InternPro Weekly Progress Update

Name	Email	Project Name	NDA/ Non-NDA	InternPro Start Date	OPT
Adharsh Prasad Natesan	anatesan@asu.edu	IT-Core Foundation Suriname	Non-NDA	2024-08-05	Yes

Progress

Include an itemized list of the tasks you completed this week.

#	Action Item/ Explanation	Total Time This Week (hours)
1	Farmland Model for ROS Gazebo	3
2	Farm land Running in Gazebo	3
3	Getting a Turtlebot Running in an Empty Gazebo World	3
4	Autonomous Navigation with ROS for a Mobile Robot in Agricultural Fields	3
5	Autonomous Navigation of a Center-Articulated and Hydrostatic Transmission Rover	3
6	GNSS Compass in Autonomous Agriculture - Advanced Navigation	3
7	Next week plan	1
8	Report writing	1
	Total hours for the week:	20

Verification Documentation:

Action Item 1: Farmland Model for ROS Gazebo – 3 hour(s).

Project Work Summary

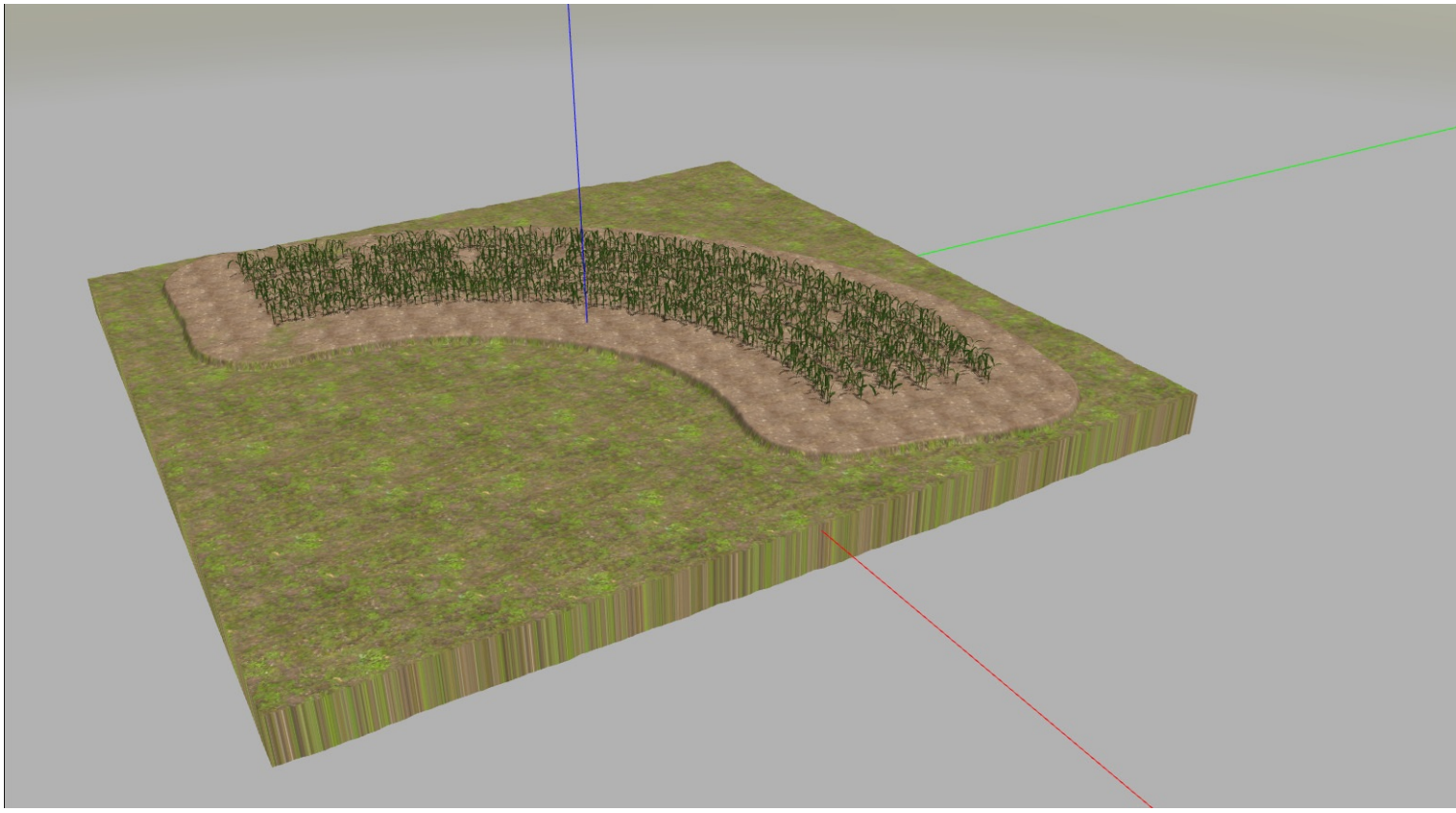
- Started by finding around for some farmland simulation options that wouldn't be a total pain to set up. First stop was checking out what Clearpath Robotics had - they've got this Agricultural World thing that works with their rover bots like Jackal and Husky.
- Then found the Virtual Maize Field package on GitHub that looked way more suited for the application. It's basically this procedural generator that spits out random corn fields with actual rows of plants - perfect for testing rovers in farm settings without the hassle of building everything from scratch.
- I went ahead and cloned their repo with:
- git clone https://github.com/FieldRobotEvent/virtual_maize_field
  Then had to install all the dependencies using rosdep:
  rosdep install virtual_maize_field
- The whole process was pretty straightforward - just had to wait for the download and dependency installation to finish. The cool thing about this package is it lets you generate different field layouts on the fly, so you're not stuck with just one boring farm setup. Way better than messing around with 3D model conversions and texture issues

```
asgard@asgard-HP-Pavilion-Gaming-Laptop-15-dk0xxx:~$ git clone https://github.com/FieldRobotEvent/virtual_maize_field.git
Cloning into 'virtual_maize_field'...
remote: Enumerating objects: 2073, done.
remote: Counting objects: 100% (342/342), done.
remote: Compressing objects: 100% (122/122), done.
remote: Total 2073 (delta 282), reused 220 (delta 220), pack-reused 1731 (from 3)
Receiving objects: 100% (2073/2073), 55.00 MiB | 14.00 MiB/s, done.
Resolving deltas: 100% (1308/1308), done.
```

Action Item 2: Farm land Running in Gazebo – 3 hour(s).

Project Work Summary

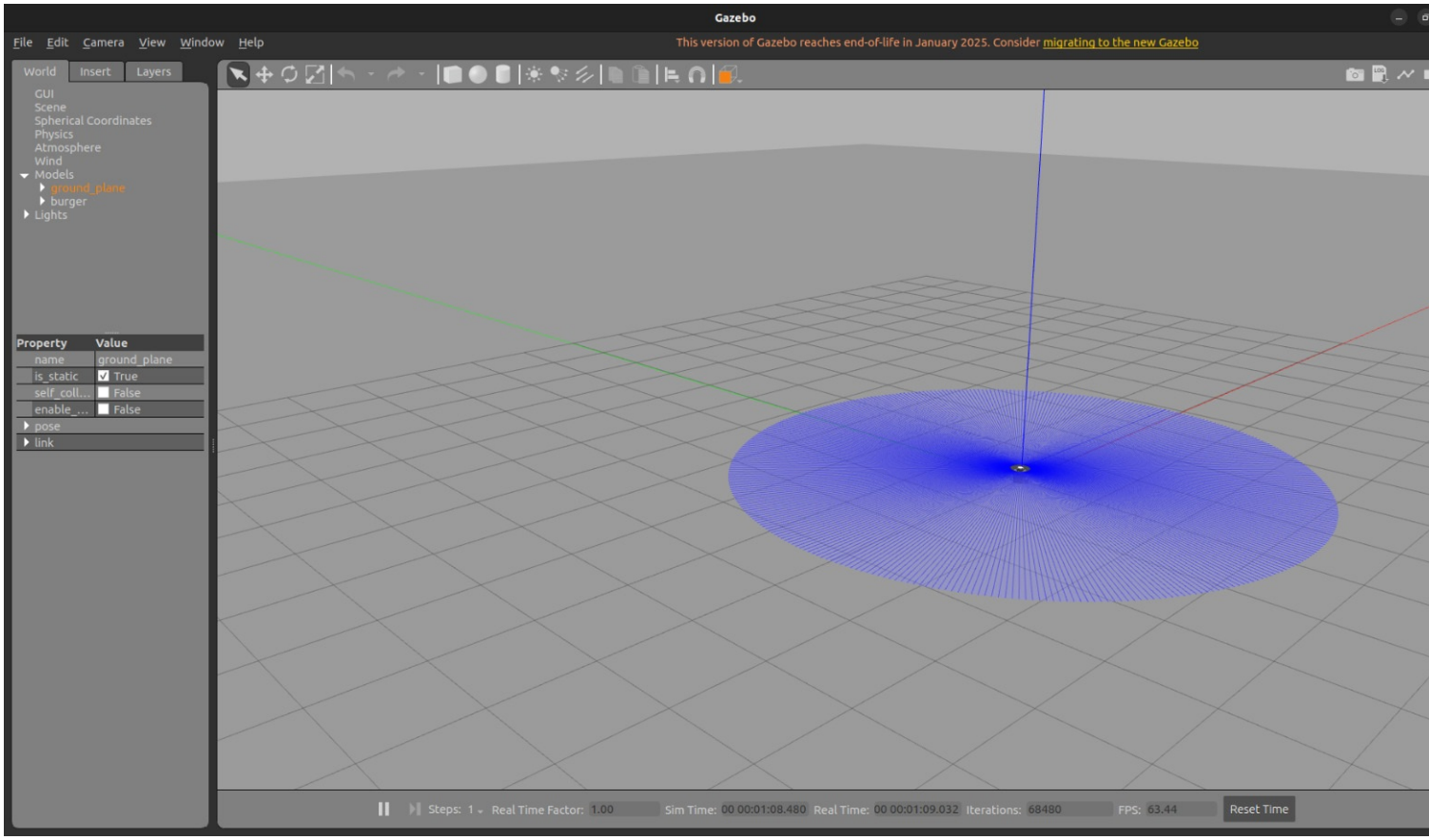
- After getting all the files downloaded and dependencies sorted, it was time to actually build the thing and see it in action. First, I had to build the package with colcon:
- bash
  cd ~/ros2_ws # or wherever your workspace is
  colcon build --packages-select virtual_maize_field
  source install/setup.bash
- The above code builds the necessary packages. Once that was done, I generated a new random field layout:
  bash
  ros2 run virtual_maize_field generate_world
  This command churned away for a bit and then dumped a world file in my ~/.ros/virtual_maize_field/ directory. Pretty neat to see it create a whole farm layout from scratch!
  Finally, the moment of truth - launching Gazebo with the new farm:
  bash
- ros2 launch virtual_maize_field simulation.launch.py
- Gazebo opens up with the map there was my corn field with nice neat rows of plants! The simulation ran pretty smoothly on my machine. I played around with the camera angles to get a good view of the whole field layout. You can actually see the individual corn plants and everything - way more detailed than I expected.

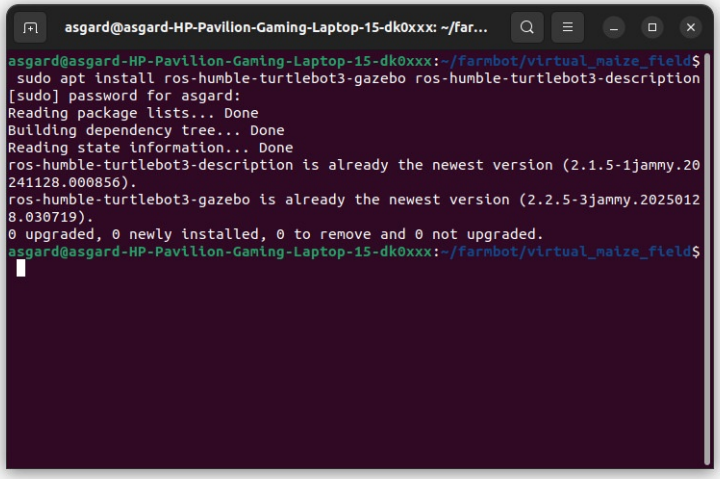


Action Item 3: Getting a Turtlebot Running in an Empty Gazebo World – 3 hour(s).

Project Work Summary

- First thing we need to do is to make sure we can add the Turtlebot packages installed on my system. I double-checked with:
- apt list --installed | grep turtlebot3
  Looked like I was missing a few, so I went ahead and installed the essentials:
- sudo apt install ros-humble-turtlebot3-gazebo ros-humble-turtlebot3-teleop
- After waiting for the installation to finish, I set up the environment variable that tells ROS which Turtlebot model to use. I went with the Burger since it's the simplest one:
  export TURTLEBOT3_MODEL=burger
  With that done, I launched the empty world simulation that comes with the Turtlebot packages:
  ros2 launch turtlebot3_gazebo empty_world.launch.py
- Gazebo opens the map and we could add Turtlebot in the empty gray world. The robot loaded perfectly - I could see all its sensors and components rendered in the simulation. The physics looked good too, with the robot sitting properly on the ground plane.
- To make sure everything was working correctly, I opened up another terminal window, set the model variable again, and launched the teleop package:
  export TURTLEBOT3_MODEL=burger
  ros2 run turtlebot3_teleop teleop_keyboard
- The terminal shows the keyboard controls, and I tested driving the robot around with the WASD keys. We could control the linear and angular velocity, and the robot moved around the empty world without any issues.
- The whole process was pretty straightforward once I had the right packages installed. This empty world setup gives me a good baseline to test basic functionality before moving on to more complex environments like that corn field we were working with earlier.





Action Item 4: Autonomous Navigation with ROS for a Mobile Robot in Agricultural Fields – 3 hour(s).

Research

- https://strathprints.strath.ac.uk/61247/1/Post_etal_ICINCO_2017_Autonomous_navigation_with_ROS_for_a_mobile_robot_in_agricultural_fields.pdf
- Autonomous Navigation with ROS for a Mobile Robot in Agricultural Field
- Summary of Report
  - They built a low-cost autonomous ground rover for farm monitoring using off-the-shelf hardware and ROS software to make it affordable for farmers
  - The team created custom ROS nodes specifically for agricultural navigation, including a simplified path planner for open farm fields
  - Their results showed the rover could successfully reach navigation points with high precision when using accurate real-time localization
- Relation to Project
  - Uses ROS architecture similar to what we're implementing for our state estimation project
  - Addresses the specific challenges of navigating in agricultural environments with limited features
  - Demonstrates sensor fusion techniques combining visual odometry, laser-based odometry, and wheel odometry
- Motivation for Research
  - Farming is becoming increasingly automated, but expensive systems are out of reach for medium scale farming
  - Agricultural environments present unique challenges for navigation (uniform terrain, changing conditions)
  - There's a need for affordable autonomous systems that can handle the uncertainty of outdoor farm environments

Action Item 5: Autonomous Navigation of a Center-Articulated and Hydrostatic Transmission Rover – 3 hour(s).

Research

- https://pmc.ncbi.nlm.nih.gov/articles/PMC7472308/ink to Article
- Autonomous Navigation of a Center-Articulated and Hydrostatic Transmission Rover
- Summary of Report
  - They developed an algorithm to control an autonomous, multi-purpose, center-articulated hydrostatic transmission rover for navigating crop rows
  - Testing involved obtaining GNSS waypoints manually and then having the rover autonomously follow those paths
  - The rover achieved impressive accuracy with mean absolute errors of 0.04m, 0.06m, and 0.09m across three passes, with slightly higher errors (0.24m) during end-of-row turns
- Relation to Project
  - Uses RTK-GNSS for high-precision navigation, which could be incorporated into our state estimation
  - Demonstrates practical implementation in actual crop rows, similar to our corn field simulation
  - Shows that even with some navigation error, the system can still perform effectively for agricultural tasks
- Motivation for Research
  - Cotton harvesting and other agricultural operations require precise navigation to avoid damaging crops
  - Autonomous systems can reduce labor costs and increase efficiency in agricultural operations
  - Testing in real-world conditions provides valuable insights for practical implementation

Action Item 6: GNSS Compass in Autonomous Agriculture - Advanced Navigation – 3 hour(s).

Research

- https://www.advancednavigation.com/case-studies/gnss-compass-keeps-the-naio-technologies-ted-agricultural-robot-accurately-tending-vineyards/ink to Article
- GNSS Compass in Autonomous Agriculture - Advanced Navigation
- Summary of Report
  - Naïo Technologies integrated a GNSS Compass into their Ted agricultural robot for vineyard operations
  - The system achieved heading accuracy to approximately 1° and made turning smoother and more stable
  - A key advantage was the ability to obtain accurate heading even when the robot was stationary, which was crucial for end-of-row turns in vineyards
- Relation to Project
  - Demonstrates the importance of accurate heading information for agricultural robots, which will be crucial for our state estimation
  - Shows how GNSS can be used effectively even in challenging environments like vineyards
  - Highlights the importance of safety systems for autonomous agricultural robots
- Motivation for Research
  - Vineyard operations require extremely precise navigation due to narrow rows and valuable crops
  - Autonomous robots need to operate unsupervised, requiring highly reliable navigation systems
  - Different soil textures can cause sliding issues that need to be addressed by robust state estimation

Action Item 7: Next week plan – 1 hour(s).

Project Work Summary

- Setup and Configuration
  - Install and configure the TurtleBot3 packages on your system
  - Set up the ROS environment variables (TURTLEBOT3_MODEL)
  - Install the robot_localization package for sensor fusion
  - Configure the GNSS sensor component for your simulation environment
- Mapping and Navigation Implementation
  - Generate a map of the corn field environment using SLAM techniques
  - Configure the Navigation2 stack for TurtleBot3
  - Set up the navsat_transform_node to transform GPS data into a frame consistent with the robot's world frame
  - Implement initial pose estimation using GNSS data
- Sensor Fusion and Communication
  - Create ROS nodes for publishing and subscribing to GNSS data
  - Implement a dual-instance approach for robot_localization:
    - One instance for continuous data (odometry, IMU)
    - Second instance that includes GNSS data
    - Set up topic-based communication between nodes for sensor data sharing
    - Configure the local and global costmaps for navigation in the agricultural environment

Action Item 8: Report writing – 1 hour(s).

Project Work Summary

- Created word document layout to write contents of the weekly progress.
- Created relevant subsections in the epicspro website and documented 20 hours of weekly progress.
- Collected relevant documents research papers, relevant links and company's objective from their portal.