RL in Drones for Operation in Fire Scenarios

Paola Rojas Domínguez
Skyspec
Oct – Dic 2025

Introduction

- Wildfires create dynamic, unstable, and dangerous environments.
- Using drones in fire scenarios requires autonomy and intelligent navigation.
- The project's objective is to build the theoretical and technical foundations for training drones using Reinforcement Learning (RL) in simulation.
- The project aims to establish a foundation for training drones using Reinforcement Learning (RL).
- Initial work includes technical exploration, simulation, and sensor integration.
- The first phase of the project focuses on studying advanced simulation tools: Omniverse, Isaac Sim, and Isaac Lab.

- Company specializing in environmental data analysis and hazardous substance detection.
- Interested in drones capable of operating in complex environments.
- The project contributes to future autonomous monitoring tools.

Theoretical Foundations

NVIDIA Omniverse

NVIDIA Omniverse is a physics-based simulation and collaboration platform for designing, testing, and validating autonomous systems. Its architecture is built on the Universal Scene Description (USD) standard, developed by Pixar, which allows for the representation of complex 3D scenes with high interoperability and modularity.

Key elements:

- USD as a unified language for describing robots, sensors, environments, materials, and animations.
- RTX and PhysX for high-fidelity physics simulation (collisions, rigid dynamics, realistic lighting).
- Modular extensions that allow for connecting multiple design tools (Blender, Maya, Unreal).
- Multi-platform interoperability, useful in projects where the drone goes through design in Blender, testing in Unity, and physics simulation in IsaacSim.

Relevance to the project.

For drone training using real-world simulation, Omniverse is the foundation that guarantees simulations with sufficient physical realism, an essential condition for achieving sim-to-real transferability. Its USD-based structure facilitates the integration of sensors, animations, complex movements, and modeling of damaged or smoky environments, relevant to fire scenarios.

Isaac Sim

Isaac Sim is NVIDIA's robotic simulation engine built on Omniverse. It allows for the physically accurate modeling of robots, sensors, cameras, and environments, integrating natively with ROS 2, computer vision pipelines, and AI systems.

Main features

- Robust physics simulation for aerial and ground robots.
- Advanced sensors: Lidar, IMU, RGB/Depth cameras, thermal cameras.
- ROS 2 connectivity for publishing point clouds, IMU, and odometry data.
- Generation of realistic environments: warehouses, factories, and uneven terrain.
- Direct import of USD models and custom extensions.

Advantages for a drone project

- It allows for the simulation of aerial dynamics without the risk of loss or physical damage.
- It facilitates experimentation with fire scenarios using fog, smoke, or reduced lighting.
- It acts as a bridge between design (Blender) and real-world simulation (Isaac Lab).

Limitations encountered

- Dependence on omni modules in certain versions.
- Variations between versions 2023–2025 that affect import compatibility and Isaac Lab actions.
- Not all USD drone models have sufficient joints for direct motor control.

Isaac Lab

Isaac Lab es un framework de entrenamiento para robots basado en aprendizaje por refuerzo (RL). Proporciona un sistema estandarizado para crear entornos, definir recompensas, configurar observaciones y entrenar políticas con algoritmos modernos.

Componentes principales

- **Tasks / Environments:** definición del comportamiento del robot en cada episodio.
- **MDP (Markov Decision Process):** manejo de estados, acciones, recompensas y terminación de episodios.
- **Gestión de políticas RL:** integración con SKRL, RSL-RL, RL Games.
- **Batch de simulaciones:** entrenamiento paralelo para acelerar el aprendizaje.

Relevancia para drones
Isaac Lab permite entrenar políticas para tareas como:

- Hover (estabilización en Z).
- Control de actitud (pitch, roll, yaw).
- Navegación hacia coordenadas (x,y,z).
- Evitación de obstáculos (con lidar).
- Navegación en laberintos y entornos dinámicos.

Desafíos encontrados en el proyecto:

- Varias clases y APIs cambiaron entre versiones, generando incompatibilidades (ej.: ActionTermCfg, ArticulationCfg).
- Falta de soporte para ciertos modos de acción (BodyForce, JointVelocity) en la versión instalada.
- Complejidad para adaptar un dron realista a un esquema de articulación controlable por RL.

Fundamentals of Reinforcement Learning

Reinforcement Learning (RL) is a paradigm where an agent learns to act in an environment through trial and error, optimizing a policy based on reward signals. It is particularly well-suited for motor control, nonlinear dynamics, and real-time decision-making.

Fundamental Components

- State(s): Information describing the environment (e.g., orientation, altitude, drone speed).
- Action(a): Commands executed (e.g., motor power).
- Policy(π): Function that determines the optimal action.
- Reward(r): Reinforcement that guides learning.
- Value and Advantage: Measures for evaluating future performance.

Modern Methods

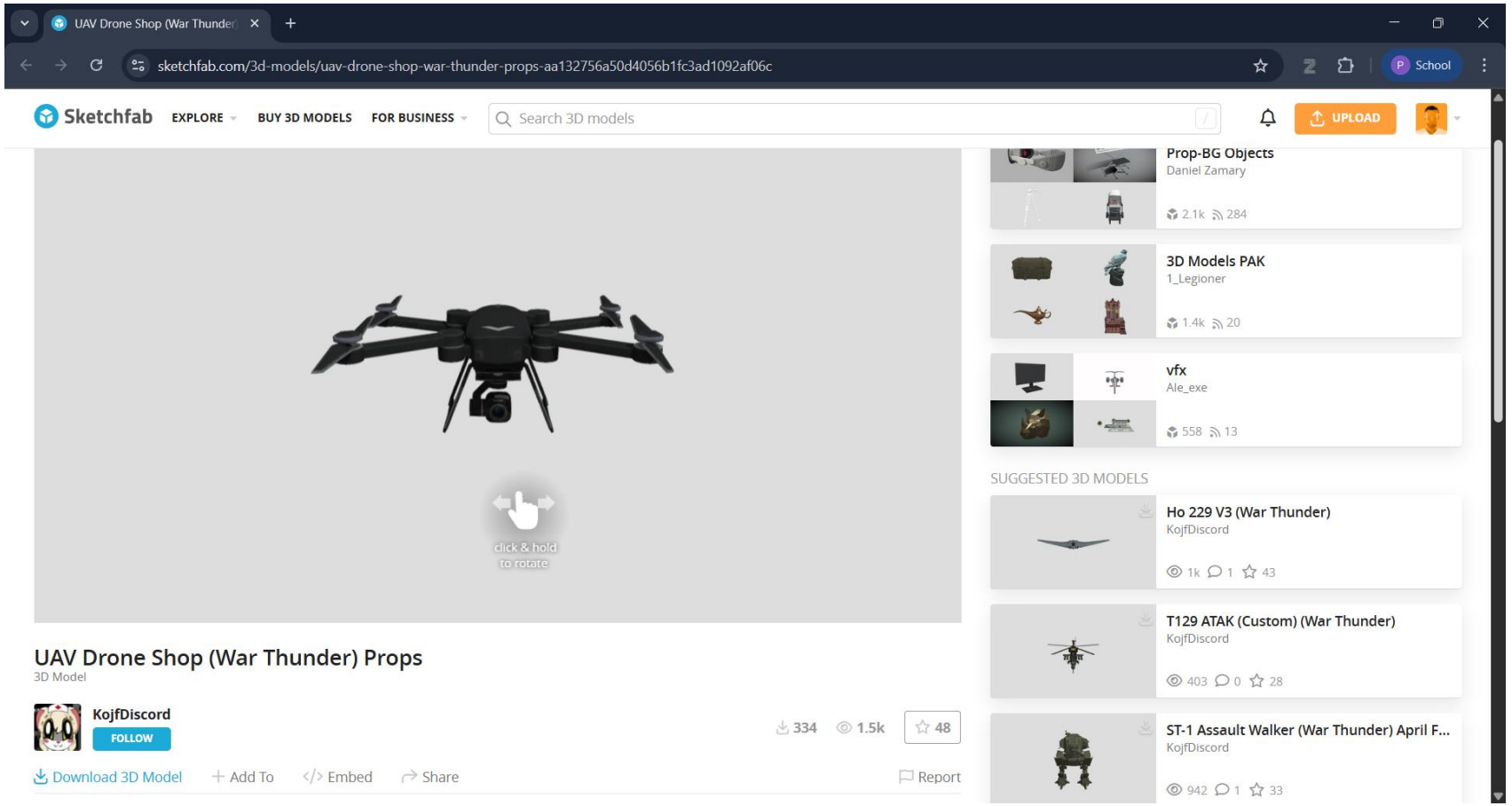
- Actor-Critical (PPO, SAC): suitable for robots with continuous dynamics.
- Hierarchical Learning (HRL): breaks down complex tasks into manageable subtasks, useful for drones in firefighting.
- Neuroscience-Inspired Learning (Kenji Doya):
  - Dopamine ≈ positive reinforcement
  - Serotonin ≈ risk control
  - Norepinephrine ≈ rapid reaction
  - Acetylcholine ≈ contextual learning

Relevance of drones for firefighting

Drones must:

- Stabilize in turbulent environments.
- Make decisions with partial information.
- Adapt to changing geometry and smoke.
- Learn optimal routes without rigid pre-programming.

Project Development



Sketchfab

Sketchfab: an online platform for sharing and downloading interactive 3D models.

Phase 1 — Modeling and animation in Blender

- Drone modeling and animation in Blender: objectives and deliverables.
- 3D asset preparation: mesh cleanup, pivots, and hierarchy. Keyframe animation: takeoff and test flight.
- Exporting to USD for Omniverse: compatibility and considerations.
- Evidence: video “3D Drone Model – Blender” (YouTube).

Phase 2 — Simulation in Unity

- Prototyping in Unity: basic manual control (WASD + spacebar). Integration of the .obj/.fbx asset and Rigidbody configuration.
- Control script in C#: applied forces and propeller rotations.
- Objectives: validate response to inputs and adjust physical parameters.
- Evidence: video “3D Drone Model - Unity” (YouTube).

Phase 3 — Sensors in Unity

- Implementation of a simulated LiDAR in Unity.
- Objectives: point cloud, obstacle detection, acquisition rate.
- Results: point cloud generation and mapping tests.
- Limitations: physical fidelity and realistic noise.
- Evidence: video “Lidar Sensor - Unity” (YouTube).

Phase 4 — Sensors in Isaac Sim

- Integration of LiDAR and IMU in Isaac Sim.
- Publishing PointCloud2 and IMU to ROS2.
- Objective: To create a realistic sensor flow for RL.
- Evidence: “Lidar Sensor - Isaac Sim” video (YouTube) and ROS logs.

Phase 5 — Construction of the project at Isaac Lab

- Drone project structure in Isaac Lab (tree summary).
- Key components: drone_env_cfg.py, mdp/rewards.py, skrl_ppo_cfg.yaml, train.py, play.py.
- Status: Partially operational environment; initial testing pending.
- Immediate goal: To train the controller to move the drone to coordinates (x,y) in the first instance.

Goals Achievement

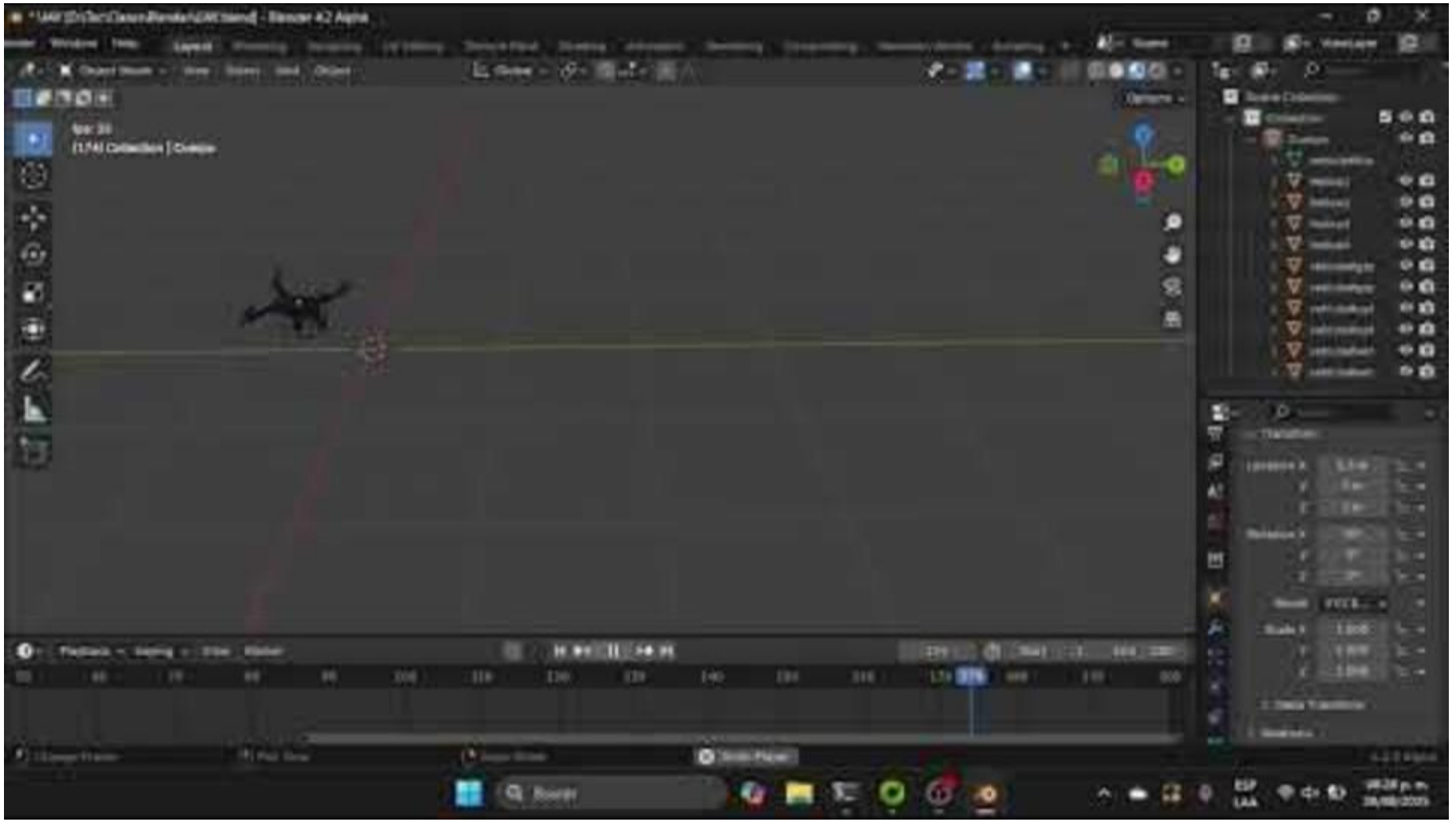
General

To establish the conceptual, technical and experimental foundations necessary to develop a reinforcement learning (RL) system applied to drones intended to operate in complex fire-related scenarios, using NVIDIA Omniverse, Isaac Sim and Isaac Lab as simulation and validation platforms.

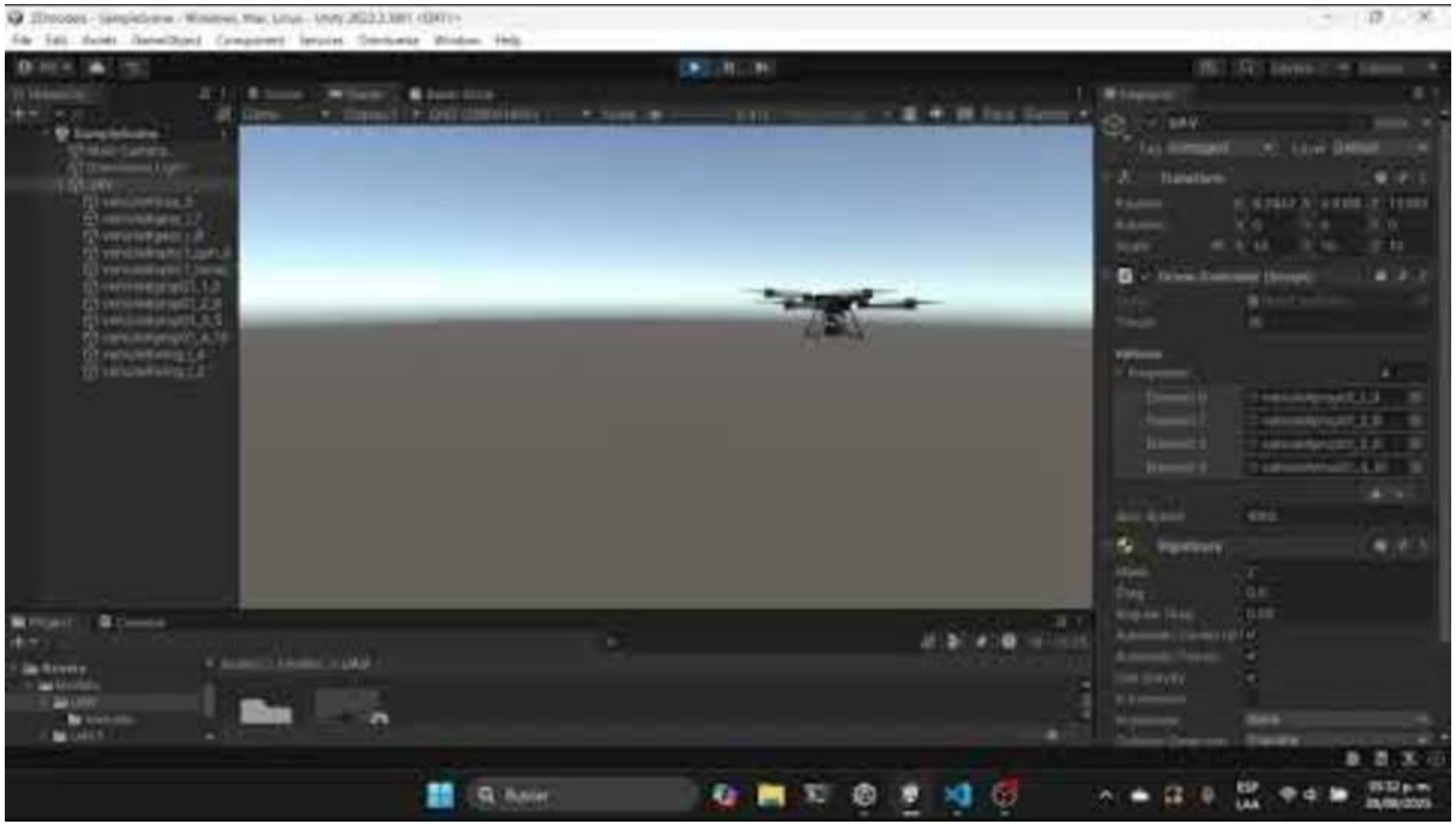
Specific

- Theoretical understanding of RL applied to aerial robotics (completed).
- Study of Omniverse, Isaac Sim, and Isaac Lab (completed).
- Implementation of simulations and sensors (partially completed).
- Preliminary design of the RL environment and configuration of the MDP (in progress).
- Policy training and transfer (pending for subsequent phases).
- Collaborative multi-agent (not yet addressed).

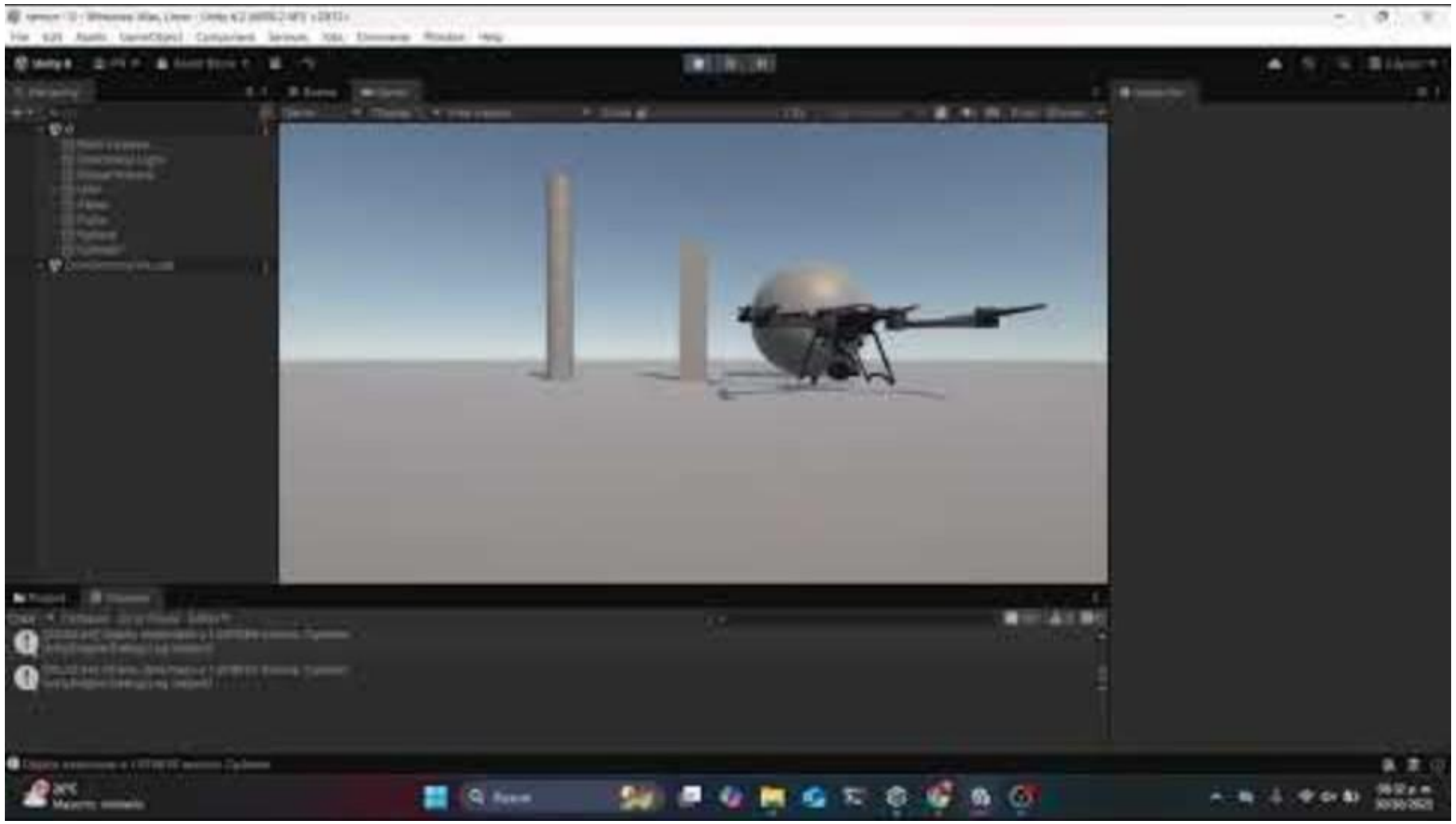
Results obtained



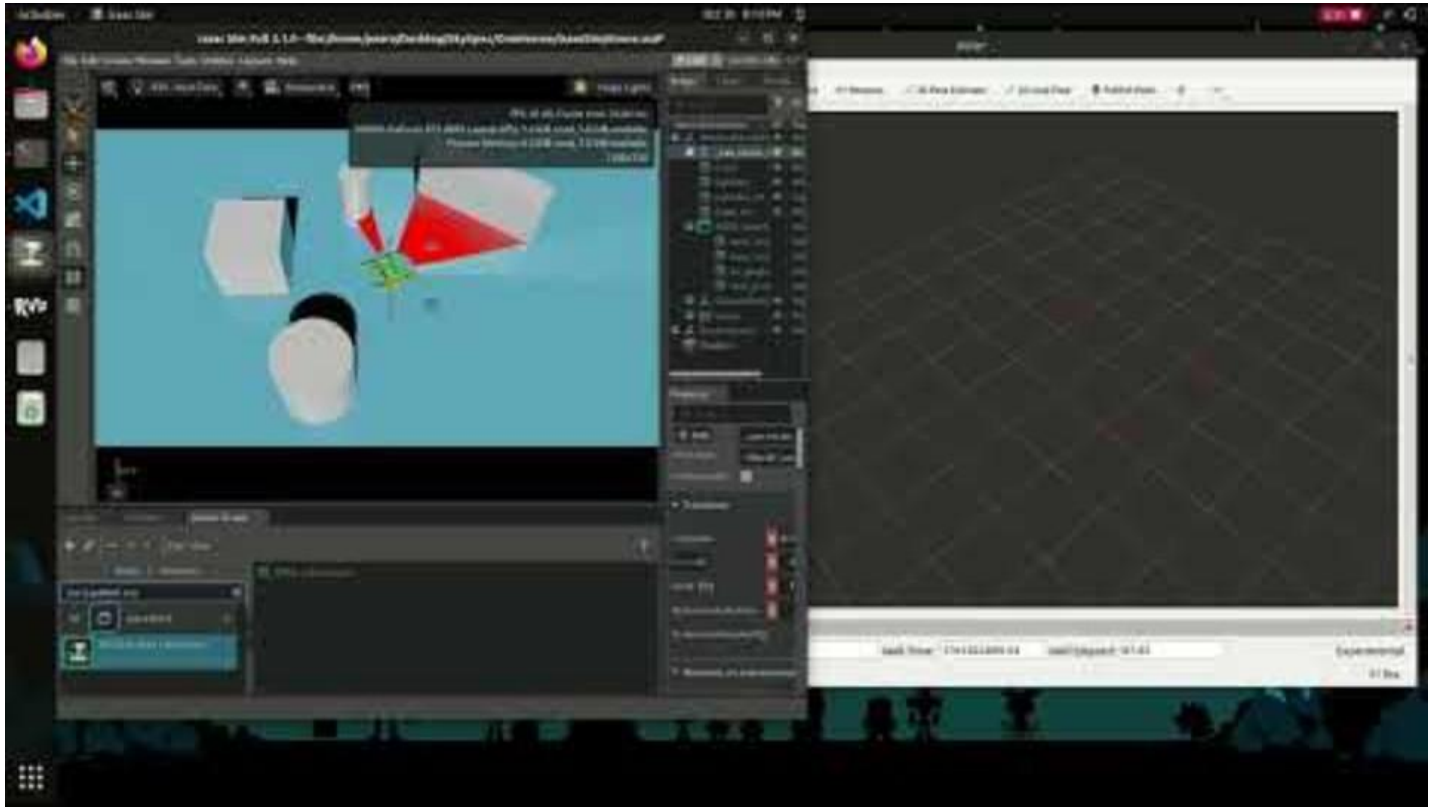
3D Drone Model - Blender



3D Drone Model - Unity



Lidar Sensor - Unity



Lidar Sensor - Isaac Sim

Conclusion

- Theoretical and technical foundations for the use of real-world robotics (RW) in drones were consolidated.
- Key tools—Omniverse, Isaac Sim, and Isaac Lab—were integrated and evaluated.
- Significant progress was made in modeling, sensory simulation, and the structure of the RW environment.
- The project is ready to begin autonomous training phases.
- The experience strengthened research and analysis skills in aerial robotics.

Future work

- Finalize the real-world environment in Isaac Lab.
- Drone training for: Navigation to (x,y) coordinates.
- 3D extension (x,y,z).
- More complex scenarios: warehouses, mazes, damaged areas.
- Multi-agent real-world exploration.