# Food Delivery with Nav2 in Custom Gazebo World

## Project Overview

This project implements a food delivery robot system using custom bot in a custom Gazebo environment. The robot navigates autonomously using the Nav2 stack and handles real-world delivery tasks such as:  
- Delivering to multiple tables  
- Receiving confirmation at delivery points  
- Canceling deliveries interactively  
- Returning to predefined locations like Kitchen or Home

## System Architecture

### 1. Gazebo Simulation Setup

The simulation is launched using a custom Gazebo world (my\_world.world).  
  
Launch file: world.launch.py   
Responsibilities:  
- Launch gzserver and gzclient  
- Spawn the custom robot  
- Set initial position (x\_pose, y\_pose)  
- Launch robot\_state\_publisher

### 2. Navigation Setup (Nav2)

Launch file: nav.launch.py   
Initializes the Nav2 stack using:  
- Predefined map YAML file  
- Parameter file based on model  
- RViz with default configuration  
  
Enables:  
- Global/local planning  
- Recovery behaviors  
- SLAM or localization

### 3. Delivery Task Handling – robot1.py

Defines a ROS2 Node named restaurant\_robot using BasicNavigator from nav2\_simple\_commander.  
  
Features:  
- Navigate to HOME, KITCHEN, TABLE1, TABLE2, TABLE3  
- Multi-table support  
- Confirmation with timeout  
- Live feedback (distance/progress)  
- Cancellations (cl, c1, c2, c3)  
- Return logic

### Default Locations

HOME: {x: -1.0, y: 0.0, theta: 0.0}  
KITCHEN: {x: 1.0, y: -0.5, theta: 0.0}  
TABLE1: {x: 4.0, y: 1.3, theta: 0.0}  
TABLE2: {x: 4.0, y: 0.0, theta: 0.0}  
TABLE3: {x: 4.0, y: -1.3, theta: 0.0}

## Launch Instructions

1. Launch Gazebo World:  
ros2 launch pranesh\_description world.launch.py  
  
2. Launch Nav2:  
ros2 launch pranesh\_description nav.launch.py use\_sim\_time:=true  
  
3. Run Delivery Script:  
ros2 run pranesh\_description robot1.py

## Robot Commands

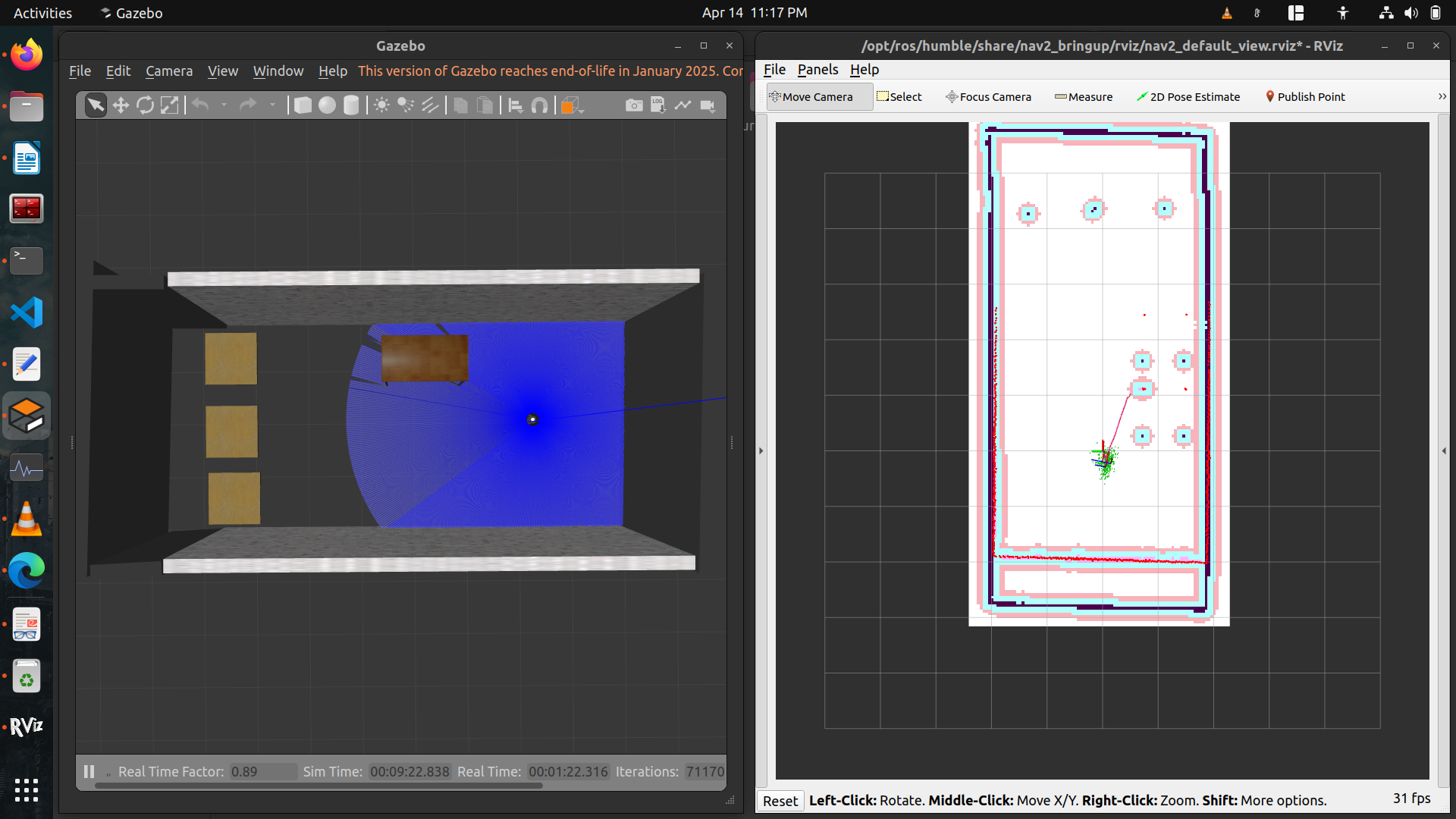
|  |  |
| --- | --- |
| Command | Description |
| t1 | Deliver to Table 1 |
| 1,2,3 or t1,t2,t3 | Multiple table delivery |
| c | Confirm delivery |
| cl | Cancel and go home |
| c1, c2, c3 | Cancel a specific table |
| home | Force return to HOME |
| status | Check current robot status |
| help | Display available commands |
| q | Quit the program |

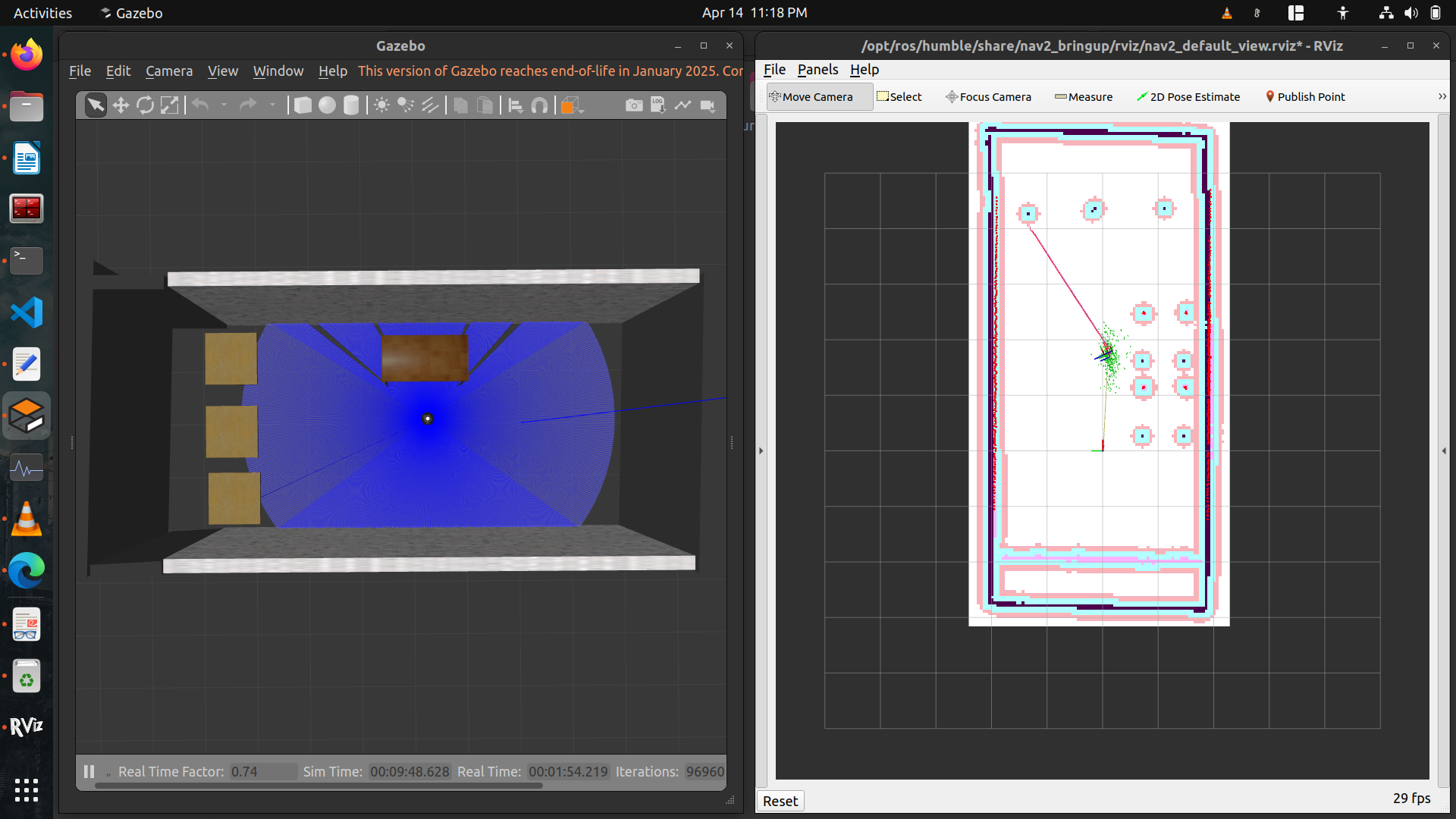
## Flow Diagram (Simplified)

START  
 └──> Spawn in Custom Gazebo World  
 └──> Nav2 Initializes (Map + Params)  
 └──> robot1.py script starts  
 └──> Accept user command (t1, t2, ...)  
 ├──> Navigate to Kitchen  
 ├──> Confirm Pickup  
 ├──> Deliver to tables (with per-table checks)  
 └──> Return to Kitchen/Home

## Key Concepts Used

- ROS 2 Nodes in Python for modular navigation and delivery logic  
- Nav2 stack for path planning and obstacle avoidance  
- PoseStamped goals for movement  
- Threading for background initialization  
- Interactive CLI for user control  
- Custom world and Custom robot design in Gazebo





## Conclusion

This assignment demonstrates a robust navigation and task-execution system using Custom robot and Nav2. The robot simulates real-world café delivery behavior using ROS2 and Gazebo.

## Drive Link:

<https://drive.google.com/file/d/1FeIHFN5liGNqi4ey2wTAjyfsXWB0OKkN/view?usp=sharing>

**Contact**:

https://github.com/pr0nesh

www.linkedin.com/in/pranesh-s-183a99202

spraneshsekar@gmail.com

9345672745