Task 4

Report

David Cohen - Mat. 0001085730

Indice

0.1 Implementation

In this project, the behaviors implemented were **walk**, **stop**, and **wandering with obstacle avoidance (WOA)**. WOA was unified due to their strong correlation, avoiding redundancy in the code. The **walk** and **stop** behaviors were designed to operate at a higher level, invoking WOA as needed.

0.2 Wandering with Obstacle Avoidance Behavior

Initially, previous subsumption **wandering** and **obstacle avoidance** behaviors were reused. However, they were too complex and hindered proper aggregation. **Complex behaviors interfere with aggregation because frequent and wider movements disrupt cohesion.**

By simplifying WOA behavior and ensuring higher-level behaviors took precedence, significant improvements in swarm aggregation were achieved. The WOA behavior was thus kept essential, focusing only on obstacle avoidance and random movement.

0.2.1 Walk and Stop Behaviors

The robot's behavior was modeled using an internal finite state automaton (FSA) with two states: **WALK** and **STOP**. Transitions between these states are probabilistic and depend on the number of nearby stopped robots. The integration of FSA with subsumption architecture provided a structured and modular approach.

- **Walk**: activated when the robot is in a walking state. The robot sets its LEDs to yellow and signals through the range and bearing system that it is moving. This behavior **calls wander** to handle movement and obstacle avoidance.
- **Stop**: Activated when the robot needs to stop. The robot sets its LEDs to red and signals through the range and bearing system that it is stopped. The **stop** behavior preempts any lower behavior, ensuring an immediate stop.

1 Performance Analysis

More focus was given to this part of the task. To ensure a more accurate performance evaluation of the swarm robotics system, several functionalities and improvements were integrated from the previous task:

- **Dynamic Parameter Configuration:** all parameters were extracted into a separate configuration file (`hyperparameters.lua`). This approach allowed for the easy testing of different parameter combinations.
- **Automation Script:** the script (`test-simulation.sh`) was refactored to generate unique configurations for each parameter combination and run them in parallel. This method enabled the efficient evaluation of a wide parameter space, significantly reducing the time required for comprehensive testing.
- **Average Distance Calculation:** each robot calculated the average distance to its neighbors within the range of the **range_and_bearing** system. The overall metric for evaluating the simulation was the average of the individual distances to other robots.

1.1 Simulation Setup

The following setup was used for the simulations using a **fixed seed**:

- **Number of Robots: 50 robots** were deployed in a rectangular arena measuring 4x5 meters.
- **Simulation Duration:** each simulation ran for **3000 steps** to allow sufficient time for behaviors to manifest and for aggregation to occur.
- **Parameters:** various sets of hyperparameters were tested to optimize performance, including variations in maximum velocity, alpha, beta, stopping probability (S), and walking probability (W).

The specific ranges for the hyperparameters tested in the simulations resulted in **81 combinations**:

- $VMAX$: 15 (fixed)
- α: 0.05, 0.1, 0.15
- β: 0.03, 0.05, 0.07
- S: 0.01, 0.02, 0.03
- W: 0.05, 0.1, 0.15

1.2 Results

Although the absence of proper null hypothesis testing and more simulations with different seeds might limit the conclusiveness of the results, the focus was on simulating a real-case scenario, applying in practice what we learned in lessons.

The most effective hyperparameter combination was found to be:

MAX_VELOCITY: 15.00, **ALPHA:** 0.15, **BETA:** 0.07, **S:** 0.01, **W:** 0.15, **Mean Average Neighbor Distance:** 27.39

The histogram in Figure 1 shows the distribution of average neighbor distances, illustrating the distribution of distances across all simulations. This visualization shows that tuning hyperparameters can significantly change the results of each simulation.

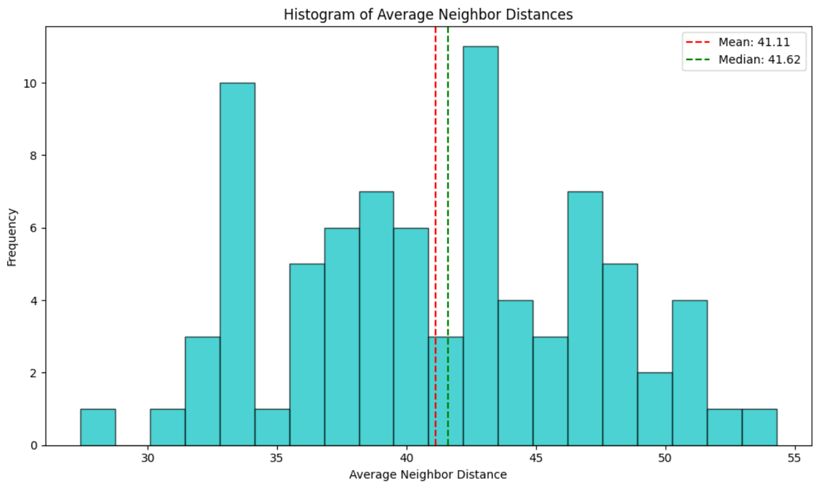

Figura 1: Histogram of Average Neighbor Distances

Figure 2 illustrates the performance of each hyperparameter set, with the best performing hyperparameter set highlighted for easy identification.

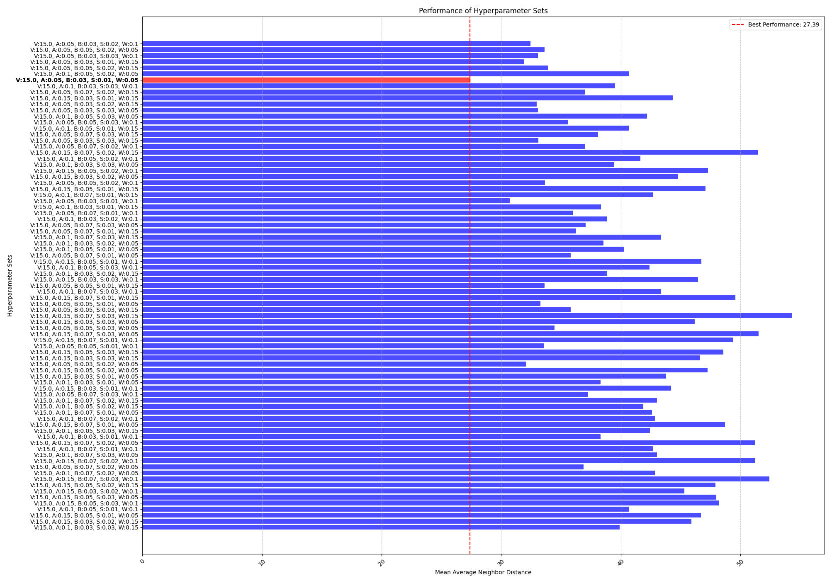

Figura 2: Performance of Hyperparameter Sets

2 Conclusion & Toughts

The results demonstrate a potential approach to analyze the aggregation task, showing the effectiveness of optimizing control parameters for achieving efficient swarm aggregation.

- In my case, the systems proved to be **not really robust**, as small changes in parameter values slightly affected the system's performance.
- Experiments with distributed behaviours show more flexibility on designing and scalability .