Shreyas Gokhale

Dual Degree MSc Student @ TU Berlin & TU Eindhoven

Address: Theodor Heuss Platz-5, 14052, Berlin, Germany
Mobile: (+49) 15223611676
DoB: 6th September 1994
Email: shreyas6gokhale@gmail.com
LinkedIn: https://www.linkedin.com/in/shreyas6gokhale/
GitHub: https://github.com/shreyasgokhale
Portfolio: https://www.shreyasgokhaleresu.me/
Blog: https://shreyasgokhale.com/

JdeMultiBot: Multi-Robot exercises for Robotics Academy In ROS2

Abstract

JdeRobot robotics academy[1] currently offers an exercise for simulating amazon indoor warehouse scenario. The goal of this project is to create a similar scenario for a fleet of robots. This will involve task and path planning for multiple robots. The project also uses ROS2 and leverages its swarm robotics capabilities.

Project

Automated warehouses play an important role in modern Industry 4.0 based factories. In companies such as Amazon, multiple robot agents coordinate with each other to optimize delivery times. When a job such as “pickup from shelves” is scheduled, a task is assigned to one of these agents based on a number of factors, such as its proximity to the location or its path towards the goal. In many cases, the agents roam freely, communicate with each other, and have to avoid obstacles and humans in their path. The purpose of this project is to teach students these real-world interactions by creating a new exercise(s) in robotics academy.

Traditionally in ROS1, handling multiple robots is achieved using one master (roscore) server[2]. However, this approach requires the robot to be on one network, making decentralized approach only achievable using non-straightforward solutions (such as using port forwarding or rospackages such as Nimbro Network[3] & FKIE Multimaster[4]). ROS2 has improved on this and many other shortcomings of ROS1 by introducing the

Middleware interface[5]. This project aims to benefit from these features by migrating existing scenarios from ROS1 to ROS2.

Existing Work

Currently, JdeRobot robotics academy hosts one exercise demonstrating autonomous robot navigation and pick-and-place logic in the warehouse. The map of the warehouse is divided into zones: input, storage, output and charging. An agent is scheduled to pick up and deliver a pallet in another zone, representing a task. Using his algorithm, a student has to navigate the agent and carry out these operations.

Proposed Solution

JdeMultiBot will extend on this to implement a scalable cooperative multi-agent task and path planning system. Therefore, in addition to simple navigation and pickup, the student also has to coordinate between the agents (robots). This will include three components:

➔ **Task planning:** *Which agent handles which task*

Task planner is mainly based on network topologies. Based on how tasks are generated, a single master planner (centralized approach) or agent bounty based planner (decentralized approach) will be selected. This also involves various sub approaches. Centralized task planner can be one fixed master, or can have variations like leader election. Decentralized task planners have similar options like TDMA or polling.

➔ **Path planning:** *How agents plan the path, avoiding the obstacles*

There are numerous algorithms which deal with path finding. A*[6] is an optimal single robot path finding algorithm with a heuristic component, which is now used widely as a go-to path planner. Theta*[7] algorithm works by connecting discrete grid points on the map through a continuous line, not limited to graph edges. Another option is Rapidly Exploring Trees (RRT) [8] and RRT*[9]. Each of these algorithms also possesses different shortcomings with respect to scalability and speed.

➔ **Behaviour Planning:** *How agents can cooperate with each other*

Co-operations of the robots with each other can lead to interesting behaviour patterns and can lead to an increase in efficiency. This is specifically important for the simple exercise mentioned in the timeline. For example, robots can split themselves in such a way that they can explore and find targets quicker.[10] [11] [11]

The decision of which algorithms are decided on various factors such as: How challenging it will be for the students? Is demo implementation available as open-source (if yes, what licence)? Are ROS2 packages necessary for the development available? Etc.

Some other components to consider are charging management, collision detection and storage management.

Deliverables

The project is roughly divided into 4 components:

1. Porting of existing exercises from ROS 1 to ROS2. The porting will be done in a modular fashion so that it can be reused elsewhere.
2. Creating a multi-agent exercise which includes coordination between the robots for task and path planning. This includes different scenarios for experimenting with test solutions. How actual solutions will be formalised is explained in the timeline below.
3. Porting Amazon Robot exercise
4. Documenting, publishing on Jekyll pages.

Timeline

Week	Dates	Milestone	Main Tasks	Sub Tasks
0	4th May - 1st June	Community bonding Period	Experimenting with ROS2	Demo codes, reading porting guidelines, going through the requirements again
1	1st June - 7th June		Exploring current exercises, Discussing Reference solutions, Discussing porting parallels between ROS1 and 2	Finding equivalent alternative packages for ● Move_base (Nav2?) ● Global_planner ● Goal Sender cmd_vel mux ● Gazebo
2	8th June - 14th June		Porting amazon exercise from ROS1 -> ROS2	Using New packages, Porting existing:: ● Interfaces (move base client etc)

				• Sensors • GUI Improving Robot and world model
3	14th June - 1st June		Porting amazon exercise from ROS1 -> ROS2	Porting: • Algorithm • Solution
4	22nd June - 28th June		Porting amazon exercise from ROS1 -> ROS2	Tying up everything together, writing test cases, documentation. Publishing on the Jekyll page with wiki.
5	29th June - 5th July	Phase 1 Evaluation	Evaluation reports, Fixing structure of new exercise	Discussing Ideas for new multi-robot exercise: • Two robots play hide and seek to find a hidden target • Game of Pacman, where robots go around and try to collect randomly dropped virtual packages • Others?
6	6th July - 12th July		Creating New Exercise: Creating path and task planner options	Implementing different task and path planners which students choose and experiment on. This will be a simple scenario of pick and place exercise.
7	13th July - 19th July		Creating New Exercise: Wrapping up in an exercise	Packaging. Adding different maps, scenarios and models. Exploration algorithms
8	20th July - 26th July		Creating New Exercise: Testing + Docu	Tying up everything together, writing test cases, documentation. Publishing on the Jekyll page with a wiki.

9	27th July - 2nd Aug	Phase 2 Evaluation	Amazon Multi-robot exercise porting: discussion	Testing different multi-robot behaviour algorithms. + Writing Evaluations
10	3rd Aug - 9th Aug		Amazon Multi-robot exercise porting: Advanced task + path planner	Implementing: ● Planners for task scheduling ● Porting path following algorithms from previous exercises + adding new
11	11th Aug - 16th Aug		Amazon Multi-robot exercise porting: Other planners and creating challenge for students	Implementing: ● Charging behaviour options ● Creating a challenge exercise (eg: 10 Deliveries within 5 mins or score = max deliveries in 5 mins etc)
12	17th Aug - 23rd Aug		Amazon Multi robot exercise porting: Testing + Documentation	Tying up everything together, writing test cases, documentation. Publishing on the Jekyll page with a wiki.
13	24th Aug - 30th Aug	Final Evaluation	Final Evaluation + Leeway for delays	+
14+	1 Sept onwards		Patching bugs, responding to pull and feature requests, contributing whenever possible	

Challenges / Design Choices

➔ Unavailability of ROS 2 packages

ROS2, is aimed for multi-robots. But on the downside, not all the packages are properly ported to ROS2 and will need modifications if we decide to use them. More and more packages are being added to the latest release and, right now, Turtlebot3

library[10] is ported and a few common packages (for example Navigation2 and rviz) seem to work fine. But if not, then we have to work on custom replacements for the packages that we need. Such decisions can be taken as GSoC goes on.

➔ Complexity of the Simulation

Ideally, the goal is to have amazon warehouse simulation, complete with multi-robot collaboration, ready for students to experiment on. However, the project has a lot of small components as explained above. As GSoC is just 12 weeks, it might not be enough to do everything. The project will start with a simple simulation of multi-agents and the complexity of the simulation will be increased as time allows. Hence, it might happen that some to-dos will have to be done as future work.

Related Work

The project I had been working on for my robotics course had exactly similar conditions, but with ROS1, MORSE Simulator[12] and Blender stack. For our project, we are aiming for ROS2, Gazebo, Qt stack.[13]

AWS robomaker has some scenarios in AWS in ROS1 and ROS2 to practice online.[14]

Robot development studio also offers students to practice their ROS skills using their online academy.[15]

Biographical Information

Studies

I am pursuing a dual degree Master of Science in Embedded Systems with minor in Innovation and Entrepreneurship at Technical University of Eindhoven (TU/e) and Technische Universität Berlin (TUB).

I have finished all of my credits from both the universities and, since 1 year, I have been pursuing my master thesis: "*Decentralized, Multi-Robot, Collaborative Mapping and Exploration*" at Fraunhofer FOKUS, Berlin. This project shares some parallels with my master thesis, however, there is no intersecting code space. The thesis is already 50% completed and will require another 3 months to finish. As the deadline got extended until the mid of September due to the recent COVID-19 outbreak, I'll be able to allocate more than 30 hours of time on the GSoC project per week.

Programming Information:

I use Linux as my daily driver OS (custom Ubuntu Mate) with occasional dual-booted Windows for gaming; Visual Studio Code / Clion as IDE; Tilix, Fish and Powerline for the terminal.

I am well versed with C, Python, C++, Java, Bash, Lua and MarkDown. On the Systems side, I have worked on various projects involving MongoDB, MySQL, Docker, BLE, LoRa, WiFi, CUDA, OpenMP and FreeRTOS. Being embedded systems developer, I program on a variety of hardware from 8bit PIC - 32 bit ARM microcontrollers & SoCs and also x86 systems. I have ROS1 (Melodic) and ROS2 (Dashing) set up, along with ROS1 Kinetic Docker containers to test any old code.

Other Software:

Created a Smart (Euro) Trip planner using Python3, MultiThreading, Docker, MongoDB and Skyscanner APIs. Wrote WraPyMongo: Python PIP package for MongoDB API Wrapper. Have written various tech and travel articles on my Jekyll based blog. My other work and resume can be viewed on my portfolio website.

I have also worked on development of Java Agent based IIoT Middleware which connects low level RoS services and devices to high level cloud based applications such as Predictive Maintenance and Machine Learning.

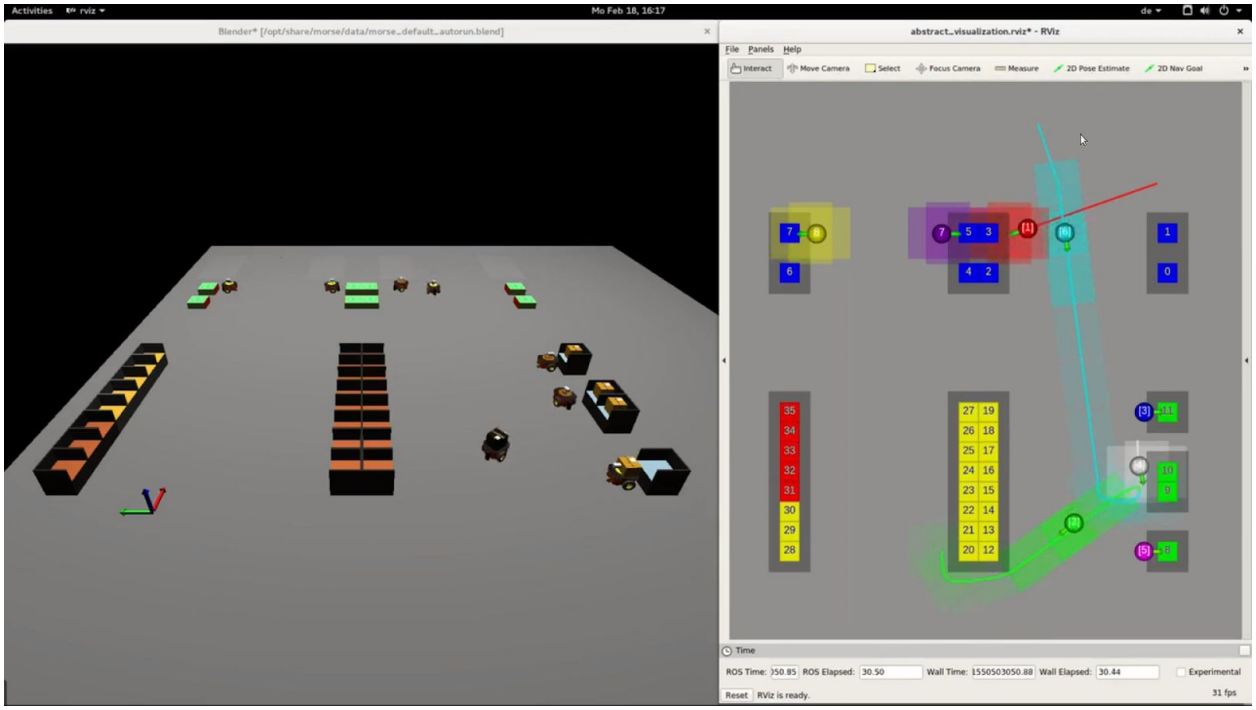
GSoC Experience:

I haven't participated in GSoC before and this is my first and only application ever.

Robotics Experience

I have ~ 1.5 years of experience in working on ROS1 and developing various packages and applications. I am quite familiar with indoor warehouse (amazon) problem and I have worked on the "Task and Path Planning" solution in a Free Roaming environment for the "*Application of Robotics*" course. We received a German grade 1 for the project and we are currently in the process of writing a conference paper based on our final report. Hence I cannot share the project repo, yet. However, you can check its presentation here, which explains the project in detail.

Screenshots of the project:



However, this project uses a completely different s/w stack for development and no components will be used for our project.

For the purpose of my master thesis, I have created an open-source, ROS1 based project which lets you simulate a large number of isolated robots on the cloud. This takes advantage of Docker and subsequently docker-compose / Kubernetes.

It works as follows:

1. A central gazebo simulation docker container starts the world, deploys robot models and generates topics for sensors (eg /scan).
2. A robot is an isolated container, which runs its own isolated ROS environment. This includes perception stack, navigation and other packages. N Robots = N containers.
3. The connection between simulator and robot containers is done by the Nimbro network package. Hence it is completely controllable and we can also simulate loss and throttling.
4. The whole deployment is done using a docker-compose file and several bash scripts and runs all the GUI as well.

The project can easily be used to run large scale simulation, complex machine learning projects and teach robotics to students remotely. The project is available on Github with How-Tos and Screenshots.

For Computer vision, I participated in a lab course for Sensor Fusion of Camera, LiDAR and RADAR for parking lot detection in autonomous vehicles.

Bibliography

[1] "JdeRobot," *JdeRobot*. http://jderobot.github.io/ (accessed Mar. 26, 2020).
[2] "ROS/Tutorials/MultipleMachines - ROS Wiki." http://wiki.ros.org/ROS/Tutorials/MultipleMachines (accessed Mar. 26, 2020).
[3] *AIS-Bonn/nimbro_network*. AIS Bonn, 2020.
[4] *fkie/multimaster_fkie*. Fraunhofer FKIE, 2020.
[5] "ROS 2 middleware interface." http://design.ros2.org/articles/ros_middleware_interface.html (accessed Mar. 26, 2020).
[6] "astarNilsson.pdf." Accessed: Mar. 26, 2020. [Online]. Available: https://www.cs.auckland.ac.nz/courses/compsci709s2c/resources/Mike.d/astarNilsson.pdf.
[7] K. Daniel, A. Nash, S. Koenig, and A. Felner, "Theta*: Any-Angle Path Planning on Grids," *J. Artif. Intell. Res.*, vol. 39, pp. 533–579, Oct. 2010, doi: 10.1613/jair.2994.
[8] "Lav98c.pdf." Accessed: Mar. 26, 2020. [Online]. Available: http://msl.cs.illinois.edu/~lavalle/papers/Lav98c.pdf.
[9] V. R. Desaraju and J. P. How, "Decentralized path planning for multi-agent teams in complex environments using rapidly-exploring random trees," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 4956–4961, doi: 10.1109/ICRA.2011.5980392.
[10] "Coordination strategies for multi-robot exploration and mapping - Carlos Nieto-Granda, John G. Rogers, Henrik I. Christensen, 2014." https://journals.sagepub.com/doi/abs/10.1177/0278364913515309 (accessed Mar. 28, 2020).
[11] "explore_multirobot - ROS Wiki." http://wiki.ros.org/explore_multirobot (accessed Mar. 28, 2020).
[12] "What is MORSE? — The MORSE Simulator Documentation." https://www.openrobots.org/morse/doc/stable/what_is_morse.html (accessed Mar. 26, 2020).
[13] S. Gokhale, *shreyasgokhale/Multi-Robot-Decentralized-Architecture*. 2020.
[14] *aws-robotics/aws-robomaker-sample-application-cloudwatch*. AWS Robotics, 2020.
[15] https://www.theconstructsim.com/.