Emergent Flocking Behaviour using Reinforcement Learning

Zakaria Narjis[1]

Institut für Neuro- und Bioinformatik, University of Luebeck
z.narjis@uni-luebeck.de

Abstract. Flocking behaviour, a widespread phenomenon in the natural world, represents coordination and collective motion observed among diverse species. Traditional approaches are mostly used to model this behaviour. However, these approaches rely on static flocking rules, limiting their adaptability to dynamic real-world scenarios. The challenge lies in effectively understanding and using this complex behaviour for practical applications. In this study, we present an approach using reinforcement learning to address this challenge. Our aim is to train autonomous agents to replicate flocking behaviour within a continuous 2D environment. The approach involves using a reward function to imitate flocking behaviour with an artificially generated flock. By overcoming these limitations, our study offers a deeper understanding of natural systems and broadens the scope for controlling swarming behaviours in various domains and environments.

Keywords: Reinforcement Learning · Swarm Robotics · Boid.

1 Introduction

Flocking and swarm behaviour is a widely seen phenomenon manifesting in animal behaviour. A notable example is bird swarming, where birds coordinate their movements, align directions, maintain cohesion, and avoid collisions. These collective actions offer great advantages such as enhanced efficiency, increased mating opportunities, and improved predator detection.

Reynolds [11] introduced a set of rules for modeling flocking behaviour, consisting of three fundamental rules representing the interactions among individuals, often referred to as 'Boids'. These rules are designed to enable individuals to exhibit coordinated group behaviour. This behaviour depends only on local information about their nearby neighbors within a certain proximity. There are three key rules: alignment, where Boids align their movement with the average heading direction of their neighboring flockmates, cohesion, where Boids move toward the center of mass of their local flockmates; and separation, where Boids actively avoid close encounters by steering away from their neighboring flockmates. Reynolds' work revealed the mechanics of flocking behaviour, allowing local interactions among Boids to give rise to complex flocking patterns.

Researchers are exploring computational and AI-based solutions, to enhance traditional approaches. The developing methods include machine learning concepts and agent-based modeling. The use of this behaviour is crucial in fields such as robotics, traffic management, and crowd control, where traditional approaches are limited in versatility.

In our paper, we use reinforcement learning to train agent to imitate behaviour of "Boids" through Reynolds' rules. Reinforcement learning as an inspired approach from evolution enables an agent get evolved with interacting with the environment. This approach enables our artificial agents to replicate the swarms behaviour observed in nature.

To present the findings of the study, the paper is organized as follows. Section 2 provides an overview of recent advancements in the study of flocking behaviour. In section 3, preliminary notions about reinforcement learning are presented. Section 4 details the design and implementation of the solution. In Section 5, the results of the study are presented. Finally, a conclusion is presented to summarize the study, emphasize the paper's perspectives, discuss the limitations of the results, and suggest potential future improvements.

2 Related Work

Many studies have been conducted to understand and imitate flocking behaviour observed in diverse species. In 2008, Morihiro et al. [10] introduced an approach to overcome the static flocking rules proposed by Reynolds. By employing reinforcement learning, they trained individual agents to adaptively adhere to alignment, cohesion, and separation principles. This was achieved by implementing a reward signal based on distances between individuals, constraining actions to attraction, repulsion, or parallel movement with respect to neighboring individuals. Their work played a vital role in addressing the limitations of static flocking rules.

Subsequently, studies in the realm of flocking behaviour continued to evolve. Hahn et al. [5] presented a reinforcement learning approach featuring multiple autonomous agents navigating a continuous space with the primary goal of evading a pursuing predator. Remarkably, their work revealed that agents, when trained using reinforcement learning techniques, can prioritize individual survival, leading to the emergence of flocking behaviour akin to the classic Reynolds simulation.

In 2023, Shadi & Kathryn [1] focused on enabling machines to recognize and classify collective motion behaviour. Their methodology involved the creation of a dataset with ground truth data obtained through a human perception survey, specifically geared toward collective motion behaviour recognition.

3 Design and implementation

In order to train the agent on flocking behaviour, we created an environment that models a population of artificial boids using Reynolds rules. The objective

of the agent is to follow the same rules at each time step within the environment to maximize the cumulative reward.

3.1 Environment

The whole artificial population (agent and generated boids) is scattered randomly at the beginning in a continuous two-dimensional space with predefined size. For simplicity, boids can move freely without collision however get propelled if they surpass rectangular borders space. They are represented as a filled circle with a surface smaller than environment space's (see Fig. 1).

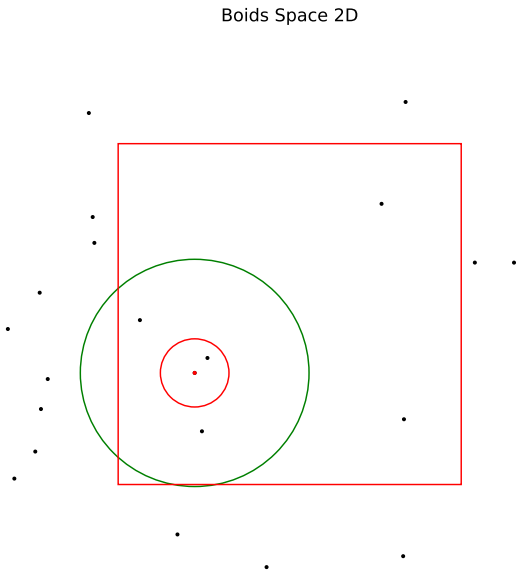

Fig. 1. Example of space with 19 Boids (black) and one agent (red). The borders are represented by the red rectangle.

Within the outer radius R_o, the agent should steer toward the average heading direction and position of its neighboring boids. Conversely, within the inner radius R_i, the agent should move away from neighboring boids. These radius are visually represented as a green circle for the outer radius and a red circle for the inner radius surrounding the agent.(see Fig. 1)

3.2 Observation Space

The observation space available to the agent is restricted to consider only its own state and those of n proximate boids residing within its designated outer radius. This paradigm aligns with biological analogies, similarly to scenarios observed in nature where, for instance, a bird within a larger flock possesses the capacity to perceive only its immediate neighbors rather than the entirety of the collective group. At each time step, the agent receives five vectors $[p_t, v_t, a_t, c_t, s_t]$ and one scalar n_t where:

- p_t : actual agent's position vector;
- v_t : actual agent's velocity vector;
- a_t : vector that points to average heading direction of neighboring boids within outer radius R_o;
- c_t : vector that points to the center of mass of neighboring boids within outer radius R_o;
- s_t : vector that points to away from the center of mass of neighboring boids within inner radius R_i;
- n_t : neighboring boids within outer radius R_o.

3.3 Action Space

Action space governing behaviour of the reinforcement learning agent comprises a discrete set of nine possible actions, each dictating the agent's velocity along the x and y axes. The velocity values can take on one of three states: -1, 0, or 1, for both the x and y dimensions. Consequently, this action space considers a range of nine distinct combinations, enabling the agent to change it's position at each time step.(see Fig. 2)

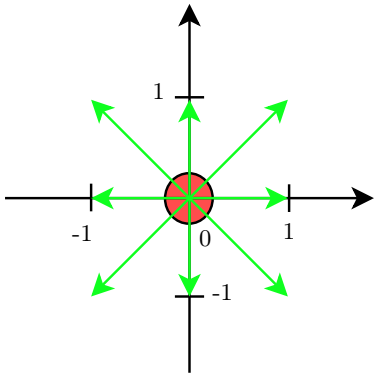

Fig. 2. Agent action space represented by the green arrows.

3.4 Reward function

The reward function is composed of flocking and grouping. The assessment of flocking considers an evaluation of the disparity between the agent's subsequent states, resulting from action a, and the states associated with an optimal action a^* that adheres to Reynolds' behavioural rules. This evaluation quantifies variations in alignment, cohesion, and separation vectors between the optimal next state and the state that follows the agent's chosen action. The quantification of this disparity for the case of 2D space is measured using mean squared error with the following formula:

$$\varepsilon_f = -\frac{1}{6}\sum_{i=1}^{6}(y_i - \hat{y}_i)^2 \tag{1}$$

Where $y = (a_x, a_y, s_x, s_y, c_x, c_y)$ for the next state following action a, and $\hat{y} = (a_x, a_y, s_x, s_y, c_x, c_y)$ for the next state following Reynolds rules.

Nevertheless, relying only on flocking principles it's not sufficient to induce the agent to exhibit flocking behaviour. This limitation arises from the agent's inclination to remain outside the proximity of other boids, thereby receiving positive rewards. To address this, the concept of grouping error is introduced, serving the purpose of compelling the agent to reassemble or regroup with its fellow boids. The grouping error is calculated with the following formula:

$$\varepsilon_g = \begin{cases} -(10 + d_{\text{center}}) & \text{if } n_t = 0 \\ 0 & \text{else} \end{cases} \tag{2}$$

The variable d_{center}, denoting the agent's distance from the spatial center. It serves the purpose of encouraging the agent to explore the central region. This emphasis is driven by the higher likelihood of encountering other boids in the central region compared to the periphery. The reward function is then expressed as weighted sum of the two error following the formula:

$$Reward = w_f\varepsilon_f + w_g\varepsilon_g \tag{3}$$

Here, w_f and w_g denote the respective weighting factors assigned to the flocking and grouping errors. These weights serve the purpose of amplifying error magnitudes to ensure accurate gradient flow.

3.5 Training

To train the agent, we utilized the Double Deep Q-Networks (DDQN) method proposed by Van Hasselt et al. [6], which was implemented within the PFRL library developed by Fujita et al. [4]. This library offers the flexibility to customize all hyperparameters required for deep reinforcement learning. Additionally, it integrates with OpenAI Gym Environments developed by Brockman et al. [2], providing an abstraction layer through the gym.env class. The spatial and boid models were realized using the AgentPy library developed by Foramitti & Joël [3], constituting the comprehensive Boids reinforcement learning environment together with the gym.env class.

Table 1. Space hyperparameters used for training.

Hyperparameters	Values
Space size	$50x50$
Episode steps	50
Population	10
Inner Radius	3
Outer Radius	10
Borders distance	10
Alignment strength	0.3
Cohesion strength	0.005
Separation strength	0.1
Border strength	0.5

Table 2. DDQN hyperparameters used for trainning.

Hyperparameters	values
Training steps	500000
Hidden layers	Linear with batch normalization
Number of hidden Layers	4
Number of hidden neurons	$[64, 32, 16, 9]$
Hidden layer activation	LeakyReLU with negative slope $\alpha = 0.01$
Discount factor γ	0.9
Optimizer	Adam
Learning rate	0.001
Replay buffer capacity	100000
Batch Size	34
Target network update interval	20000 steps
Exploration	Linear Decaying Epsilon
ϵ_{start}	0.9
ϵ_{end}	0.01
Decay steps	250000

4 Results

In this subsection, we present results of training and simulation of the agent. Our agent underwent training over 500,000 time steps and was subject to periodic evaluations every 10,000 steps, during which the mean episodic reward over 1,000 steps was computed. The results, as depicted in Figure 3, provide a compelling insight into the learning dynamics of the agent. Notably, our findings reveal a significant convergence point at approximately 200,000 training steps. This convergence signifies that the agent successfully learned to exhibit emergent flocking behaviour following Reynolds rules (see Figure 4). The presented results highlight the effectiveness of the DDQN algorithm in enabling artificial agents to acquire complex and adaptive behaviours, mirroring those observed in nature.

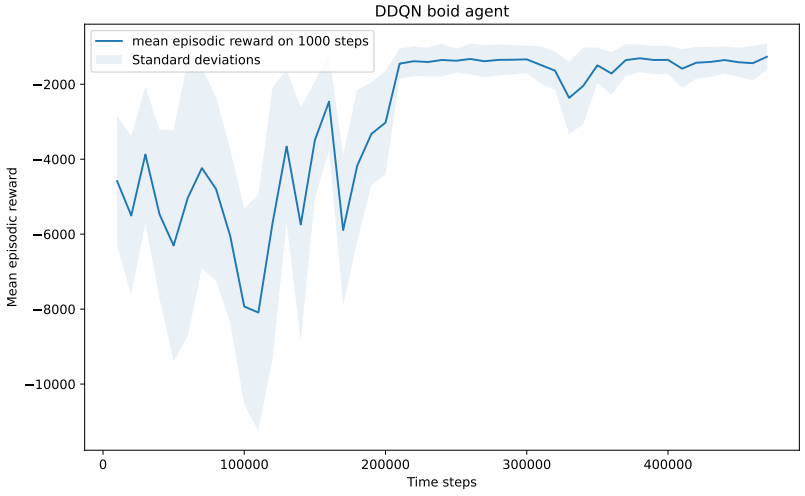

Fig. 3. Mean episodic reward of each evaluation step on DDQN algorithm.

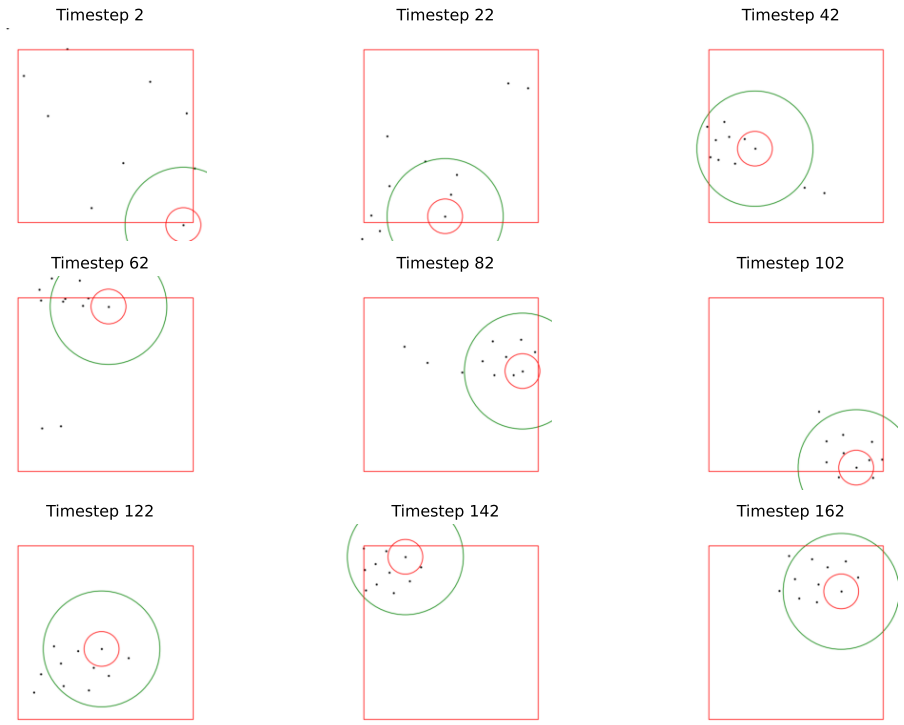

Fig. 4. Flocking agent dynamics: Evolution over 200 time steps.

5 Conclusion and Future Work

In this study, we trained a reinforcement learning agent to replicate the flocking behaviour observed in natural systems, inspired by Reynolds' rules. Our agent underwent extensive training, accumulating 500,000 time steps and undergoing periodic evaluations every 10,000 steps. Notably, it exhibited significant convergence at around 200,000 training steps, successfully emulating emergent flocking behaviour.

However, our DDQN rewards did not reach higher values, primarily due to the discrete action space employed. To address this limitation, future work should explore algorithms designed for continuous action spaces, such as deep deterministic policy gradient (DDPG) proposed by Lillicrap et al. [9]. This approach has the potential to enhance both performance and reward convergence. Moreover, it is worth noting that further investigation into our reward function may enhance our agent's performance.

Additionally, the partial observability of our problem presents an intriguing challenge. To tackle this, we recommend investigating the integration of Deep

Recurrent Q-Learning Networks (DRQN) by Hausknecht & Stone [7]. By incorporating memory and sequential information processing, DRQN could empower our agent to make more informed decisions, potentially achieving superior results in emulating complex flocking behaviours.

Furthermore, to improve the agent's ability to understand the heading and speed of flocks of birds, we propose extending our research by training the agent on variable inner and outer radius parameters. This modification can be applied to any reinforcement learning agent equipped with a memory architecture, such as HCAM (Hierarchical Chunk Attention Memory) proposed by Lampinen et al. [8], or DRQN. By enabling the agent to benefit from previous observations even when there are no neighboring birds or only a few present, this approach could provide valuable insights into the behaviours of distant flocks.

References

1. Abpeikar, S., Kasmarik, K.: Motion behaviour recognition dataset collected from human perception of collective motion behaviour. Data in Brief **47**, 108976 (2023). https://doi.org/https://doi.org/10.1016/j.dib.2023.108976, https://www.sciencedirect.com/science/article/pii/S235234092300094X
2. Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W.: Openai gym (2016). https://doi.org/10.48550/arXiv.1606.01540
3. Foramitti, J.: Agentpy: A package for agent-based modeling in python. Journal of Open Source Software **6**, 3065 (06 2021). https://doi.org/10.21105/joss.03065
4. Fujita, Y., Nagarajan, P., Kataoka, T., Ishikawa, T.: Chainerrl: A deep reinforcement learning library. Journal of Machine Learning Research **22**(77), 1–14 (2021), http://jmlr.org/papers/v22/20-376.html
5. Hahn, C., Phan, T., Gabor, T., Belzner, L., Linnhoff-Popien, C.: Emergent escape-based flocking behavior using multi-agent reinforcement learning (2019). https://doi.org/10.48550/arXiv.1905.04077
6. van Hasselt, H., Guez, A., Silver, D.: Deep reinforcement learning with double q-learning (2015). https://doi.org/10.48550/arXiv.1509.06461
7. Hausknecht, M., Stone, P.: Deep recurrent q-learning for partially observable mdps (2017). https://doi.org/10.48550/arXiv.1507.06527
8. Lampinen, A.K., Chan, S.C.Y., Banino, A., Hill, F.: Towards mental time travel: a hierarchical memory for reinforcement learning agents (2021). https://doi.org/10.48550/arXiv.2105.14039
9. Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning (2019). https://doi.org/10.48550/arXiv.1509.02971
10. Morihiro, K., Nishimura, H., Isokawa, T., Matsui, N.: Learning grouping and anti-predator behaviors for multi-agent systems. pp. 426–433 (01 2008). https://doi.org/10.1007/11893011_89
11. Reynolds, C.W.: Flocks, herds and schools: A distributed behavioral model. In: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques. p. 25–34. SIGGRAPH '87, Association for Computing Machinery, New York, NY, USA (1987). https://doi.org/10.1145/37401.37406, https://doi.org/10.1145/37401.37406