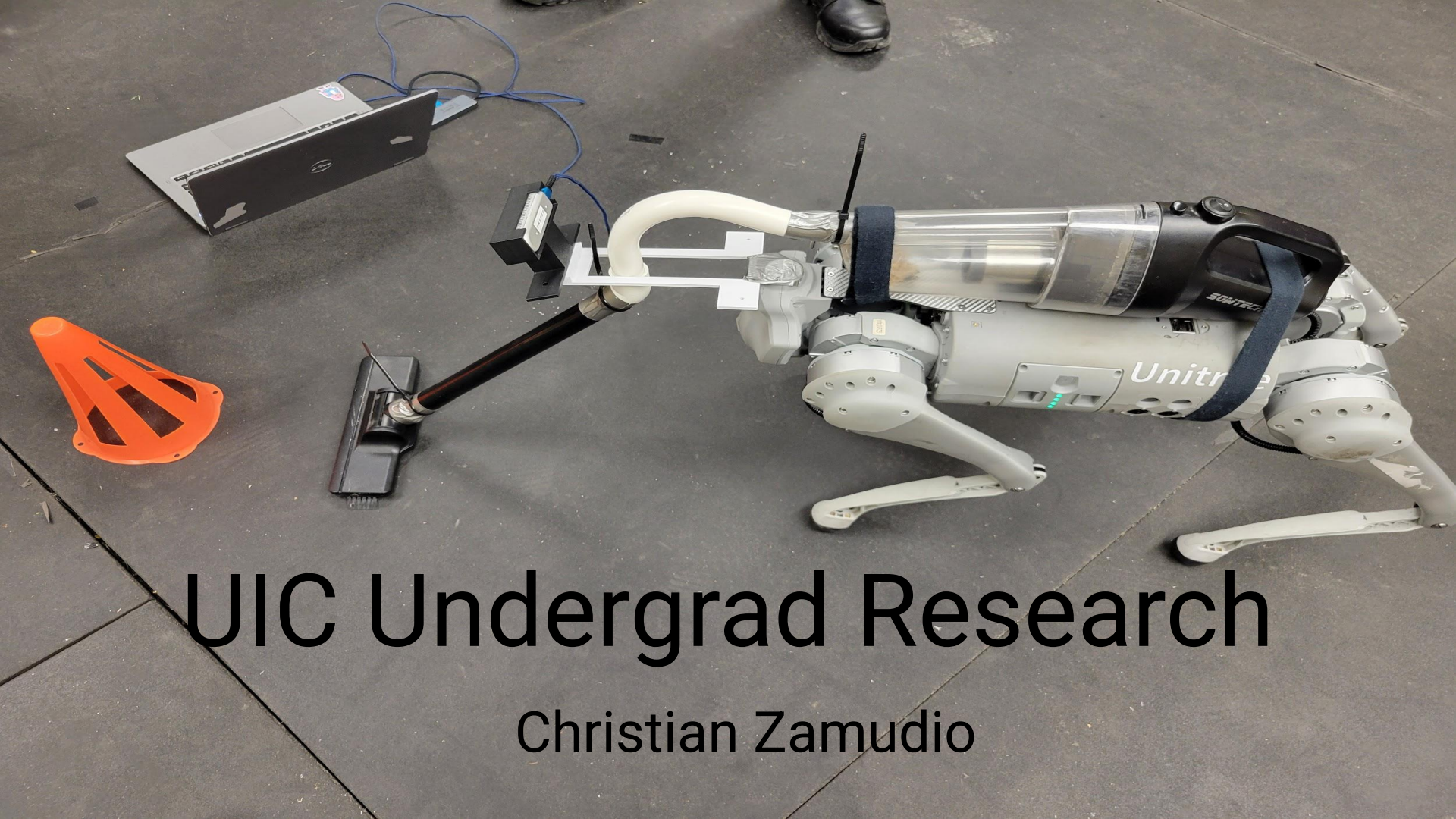
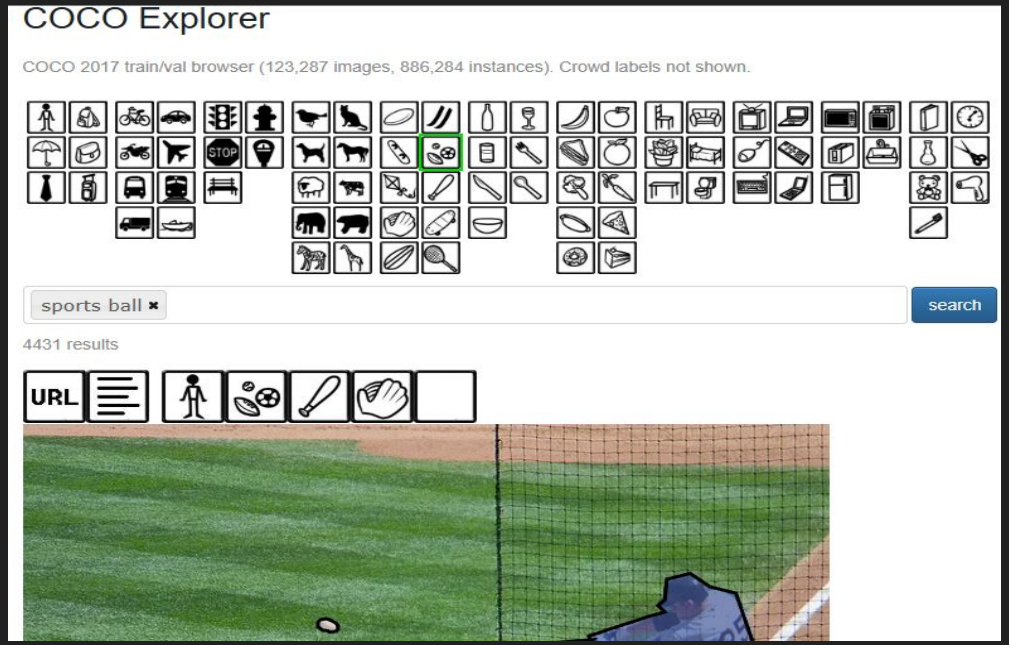


UIC Undergrad Research

Christian Zamudio

Computer Vision

YOLO object detection using COCO dataset allows for specific class detection.
[COCO Dataset]



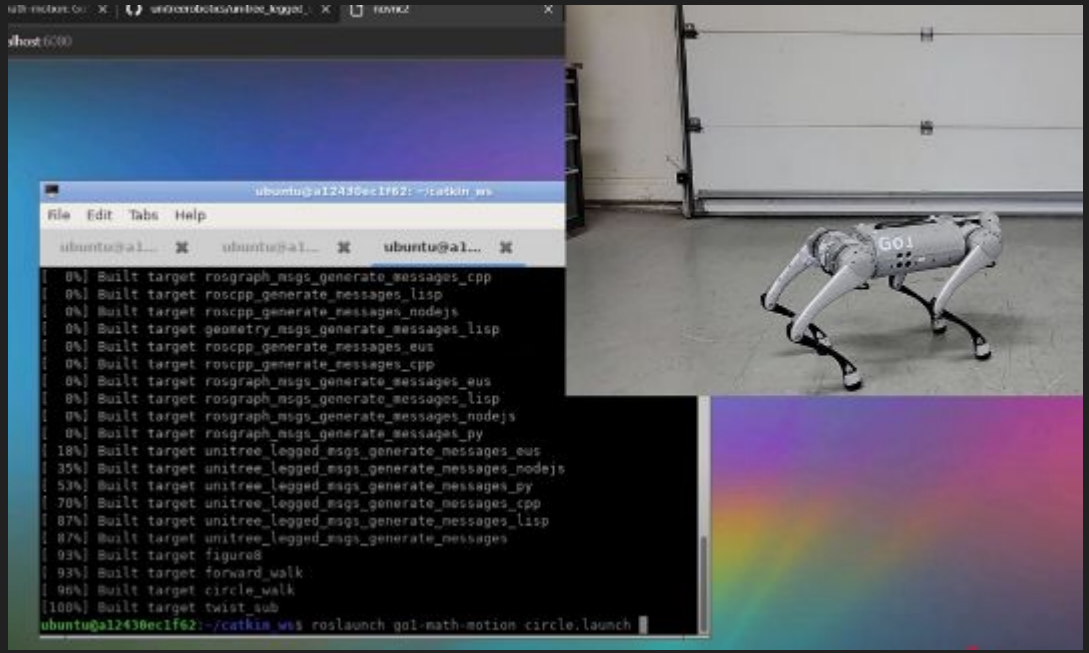
YOLOV8 ROS2 Node

XY input can be used to detect an object at specific location

```
item_dict[f'item_{n}'] = {
    'class': detection.names[detection_class[n]],
    'confidence': round(float(detection_conf[n]), 3),
    #'bbox': [round(float(coord), 3) for coord in object_boxes[n]],
    'position_xyz': [round(val, 3) for val in center_xyz],
    'estimated_width_m': round(width_m, 4)
```

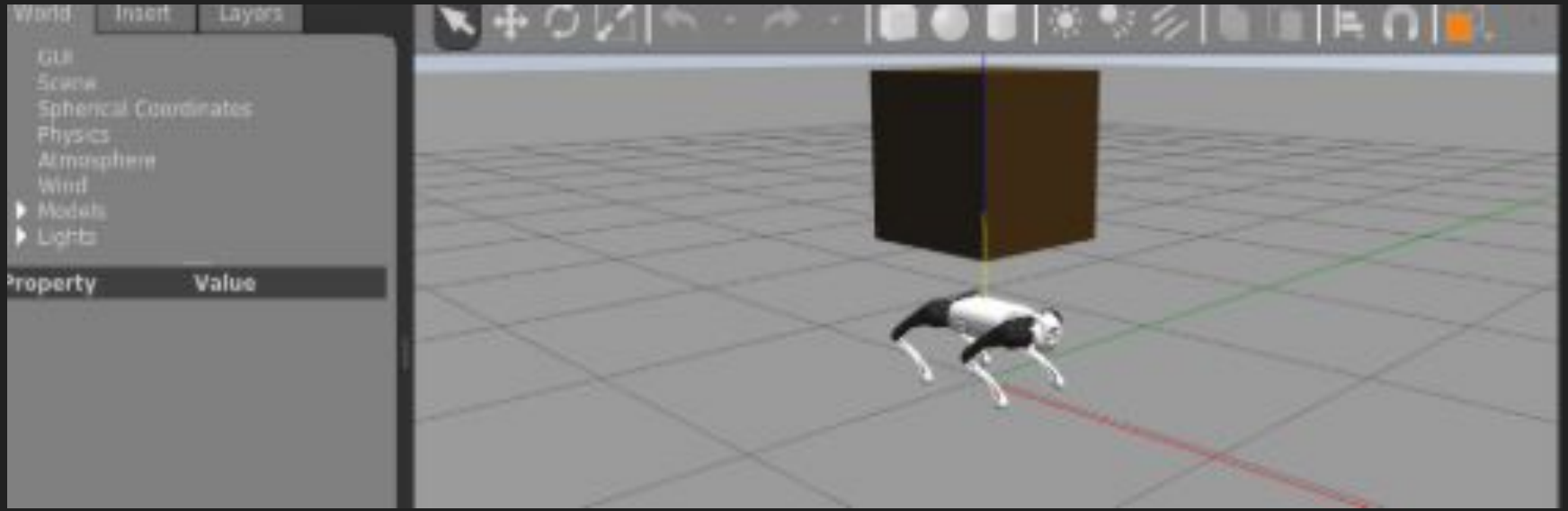
GO1 High Level Control

This example allows GO1 to walk around in a circle using ROS commands. Code can be modified for GO1 to go to XY psoition of object. [GO1 Video]



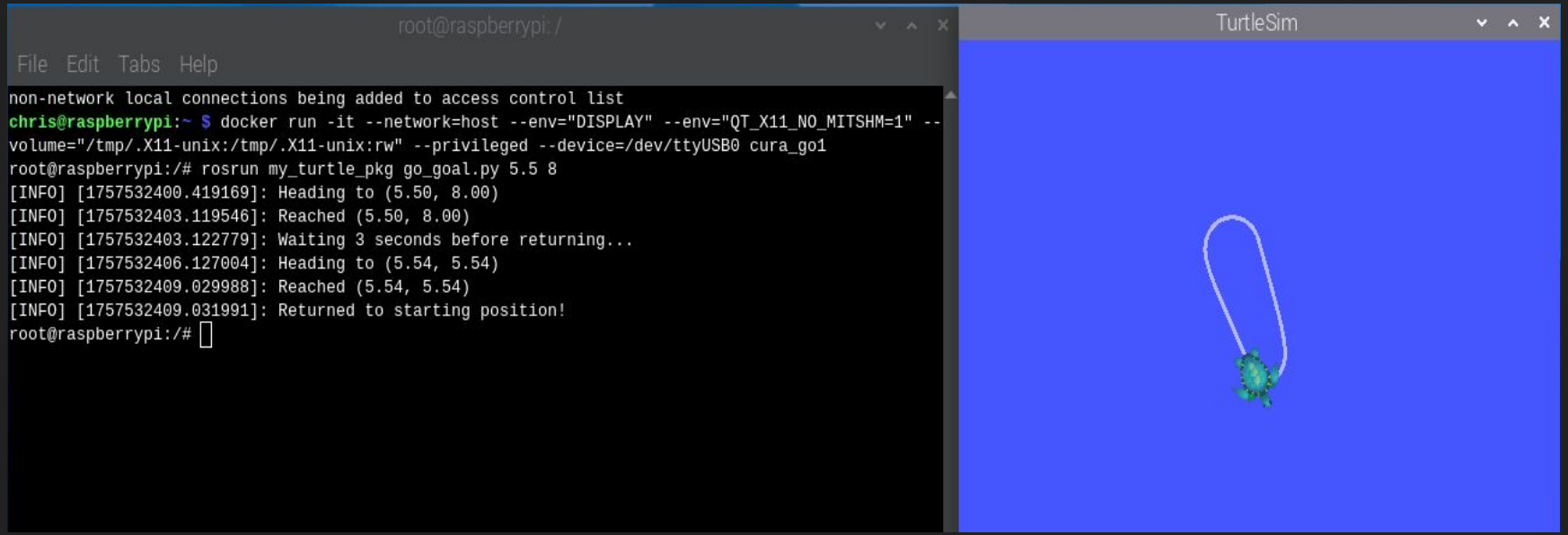
GO1 and A1 Simulations

Testing simulations for both robots using Gazebo, Unitree provides the ROS packages. [Github Unitree]



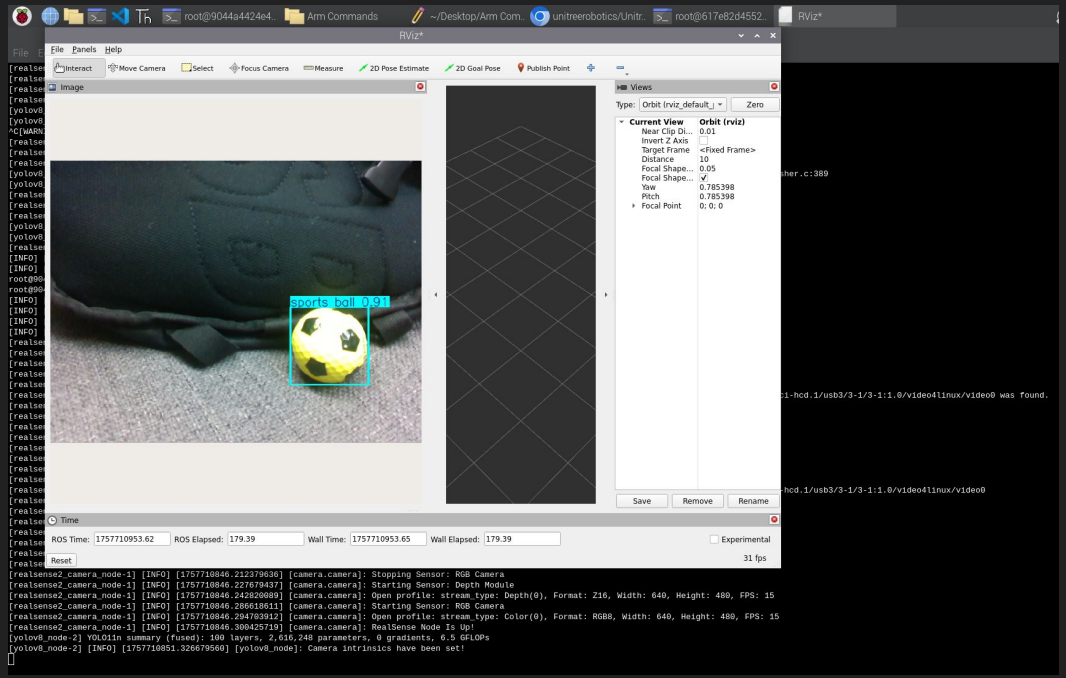
Turtlesim Simulation

Tested with turtlesim to simulate what GO1 would do when code is ran. XY location is 5.5 8. Returns back to starting position. Perfect for velocity commands.



Object Detection

ROS node that uses YOLO to detect a golf ball.



Object Detection Values

This string shows what the Intel Realsense is detecting. The XYZ values are in meters. The x value shows 0.329 which is around 1 foot distance from object to camera.

```
data: '{"item_0": {"class": "sports ball", "confidence": 0.661, "position_xyz": [0.329, -0.017, 0.015],
---
data: '{"item_0": {"class": "sports ball", "confidence": 0.657, "position_xyz": [0.329, -0.017, 0.016],
---
data: '{}'
---
data: '{"item_0": {"class": "sports ball", "confidence": 0.691, "position_xyz": [0.329, -0.017, 0.016],
---
data: '{"item_0": {"class": "sports ball", "confidence": 0.655, "position_xyz": [0.329, -0.017, 0.016],
---
```

ROS Unitree Package

This package allows for ROS2 to be used on the GO1 robot. Will use a python script for moving GO1 to object position. [ros_unitree]

Unitree Go1 ROS2 Driver

Humble CI no status | Iron CI no status | Jazzy CI no status

A ROS2 package which can be used to control the legged robot **Unitree Go1 Edu** using ROS topics.

This package acts has middleware between ROS2 and `unitree_legged_sdk`, which enables anyone to control the robot with velocity commands as well as receive back the robot state. Furthermore, additional features are also available such as standing up/down the robot and use head LEDs for some status information.

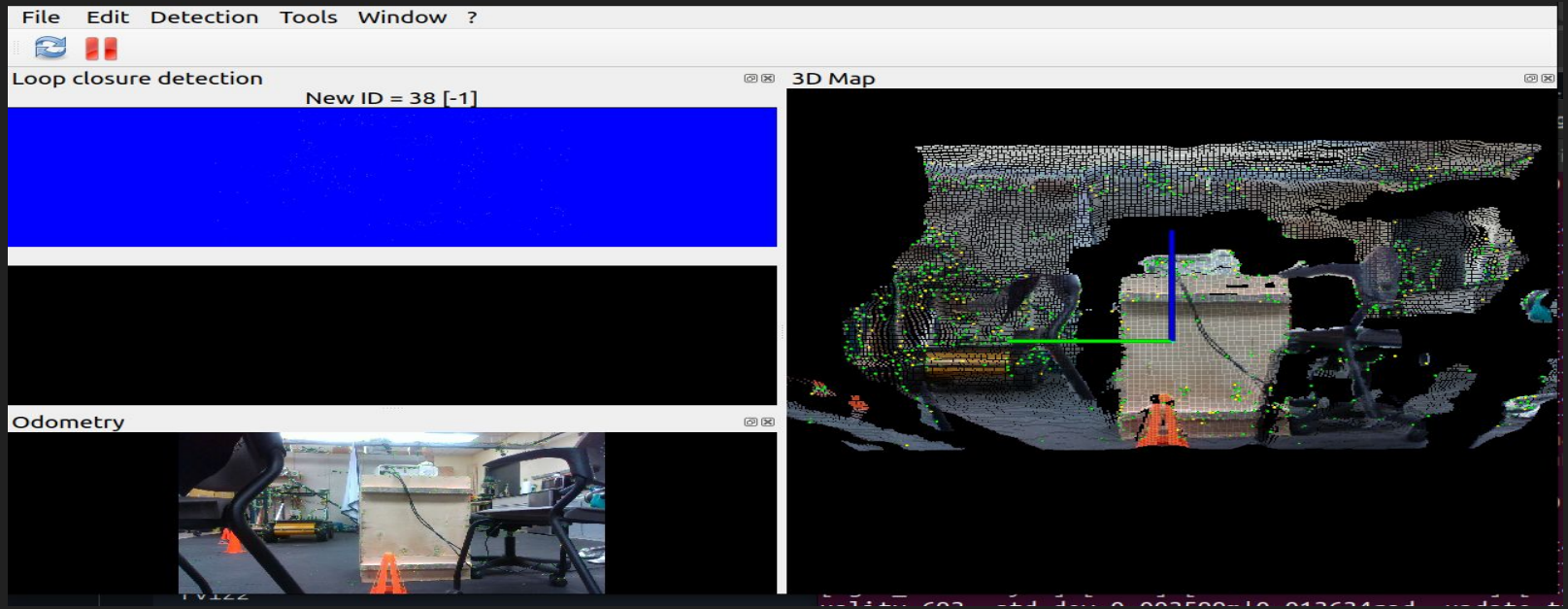
Testing of Python Code

GO1 experiences drift when publishing velocity commands, to fix this, external sensors from the realsense camera are necessary.

Demo shows GO1 supposed to be moving forward x direction.

Realsense SLAM

Rtab_slam demo, uses realsense IMU data. This data is fused with the odom of robot to solve drifting problem.

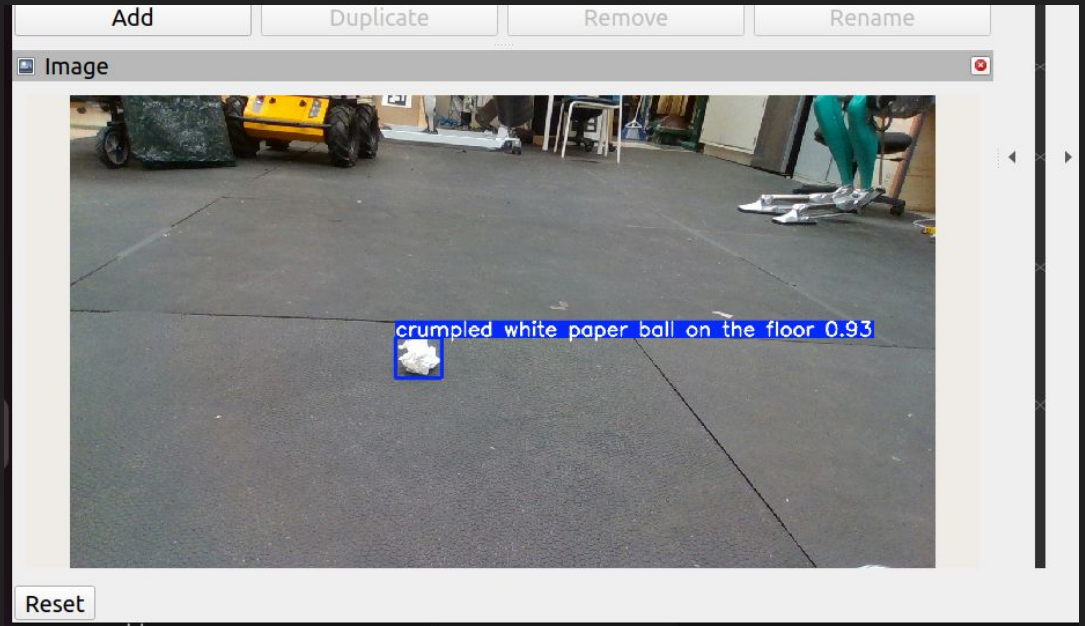


Unitree GO1 Ball Demo

Using the RealSense camera, the xy coordinates of the object are obtained, allowing GO1 to move to the target position.

Ultralytics YOLO-World

YOLO-World enables the creation of text prompts for zero-shot object detection, allowing objects to be detected without prior training.

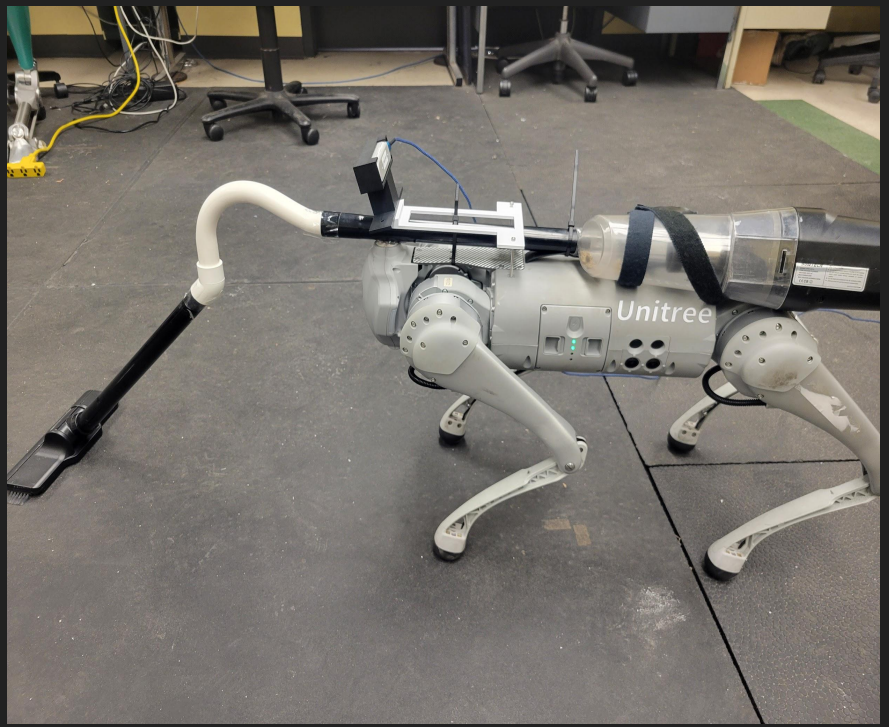


Attaching Vacuum to GO1

Possible demo setup for project, zip ties holding vacuum tube in place.

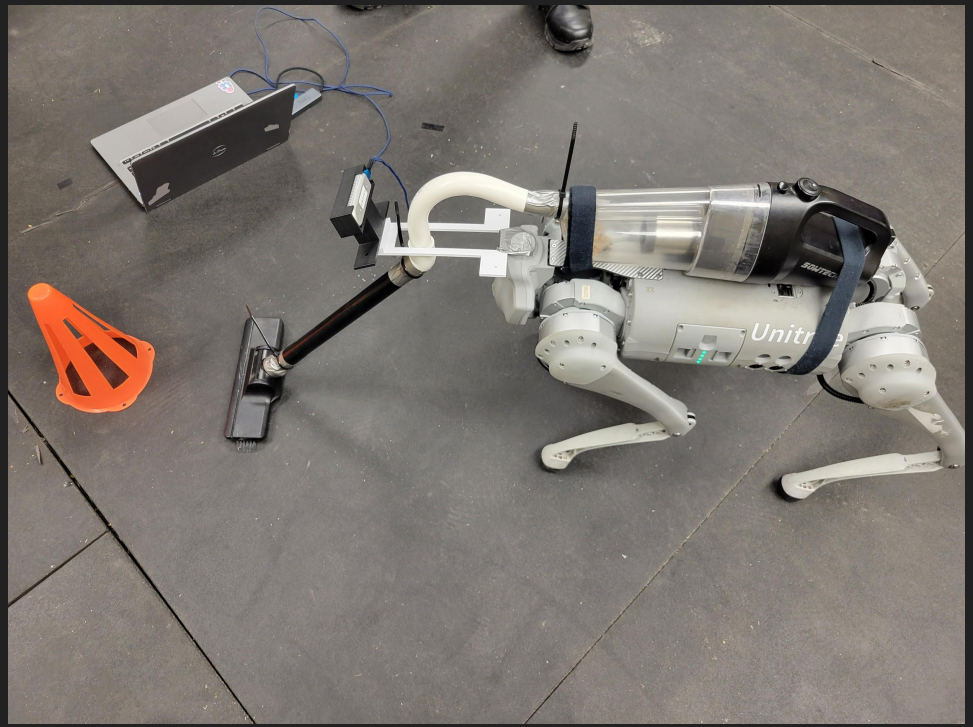
New Vacuum Attachment

Previous vacuum did not properly pick up dust on the ground, this attachment is rigid and worked far better in testing.



Final Design

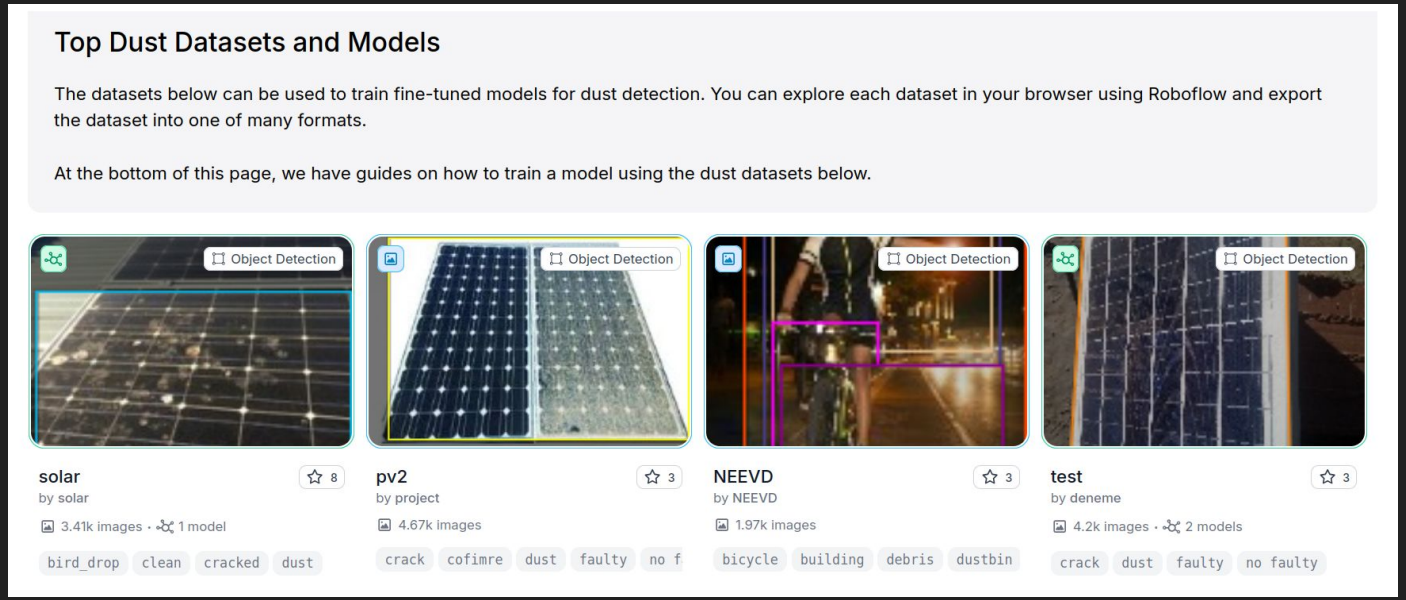
The camera mount was moved further down to avoid any obstruction.



Camera Localization Test

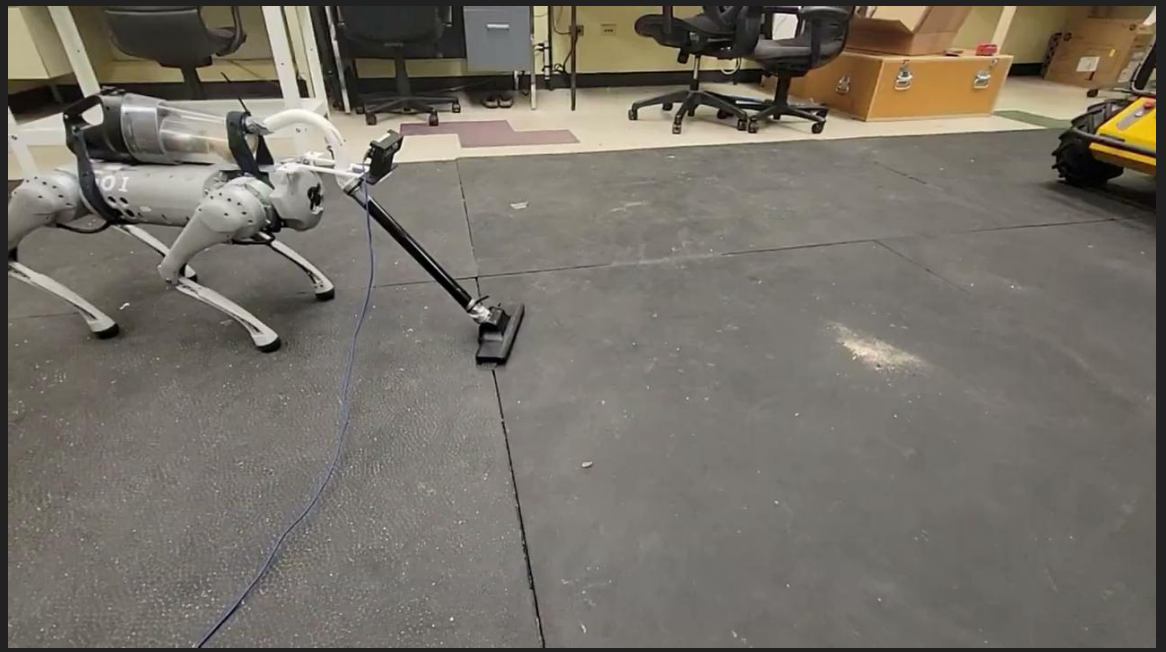
Feedback: Vision Model

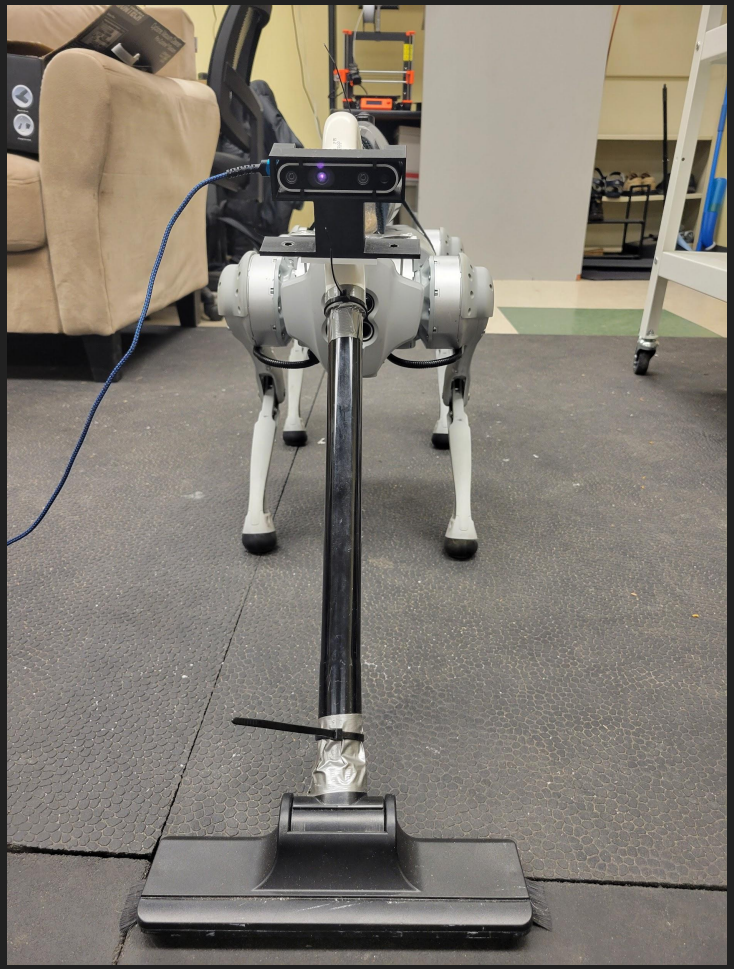
YOLO-World model is not as effective as a custom dust model in detections. Training a new model is necessary to collect dust. Roboflow is great for creating these datasets before training.

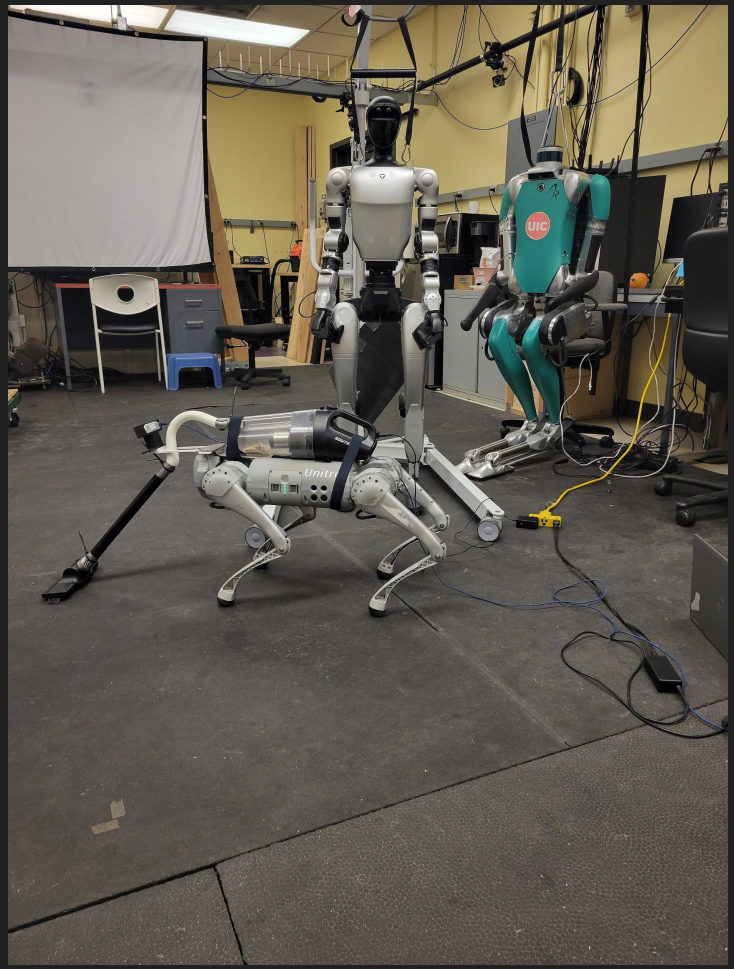


Feedback: Dust Collection

In order to get the vacuum to pick up almost all dust on the ground, a scrubbing backward and forward motion was added to end of the initial detection movement.






UIC

Final Testing