Technical Report: Evaluation and Selection of Simulators for the SkyVision Multi-Drone Project

Ahmed, Belal, Abdllah, Ammar, Yousef
Al Alamein International University, New Alamein City, Egypt

December 8, 2025

Abstract

This technical report documents our evaluation process and findings regarding several open-source simulators and middleware for the SkyVision multi-drone project. We systematically explored Gazebo, AirSim, and Isaac Sim as our simulators, and PX4, QGroundControl, and ROS2 as middleware, comparing their features, integration complexity, sensor and physics realism, and suitability for collaborative mapping and control tasks specific to SkyVision. The report details our hands-on experiments, highlights the strengths and limitations of each platform, and explains our decision to adopt AirSim with ROS2 as the foundation for SkyVision's simulation and development work. Our experience aims to guide other drone researchers and students in selecting appropriate simulation tools for aerial robotics projects.

1 Introduction

SkyVision is a specialized graduation project in collaboration with JMU, focused on developing an autonomous swarm of Unmanned Aerial Vehicles (UAVs) for rapid disaster response. The system leverages Artificial Intelligence to coordinate multiple drones for search-and-rescue operations, utilizing real-time Computer Vision (CV) and Simultaneous Localization and Mapping (SLAM).

This report documents the selection, configuration, and evaluation of the simulation environment used to develop and test SkyVision before physical deployment.

2 Project Requirements

To successfully simulate the SkyVision swarm, the simulation platform needed to meet specific criteria:

- **Multi-Vehicle Support:** Capability to spawn and control multiple drones simultaneously (swarm functionality).
- **ROS2 Integration:** Native compatibility with ROS2 (Humble/Jazzy) for communication between agents. We are currently using Humble for better compatibility.
- **Sensor Simulation:** High-fidelity simulation of RGB cameras, LiDAR, and IMUs for Computer Vision and SLAM tasks.
- **Physics Fidelity:** Accurate flight dynamics compatible with the PX4 Autopilot stack.
- **Customizability:** Ability to modify drone models (e.g., Holybro X500) and environments (disaster scenarios).
- **Visualization:** Real-time visualization of drone states, sensor data, and mapping results.
- **AI and Machine Learning:** Support for advanced AI workflows, including:
  - Computer Vision (CV) for perception and mapping
  - Reinforcement Learning (RL) for autonomous navigation and control
  - Federated Learning for distributed model training across multiple drones
  - Large Language Model (LLM) integration for advanced decision-making or mission planning
- **Middleware Flexibility:** Ability to integrate with a wide range of middleware, including ROS2, PX4, MAVSDK, QGroundControl, and potentially DDS or other robotics frameworks.

3 Simulator and Middleware Options

We evaluated three primary candidates for the SkyVision simulators:

- **AirSim:** Built on Unreal Engine by Microsoft, offering photorealistic rendering and APIs for controlling drones.
  Pros: Extremely high visual fidelity (great for CV).
  Cons: Deprecated/archived by Microsoft; heavy resource usage; difficult setup with modern ROS2 versions. Some community-maintained forks are compatible with newer versions of Unreal and ROS2.
- **Gazebo (Harmonic):** The open-source standard for robotics simulation, previously known as Ignition. Default simulator for ROS2.

Pros: Native ROS2 integration; lightweight; official support from PX4 Autopilot; large community.
Cons: Visuals are less realistic than Unreal Engine (though Harmonic is significantly improved).

- **NVIDIA Isaac Sim:** Scalable robotics simulation based on NVIDIA Omniverse.
  Pros: Photorealistic physics and visuals; GPU-accelerated.
  Cons: Requires high-end NVIDIA hardware; steeper learning curve for custom drone integration compared to Gazebo.

4 Middleware

For middleware and autopilot, we considered:

- **PX4:** Open-source autopilot stack for drones, widely supported in simulation and real hardware.
- **QGroundControl:** Ground station software for monitoring and controlling PX4-based drones.
- **ROS2:** Modern robotics middleware for distributed communication, sensor integration, and control.
- **MAVSDK:** Modern SDK for MAVLink-based drone control, supporting both Python and C++ APIs. Used for programmatic mission control and telemetry.
- **DDS:** Data Distribution Service, a middleware standard for scalable, real-time, and high-performance data exchange, used in some advanced robotics systems.
- **Other Options:** We also explored other middleware and APIs as needed for specific AI or distributed learning tasks.

5 Evaluation Process

We compared the simulators and middleware based on the following metrics:

- **Integration Ease:** How easily does it connect with PX4, ROS2, and MAVSDK?
- **Swarm Scalability:** Can it run 3+ drones without crashing the host PC?
- **Sensor Data:** Ease of accessing camera/LiDAR streams in Python/OpenCV, and compatibility with AI pipelines.
- **AI Capabilities:** Support for computer vision, reinforcement learning, and custom neural network integration. Ability to run perception and control algorithms efficiently.
- **Computation Cost:** Resource requirements for running advanced AI tasks (e.g., RL, CV) and multi-agent scenarios.
- **Long-term Viability:** Is the software actively maintained?
- **Community Support:** Availability of documentation, forums, and troubleshooting resources.

6 Installation Guide

For detailed installation and setup instructions, please refer to the project README and the provided PDF installation guide. These documents cover all steps for setting up the simulation environment, middleware, and dependencies on Ubuntu 22.04 and ROS2 Humble.

extbfGitHub Fork: The latest code and updates for our AirSim-based SkyVision platform are available at:
`https://github.com/ahmedislamfarouk/AirSim`

7 Final Choice and Rationale

After extensive testing and evaluation, we ultimately chose **AirSim** as our main simulator for SkyVision. The key reasons for this decision are:

- **Best Visuals:** AirSim, built on Unreal Engine, provides photorealistic rendering, which is crucial for developing and testing advanced computer vision and AI perception algorithms.
- **AI and RL Support:** AirSim natively supports reinforcement learning workflows and integration with popular AI frameworks, making it ideal for training and evaluating autonomous behaviors.
- **Efficient Computation:** While Isaac Sim offers similar AI capabilities, it is much more resource-intensive. AirSim achieves a good balance between performance and realism, making it more accessible for our hardware.
- **Better Than Gazebo for Vision:** Gazebo Harmonic is excellent for physics and ROS2 integration, but its visuals and AI support lag behind AirSim, especially for vision-based tasks.
- **GazeboDrone Integration:** AirSim also supports the GazeboDrone plugin, allowing us to leverage Gazebo's physics and multi-drone features within the AirSim environment.
- **Community Forks:** We COULD USE a community-maintained fork that is much newer and more compatible.

8 Lambda Server Experiments

To scale our experiments and test cloud-based simulation, we attempted to launch both Isaac Sim and AirSim on a Lambda Labs SSH server. This allowed us to:

- Run heavy AI workloads remotely, including reinforcement learning and multi-agent training.
- Benchmark simulator performance on high-end GPUs and CPUs.
- Test remote visualization and control via SSH and web streaming.
- Identify deployment challenges for cloud-based robotics research.

some success was made but AirSim and Isaac Sim require more specialized hardware and configuration. We are still testing the streaming capabilities to see if this is feasible or not.

9 Project Work Summary

This section summarizes the major work completed so far, organized by simulator and platform:

9.1 Gazebo

- Set up Gazebo Harmonic with PX4 and ROS2 integration for multi-drone simulation.
- Developed and tested custom SDF models for the Holybro X500 frame, including simulated sensors (LiDAR, RGB camera).
- Implemented SITL bridging and swarm launching scripts for multiple drones.
- Used RViz2 and QGroundControl for real-time monitoring and control.

9.2 Isaac Sim on Lambda Server

- Deployed Isaac Sim on the on-campus Lambda Labs to leverage high-end GPU resources.

9.3 AirSim (Main Platform)

- Adopted AirSim as the primary simulator for its superior visuals, AI/vision support, and efficient computation.
- Currently working on integrating RTAB-Map SLAM with AirSim and ROS2 for real-time multi-drone mapping and localization. This includes collaborative point cloud generation and visualization in RViz2.
- Integrated AirSim with ROS2, PX4, and MAVSDK for full-stack multi-drone control.
- Developed Python scripts for multi-drone control, path planning, and SLAM integration.
- Implemented and validated LiDAR-based localization and collaborative mapping, enabling multiple drones to build maps together in real time.
- Successfully simulated collaborative mapping missions with 5 drones, using LiDAR for localization and map building.
- Validated sensor data streams (LiDAR, camera) for real-time computer vision and SLAM tasks.
- Added and tested ROS2 topics for sensor data, localization, and control, facilitating seamless integration with our AI and SLAM pipelines.
- Integrated YOLO for real-time object detection and disaster marker identification (snow, rain, fire, etc.).
- Used RViz2 extensively for visualization of drone states, sensor data, and mapping results in AirSim.
- Initiated work on federated learning for distributed model training across drones; this will be detailed in a separate report.
- Planns to use LLMs for advanced mission planning and decision-making.
- Plans to leverage the GazeboDrone plugin for enhanced physics and multi-drone support.
- Documented all workflows and troubleshooting for reproducibility.

10 Demo and Visualization

We prepared several demo scenarios to showcase the capabilities of our AirSim-based system. Below are screenshots from our experiments, each illustrating a key aspect of our multi-drone mapping and SLAM workflow:

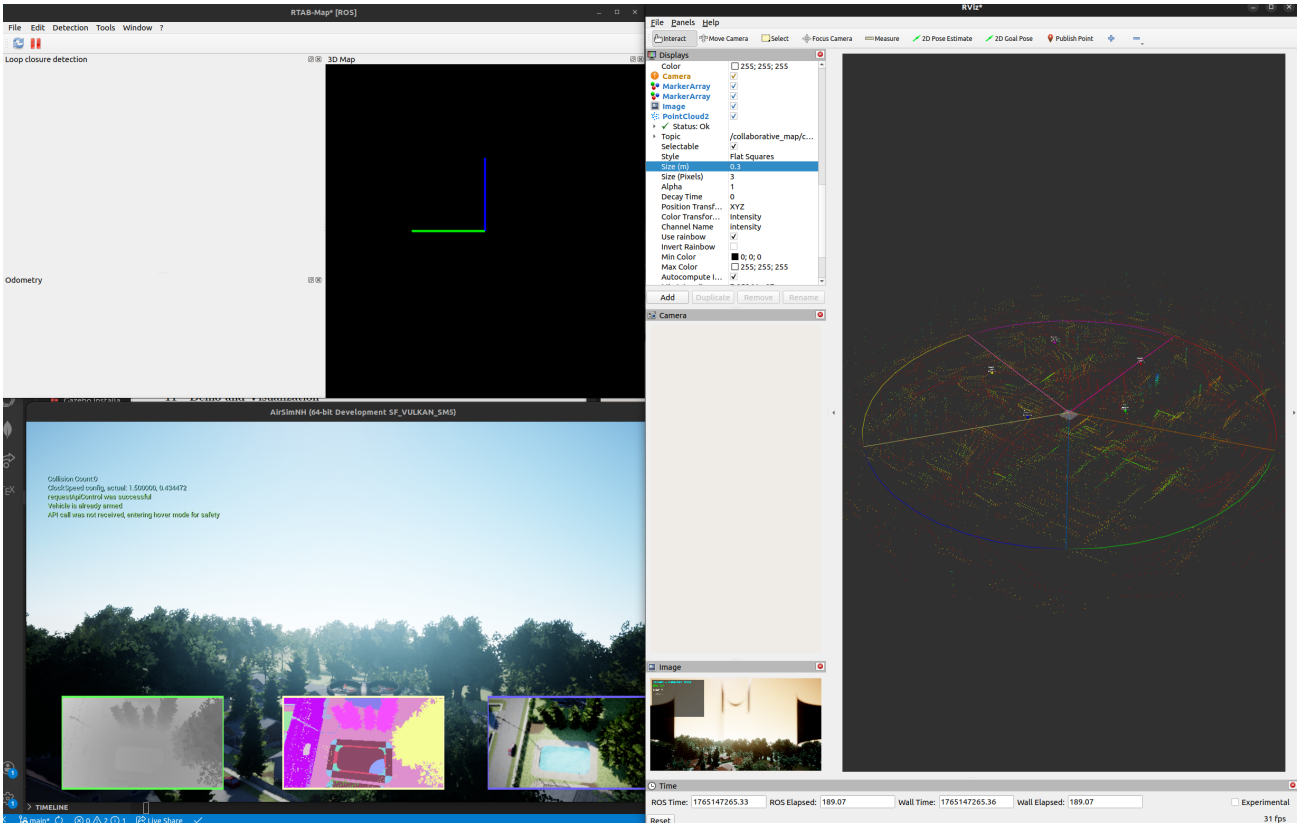

Figure 1: *

RTAB-Map and RViz2 Integration: This screenshot shows the RTAB-Map GUI (top left) for loop closure detection and odometry, alongside RViz2 (right) visualizing the collaborative point cloud map generated by multiple drones. The bottom panel displays AirSim running the simulated environment with real-time sensor feeds.

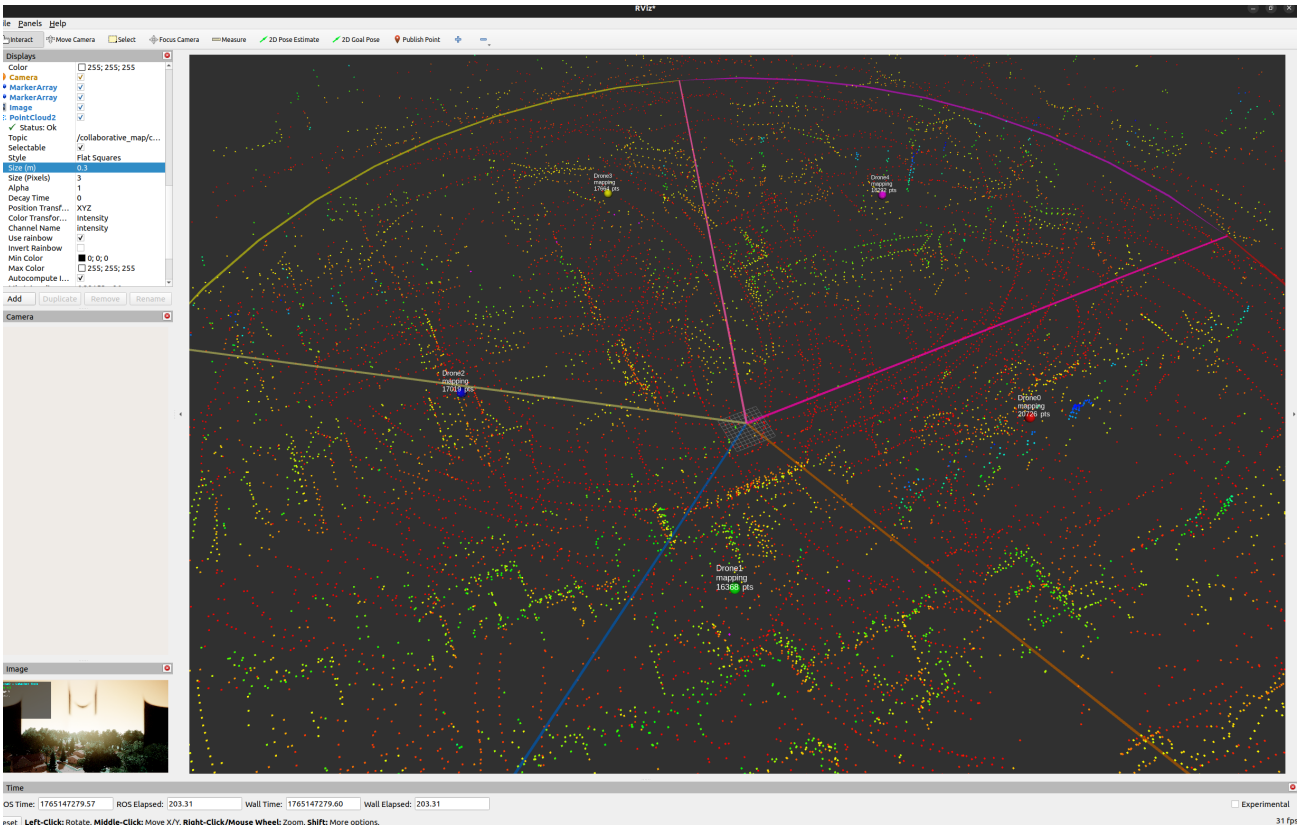

Figure 2: *

Collaborative Mapping in RViz2: Here, RViz2 displays the colored point cloud map from several drones, with each drone's mapping progress and position labeled. This demonstrates the effectiveness of our multi-agent SLAM pipeline.

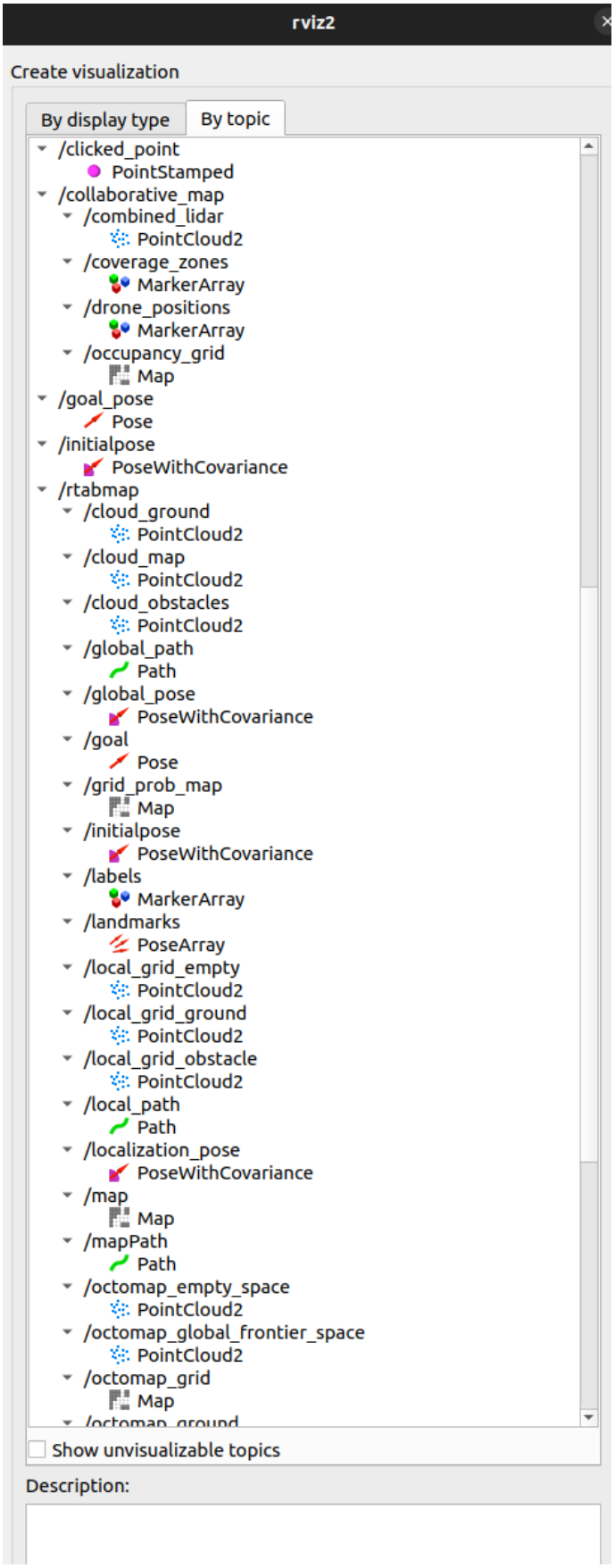

Figure 3: *

RViz2 Topic Visualization: The RViz2 topic list shows the various data streams available, including collaborative maps, drone positions, coverage zones, and SLAM outputs from RTAB-Map.

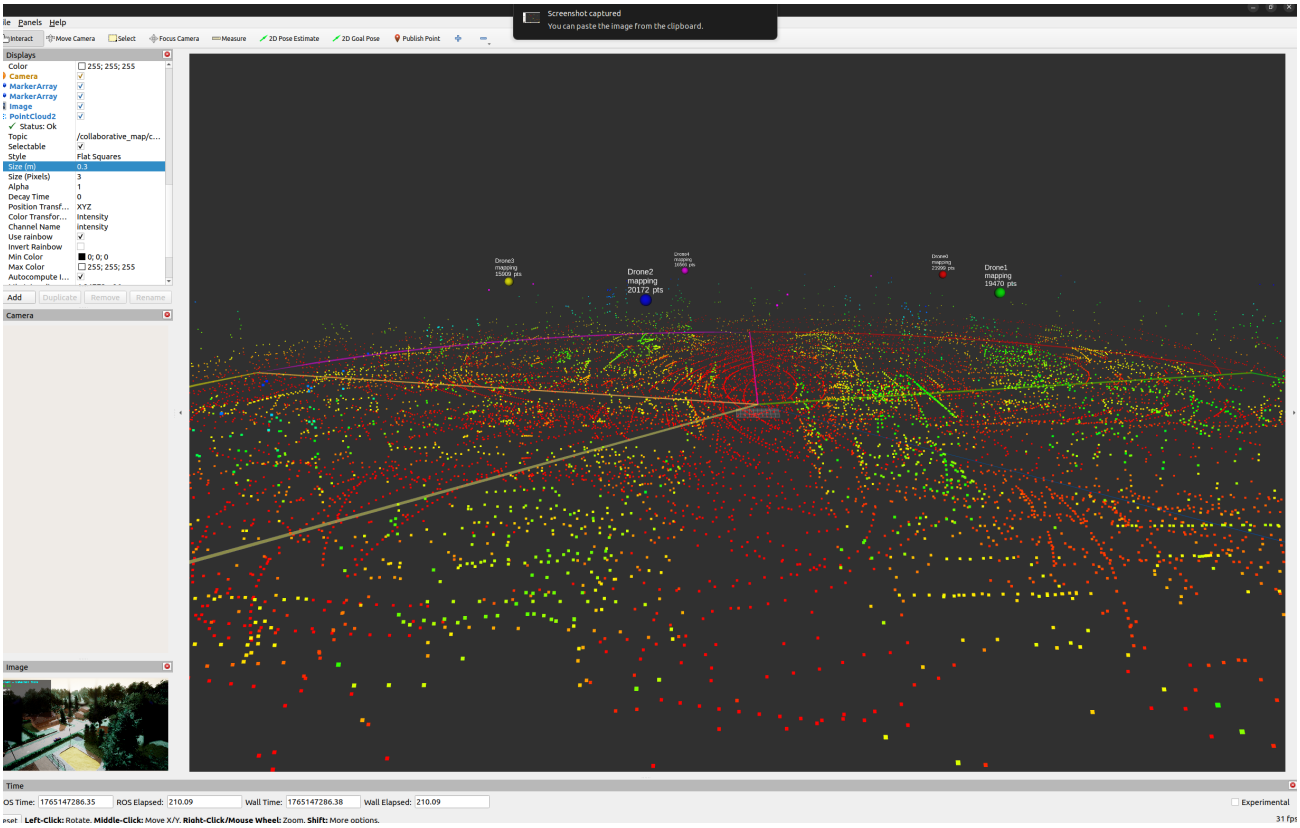

Figure 4: *

Multi-Drone Point Cloud Mapping: A wide-angle view in RViz2 of the point cloud generated by several drones, highlighting the spatial coverage and mapping density achieved in our experiments.

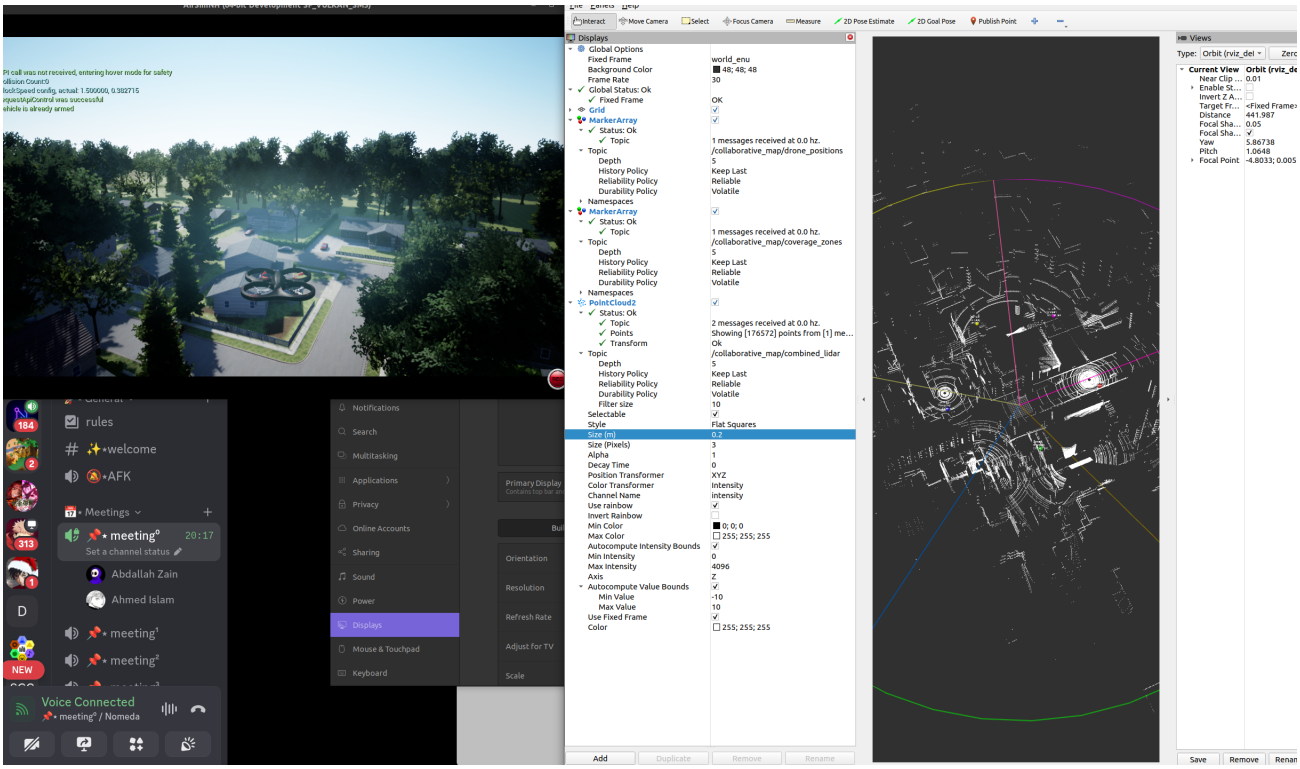

Figure 5: *

AirSim and RViz2 in Action: This screenshot shows AirSim simulating a drone swarm in a photorealistic environment (left), while RViz2 (right) visualizes the combined LiDAR data and drone positions.

11 Conclusion

This report summarizes our evaluation and selection of simulation tools for the SkyVision multi-drone project. By choosing Gazebo Harmonic, PX4, and ROS2, we established a robust, scalable, and well-supported simulation environment for developing autonomous drone swarms. Our experience and documentation aim to assist future teams in making informed decisions for aerial robotics research.

Acknowledgments

We thank Al Alamein International University and JMU for their support and collaboration, and the open-source communities behind Gazebo, PX4, ROS2, and AirSim for their invaluable resources.