Phase 1 - Environment Setup & Simulation Initialization

Project: *Autonomous Parking - 2D + 3D Simulation Environment*

Author: Vishwaksena Dingari

UID: 121113764

Course: MSML 642 - Final Project

Date: November 6, 2025

1 Objective

This project implements an **autonomous parking system** using both **Gazebo (3D)** and **custom 2D kinematic environments**. The goal is to train and evaluate reinforcement learning or hybrid controllers for realistic parking scenarios.

Phase 1 demonstrates that all **simulation components are operational** and ready for algorithm integration.

2 Environment Setup

Tools Installed & Configured

Category	Tools / Frameworks
ROS 2	**Humble Hawksbill** (`/opt/ros/humble`)
Simulation	**Matplotlib**, Gazebo Classic (final rendering check)
Languages	Python 3.10+
Visualization	matplotlib (custom 2D renderer)
Libraries	NumPy, PyYAML

Workspace Structure

```
~/autonomous_parking_ws/src/autonomous_parking/
├── autonomous_parking/
│   ├── __init__.py
│   ├── config_loader.py
│   ├── keyboard_drive_2d.py
│   ├── print_bays.py
│   ├── test_env2d.py
│   └── env2d/
│       ├── __init__.py
│       ├── parking_env.py
│       └── test_env2d.py
├── config/
│   └── bays.yaml
├── launch/
│   ├── parking_lot_a.launch.py
│   └── parking_lot_b.launch.py
├── worlds/
│   ├── parking_lot_a.world
│   ├── parking_lot_b.world
│   └── parking_lot_parallel.world
├── resource/
├── test/
├── package.xml
├── setup.py
├── setup.cfg
└── readme.md
```

Build verification:

```
cd ~/autonomous_parking_ws
colcon build --symlink-install
source install/setup.bash
```

3 Simulation Proofs

3.1 Gazebo 3D Parking Lots

Two parking-lot environments were created to match realistic layouts:

Lot	Layout	Key Features
lot_a	Horizontal 2-row (A1-A6 facing B1-B6)	Center road between rows, aligned spacing
lot_b	L-/T-shaped (H1-H5 + V1-V5)	Intersection road, vertical + horizontal lanes

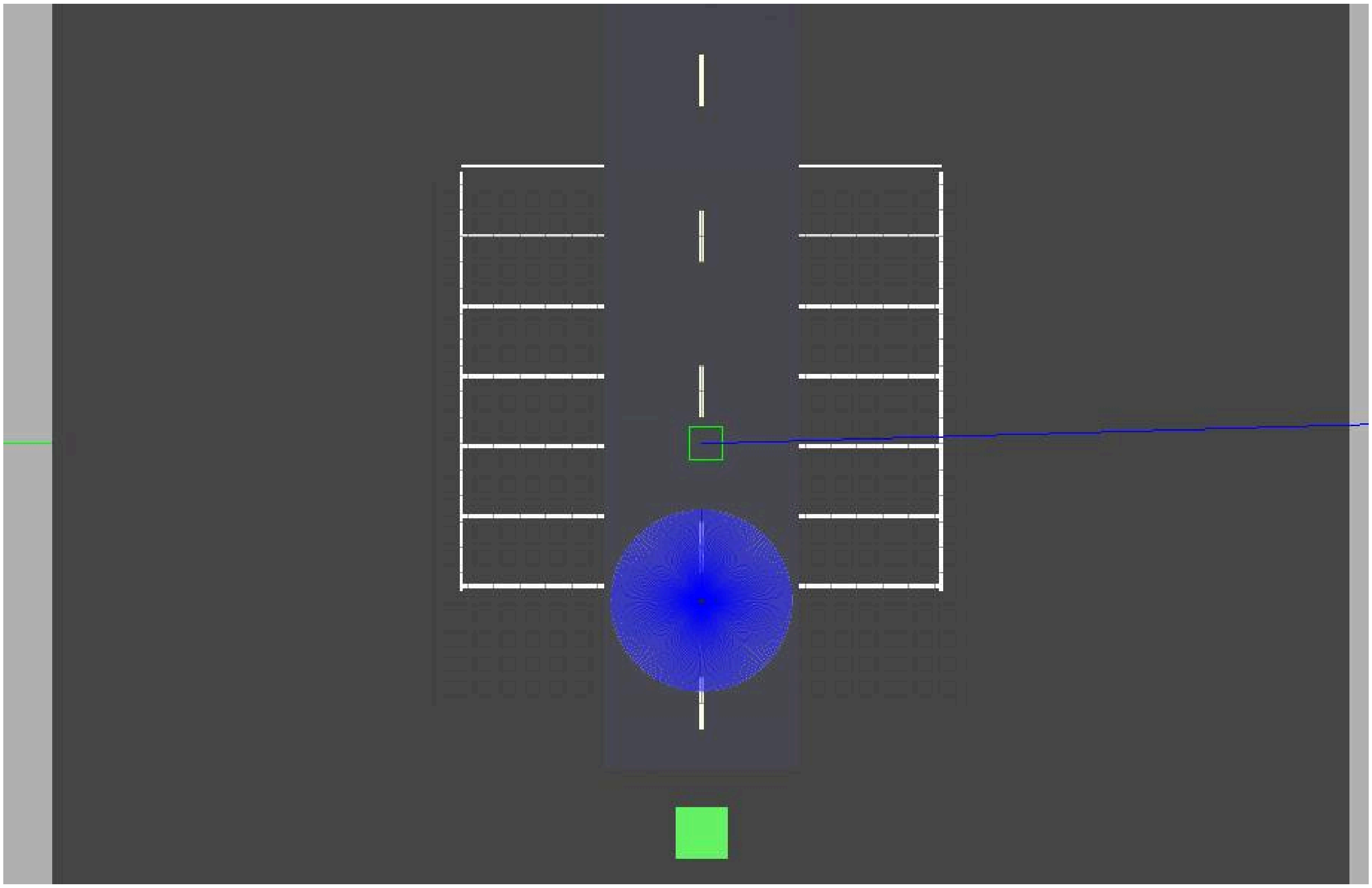
Launch commands:

```
ros2 launch autonomous_parking parking_lot_a.launch.py
ros2 launch autonomous_parking parking_lot_b.launch.py
```

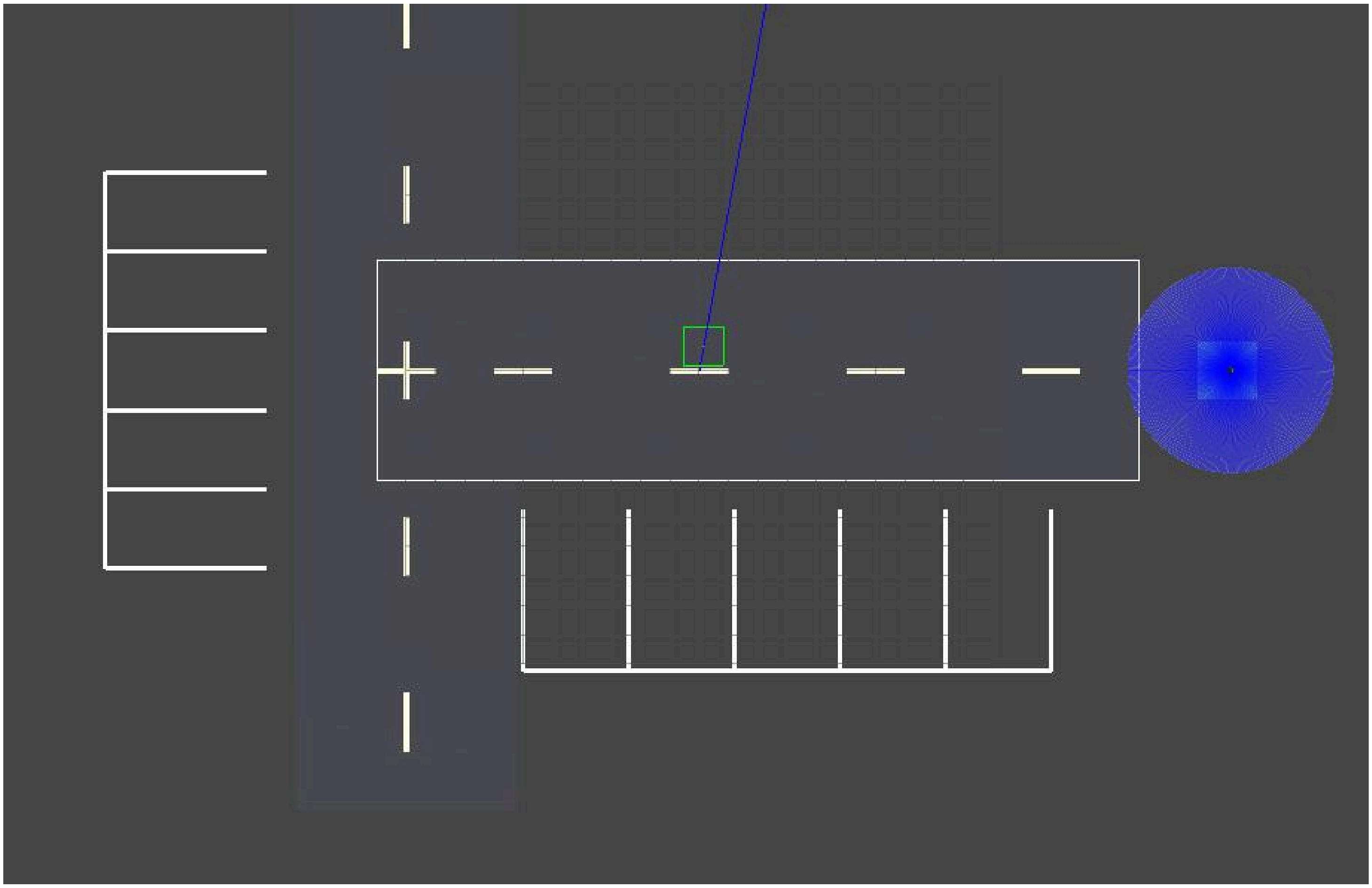
Each world includes:

- Properly scaled parking bays (5.5 × 2.7 m)
- Asphalt road sections (6 m wide)
- Vehicle spawned at entrance orientation

Images:
Gazebo Lot A - Top View:



Gazebo Lot B - Top View:



3.2 2D Kinematic Simulation

The Python environment `parking_env.py` reproduces identical geometry in 2D with dynamic visualization.

Run examples:

```
ros2 run autonomous_parking keyboard_drive_2d --lot lot_a
ros2 run autonomous_parking keyboard_drive_2d --lot lot_b
```

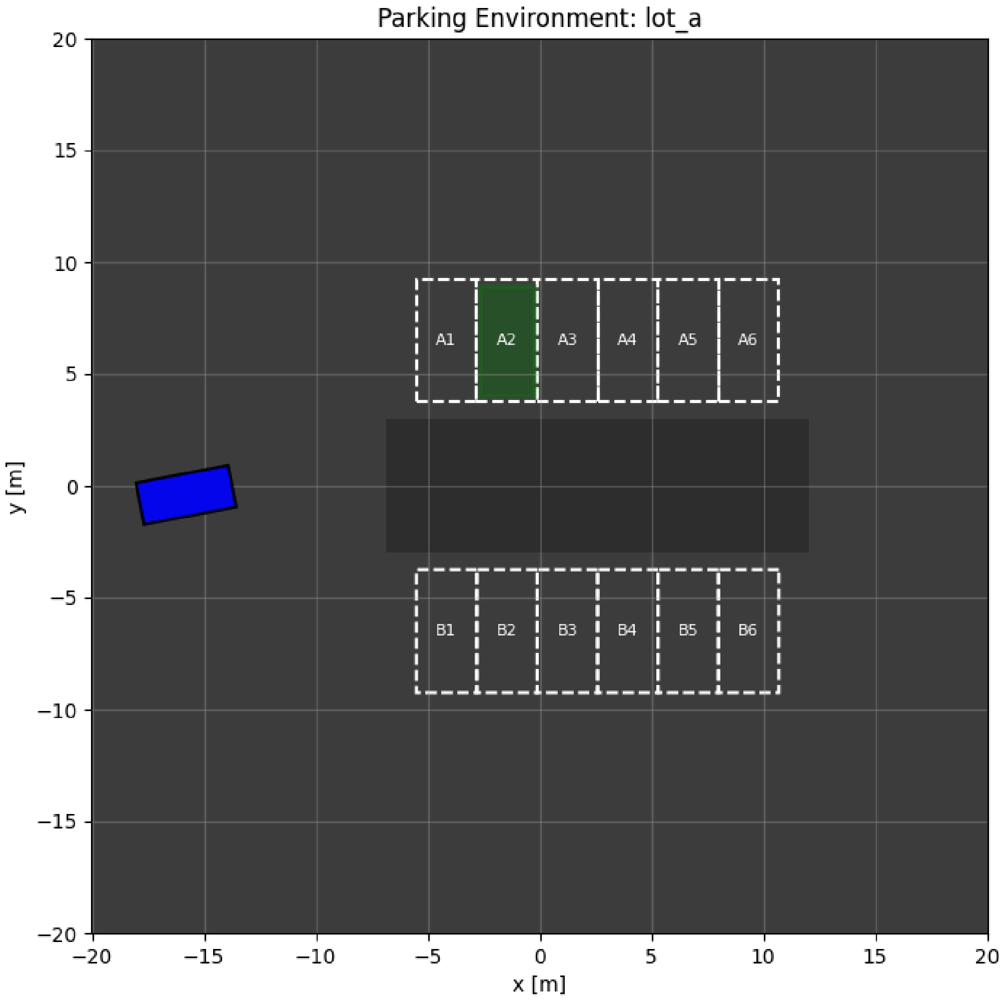
Keyboard Controls

Key	Action
W / ↑	Move forward
S / ↓	Reverse
A / ←	Steer left

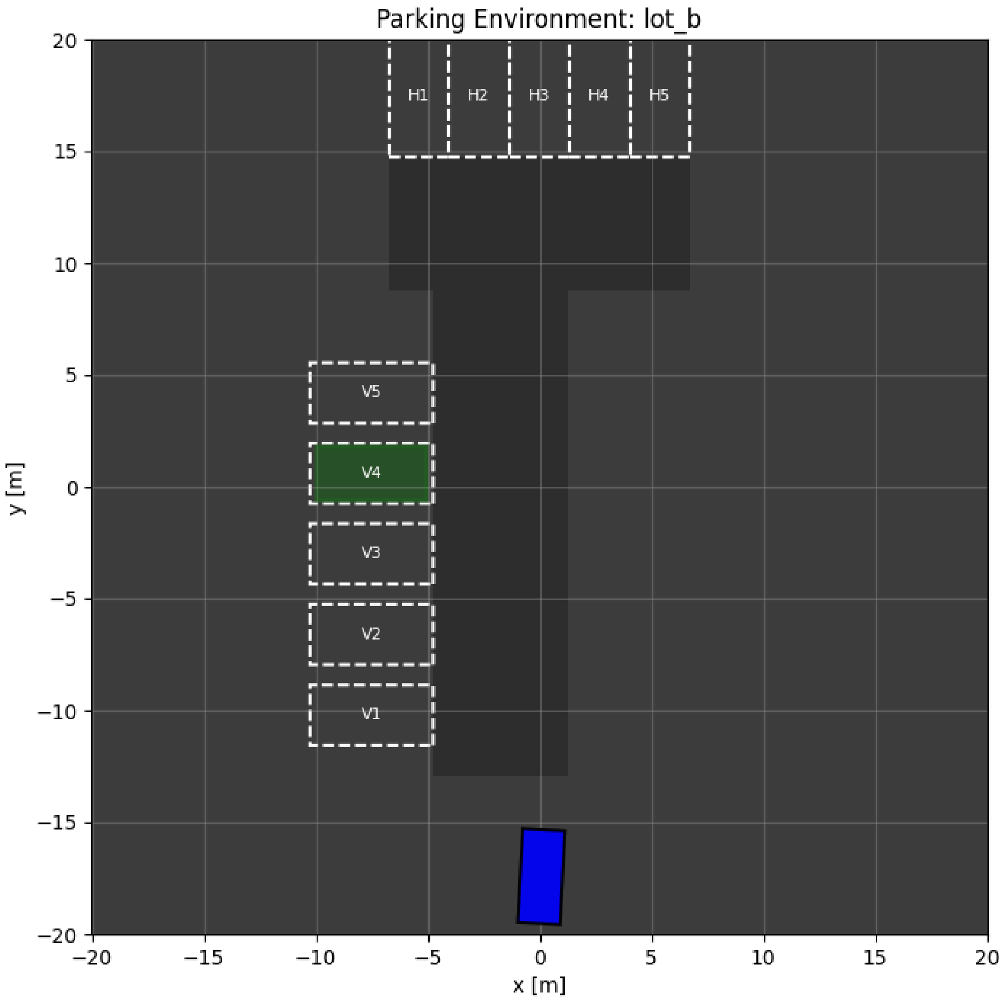
Key	Action
D / →	Steer right
Space	Brake / stop
R	Reset to new starting pose
B	Cycle target goal bay
Q	Quit simulation

Images:

2D Lot A - Matplotlib Render:



2D Lot B - Matplotlib Render:



4 Quick Usage Guide

Git Workflow

```
# Before working
git pull origin master

# After changes
git add -A
git commit -m "descriptive message"
git push origin master
```

Ubuntu - Full ROS + Gazebo + 2D

Setup (once per terminal)

```
source /opt/ros/humble/setup.bash
cd ~/autonomous_parking_ws
colcon build --symlink-install
source install/setup.bash
```

Run Gazebo 3D

```
ros2 launch autonomous_parking parking_lot_a.launch.py
# or
ros2 launch autonomous_parking parking_lot_b.launch.py
```

Run 2D Simulation

```
# Test environment
python -m autonomous_parking.env2d.test_env2d

# Keyboard control
python -m autonomous_parking.keyboard_drive_2d --lot lot_a
```

Ubuntu / macOS - 2D Only (No ROS)

```
cd ~/autonomous_parking_ws

# Create virtual environment
python -m venv .venv
source .venv/bin/activate

# Install package
pip install -e src/autonomous_parking
pip install numpy pyglet matplotlib pyyaml

# Test environment
python -m autonomous_parking.env2d.test_env2d

# Keyboard control
python -m autonomous_parking.keyboard_drive_2d --lot lot_a
```

This allows **cross-platform development**: Ubuntu handles ROS + Gazebo + 2D, while macOS supports 2D-only workflows using the same codebase and configuration files.

5 Verification Summary

Component	Notes
Gazebo Lot A Two rows + center road correctly aligned	
Gazebo Lot B	L-shaped layout with connected roads
2D Lot A	Road and bays match Gazebo geometry
2D Lot B	Correct T-shape; minor alignment refinement
Config Integration	Fully YAML-driven, no hardcoded values
Rendering	Real-time matplotlib animation functional
Build/Launch	Clean builds, no runtime errors

6 Minor Refinements Pending

1. Fine-tune `lot_b` vertical road alignment (~0.5 m shift for perfect Gazebo match)
2. Optional: Add lane markings and directional arrows in 2D
3. Implement shared pose publisher for synchronized Gazebo + 2D visualization
4. Minor tweaks in visualization and physics for the robot.

7 Conclusion

Phase 1 environment setup is **fully operational**. Both 3D Gazebo and 2D matplotlib simulations accurately represent the parking lot layouts and support vehicle control. The system is ready for algorithm development and reinforcement learning integration in subsequent phases.