Planner-Guided Robot Swarm Demonstration with Natural Language Control

Michael Schader(✉) and Sean Luke

George Mason University, Fairfax, VA 22030, USA
{mschader,sean}@gmu.edu
https://cs.gmu.edu/robotics

Abstract. Robot swarms have been proposed as a way to take advantage of the scalability, robustness, and adaptability of natural large-scale multiagent systems in order to solve engineering challenges. However, accomplishing complex tasks while remaining flexible and decentralized has proven elusive. Our prior work on planner-guided robot swarms successfully combined a distributed swarm algorithm implementing low-level behaviors with automated parallel planners and executives selecting high-level actions for the swarm to perform as a whole, but had only been tested in simplistic grid-world simulations. Here we demonstrate our approach on physical robots augmented with experiments in continuous-space simulation, showing that it is an effective and efficient mechanism for achieving difficult task objectives to which swarms are rarely applied. We also use a Large Language Model prompted with the planning domain definition and a natural language goal statement to generate the formal problem definition, enabling non-expert users to control the swarm.

Keywords: Multi-robot systems and real world robotics · Agent cooperation and negotiation · Human agent interaction

1 Introduction

The field of swarm robotics prizes three cardinal virtues. The first virtue is scalability, thanks to potentially large numbers of inexpensive robots. The second is robustness, the ability to withstand the loss of members and to accommodate the addition of new ones. The third is adaptability, the appropriate response to changing conditions in the environment. These virtues take inspiration from natural systems such as ant colonies, flocks of birds, schools of fish, and so on. To achieve these goals, swarm robotics designs have historically taken the form of potentially large numbers of simple and usually homogenous robots, with limited and typically local interaction and communication, and with loosely coupled or entirely separated decision-making.

P. Mathieu and F. De la Prieta (Eds.): PAAMS Workshop 2025, LNAI 16031, pp. 372–377, 2026.
https://doi.org/10.1007/978-3-032-07638-0_33

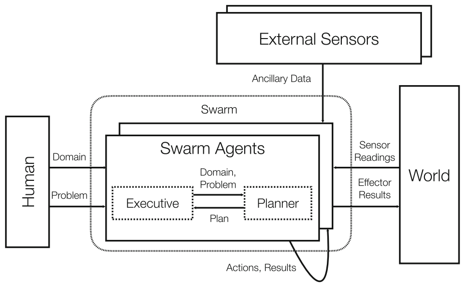


Fig. 1. Planner-guided swarm architecture.

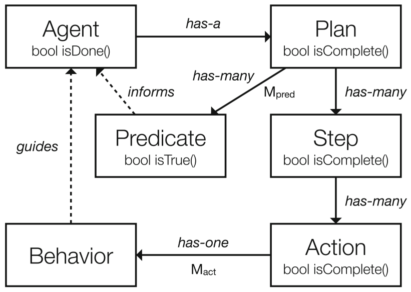


Fig. 2. Component relationships.

However, the highly decoupled and distributed nature of a typical robot swarm, valued for these virtues, has also proven difficult to control. As the survey of Brambilla et al. [2] noted, "[d]ue to the lack of a centralized controller, it is in general very difficult to effectively control a swarm once it starts operating." Because they are loosely coupled, swarms by design cannot easily coordinate to do synchronized, interleaved, or nontrivial collaborative tasks. Rather, swarm robotics dogma often turns to *emergent behavior*, arguing that swarms can achieve complex macro-level behavior through the micro-interactions of many agents. But while it is feasible, through simulation, to predict the resulting macrophenomena arising from these interactions, a critical inverse problem—identifying which micro-behaviors will achieve a desired macrophenomenon — is generally unsolved and perhaps unsolvable. Collective behavior involving synchronization and coordination has proven elusive. In short: researchers have succeeded in getting swarms to forage, patrol, distribute themselves, and form shapes, but swarms have not shown promise in working together to build a house.

The tension here is between coordination and decentralization. The classic method for identifying, solving, and executing synchronized and collaborative robot tasks is to use a (normally centralized) task planner and executive with tight coupling. But when doing so, a swarm degenerates into a single-agent system with multiple effectors (the swarm robots), hurting scalability due to network complexity, and damaging robustness by relying on a single point of failure. Global knowledge held by every agent would not scale well and would limit adaptability, and requiring long-distance communication among robots would violate the swarm-style focus on having only local interactions.

We are interested in endowing swarm architectures with sophisticated collaboration and synchronization. To this end, we have developed *planner-guided robot swarms* as a novel solution to these problems. In our method, the mission for the swarm is specified in automated planning terms. Each agent has its own planner, and all use the same algorithm and inputs, yielding identical results. The swarm is treated as a set of one or more *virtual agents*, each composed of many real ones and responsible for the parallel execution of the actions in the plan steps. An *a priori* mapping of virtual agent actions to real agent behaviors is the bridge that leads to emergent behavior in service of the mission objectives. Our approach does not require tight synchronization among swarm members but is still robust to retrograde behavior among out-of-sync robots. The method also seeks to ensure that their plans will ultimately synchronize and align (Fig. 1 and Fig. 2).

2 Main Purpose

In our early work in this area [8,9], we assumed ideal and simplistic conditions in a trivial simulated grid-world, with predictable communications and none of the sensor noise or action failures associated with actual physical robots. In this work we remedy that deficiency, showing that potentially large groups of physical robots can collectively perform synchronized and planned actions. The robots are able to do these tasks while overcoming physical crowding and interference, significant difficulties in localization and wayfinding, and physical challenges inherent in object detection, grasping, and manipulation. We further show that the method scales and that it can adapt to dynamic changes in the environment and in the nature of the robot swarm.

In this demonstration we exercise the planner-guided swarm approach in real-world conditions with groups of physical robots, as well as the noise, localization and wayfinding difficulties, physical crowding, and interference that comes with them. The robots are nonetheless able to perform a coordinated and planned task using this technique. We also show via simulation that the method still exhibits qualities ascribed to swarms, namely that it scales with the number of agents, that it can deal with changes in the makeup of the swarm, and that it can adapt to dynamic changes in the environment.

We also address a practical limitation in the architecture: the need for a human to specify the initial conditions and the goal conditions using Planning Domain Definition Language (PDDL) [5]. Many potential use cases involve a non-technical user directing a robot swarm to accomplish some multistep task. Requiring that user to express the initial and goal states in formal PDDL would make the system more difficult to use and limit its possible reach. We show that a Large Language Model (LLM) can be used to convert the user's natural language description of the conditions into predicates for the planner to process, thus expanding the potential applications of the system (Fig. 4 and Fig. 5).

3 Demonstration

We performed three experiments using physical robots and in simulation, all using a scenario called "Brick Layering". The first was a baseline experiment that demonstrated that the method could scale to large numbers of agents (we tested up to 64). The second experiment showed that the method was robust to unexpected changes in the number of agents in the swarm, both tolerating loss of agents and accepting new ones. The third experiment demonstrated that

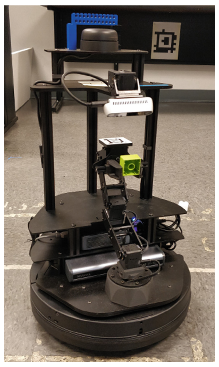


Fig. 3. Locobot with brick.

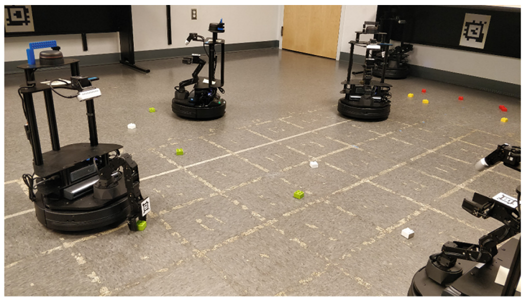
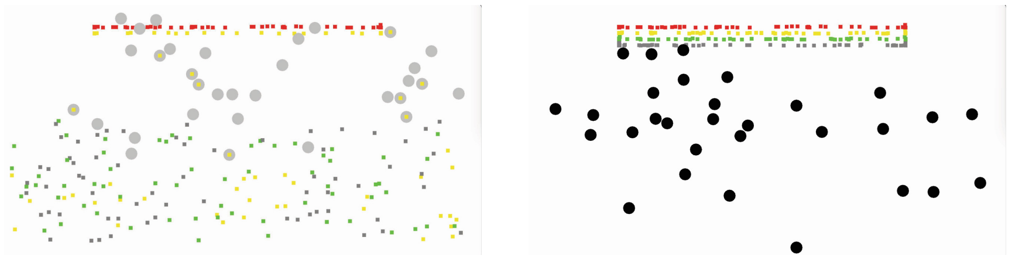


Fig. 4. Robots at work on a Brick Layering scenario. Red and yellow bricks are at their destinations in the top right; green and white ones are still in the field. (Color figure online)



(a) Agents (gray disks) explore the field for bricks (colored rectangles).

(b) All four rows are correctly filled in; the agents have declared the job complete.

Fig. 5. Stages of the Brick Layering scenario in simulation.

the method could deal with noise and unexpected state changes in the robot environment.

Our simulations were created with the MASON multiagent simulation toolkit [4] using two-dimensional continuous space. For the physical robot implementation we used the Trossen Robotics PX100, a robot platform based on the open-source LoCoBot design (Fig. 3). We positioned four AprilTag [6] two-dimensional barcodes on the sides of the experiment area to add visual localization assistance to the onboard odometry, and incorporated YOLOv5 object detection [3] trained on a custom set of images of colored Duplo bricks on the floor.

All code for the abstract swarm operations, including PDDL definitions, parallel planning, success token management, and completion criteria, was shared between the physical robot and simulation implementations. We used our own custom implementation of the GraphPlan algorithm [1] built using the PDDL4J planning toolkit [7](Fig. 6,Fig. 7 and Fig. 8).

For the LLM exploration [10], we created a system prompt beginning with, “You are a PDDL expert. In your answers, only respond with PDDL code...” Then we supplied the PDDL domain definition along with a natural language description of the initial and goal states, e.g. “Write a PDDL problem definition that starts with all areas being empty and ends with item1 in area4, item2 in

Robots	*Minutes to completion*	*Speedup vs one*
1	45.1	1.00
2	24.1	1.87
3	17.6	2.56
4	16.4	2.75

Fig. 6. Scalability results with physical robots.

Robots	*Deliveries per minute*	*Speedup vs one*
1	0.53	1.00
2	0.73	1.39
3	0.81	1.53
4	1.26	2.39

Fig. 7. Robustness results with physical robots.

Notif. step	*Mins to compl.*	*Add'l mins needed*
2	29.2	5.1
3	31.9	7.8
4	33.7	9.6

Fig. 8. Adaptability results with physical robots.

area3, item3 in area2, and item4 in area1." Smaller LLMs did not usually produce usable output, but flagship models such as GPT and Claude were consistently successful, and self-hosted CodeGemma performed well too.

4 Conclusions

Implementing the planner-guided swarm algorithm on physical robots exposed many challenges that were not readily apparent when working in simulation. In the course of conducting the experiments, we made a number of observations relevant to practical swarm robot engineering.

Crowding and Interference. Robots are not point objects. With real robots, crowding is a much bigger problem than in simple simulations. In the physical world, crowding can become a dominant factor even with small numbers of agents (four in our lab). Robots frequently bump into each other, causing them to be nudged out of position, miss their destinations, and fail to pick up bricks. Furthermore, often two robots will detect the same brick, then spend time aligning, approaching, and targeting it, only to have one grab it and the other miss, or for both to miss due to an arm collision (which can and does damage hardware). We mitigated this problem to some extent by having each robot announce when it's about to attempt a pickup and delay arm movements if another is grabbing, but this de facto mutex lock either works only occasionally due to timing issues, or introduces long delays if timeout periods are extended.

Localization. The agents in our previous grid-world simulations needed no absolute positioning knowledge because they navigated using pheromones and could sense when they were in a pickup or dropoff zone. Our real robots lack this, and so must rely on other, noisy localization methods (odometry and AprilTags in our work) to grab and drop bricks in the correct places and to prevent retrograde behavior (e.g. one removing bricks from where another has just correctly placed them). Similarly, without pheromones the robots need other mechanisms for wayfinding and deconfliction. Lacking help from a pheromone gradient, they employ arc-turn maneuvers to escape from head-on encounters. These motions are usable but imperfect solutions to problems that don't even exist in the grid-world simulations.

Delay. The physical robots experience inertial delays in their rotation and significant delays due to repeated misses in grasping bricks and maneuvers to avoid interference among arms. Most importantly, the data pipeline that provides our real robot code with the image location of a brick runs with an approximately one-second latency. This small delay has implications for every aspect of the system. While rotating to scan for bricks in the area, we maintain a first-in/first-out queue of historical pose measurements to pair with the delayed detection notifications so the robot can turn back to where it spotted a target. When preparing to move the arm to pick up a brick, we need to insert delays to ensure the robot has an up-to-date image with which to plan the arm motion.

In this demonstration, we built on our prior work with planner-guided robot swarms, showing the system running on physical robots and in continuous-space simulation for the first time. We demonstrated that the technique is scalable, robust, and adaptable in real swarm and multi-robot conditions, and that an LLM can ease the work of specifying initial and goal conditions. This will help pave the road from research to real-world applications for this powerful, general approach to swarm engineering.

References

1. Blum, A.L., Furst, M.L.: Fast planning through planning graph analysis. Artif. Intell. **90**(1–2), 281–300 (1997)
2. Brambilla, M., Ferrante, E., Birattari, M., Dorigo, M.: Swarm robotics: a review from the swarm engineering perspective. Swarm Intell. **7**(1), 1–41 (2013)
3. Jocher, G., et al.: ultralytics/yolov5: v3.1 - bug fixes and performance improvements (2020). https://doi.org/10.5281/zenodo.4154370
4. Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., Balan, G.: MASON: a multiagent simulation environment. SIMULATION **81**(7), 517–527 (2005)
5. McDermott, D., et al.: PDDL: the planning domain definition language (1998)
6. Olson, E.: Apriltag: a robust and flexible visual fiducial system. In: 2011 IEEE International Conference On Robotics And Automation, pp. 3400–3407. IEEE (2011)
7. Pellier, D., Fiorino, H.: PDDL4J: a planning domain description library for Java. J. Exp. Theor. Artif. Intell. **30**(1), 143–176 (2018)
8. Schader, M., Luke, S.: Planner-guided robot swarms. In: International Conference on Practical Applications of Agents and Multi-Agent Systems, pp. 224–237. Springer (2020)
9. Schader, M., Luke, S.: Fully decentralized planner-guided robot swarms. In: International Conference on Practical Applications of Agents and Multi-Agent Systems, pp. 241–254. Springer (2021)
10. Smirnov, P., Joublin, F., Ceravola, A., Gienger, M.: Generating consistent PDDl domains with large language models. arXiv preprint arXiv:2404.07751 (2024)