Improving Cooperative Swarm Navigation - Team 36

Nolan Groves **Landry Tun**

Vineeth Veligeti **Nitin Yarava** **Satyanarayana Mulagala**

1 Introduction

This project is about cooperative navigation in robotics swarms. The presence of many robots in an environment presents unique challenges and opportunities for navigation and can allow for navigation without reliance on pre-existing maps or external guidance systems. We focus on a scenario where a single robot must locate and navigate toward a target within an environment of randomly moving robots going about other tasks but assisting in navigation. First, we set up a communication-based navigation system that allows us to exchange navigation information among robots, facilitating the searcher's task. To achieve this, we use robots equipped with Infrared Range and Bearing (IrRB) sensors, which allow the transition of information over line-of-sight while also knowing the distance and direction the message was sent from. The navigation information is stored in a data structure called a 'navTable' within the robot. Each entry in this table represents a target and includes the age of the information and an estimated distance to that target. This allows robots to keep track of and update their knowledge about the targets' locations based on new information received from other robots in the swarm. Since the information is transmitted over line-of-sight, it guarantees an obstacle-free path, and a robot can follow the chain of information back to the target The problem we are solving showcases the system's ability to function without any type of mappings, but merely using wireless communication for real-time data sharing, which is crucial for making decisions in unstructured environments. By using a system that relies on real-time communication rather than a fixed map, robots can adapt to dynamic environments, and work together without having to coordinate complex mapping data.

2 Related Work

Optimizing the navigation in robotic swarms is done on several fronts, such as building better communication algorithms (ex: information sharing through modifying environments like ants [1]), building algorithms that use swarms to localize and map the environment and use the information to navigate properly (ex: a robotic bat algorithm that uses sensors to gather information of surroundings and use the shared information to build a map and navigate [4]), or building better simulation software to bridge the gap between simulation and the real-world [7]. Some algorithms are built to specifically solve certain domains of problems such as maze solving [4], and the transport of swarms to several event points [1]. Strategies for maze solving include remembering the particular obstacles in a memory module and the angle to turn at those locations. This information is then shared among the whole swarm. these strategies when applied to a whole swarm will lead to optimization in the navigation behavior. Another approach is building strategies for better information sharing by building hierarchies among the swarm [6] and allocating different roles in information gathering and sharing. This hierarchy can be entirely virtual or can be physically implemented (robots with better mapping sensors being on the boundary, and ones with better communications sensors and compute power can be in the middle of the swarm. the middle ones can help in better decision-making and delegating tasks.)

The paper we are building upon is focused on using the swarm to navigate with incomplete information. By simply using the presence of other bots in the environment and having them relay a minimal

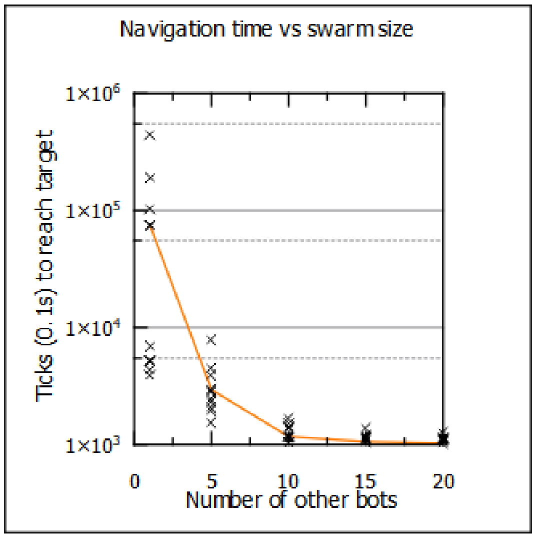


Figure 1: Results of running numerous simulations timing a single robot navigating an empty room using cooperative swarm navigation, with varying numbers of other bots assisting

amount of information, other bots in the swarm area can navigate without having to use any complex mapping or sensors.

3 Baseline Results

Our project focuses on recreating and improving the Single Robot Navigation results from the original paper[1]. Using the ARGoS simulator[2], we created a swarm of Footbots that communicated navigational data using IrRB and implemented the navigational algorithm as described by [1]. For our initial experiment, the bots were placed in a 4 by 4-meter walled arena, and the range of communication was limited to 1.5m to simulate a sparser swarm setup without requiring simulating a larger area. A single target robot remains stationary in the top right corner, while a "navigator" (or "nav") robot starts in the bottom left. The rest of the arena is filled (randomly and uniformly) with assistant bots that wander aimlessly, turning around when they bump into obstacles. In our initial experiment, we used the **Navigation with Stopping (NwS)** algorithm, where the navigator robot would stop moving once it reached its last known navigational coordinate. The number of other bots was changed between values of 1, 5, 10, 15, and 20, then the time it took for the navigator to successfully find the target bot was recorded. (Figure 1)

While the larger swarms quickly converged on a navigation time of around 1000 ticks (100 seconds), the sparser swarm navigation times compose a long-tailed distribution that sometimes took several hours to find the target. Additionally, the algorithm was successfully shown to be able to navigate an environment with randomly placed obstacles scattered throughout.

The NwS algorithm was only used for our initial testing, to ensure that the algorithm was working properly and produced a similar distribution to the original paper. To more closely match the paper's results, we also implemented the **Navigation with Random (NwR)** variant of the algorithm. In NwR, if the robot runs out of navigation data, it begins moving around randomly, choosing a random direction and moving in that direction for a length of time drawn from an exponential distribution. The NwR algorithm outperformed the NwS algorithm in all aspects in the original paper, so it is the algorithm we worked to improve upon and NwS was ignored.

Additionally, we also altered the arena size and layout to match the original. The arena was increased in size to 10m x 10m, and two main layouts were used for testing; an empty arena with no obstacles except for the walls, and an arena with four sections of wall that form a plus shape (Figure 2)

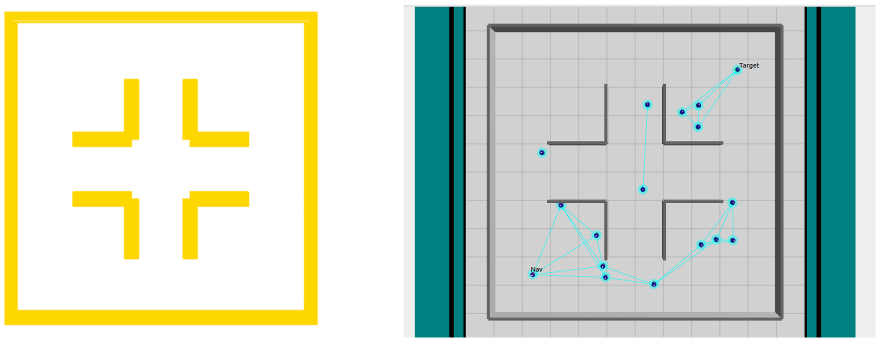


Figure 2: Left: The arena layout from the original paper. Right: A screenshot of the recreated arena in our Argos simulator. The blue lines show robots that are in range to communicate with each other.

4 Our Contribution

We picked three separate areas to improve upon the existing algorithm. Our first improvement was to add a new variant of the navigational strategy. [1] has shown the performance comparisons between two navigation strategies, namely: **Navigation with stopping (NwS)** & **Navigation with random (NwR)**. We employed a different strategy called **Navigation with Direction (NwD)**, where rather than wandering aimlessly should they run out of information, the robots keep track of the direction information is coming from, and travel in this direction rather than a random one. The second improvement was adding a parity bit to the navigational data being transmitted, allowing it to be more robust to environmental noise. The third improvement was a new strategy based on NwD where the robot attempts to estimate a more direct path to the target. However, the second and third improvements did not result in improvements to the baseline, so the paper will focus on the first algorithm for the sake of brevity and the results of the other improvement ideas will be left to the appendix.

4.1 Implementing Navigation with Direction

Whenever any bot receives new navigational information, the direction that this data is received from is also now stored in the navigation table (and updated as the bot moves). When the nav robot receives good information and begins heading towards a roaming robot (let's call it robot B), robot B remembers the direction it received the navigation information from (which should be in a direction closer to the target), and should the nav robot not receive any newer information before reaching B, it can travel in that saved direction rather than a random one. However, this information does not fit into the 10 bytes allocated for messages, and without a fixed reference point the information is not directly helpful, as each robot only knows directions relative to its current heading. Thus, we implemented a 3 step process for transmitting this directional information. (Figure 3)

Whenever the nav robot receives navigational data that is better than its current data, it will now send a message requesting directional information. This allows robot B to know the relative direction of the nav robot, and can thus calculate the saved direction with respect to the direction of the nav robot. It then sends back a message containing the saved direction, relative to the facing that the nav robot would have after it reaches B's current location. If the nav robot reaches B and does has not received better info by that time, it can then turn toward the saved direction directly without additional calculations. Since messages sent with the IrRB equipment are indiscriminate, the nav robot will receive directional information from all nearby robots upon making a request; it simply ignores any that did not come from the same direction as the original navigation information.

5 Results

The Navigation with Direction algorithm shows considerable improvement over the baseline NwR results at low swarm sizes, taking a shorter time to reach the target and with less variance (Figure 4). At larger swarm sizes, particularly in an arena without obstacles, there is never a disconnected enough swarm that there is no navigation information, and thus the two algorithms perform identically under

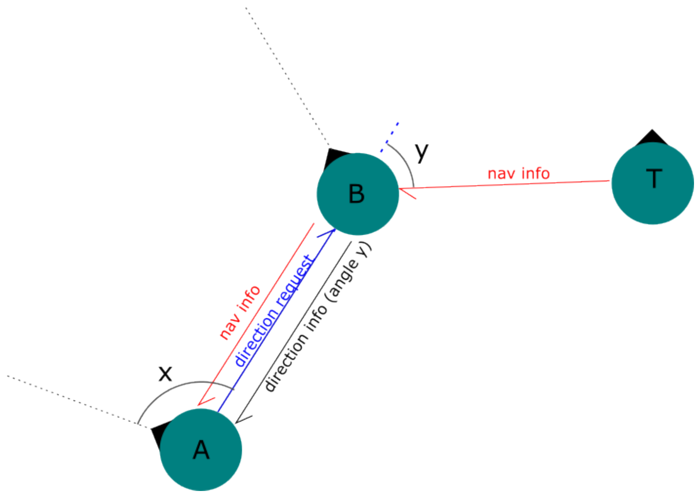


Figure 3: Diagram showing the 3-part exchange of directional information between a nav robot (A), a wandering swarm robot (B) and another robot (T), either representing the actual target or simply another link in the chain of bots that is closer to the target.

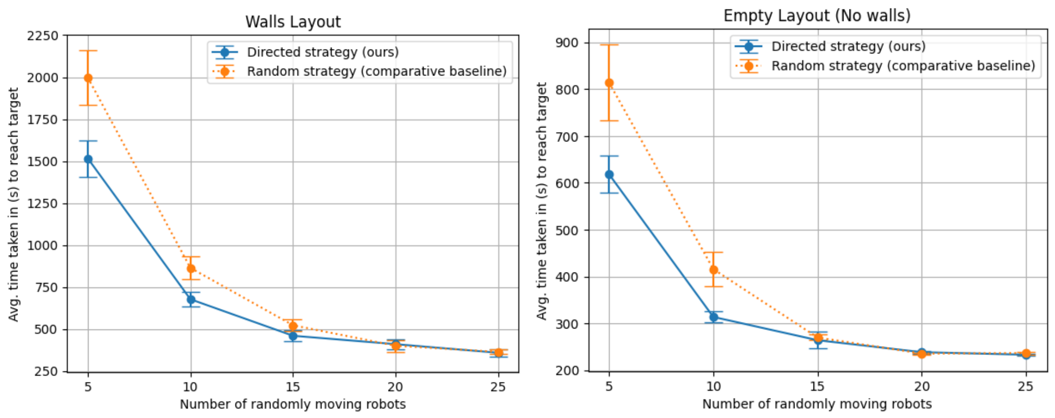


Figure 4: Comparison of our NwD algorithm vs the NwR baseline on two different arenas, one with the walls in a plus shape, and an empty arena. Each of the data points represents the average of 100 trials under that specific combination of arena, algorithm, and swarm size.

those conditions. This shows that our algorithm does meaningfully improve upon the performance of the original algorithm in situations where the spread of navigational information is more sparse.

6 Conclusion

The original paper proposes an algorithm that allows for pathfinding without the use of traditional techniques such as vision or stationary waypoints. However, in more disconnected scenarios, the original algorithm falls back on random movement to blindly path it's way forward. By keeping track of the flow of navigation information, even if it's just a rough approximation due to the robot's limited localization ability, we are able to improve significantly over random chance in these scenarios. This improvement maintains the robustness of the original algorithm, only using our approximate direction when no data is available, and falls back to the original random motion should the saved direction not pan out.

References

[1] Ducatelle, F., Di Caro, G.A., Förster, A. et al. Cooperative navigation in robotic swarms. *Swarm Intell* 8, 1–33 (2014). https://doi.org/10.1007/s11721-013-0089-4

[2] Carlo Pinciroli, Vito Trianni, Rehan O'Grady, Giovanni Pini, Arne Brutschy, Manuele Brambilla, Nithin Mathews, Eliseo Ferrante, Gianni Di Caro, Frederick Ducatelle, Mauro Birattari, Luca Maria Gambardella, Marco Dorigo. 2012. **ARGoS: a Modular, Parallel, Multi-Engine Simulator for Multi-Robot Systems.** Swarm Intelligence, volume 6, number 4, pages 271-295. Springer, Berlin, Germany.

[3] Khalil, A. (2017).Swarm robotics: Cooperative navigation in unknown environments [Master's Thesis, the American University in Cairo]. AUC Knowledge Fountain.

[4] Youssefi, K. A. R., Rouhani, M. (2021). Swarm intelligence based robotic search in unknown maze-like environments. Expert Systems With Applications, 178, 114907. https://doi.org/10.1016/j.eswa.2021.114907

[5] Junior, L. S., Nedjah, N. (2016). Efficient Strategy for Collective Navigation Control in Swarm Robotics. Procedia Computer Science, 80, 814-823. https://doi.org/10.1016/j.procs.2016.05.371

[6] Benavidez, Patrick Nagothu, Kranthimanoj Ray, Anjan Shaneyfelt, Ted Kota, Satyanarayana Behera, Laxmidhar Jamshidi, Mo. (2008). Multi-domain robotic swarm communication system. 1 - 6. 10.1109/SYSOSE.2008.4724189.

[7] Calderón-Arce, C.; Brenes-Torres, J.C.; Solis-Ortega, R. Swarm Robotics: Simulators, Platforms and Applications Review. Computation 2022, 10, 80. https://doi.org/10.3390/computation10060080

7 Appendix

7.1 Interpolated Movement using Directional Information

In Section 3.1, we described the algorithm we used to get the relative direction of the next robot along the path to the target. The idea here is to use that direction, to skip reaching the nearby robot and directly go towards the next robot. As shown in figure 5, the robot will move in curved interpolated paths until it reaches the target. The Expectation is that the robot travels less distance and reaches the target faster. We implemented this by combining the directions of the robots A and B in the figure. In practice, weighing the direction of B made the robot rotate in the same place without moving further, so we used a low weight for B's direction.

The Interpolated movement algorithm performed worse than the baseline on all swarm sizes. Though it takes less time with large swarm sizes, results are considerably worse than baseline. We have done about 100 trials for each swarm size, the results are shown in the figure5.

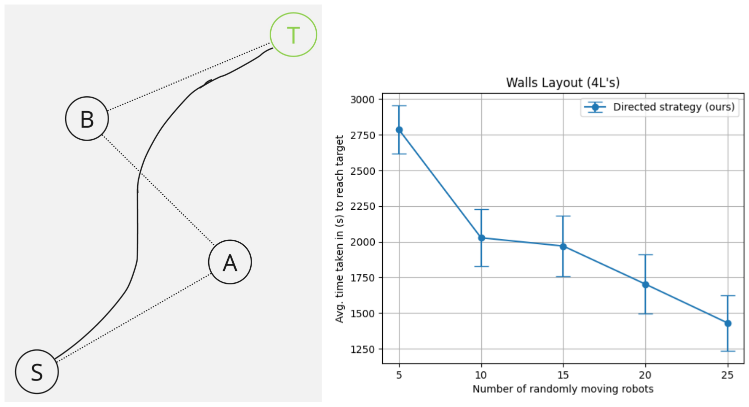


Figure 5: source robot moving in an interpolated path using relative direction of the next robot along the path. results show time taken to reach target robot for different swarm sizes.

7.2 Interpolated Movement using Mean Heading Direction of Multiple Robots

In this section, we try to estimate the approximate direction towards the target by using the next heading directions of multiple robots. With the presence of multiple robots in the swarm having information to reach the target, using the average estimate to interpolate the robot direction might be efficient instead of relying on the best navigation direction.

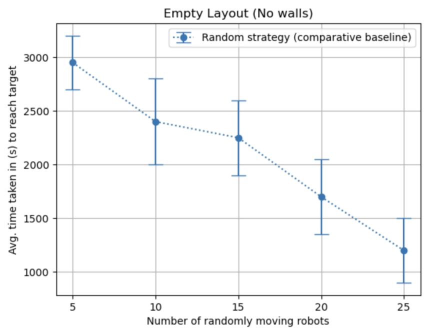
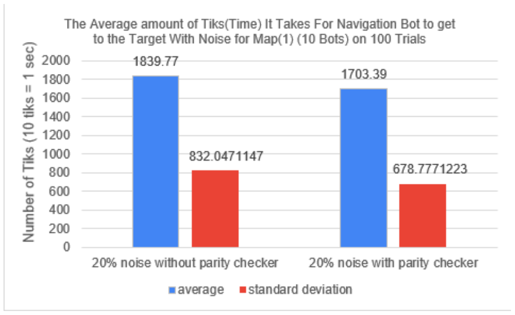


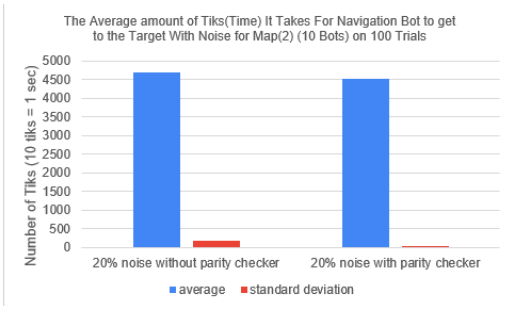
Figure 6

7.3 Error checking

Bots sending and receiving messages perfectly without any interference is not a very realistic scenario. There will be times when communication is hindered or misinterpreted due to the environment or other factors. With that being said, we added two kinds of noise. The first one is the built-in noise from Infrared Range and Bearing. This is used to simulate the effect of environmental and sensor noise on the range and bearing readings. We initially have the noise set to Gaussian noise with a standard deviation of 1.0. This noise model assumes that the errors in sensor readings are normally distributed, which is a common assumption for many real-world sensors. The second type of noise we added is a bit-flipping noise. We make it in a way that there is a twenty percent chance a bit will flip in the process of sending messages. We planned to tackle this by putting an even parity bit at the lowest bit of the message so that we know if the message is corrupted or not. If it is, we ignore it and reuse the initial message to reach the target until it receives a new one that is not corrupted.



(a) The Average amount of Ticks(Time) It Takes For Navigation Bot to get to the Target With Noise for Map(1) (10 Bots) on 100 Trials



(b) The Average amount of Ticks(Time) It Takes For Navigation Bot to get to the Target With Noise for Map(2) (10 Bots) on 100 Trials

Figure 7

For the results on Navigation with parity checker, navigating through noise, do not show a significant results to tackle communication noise (figure 7a). Both images in figure 7 are the same except they navigate through different placments of obstacles.