t)} - LS_7^{(t-1)})}{2} + r_{\text{box}}(t), \tag{1}$$

$$r_{\text{box}}(t) = \begin{cases} 1.1 & \text{if } LS_0^{(t)} > \text{FindThreshold}_{\text{searching}} \\ 1.1 & \text{if } LS_7^{(t)} > \text{FindThreshold}_{\text{searching}} \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

where $LS_0^{(t)}$, $LS_7^{(t)}$ – the normalized current readings of light sensors 0 and 7, respectively, at time t,
$LS_0^{(t-1)}$, $LS_7^{(t-1)}$ – the previous normalized readings of light sensors 0 and 7, respectively, at time $t-1$,
$\text{FindThreshold}_{\text{searching}}$ – the threshold value of the light sensor where the box is found. The normalized readings sensors are more than 0.85 that means the robot reaches the boundray of the box. It is defined experimentally,
$r_{\text{box}}(t)$ – the additional reward when the robot finds the box. The common approach to choose values of rewards like1.1 are defined experimentally to fit the environment.

3.3.2. PPO in Transporting. For transporting phase, the inputs of the PPO network are the robot's current distance and angle relative to the nest, with outputs modifying the wheel velocities to navigate the nest effectively. Rewards are sparse for successful delivery as in Equation 3, and the shaping method for leveraging the experience each time step to speed up the learning

process by considering the angle and the distance to know if the robot is in the direction of the nest or not as in Equation 4.

– **Reward in case of collaboration is not required (one robot is enough to transport a small box to the nest)**:

$$r_{\text{nest}} = \begin{cases} 0.1 & \text{if } d_{\text{current}} < \text{FindThreshold} \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

$$\text{reward} = (d_{\text{prev}} - d_{\text{current}}) + r_{\text{nest}} + \frac{\cos(\alpha_{\text{current}})}{1000}, \tag{4}$$

where r_{nest} – the obtained reward when robot reaches the nest.
FindThreshold – the Threshold to consider that the robot inside the nest. When the value of Threshold is less than 0.2. The nest circle has a radius 0.2 so when the distance between the robot and the center of the nest less than 0.2, the robot is in the nest,
d_{current} – the normalized distance between the robot and the nest at time t with $[0,3]$,
d_{prev} – the normalized distance between the robot and the nest at time t-1 with $[0,3]$,
α_{current} – the angle between the robot and the nest with $[0,2\pi]$.

This overall reward is designed to incentivize the robot to decrease its distance to the target (higher reward for reduced distance) and to orient itself towards the target (using the cosine of the angle).

– **Reward in case of collaboration is required (two robots have to transport a big box to the nest together**: Distance reward (dis_{reward}): When another robot (a "friend") is present within a certain distance range, a positive or negative reward is given based on the distance between the two robots (d_{robots}), as in Equation 5.

$$dis_{reward} = \begin{cases} 0.01 & \text{if } 0.035 \leqslant d_{\text{robots}} \leqslant 0.1 \\ -0.001 & \text{if } d_{\text{robots}} < 0.035 \\ -\frac{d_{\text{robots}}}{100} & \text{otherwise} \end{cases}, \tag{5}$$

when the calculated Euclidian distance between two robots is between 0.035 and 0.1, the two robots are rewarded for learning to stay together. They are punished if they get closer to less than 0.035 or go farther from each other. All numerical values are chosen by the trial-error approach. Nest Reward (r_{nest}): a

reward is given when the current distance to the nest ($d_{current}$) is less than a defined threshold (Threshold), as in Equation 6.

$$r_{\text{nest}} = \begin{cases} 0.15 & \text{if } d_{\text{current}} < \text{Threshold and friend is present} \\ 0 & \text{otherwise} \end{cases}. \quad (6)$$

The overall reward includes the difference in distance to the nest between the previous and current time step, the nest reward, the cosine of the current angle to the nest, and the distance reward,as in Equation 7. Let d_{prev} be the previous distance to the nest, and $d_{current}$ be the current distance to the nest, and $\alpha_{current}$ be the current angle to the nest.

$$\text{reward} = (d_{\text{prev}} - d_{\text{current}}) + r_{\text{nest}} + \frac{\cos(\alpha_{\text{current}})}{1000} + dis_{reward}. \quad (7)$$

3.4. PSO for comparative analysis. We implemented PSO to compare the performance. PSO is a biologically inspired computational algorithm that simulates the social behavior of organisms. Robots adjust their trajectory based on the collective movement of the swarm, aiming to find the optimal path by sharing information about individual successes. The PSO pseudo-code is shown in Algorithm 1, where $r1, r2$ – random values generally used to maintain diversity in the swarm's search behavior:

$$v_{rightmotor} = V + W + V_r, \quad (8)$$

$$v_{leftmotor} = V + W + V_l, \quad (9)$$

V_r – Avoiding speed for right wheel.
V_l – Avoiding speed for left wheel.
W – Turning velocity.
V – Linear velocity.

Algorithm 1. PSO algorithm

Initialize the environment and robots with positions and velocities
Define fitness function ← The average of the values of the sensors $[LS_0, LS_1, LS_6, LS_7]$.
repeat
  for each robot **do**
    Evaluate robot fitness using fitness function
    if current robot fitness is better than previous best robot fitness **then**
      Update robot best position to current position
    end if
    GET all robots fitness and positions
    Update global best position according to best robot fitness
  end for
  for each robot **do**
    Define $w = 0.7$, $c1 = 2$, $c2 = 2$
    Update robot velocity using equation:
    $v = w \times v + c1 \times r1 \times (\text{robot best} - \text{current position}) + c2 \times r2 \times (\text{global best} - \text{current position})$
    Update robot position using equation:
    position = position + velocity
    Calculate the linear velocity $V \leftarrow$ Distance to the new position
    Calculate the turning velocity $W \leftarrow$ Angle to the new position
    Calculate the avoiding speed for each wheel Vl, Vr
    Apply left motor speed $\leftarrow V + W + Vl$
    Apply right motor speed $\leftarrow V + W + Vr$
  end for
until robot reaches the box

4. Results and discussion. The proposed modular design, as outlined in the flowchart in Figure 2, leverages the application of PPO and PSO to enhance decision-making during the searching and navigation. Simultaneously, it maintains simplicity for less computation-intensive tasks, like gripping or waiting. This approach not only increases computational efficiency but also allows for specialized optimization when necessary. The modular approach offers multiple advantages:

– Specialized Optimization: PPO is deployed in modules that significantly influence task performance, such as search and transportation. This approach ensures that PPO's strengths are effectively utilized.

– Computational Efficiency: As a computationally intensive algorithm, the selective implementation of PPO optimizes the computational load, which is crucial for managing a large swarm of robots with limited processing power.

– Simplicity in Less Complex Modules: Certain tasks that do not require complex decision-making, such as gripping or releasing objects, benefit from simpler control mechanisms, facilitating ease of programming and system maintenance.

– Reduced Overfitting Risk: Limiting the usage of PPO to more complex tasks mitigates the risk of overfitting, ensuring that the model remains generalizable and applicable to diverse scenarios.

– Faster Training Time: By focusing on specific modules, PPO reduces the overall training time, speeding up the system deployment and adaptation.

– Maintaining Predictability in Certain Behaviors: Some modules prioritize predictability and reliability over adaptability, where rule-based behaviors are more appropriate.

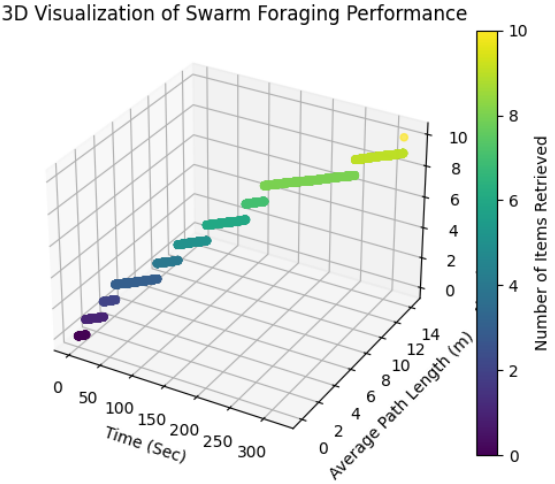
– Reward Design: The reward structure is carefully crafted to align with the objectives of each module. Designing the reward function just for two phases ensures that the main objective of the system will be achieved. It will prevent the system from engaging in unintended behaviors.

– End-to-end System Autonomy: An end-to-end system that includes one deep RL architecture to learn all behaviors like navigation, searching, gripping, and others for all robots' swarm. Relying solely on autonomous decision-making may not yield optimal efficiency. Therefore, combining autonomous and rule-based modules can create a more resilient system.

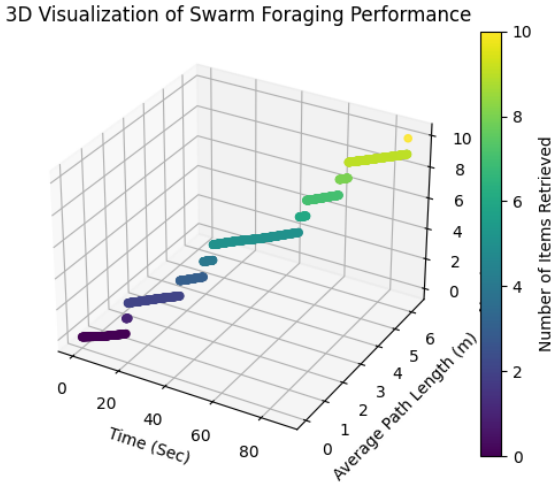
4.1. Foraging performance (RL vs PSO). The provided 3D visualizations in Figure 4 demonstrate the foraging behavior's characteristics of the swarm driven by PSO-PPO and PPO-PPO in addition to neumircal samples given in the Table 2.

Table 2. Foraging performance metrics

Retrieved items	PPO-PPO		PSO-PPO	
	Time (sec)	Average Path (m)	Time (sec)	Average Path (m)
1	13.024	0.726	7.552	0.377
2	13.696	0.774	25.024	1.173
3	27.712	1.767	36.928	1.719
4	33.92	2.243	78.08	3.513
5	36.512	2.434	98.112	4.451
6	59.488	4.129	125.92	5.725
7	61.408	4.268	163.776	7.436
8	69.92	4.942	181.6	8.256
9	72.192	5.126	272.64	12.051
10	88.096	6.305	320.096	14.211



a) PSO-PPO-driven swarm



b) PPO-PPO-driven swarm

Fig. 4. Foraging performance

In both graphs 4-a and 4-b, number of retrieved items N in the proposed problem is in the range $[0, 10]$. ΔT is the time required to collect all items and transport them to the nest. P_N is the average path length of the entire swarm needed to retrieve all items. Efficiency E is conceptualized as the number of items retrieved per unit of time and effort. To calculate efficiency, we can use the change in time ΔT, the average path lenght P_N, and the number of items retrieved N, as in Equation 10.

$$E = \frac{N}{\Delta T \times P_N}. \tag{10}$$

Based on Equation 10:

$$E_{PPO-PPO} = 10/(88.096 \times 6.31) = 18 \times 10^{-3},$$

$$E_{PSO-PPO} = 10/(320.096 \times 14.21) = 2.19 \times 10^{-3}.$$

The PPO-PPO-driven swarm has outperformed the PSO-PPO-driven swarm in terms of efficiency. Based on the data, it seems that the PPO model allows the swarm to retrieve items faster, as indicated by the steeper slope of the number of items over time. Additionally, the average path lengths taken are shorter for the PPO system. On the other hand, the PSO graph demonstrates a less steep slope, indicating a slower completation time in the foraging task. These results highlight the superiority of the PPO in rapidly adapting and efficiently solving the foraging task. The PPO not only learns faster but also appears to sustain its performance. This is due to PPO's policy gradient optimization, which allows fine-tuning adjustments to the robot's actions based on the received rewards, leading to a refined and more effective strategy. PSO, on the other hand, tends to converge to local optima and lacks the ability to fine-tune. Another reason is that PSO relies on collective swarm dynamics, which can also be a limitation if individual robots do not effectively follow the swarm's behavior or share information.

4.2. Dynamic behavior and autonomy. Based on the two sets of figures provided in Figure 5 and Figure 6, each shows the behavior of a swarm in a dynamic foraging task to collect two moving boxes. In the proposed dynamic situation with moving green boxes, the robots follow each box until they grip it. The box turns its color to red as an indicator of gripping and stopping its dynamic nature. Then, it is transported to the nest (yellow area).

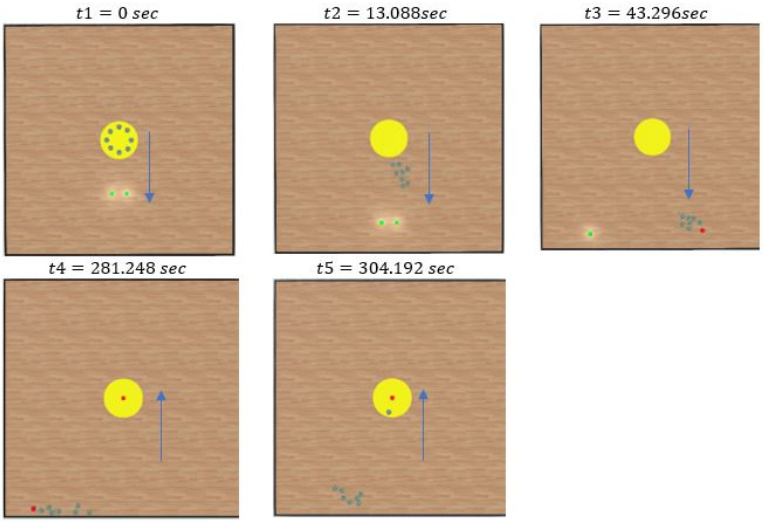


Fig. 5. Dynamic Forgaing performance/ PSO-driven swarm

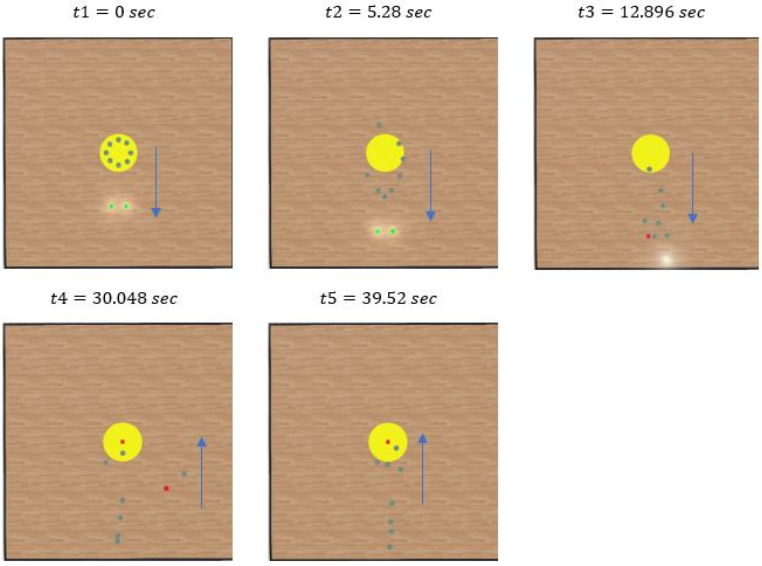


Fig. 6. Dynamic Forgaing performance/RL-driven swarm

Behavior analysis for PSO:

– Initial response (t1-t2): The swarm starts in the nest as an initial position and quickly follows one moving box, indicating a strong initial collective drive and less autonomy.

– Mid-phase (t3): As the swarm finds the green box, it clusters around it. Robots' movements are heavily influenced by their surroundings.

– Gripping action (t4-t5): When the box is gripped (indicated by a change in color to red), the robot that catches the box returns to the nest. The robot releases the box and returns to PSO mode. This demonstrates that PSO can localize and transport the target in dynamic scenarios.

Behavior analysis for RL:

– Initial Response (t1-t2): The RL swarm starts with a distributed, exploratory pattern with no immediate convergence, suggesting an exploratory approach. Which allows the swarm to follow both moving boxes at the same time with a high level of autonomy.

– Mid-phase (t3): The swarm gradually adjusts to the moving target, taking less time to locate the box compared to the PSO, which represents a better response to changes in the target's movement.

– Gripping action (t3-t4-t5): Once the box has been gripped, the robot navigates to the nest with a gripped box. It releases it and returns to the PPO searching box.

5. Conclusion. This study addresses a dynamic foraging task for a swarm of mobile robots. The proposed solution combines a modular design for handling processes like gripping, waiting for aid in carrying the big box, and releasing the box in the nest, with an intelligence algorithm for driving the searching and transportation processes, such as deep RL and PSO. This model maintains simplicity to allow for specialized optimization when necessary, like searching and transporting in a continuous environment while preventing overfitting. It introduces a module for testing various algorithms. Therefore, a comparative analysis was conducted on PPO and PSO. The results revealed that PPO achieved a faster retrieval rate, as well as better efficiency due to its high adaptability and autonomy. In contrast, PSO did not demonstrate the same level of efficiency or autonomy. The findings of this research emphasize the importance of selecting appropriate optimization algorithms depending on the specific task requirements. For tasks that require rapid adaptation and sustainable performance in dynamic environments, RL-PPO stands out as the most effective technique. The study also emphasizes the benefits of a modular approach to swarm robotics, offering a foundation for future developments in this field that require both efficiency and flexibility.

References

1. Cheraghi A., Shahzad S., Graffi K. Past, present, and future of swarm robotics. In Intelligent Systems and Applications: Proceedings of the 2021 Intelligent Systems Conference (IntelliSys). Springer International Publishing. 2022. vol. 3. pp. 190–233.
2. Brambilla M., Ferrante E., Birattari M., Dorigo M. Swarm robotics: a review from the swarm engineering perspective. Swarm Intelligence. 2013. vol. 7. pp. 1–41.
3. Schranz M., Umlauft M., Sende M., Elmenreich W. Swarm robotic behaviors and current applications. Frontiers in Robotics and AI. 2020. vol. 7. DOI: 10.3389/frobt.2020.00036.
4. Li J., Tan Y. A probabilistic finite state machine based strategy for multi-target search using swarm robotics. Applied Soft Computing. 2019. vol. 77. pp. 467–483.
5. Iskandar A., Kovacs B. A Survey on Automatic Design Methods for Swarm Robotics Systems. Carpathian Journal of Electronic and Computer Engineering. 2021. vol. 14. no. 2. pp. 1–5.
6. Jin B., Liang Y., Han Z., Ohkura K. Generating collective foraging behavior for robotic swarm using deep reinforcement learning. Artificial Life and Robotics. 2020. vol. 25. pp. 588–595.
7. Kakish Z., Elamvazhuthi K., Berman S. Using reinforcement learning to herd a robotic swarm to a target distribution. In Distributed Autonomous Robotic Systems: 15th International Symposium. Springer International Publishing. 2022. pp. 401–414.
8. Na S., Roucek T., Ulrich J., Pikman J., Krajnik T., Lennox B., Arvin F. Federated reinforcement learning for collective navigation of robotic swarms. IEEE Transactions on Cognitive and Developmental Systems. 2023. vol. 15. no. 4. pp. 2122–2131.
9. Wang Y., Damani M., Wang P., Cao, Y., Sartoretti G. Distributed reinforcement learning for robot teams: A review. Current Robotics Reports. 2022. vol. 3. no. 4. pp. 239–257.
10. Blais M., Akhloufi M. Reinforcement learning for swarm robotics: An overview of applications, algorithms and simulators. Cognitive Robotics. 2023. vol. 3. pp. 226–256. DOI: 10.1016/j.cogr.2023.07.004.
11. Dias P., Silva M., Rocha Filho G., Vargas P., Cota L., Pessin G. Swarm robotics: A perspective on the latest reviewed concepts and applications. Sensors. 2021. vol. 21. no. 6. DOI: 10.3390/s21062062.
12. Orr J., Dutta A. Multi-agent deep reinforcement learning for multi-robot applications: a survey. Sensors. 2023. vol. 23. no. 7. DOI: 10.3390/s23073625.
13. Aznar F., Pujol M., Rizo, R. Learning a swarm foraging behavior with microscopic fuzzy controllers using deep reinforcement learning. Applied Sciences. 2021. vol. 11. no. 6. DOI: 10.3390/app11062856.
14. Loffler R., Panizon E., Bechinger C. Collective foraging of active particles trained by reinforcement learning. Scientific Reports. 2023. vol. 13. no. 1. DOI: 10.1038/s41598-023-44268-3.
15. Alaa I., Bela K. Curriculum learning for deep reinforcement learning in swarm robotic navigation task. Multidiszciplinaris Tudomanyok. 2023. vol. 13. no. 3. pp. 175–187.
16. Altshuler Y. Recent Developments in the Theory and Applicability of Swarm Search. Entropy. 2023. vol. 25. no. 5. DOI: 10.3390/e25050710.
17. Lee W., Vaughan N., Kim D. Task allocation into a foraging task with a series of subtasks in swarm robotic system. IEEE Access. 2020. vol. 8. pp. 107549–107561.
18. Adams S., Jarne O, Mazo J. A self-guided approach for navigation in a minimalistic foraging robotic swarm. Autonomous Robots. 2023. vol. 47. no. 7. pp. 905–920.
19. Lee K., Kong F., Cannizzaro R., Palmer J., Johnson D., Yoo C., Fitch R. An upper confidence bound for simultaneous exploration and exploitation in heterogeneous multi-robot systems. In 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021. pp. 8685–8691.

20. Talamali M., Bose T., Haire M., Xu X., Marshall J., Reina A. Sophisticated collective foraging with minimalist agents: A swarm robotics test. Swarm Intelligence. 2020. vol. 14. no. 1. pp. 25–56.
21. Wang X., Guo H. Mobility-aware computation offloading for swarm robotics using deep reinforcement learning. In 2021 IEEE 18th Annual Consumer Communications and Networking Conference (CCNC). IEEE, 2021. pp. 1–4.
22. Michel O. Cyberbotics ltd. webots™: professional mobile robot simulation. International Journal of Advanced Robotic Systems. 2004. vol. 1. no. 1. DOI: 10.5772/5618.

Hammoud Ali — Ph. D. student, Faculty of applied informatics, Federal State Budgetary Educational Institution of Higher Education "Kuban State Agrarian University named after I.T. Trubilin". Research interests: multi-agent systems and decision-making. The number of publications — 1. ali-hammoud@mail.ru; 13, Kalinina St., 350044, Krasnodar, Russia; office phone: +7(861)221-5942.

Iskandar Alaa — Ph. D. student, Faculty of mechanical engineering and informatics (Istvan Salyi Doctoral School of Mechanical Engineering Sciences – mathematic institute), University of Miskolc. Research interests: reinforcement learning for swarm robotics, Navigation, and foraging behaviors. The number of publications — 3. iskandar.alaa@student.uni-miskolc.hu; Egyetemvaros, 3515, Miskolc city, Hungary; office phone: +(36)46-565-111.

Kovács Béla — Ph.D., Dr.Sci., Associate professor, Faculty of mechanical engineering and information, Institute of mathematics department of analysis, University of Miskolc. Research interests: mechanical engineering, differential equations. The number of publications — 99. matmn@uni-miskolc.hu; Egyetemvaros, 3515, Miskolc city, Hungary; office phone: +(36)46-565-111.

УДК 006.72 **DOI** 10.15622/ia.24.1.3

А. Хаммуд, А. Искандар, Б. Ковач

ДИНАМИЧЕСКОЕ ФУРАЖИРОВАНИЕ В РОЕВОЙ РОБОТОТЕХНИКЕ: ГИБРИДНЫЙ ПОДХОД С МОДУЛЬНОЙ КОНСТРУКЦИЕЙ И ГЛУБОКИМ ОБУЧЕНИЕМ С ПОДКРЕПЛЕНИЕМ

Хаммуд А., Искандар А., Ковач Б. **Динамическое фуражирование в роевой робототехнике: гибридный подход с модульной конструкцией и глубоким обучением с подкреплением.**

Аннотация. В этой статье предлагается гибридный подход, который объединяет интеллектуальные алгоритмы и модульную конструкцию для решения проблемы фуражирования в контексте роевой робототехники. Глубокое обучение с подкреплением (RL) и оптимизация роя частиц (PSO) используются в предлагаемой модульной архитектуре. Они используются для поиска множества ресурсов, которые различаются по размеру и демонстрируют динамическую природу с непредсказуемыми движениями. Кроме того, они транспортируют собранные ресурсы в гнездо. Рой состоит из 8 мобильных роботов E-Puck, каждый из которых оснащен датчиками света. Предлагаемая система построена на трехмерной среде с использованием симулятора Webots. С помощью модульного подхода мы решаем сложные проблемы фуражирования, характеризующиеся нестатичной средой и целями. Эта архитектура повышает управляемость, снижает вычислительные требования и упрощает процессы отладки. Наше моделирование показывает, что модель на основе RL превосходит PSO по времени выполнения задач, эффективности сбора ресурсов и адаптивности к динамическим средам, включая движущиеся цели. В частности, роботы, оснащенные RL, демонстрируют улучшенные способности к индивидуальному обучению и принятию решений, обеспечивая уровень автономии, который способствует коллективному интеллекту роя. В PSO коллективные знания роя в большей степени влияют на индивидуальное поведение роботов. Полученные результаты подчеркивают эффективность модульной конструкции и глубокого RL для продвижения автономных роботизированных систем в сложных и непредсказуемых условиях.

Ключевые слова: роевая робототехника, задача поиска пищи, модульное проектирование, обучение с подкреплением, оптимизация роя частиц.

Литература

1. Cheraghi A., Shahzad S., Graffi K. Past, present, and future of swarm robotics. In Intelligent Systems and Applications: Proceedings of the 2021 Intelligent Systems Conference (IntelliSys). Springer International Publishing. 2022. vol. 3. pp. 190–233.
2. Brambilla M., Ferrante E., Birattari M., Dorigo M. Swarm robotics: a review from the swarm engineering perspective. Swarm Intelligence. 2013. vol. 7. pp. 1–41.
3. Schranz M., Umlauft M., Sende M., Elmenreich W. Swarm robotic behaviors and current applications. Frontiers in Robotics and AI. 2020. vol. 7. DOI: 10.3389/frobt.2020.00036.
4. Li J., Tan Y. A probabilistic finite state machine based strategy for multi-target search using swarm robotics. Applied Soft Computing. 2019. vol. 77. pp. 467–483.
5. Iskandar A., Kovacs B. A Survey on Automatic Design Methods for Swarm Robotics Systems. Carpathian Journal of Electronic and Computer Engineering. 2021. vol. 14. no. 2. pp. 1–5.

6. Jin B., Liang Y., Han Z., Ohkura K. Generating collective foraging behavior for robotic swarm using deep reinforcement learning. Artificial Life and Robotics. 2020. vol. 25. pp. 588–595.
7. Kakish Z., Elamvazhuthi K., Berman S. Using reinforcement learning to herd a robotic swarm to a target distribution. In Distributed Autonomous Robotic Systems: 15th International Symposium. Springer International Publishing. 2022. pp. 401–414.
8. Na S., Roucek T., Ulrich J., Pikman J., Krajnik T., Lennox B., Arvin F. Federated reinforcement learning for collective navigation of robotic swarms. IEEE Transactions on Cognitive and Developmental Systems. 2023. vol. 15. no. 4. pp. 2122–2131.
9. Wang Y., Damani M., Wang P., Cao, Y., Sartoretti G. Distributed reinforcement learning for robot teams: A review. Current Robotics Reports. 2022. vol. 3. no. 4. pp. 239–257.
10. Blais M., Akhloufi M. Reinforcement learning for swarm robotics: An overview of applications, algorithms and simulators. Cognitive Robotics. 2023. vol. 3. pp. 226–256. DOI: 10.1016/j.cogr.2023.07.004.
11. Dias P., Silva M., Rocha Filho G., Vargas P., Cota L., Pessin G. Swarm robotics: A perspective on the latest reviewed concepts and applications. Sensors. 2021. vol. 21. no. 6. DOI: 10.3390/s21062062.
12. Orr J., Dutta A. Multi-agent deep reinforcement learning for multi-robot applications: a survey. Sensors. 2023. vol. 23. no. 7. DOI: 10.3390/s23073625.
13. Aznar F., Pujol M., Rizo, R. Learning a swarm foraging behavior with microscopic fuzzy controllers using deep reinforcement learning. Applied Sciences. 2021. vol. 11. no. 6. DOI: 10.3390/app11062856.
14. Loffler R., Panizon E., Bechinger C. Collective foraging of active particles trained by reinforcement learning. Scientific Reports. 2023. vol. 13. no. 1. DOI: 10.1038/s41598-023-44268-3.
15. Alaa I., Bela K. Curriculum learning for deep reinforcement learning in swarm robotic navigation task. Multidiszciplinaris Tudomanyok. 2023. vol. 13. no. 3. pp. 175–187.
16. Altshuler Y. Recent Developments in the Theory and Applicability of Swarm Search. Entropy. 2023. vol. 25. no. 5. DOI: 10.3390/e25050710.
17. Lee W., Vaughan N., Kim D. Task allocation into a foraging task with a series of subtasks in swarm robotic system. IEEE Access. 2020. vol. 8. pp. 107549–107561.
18. Adams S., Jarne O, Mazo J. A self-guided approach for navigation in a minimalistic foraging robotic swarm. Autonomous Robots. 2023. vol. 47. no. 7. pp. 905–920.
19. Lee K., Kong F., Cannizzaro R., Palmer J., Johnson D., Yoo C., Fitch R. An upper confidence bound for simultaneous exploration and exploitation in heterogeneous multi-robot systems. In 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021. pp. 8685–8691.
20. Talamali M., Bose T., Haire M., Xu X., Marshall J., Reina A. Sophisticated collective foraging with minimalist agents: A swarm robotics test. Swarm Intelligence. 2020. vol. 14. no. 1. pp. 25–56.
21. Wang X., Guo H. Mobility-aware computation offloading for swarm robotics using deep reinforcement learning. In 2021 IEEE 18th Annual Consumer Communications and Networking Conference (CCNC). IEEE, 2021. pp. 1–4.
22. Michel O. Cyberbotics ltd. webots™: professional mobile robot simulation. International Journal of Advanced Robotic Systems. 2004. vol. 1. no. 1. DOI: 10.5772/5618.

Хаммуд Али — аспирант, факультет прикладной информатики, Федеральное государственное бюджетное образовательное учреждение высшего образования «Кубанский государственный аграрный университет имени И.Т. Трубилина». Область научных

интересов: мультиагентные системы и принятие решений. Число научных публикаций — 1. ali-hammoud@mail.ru; улица Калинина, 13, 350044, Краснодар, Россия; р.т.: +7(861)221-5942.

Искандар Алаа — аспирант, факультет машиностроения и информатики (докторантура машиностроительных наук иштвана сали – математический институт), Университет Мишкольца. Область научных интересов: обучение с подкреплением для роевой робототехники, навигации и поиска пищи. Число научных публикаций — 3. iskandar.alaa@student.uni-miskolc.hu; Эгьетемварош, 3515, Мишкольц, Венгрия; р.т.: +(36)46-565-111.

Ковач Бела — Ph.D., Dr.Sci., доцент, факультет машиностроения и информатики института математики, кафедра анализа, Университет Мишкольца. Область научных интересов: машиностроение, дифференциальные уравнения. Число научных публикаций — 99. matmn@uni-miskolc.hu; Эгьетемварош, 3515, Мишкольц, Венгрия; р.т.: +(36)46-565-111.