Decentralized Autonomous Rovers

*Freedom Rover Units

1st Jordy A. Larrea Rodriguez
Department of Electrical and Computer Engineering
University of Utah
Salt Lake City, USA
Jordy.larrearodriguez@gmail.com

2nd Brittney L. Morales
Department of Electrical and Computer Engineering
University of Utah
Salt Lake City, USA
brittneymrls@gmail.com

3rd Misael Nava
Department of Electrical and Computer Engineering
University of Utah
Salt Lake City, USA
misaelnava812@gmail.com

***Abstract*—The state of the art in autonomous swarms employs a decentralized model consisting of multi-agent networks. These multi-agent systems hold the potential to adapt to new environments and optimize individual performance to specific tasks without having to deal with global systems prone to single points of failure. Our team's focus lies therein in developing a decentralized multi-agent system. The decentralized multi-agent system will incorporate three to five two-wheel drive rovers interfaced through the Robot Operating System (ROS) by raspberry pi 4 companion computers for edge-computing specifications. A central system is still beneficial for the assignment of global objectives; thus, our proposed decentralized system will employ a central bay station to communicate objectives for the agents to complete (carefully designed demos). LiDAR, simple infrared sensors, and low-resolution cameras will be procured to facilitate real-time (RT) decision-making by the agent(s) in response to the environment. Our development stack will leverage ROS for project management, simulation capabilities, navigation libraries, and native server-client model in robotics applications. The rovers AI will incorporate simultaneous localization and mapping (SLAM) techniques for RT positioning based on a priori grid or map; thus, facilitating navigation through improved state space mapping. Decentralized swarms allow for a set of agents to act as more than distributed actuators: i.e., more like your white blood cells rather than your limb as observed by central systems. Furthermore, drone Swarms hold the potential to gather large quantities of data for monitoring and area mapping that would otherwise prove too costly to collect, and can greatly simplify and reduce manpower in search-and-rescue, disaster recovery, or security scenarios. Swarms essentially hold the capability to replace humans in potentially dangerous and normally costly tasks.**

***Index Terms*—Decentralized Communication, Swarm Communication, Multi-agent, ROS, Gazebo**

I. Introduction and Background

As all forms infrastructure continue to evolve and grown in complexity, the pressing need for technology that can function beyond natural human capabilities. Consider industrial accidents, natural disasters, and other catastrophic accidents. Increasing global human population densities risk exacerbated potential fatalities and overwhelmed local services [2]–[4], [6]. Robots could prove versatile in situations that require human involvement due to the mass-manufacturable nature of cyber-physical systems, advancements to AI, and the negated risk to live human agents. Drones unlike their human counterparts do not need to meet biological requirements for sustained action; thus, drones can be deployed for a substantial amount of time without rest, can navigate spaces hazardous to humans, can loco-mote via flight, can reach high speeds, can inexpensively monitor expansive industrial or rural areas, and can gather large amounts of data fast and cost effectively [5]. In order for drone swarms to reach the level of versatility described above, every agent must make its own decisions in response to the environment and global criteria/task assignments in a collaborative manner with other agents (CITE Basically Everybody). These so called multi-agent systems (MAS) eliminate single points of failures and can complete tasks collaboratively and independently of other agents. Like divide and conquer techniques in modern software solutions, MAS systems can delegate and work together to complete a task without central nodes bottle-necking task completion or MAS response to dynamic environments [3]. Thus, our motivation to design and develop a MAS comes from the novel and useful applications of decentralized swarms. Currently, ROS and Gazebo have been used widely for simulation in robotics and MAS projects. ROS itself was born due to an inherent need for dynamic and adaptive control from task-specific industrial robots and to decrease the development time of robotics projects. ROS incorporates a server-client model where each 'robot' running a ROS program is a node that can 'subscribe' or listen to certain topics where messaging can then occur. Furthermore, packages and tool-boxes that ship with or are compatible with ROS incorporate path-planning techniques, search and navigation, mapping techniques, control theory, and automation algorithms. MAS projects hold the disadvantage of long debugging times and a necessity for hardware and software to function seamlessly. Three-dimensional simulation

software like Gazebo allow for reconstruction of environments and deployment of agents running on experimental software [4]. Our team intends on simulating, testing, debugging, and validating our automated agents via Gazebo before deploying them in the real world. The aforementioned approach is recommended both in literature and by industry experts that we spoke with. Computers meant to run complex operating systems like the raspberry pi are inadequate to handle RT control of motors, actuators, and sensors (poor interfacing/port options). Thus, our rovers will incorporate a micro-controller unit (MCU) to run proportional, integral and derivative (PID) control to meet state spaces communicated by a 'companion computer' such as a raspberry pi. The rovers will incorporate edge-computing to run their ROS programs on the Raspberry pi, allowing for complex AI algorithms to run within close proximity to the micro-controller: i.e., allowing the MCU to act as the peripheral nervous system of the rover, and the raspberry pi as the brain of the rover. Edge-computing places AI, cloud computing, or data driven models closer to the agent for improved performance and native high level computation. Our approach, schedule, and planned system demonstrations are highlighted in the sections below.

A. Scope of Project and Design

The scope/goal of the project is to create a small group(our swarm) of semi-decentralized rovers that achieve a common goal. Our swarms will consist of an aim of 3+ rovers which although does not constitute an ideal amount for a swarm, however this will be enough for a proof of concept for the future that will use a larger group of individuals. As for the common goal, the team of robots should be able to work together to put on a little synchronized dance performance or to push a block from point A to another point in space. While the concept of these rovers being decentralized means that they are able to act independently with assistance from a central leader. This now leads into the question of how these autonomous rovers are going to be made and how they will be aware of their environment. There are multiple ways we can get the rovers to perceive their surroundings, in the case of the project there are three options. One option is to use a 2D lidar sensor. The lidar will give the robot an image of the objects currently around it and the distance from the unit itself. With the implementation of lidar it will open up the avenue to use SLAM(Simultaneous Localization and Mapping). For the actual build of the rovers the plan is to use a premade chassis for the outer shell of the robot. It will also use two stepper motors and a caster wheel to enable movement. As for powering/controlling the system, it will use a custom PCB power module for powering the different components with another custom PCB motor driver to control the stepper motors. As mentioned the chassis are premade, meaning the idea of making a rover swarm is not original.

II. Methods and Planning

A. Initial Timeline and Project Tasks

Here is the initial timeline for the project broken down into individual tasks:

- May: Assemble Rover/PCB
  - Main goal is to make the custom PCB for power module and motor drivers
  - Buy parts: Motor, Chassis, LiPo Battery, development boards, environment sensing parts
  - Test PCB and parts to make sure they are functional and understand how to interface with them
- June: Setup ROS/Code Framework
  - Use ROS to create initial code for the multi-agent server to assist Rovers
  - Also use ROS to manipulate and control motors to allow Rover to move
  - Begin interfacing with sensors and other components
- July: Setup ability to see environment
  - Setup Lidar/Computer Vision
  - Get Rovers to build map of their environment SLAM
  - Ensure rovers can move through environment without colliding
- August: Get Rover to move in consistent manner
  - Finish up main part of Rover framework
  - Get the Rovers to move in a set pattern
- September: Setup central server/Test Communication
  - Finish creating ROS multi-agen server
  - Get the server to communicate with Rovers
  - Test reaction time between server and Rovers
- October: Rover will work together on a task
  - Get Rover to push box
  - Get Rovers to dance around
  - Or get Rovers to complete another demo that is list later
- November/December: Debug and finish up Project
  - These are free months to use to buffer if the project falls behind or need extra time to debug

B. Project Resources

In order to finish the project on time we will need to deploy the use of external tools to assist in its completion. Luckily there are many tools at disposable to get the rovers on their wheels and working together. The first and biggest resource the project will be using is the Robot Operating System, ROS for short. ROS is a tool that is made to interface well with robotics, offering a multitude of tools to help simplify the process. These tools range from helping enable the rovers to use SLAM to even PID control for controlling the motors. Raspberry Pi and ESP32 can be flashed with versions of ROS. With the assistance of ROS implementing control/deploying the rovers will be simplified to focus on other aspects of the project. Although the use of ROS enables an easier deployment of the tools needed for the project, simulations are an integral

part of ensuring that the rovers operate as intended. To create simulations of the swarm, Gazebo will address this issue. Gazebo is 3D simulation software that integrates well with ROS. However, simulations of how the rover moves is only one part of the testing process.

III. Discussion

Testing will be done regularly from the first-day our components arrive to assure quality and functionality. As we assemble each component together, testing will help in debugging and localizing the issue in the system. Creating tests for each component helps with creating a good understanding on how that device individually works as well as how it interacts with other devices and software. On the topic of software, testing is of critical importance, without testing the software integration could lead to unforeseen behavior within the system as a whole. If our tests were done correctly, on demo day it will present our flawless creation, the freedom rover units to the world.

IV. Demo Description

Our goal is to demonstrate one of these three demos on demo day. The First demo shows each drone/rover coordinating with one another to align themself to make a letter. An example of this demo: from the server we give the task of creating the letter 'L', with 5 drones scattered on the map, each rover will represent a pixel and will move to that spot. If there were two rovers that are the same distance away from a spot, one rover will have higher priority over the other and the rover with the higher priority will get that spot. As for the other rover it will go to the next closest open spot. Another option for the First demo, is that each rover will have multiple different LED lights to make a light show on the ground.

The Second demo is to have the rovers coordinate with one another to move an object from one location to another. An example: 3 rovers scattered around the map. The map is 5' by 5', where (0,0) is located at the bottom left corner and (5,5) is located at the top right corner. Each rover get the task to move a cardboard box from (1,1) to (4,3). All 3 rovers will communicate and align themselves around the box, to move it to the said location. Once done, they will wait for their next instruction which may be to move it to a different location.

As for our last demo, the Third demo shows drones moving in a line around the map. For example: there are 2 tasks, the first task is to follow behind another drone and the second task is to move around the map. The second task will be given to one rover only and the first task will be given to all the other drones on the map. In this demo, there are 5 drones scattered throughout map and drone named #2 will get the second task. As for the other Drones: #1,#3,#4,#5, they will get the first task. Let's say the order of drone conga line is : #2,#3,#1,#5,#4. If we were to separate #1 from the conga line and place it elsewhere on the map, it will find its way behind the last rover in the congo line which in this case is #4. Thus conga line will look like: #2,#3,#5,#4,#1.

References

[1] R. C. Cardoso and A. Ferrando, "A review of agent-based programming for multi-agent systems," *Computers*, vol. 10, no. 2, p. 16, 2021.

[2] W. Chen, J. Liu, H. Guo, and N. Kato, "Toward robust and intelligent drone swarm: Challenges and future directions," *IEEE Network*, vol. 34, no. 4, pp. 278–283, 2020.

[3] R. Cooley, S. Wolf, and M. Borowczak, "Secure and decentralized swarm behavior with autonomous agents for smart cities," in *2018 IEEE International Smart Cities Conference (ISC2)*. IEEE, Conference Proceedings, pp. 1–8.

[4] D. S. Drew, "Multi-agent systems for search and rescue applications," *Current Robotics Reports*, vol. 2, pp. 189–200, 2021.

[5] S. Engebråten, K. Glette, and O. Yakimenko, "Field-testing of high-level decentralized controllers for a multi-function drone swarm," in *2018 IEEE 14th International Conference on Control and Automation (ICCA)*. IEEE, Conference Proceedings, pp. 379–386.

[6] M. S. Güzel, V. B. Ajabshir, P. Natttharith, E. C. Gezer, and S. Can, "A novel framework for multi-agent systems using a decentralized strategy," *Robotica*, vol. 37, no. 4, pp. 691–707, 2019.

[7] T.-K. Hu, F. Gama, T. Chen, Z. Wang, A. Ribeiro, and B. M. Sadler, "Vgai: End-to-end learning of vision-based decentralized controllers for robot swarms," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Conference Proceedings, pp. 4900–4904.

[8] S. Khaliq, S. Ahsan, and M. D. Nisar, "Multi-platform hardware in the loop (hil) simulation for decentralized swarm communication using ros and gazebo," in *2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, Conference Proceedings, pp. 310–315.

[9] A. Pandit, A. Njattuvetty, and A. K. Mulla, "Ros-based multi-agent systems control simulation testbed (mascot)," *arXiv preprint arXiv:2212.12657*, 2022.

[10] S. Qamar, S. H. Khan, M. A. Arshad, M. Qamar, and A. Khan, "Autonomous drone swarm navigation and multi-target tracking in 3d environments with dynamic obstacles," *arXiv preprint arXiv:2202.06253*, 2022.