ROBOTICS ACADEMY: MIGRATION TO GAZEBO FORTRESS

JdeRobot
GSoC 2024 Proposal

Submitted by :

Prajyot Jadhav

Robotics-Academy: Migration to Gazebo Fortress

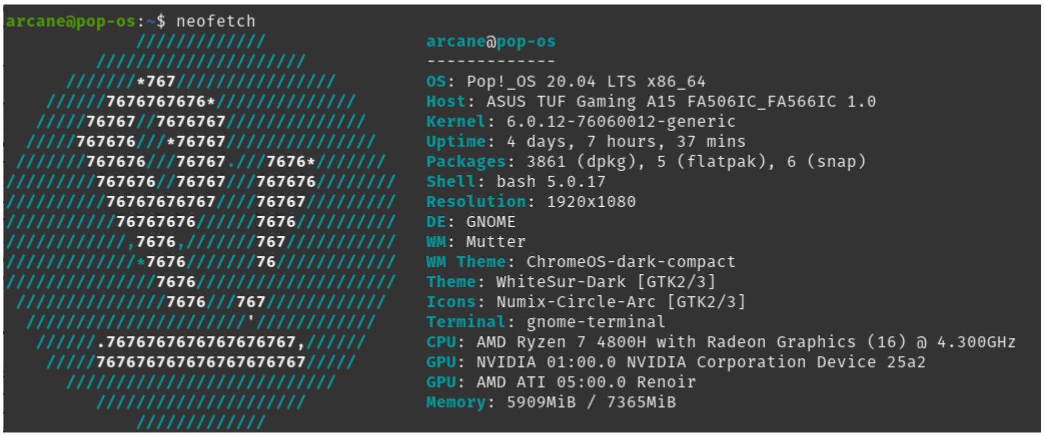
Contact Details

I am a final year undergraduate at Visvesvaraya National Institute of Technology (VNIT), Nagpur, India, pursuing my bachelor's degree in Electronics and Communication Engineering.

Name	Prajyot Jadhav
Country	India
Primary Email	prajyotj04@gmail.com
Student Email	bt20ece046@students.vnit.ac.in
Resume	Resume
GitHub	Arcane-01
Linkedin	Prajyot Jadhav

Programming Background

My working setup includes an ASUS TUF A15 with a Ryzen 7 4800H(8-core/16-thread) processor, 8 GB of DDR4-3200MHz RAM, and a 4GB Nvidia RTX 3050 graphics card. I use Pop!_OS 20.04 (Linux distribution, based on Ubuntu) as my operating system.



```
arcane@pop-os:~$ neofetch
             /////////////                 arcane@pop-os
         /////////////////////             -------------
      ///////*767////////////////          OS: Pop!_OS 20.04 LTS x86_64
    //////7676767676*//////////////        Host: ASUS TUF Gaming A15 FA506IC_FA566IC 1.0
   /////76767//7676767//////////////       Kernel: 6.0.12-76060012-generic
  /////767676///*76767///////////////      Uptime: 4 days, 7 hours, 37 mins
 ///////767676///76767.///7676*///////     Packages: 3861 (dpkg), 5 (flatpak), 6 (snap)
/////////767676//76767///767676////////    Shell: bash 5.0.17
//////////76767676767////76767/////////    Resolution: 1920x1080
///////////76767676//////7676//////////    DE: GNOME
////////////,7676,///////767///////////    WM: Mutter
/////////////*7676///////76////////////    WM Theme: ChromeOS-dark-compact
///////////////7676////////////////////    Theme: WhiteSur-Dark [GTK2/3]
 ///////////////7676///767////////////     Icons: Numix-Circle-Arc [GTK2/3]
  //////////////////////'////////////      Terminal: gnome-terminal
   //////.76767676767676767676767,//////   CPU: AMD Ryzen 7 4800H with Radeon Graphics (16) @ 4.300GHz
    /////767676767676767676767/////        GPU: NVIDIA 01:00.0 NVIDIA Corporation Device 25a2
      ///////////////////////////          GPU: AMD ATI 05:00.0 Renoir
         /////////////////////             Memory: 5909MiB / 7365MiB
             /////////////
```

I've been using ROS1 Noetic Ninjemys and ROS2 Foxy Fitzroy on my system, and they both operate smoothly when sourced separately. I am proficient in Python and am also familiar with C++. For the hardware deployment of my projects, I've worked with microcontrollers like ESP32, development boards including ODROID-XU4 and Raspberry Pi, as well as flight controllers like Cube Orange and Cube Black.

I also have experience working with PX4-Autopilot and the MAVROS package, which will help me understand the codebase used in drone-based exercises.

Experience

1. Undergraduate Research Intern, BRAIN LAB, VISTEC
   Duration: January - July 2023
   - Worked under Prof. Poramate Manoonpong in the domain of aerial robotics.
   - Developed a customized control algorithm that effectively maintains the stability of a quadrotor on a curved surface, safeguarding it against potential falls even in the presence of external disturbances.
   - Following successful results in simulation, validated and tested the developed controller on physical platforms to assess its performance in real-world environments.
   - **Software stack:** Python, ROS, RViz, Gazebo-Classic, PX4-Autopilot, QGroundControl
   - **Hardware stack:** ODROID-XU4, CubePilot Cube Orange, OptiTrack
   - **NOTE:** The code or results cannot be disclosed as the work has not yet been published.
2. Undergraduate Student Researcher, IvLabs, VNIT
   Duration: July 2021 - Present
   - Worked under Prof. Shital S. Chiddarwar in the domain of aerial robotics and deep learning.
   - Implemented a cascaded PD controller to control the motion of the quadrotor in the 2D and 3D environments for following the desired path while minimising position error.
   - Implemented a paper titled " Taut Cable Control of a Tethered UAV " to study the dynamics and control of a 2D quadcopter tethered by a cable. Deployed a custom control architecture for stable flight of the quadcopter under taut cable constraint in simulation.
   - Implemented a denoising autoencoder from scratch using PyTorch on MNIST dataset.
   - **Non-Technical:** Conducted IEEE workshops on Basic Electronics and Python.

- **Software stack:** Python, PyTorch, MATLAB, SIMULINK, OpenCV
- **Code:** Image Denoising Autoencoder, Robust Quadcopter Control, 2D Taut Cable Control Of Tethered UAV

Open Source Contributions to JdeRobot

Pull Requests	
Title	**Status**
Updated the Instructions for Developers	Closed
Fixed AttributeError for init_beacons in Position Control	Merged
Issues	
Title	**Status**
Error on following the Installation for Developers(using Docker Compose)	Closed
AttributeError encountered when attempting to initialize beacons in the Position Control exercise	Closed
Empty world launched for the Visual Lander exercise	Closed
Few exercises not available for ROS Noetic (RADI version 3.4.28)	Closed
Discussions	
Title	**Status**
Need help for running the RoboticsAcademy exercises locally	Closed
Need help for accessing the Follow Turtlebot exercise	Closed

Statement of Intent

- **Why JdeRobot?**
  I have a strong passion for Robotics and Software Development. I have been working in the field of robotics since my sophomore year of undergraduate, majorly in control of various robotics systems. I came across JdeRobot while exploring about PX4-Autopilot through the blogs of previous GSoC contributors. A significant challenge for aspiring roboticists is the tedious setup process for related software and simulators. However, the dockerized containers and web templates provided by Robotics Academy offer cross-platform functionality, eliminating the need for multiple setups and making it easier for learners to get started. This approach allows robotics beginners to focus on core concepts by directly testing their algorithms without being hindered by setup complexities. I've personally used Robotics Academy for some exercises to bypass the setup process. I am highly motivated to contribute to this project and believe that Google Summer of Code 2024 presents an ideal opportunity.

Project Details

Robotics-Academy is a framework for learning robotics and computer vision. The exercises come ready to use in the RoboticsAcademy docker image (RADI). Currently, the RADI is based in Gazebo11 version. The main goal of the project is to migrate the RADI to Gazebo Fortress. This migration will involve running a couple of exercises in ROS2 with the updated RADI. Another goal of this project is to replace PX4 with Aerostack2 Gazebo platform for drone-based exercises. The current branch of the RADI 4.4.x (ROS Version = Humble) includes the following exercises: Basic Vaccum Cleaner, Rescue People, Autoparking, Amazon Warehouse, Follow Person, Follow line, Localised vaccum cleaner, Montecarlo Laser Loc. Also, there is another migration in progress related to the exercise on-run management(Migrate all exercises to new 4-step RAM and superthin templates).

gazebo_ros_pkgs is the package that provided launch files, plugins, and other utilities for using Gazebo classic with ROS 2. The equivalent for the new Gazebo is ros_gz, a meta-package that contains a few packages (eg. - ros_gz_bridge; which provides a network bridge that enables the exchange of messages between ROS 2 and Gazebo Transport, ros_gz_sim which provides launch files and other utilities that help with starting Gazebo and spawning models).

The changes that need to be made are:

- **Updating dependencies**
  - Replacing references to gazebo, gazebo_ros_pkgs, etc with packages from ros_gz
  - Updating the package.xml and CMakeLists.txt files accordingly.
- **Editing files that start Gazebo or spawn models (RAM launchers)**
  - Replacing gazebo_ros with ros_gz_sim for launching the simulator.
  - Adapting launch files to use the new ros_gz packages to spawn models correctly.
  - Updating parameters and configurations as required.
- **Editing the SDFormat files**
  - Updating the world SDFormat file to use plugins from ros_gz instead of gazebo_ros.
  - Adjusting plugins, parameters, and configurations in model files to be compatible with the new Gazebo setup.
  - Replacing plugin names and adjusting plugin parameters for sensors, actuators, and other functionalities.

- **Bridging ROS topics**
  - Setting up topic bridges between ROS and gz-transport topics using ros_gz_bridge.
  - Configuring the bridge node with appropriate YAML files to map ROS topics to Gazebo topics and vice versa.

Expected Results from these changes:

- Successful launch of the simulation world without errors. This includes running Gazebo and spawning the robot model, which would function according to its configuration.
- The bridge nodes will enable communication between ROS 2 topics and Gazebo topics which would allow to control the robot in the simulation using ROS 2 messages and receive sensor data from the simulation.
- Visualization of the simulation of the robot and its environment in the Gazebo GUI.

References: Migrating ROS 2 packages that use Gazebo Classic, Migration from Gazebo classic: SDF, ROS 2 Integration

Timeline

- **Before May 27th**
  - Community bonding
  - Refining and finalizing potential project timeline.
  - Getting well acquainted with running, building and debugging Docker Images.
  - Setting up the GSoC Blog.
- **Phase 1 (May 27th - June 5th)**
  - Going through the RoboticsAcademy, RoboticsInfrastructure and Robotics Application Manager repositories.
  - Understanding how RAM works and the process involved in creating the RADI.
  - Developing a standardized approach with fundamental steps for migrating exercises to Gazebo Fortress.
- **Phase 2 (June 5th - June 30th)**
  - Migrating the Rescue People exercise to Gazebo Fortress.

  - Replacing PX4 with the lighter Aerostack2 Gazebo platform.
  - Testing the migration and investigating/resolving any issues.
  - Creating comprehensive documentation outlining the migration process.

  Expected Results: Gazebo Fortress on docker and new launcher in RAM for the Rescue People exercise.

- **Phase 3 (June 30th - July 31st)**
  - Migrating the Autoparking and Follow Person exercises to Gazebo Fortress.
  - Testing the migration and investigating/resolving any issues.
  - Creating comprehensive documentation outlining the migration process.

  Expected Results: Gazebo Fortress on docker and new launcher in RAM for the Autoparking and Follow Person exercises.

- **Phase 4 (August 1st - August 15th)**
  - Resolving any pending issues.
  - Running and debugging the migrated exercises.

Final Expected Results: Gazebo Fortress on docker and new launcher in RAM for the three exercises: Rescue People, Autoparking and Follow Person.

Acknowledging the Commitment

I fully understand that participating in Google Summer of Code is a significant commitment, equivalent to a full-time paid summer internship or job. I am prepared to dedicate the necessary time and effort to successfully complete the project. I do not have any time conflicts until 15^{th} August that would interfere with my participation in the official coding period. I will have a job orientation scheduled during the second half of August. I am fully committed to completing the project within the allocated timeframe and will manage my time effectively.

GSoC participation

This is my first time applying for the Google Summer of Code program, and I won't be submitting another proposal to a different organization for GSoC 2024.

Post GSoC

I would like to keep contributing to Robotics Academy's ongoing development and improvement even after the GSoC coding period ends. I really like the collaborative spirit of the open-source community and the valuable learning experiences it offers. I am looking forward to continue contributing to JdeRobot and giving back to the community.