Developing a Robotic Learning System Using Pololu Romi Robot

Charlie Monroe

Advisor: Dr. Jim Conrad

UNIVERSITY OF NORTH CAROLINA CHARLOTTE

ELECTRICAL AND COMPUTER ENGINEERING

Background and Motivation

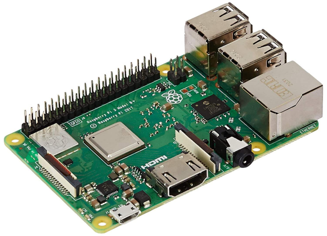
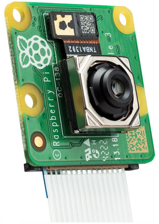
Previous Work

- TI-RSLK (Robotic Systems Learning Kit)
    - Communication limitations due to microcontroller and software capabilities.
    - The RSLK kit has been discontinued.

Motivation

- Building a high powered robotic learning system:
    - Pololu Romi with Raspberry Pi offers enhanced processing power, software flexibility, and educational applications for learning ROS.
    - A flexible framework for building robot applications.
    - Provides the tools and libraries to build and reuse software components.
    - Peer-to-peer network communication infrastructure.

Objectives

Integrate systems into the Pololu Romi robotic system:

- **Raspberry Pi** - processing power
- **Camera** - image processing
- **LiDAR** - object avoidance and SLAM

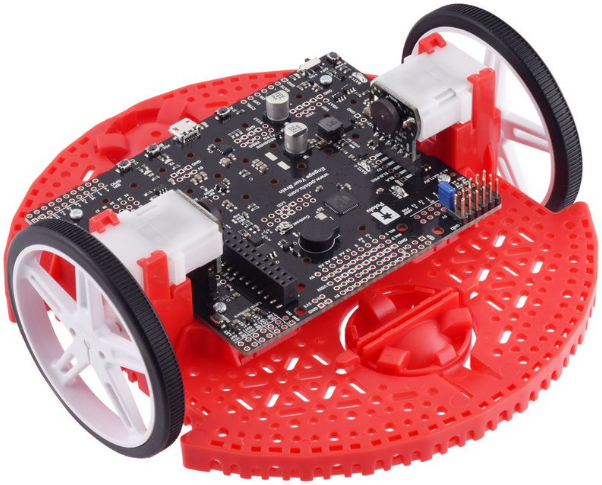
Develop and integrate ROS nodes for control and sensor data acquisition:

- RViz - robotic movement and environmental perspective

Test and evaluate the system's performance:

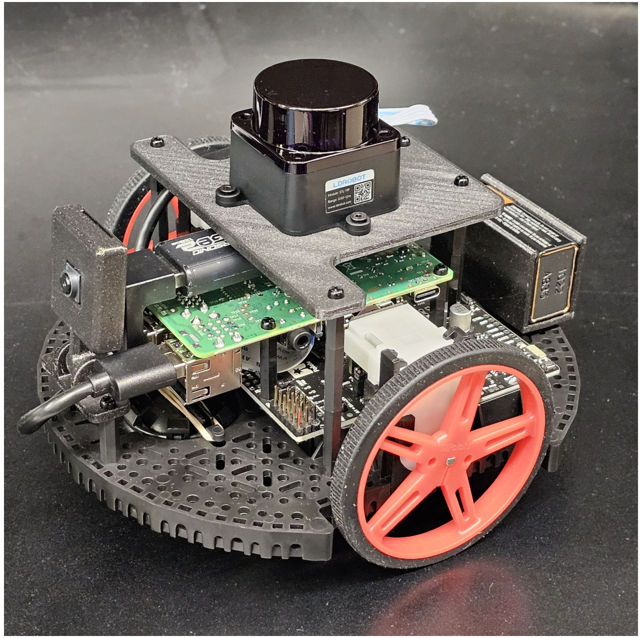
- Power consumption
- Encoder and position accuracy
- Evaluate object detection and SLAM algorithm

Robot Hardware



Romi Robot:

- Driver board - ATmega32U4
- Motor pair with encoders
- Arduino



Modified Robot:

- Romi base
- Raspberry Pi
- Additional sensors
- 3D printed parts

Hardware Design

Pololu Romi Chassis and Romi32U4 Control Board

- Manages low-level motor control and integrates with the Raspberry Pi.

Raspberry Pi 4B

- Capable of running various software environments, including Ubuntu/ROS.

LD19 LiDAR Module

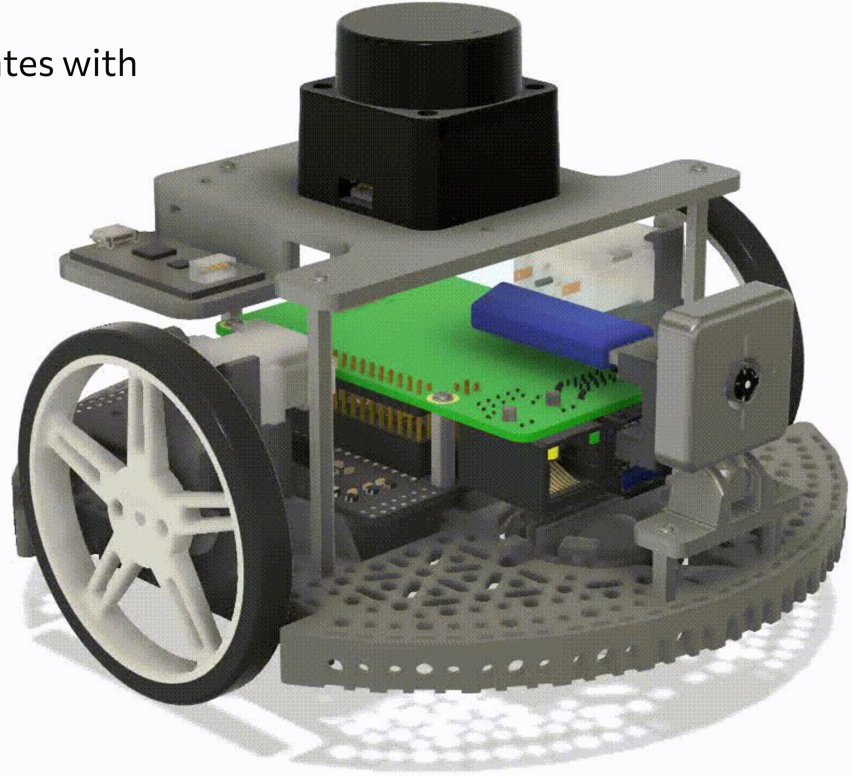
- 0.02~12m 360° range for Simultaneous Localization and Mapping (SLAM.)

Raspberry Pi Camera v2.1

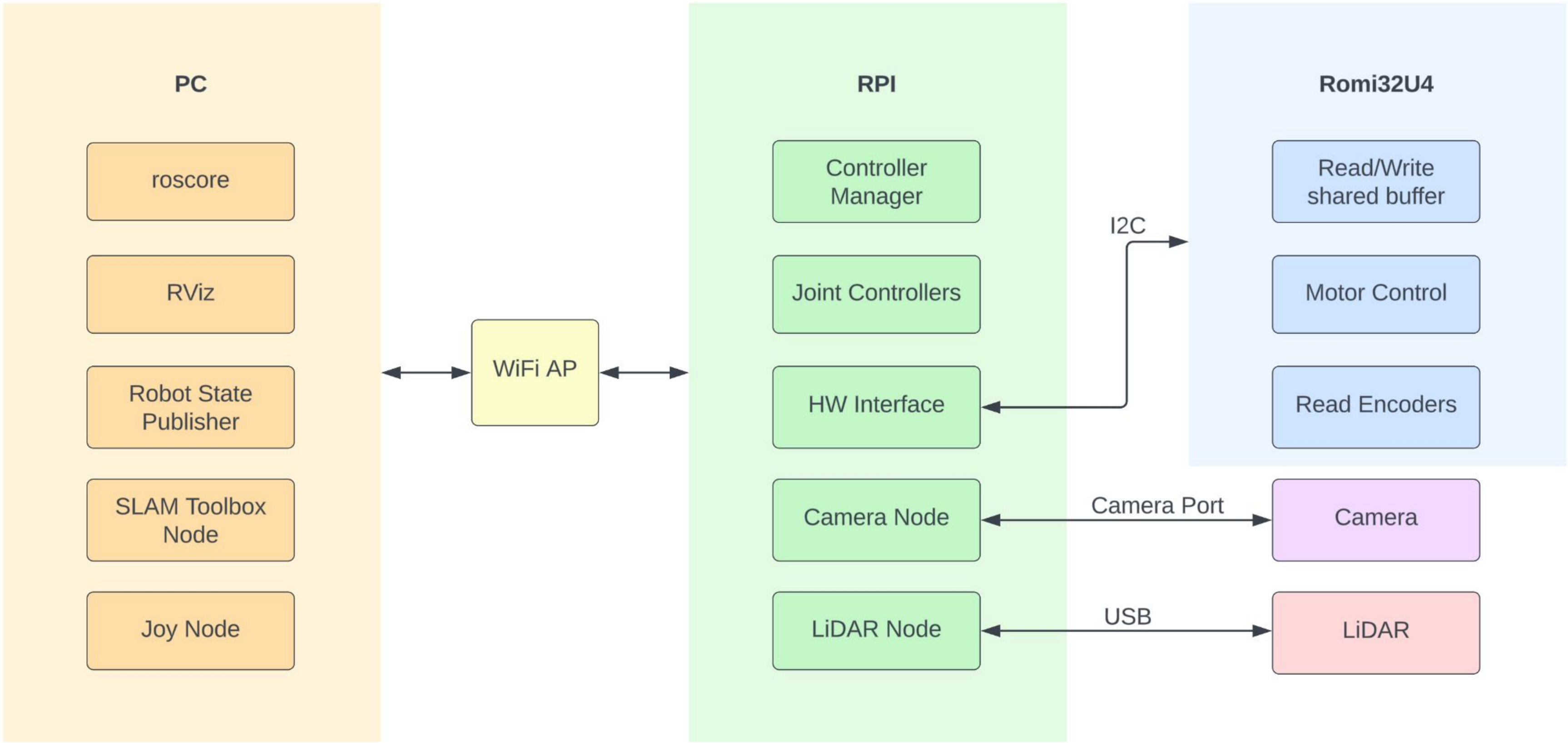
- 8MP, 1080p30 video for vision-based tasks.

USB Flash Drive/MicroSD Card

- Provides storage for the Raspberry Pi

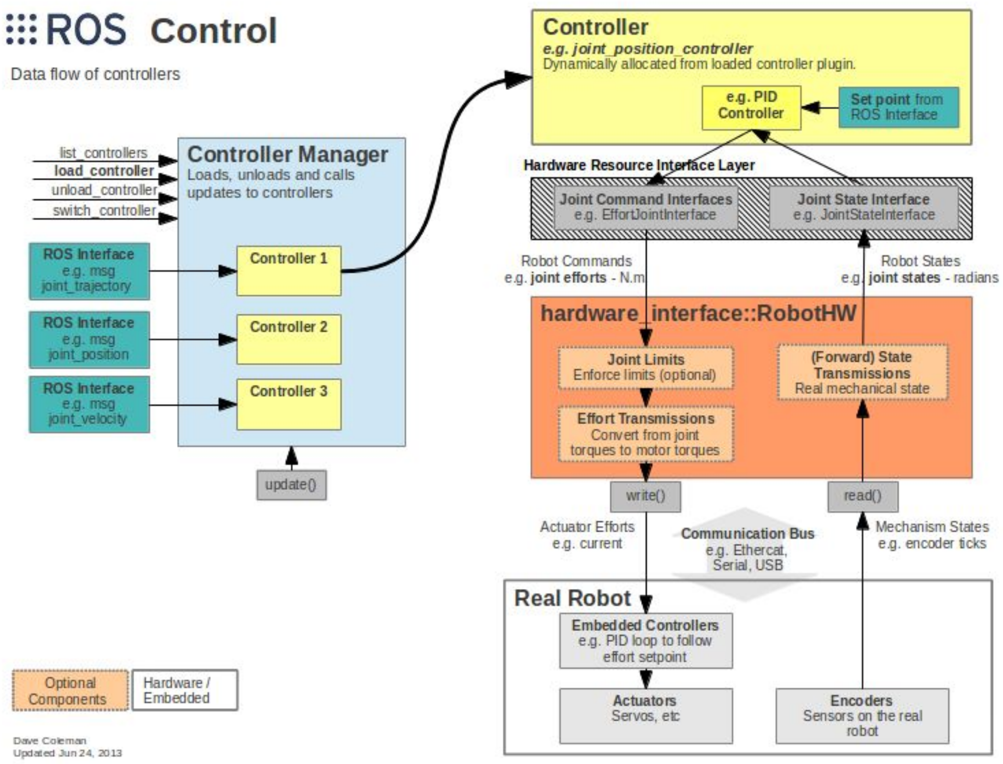


System Design

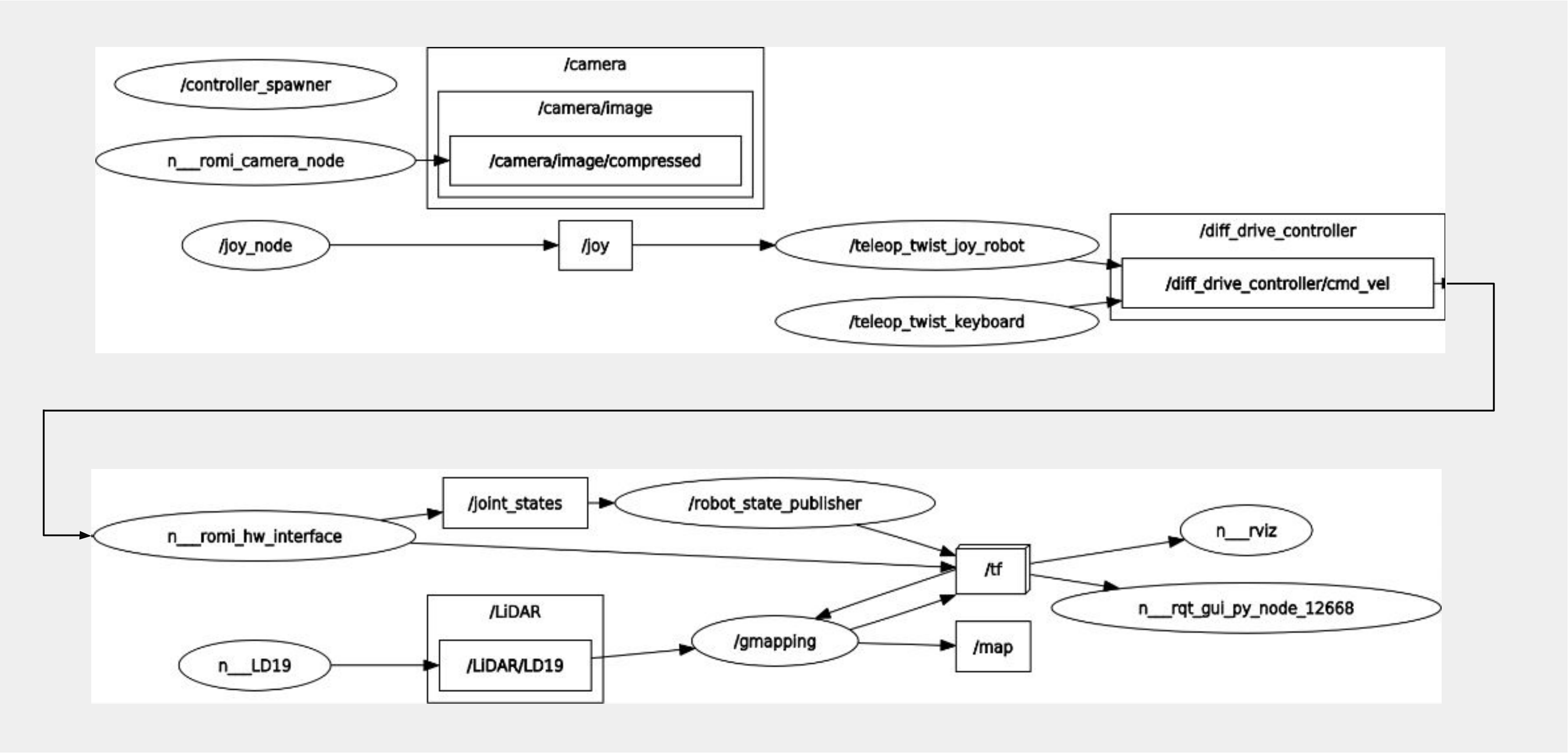


Software Architecture

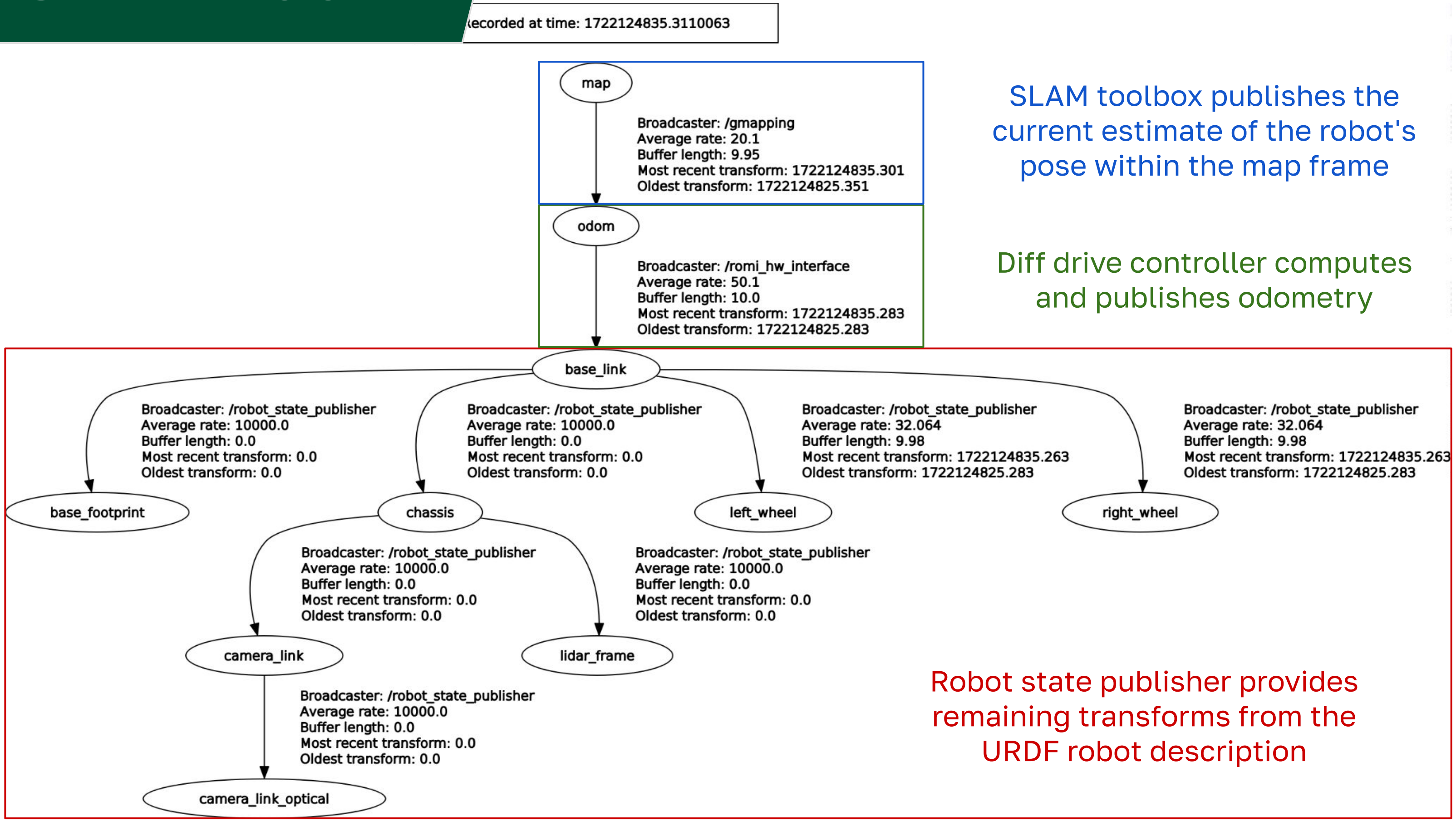
- Ubuntu 20.04 and ROS Noetic
- Unified Robotics Description Format (URDF) for physical robot description
- ROS control stack
  - Create controller manager
  - ROS differential drive controller
    - Joint state and joint velocity interfaces to wheel joints
  - C++ hardware interface class
    - Read and write methods
    - Class for I2C communication
    - Dead man’s switch
- Camera: OpenCV and image_transport
- LiDAR: LDROBOT LD19 and SLAM_toolbox



ROS RQT Graph



ROS TF Tree



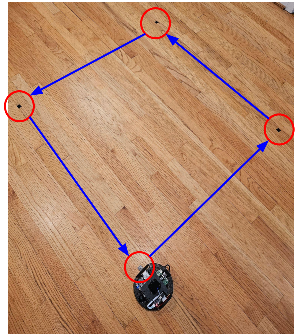
SLAM toolbox publishes the current estimate of the robot's pose within the map frame

Diff drive controller computes and publishes odometry

Robot state publisher provides remaining transforms from the URDF robot description

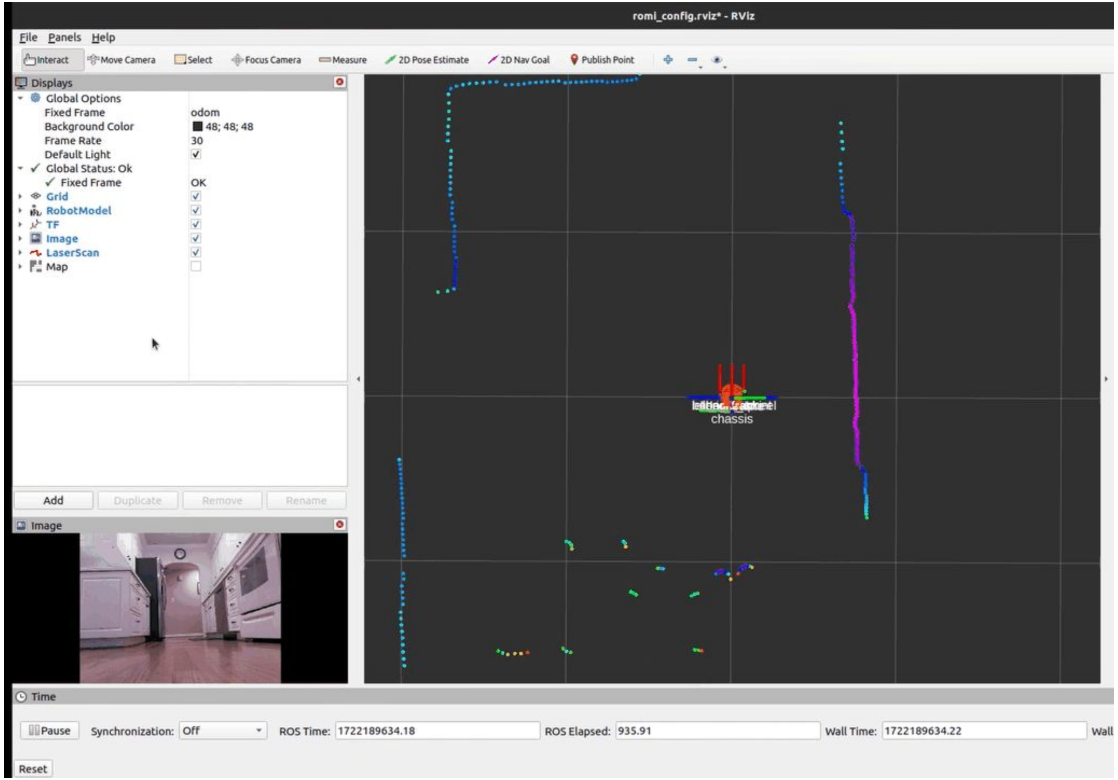
Accuracy Testing

- 1 meter test square
- Measure error in Rviz



Driving forward 1 meter

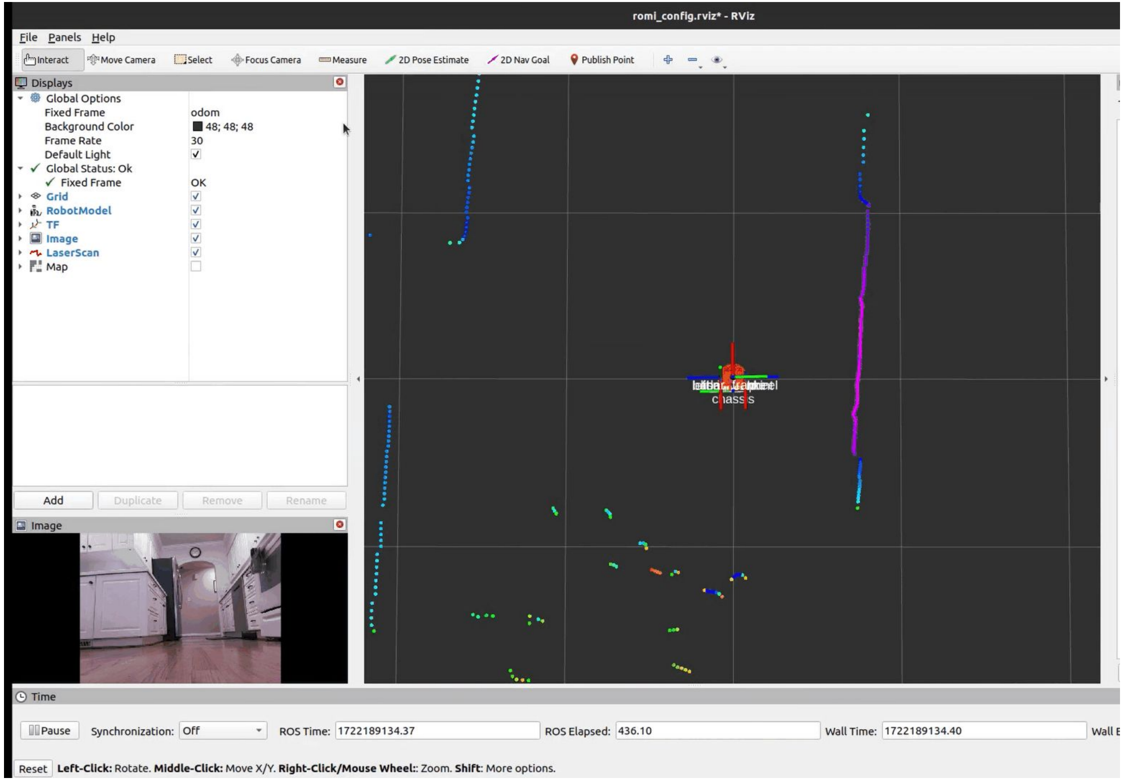
Test #	Error (cm)
1	<1
2	<1
3	<1
4	<1
5	<1
AVG	<1



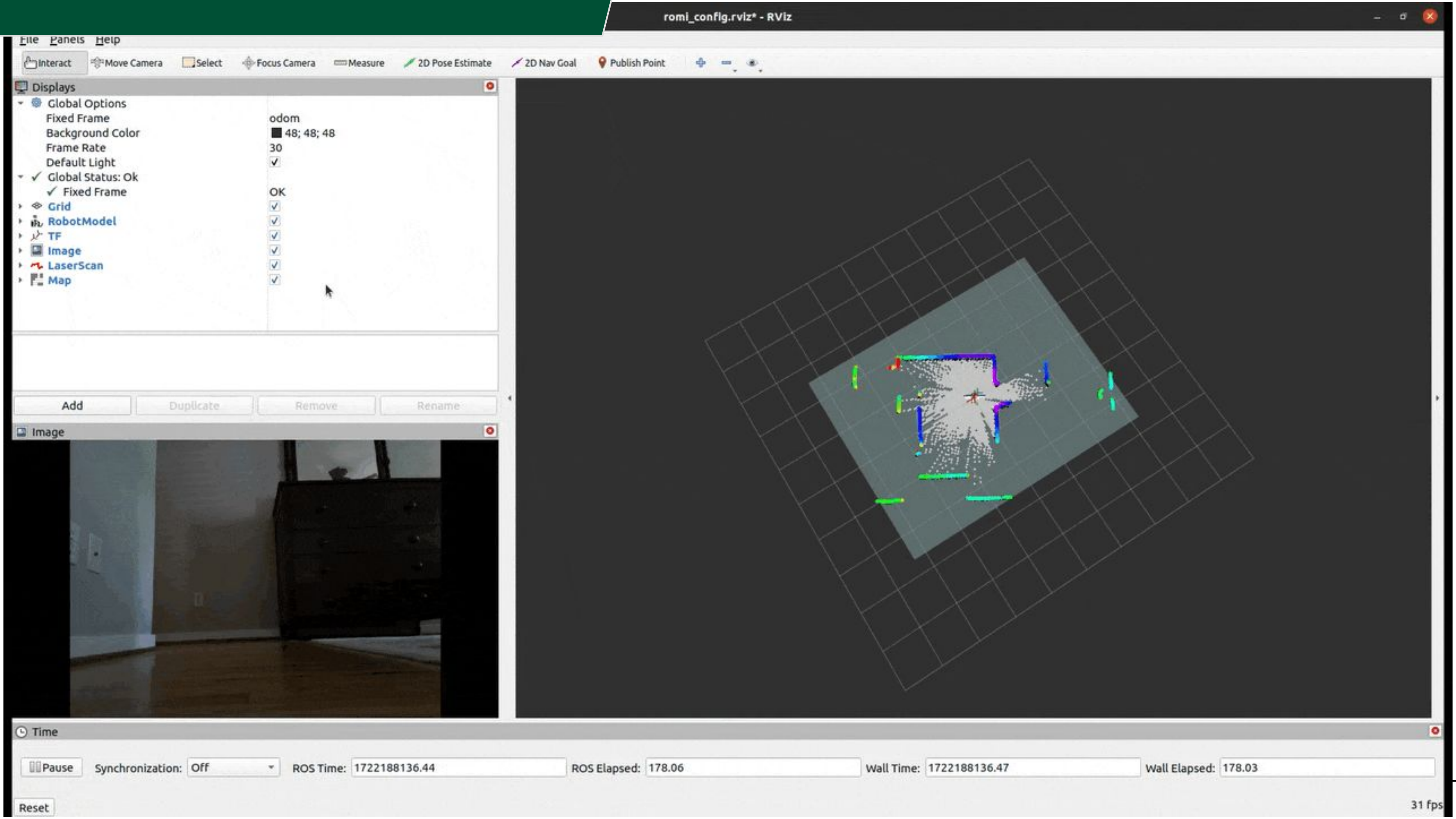
Accuracy Testing

Driving 1 meter square

Test #	Error (cm)
1	2.5
2	3.1
3	1.5
4	2.9
5	1.2
AVG	2.24



Results - SLAM



Results - Power

- Modified 40-pin extension header
- USB current meter

Battery Type	Battery Size (mAh)	Battery Voltage (V)	Battery Power (mWh)
Rechargeable NiMH	2500	7.2	18000
Alkaline	2500	9	22500
Lithium	3000	9	27000

	Component	Current Draw (mA)	Supply Voltage (V)	Power Consumption (mWh)	Rechargeable Runtime (hours)	Alkaline Runtime (hours)	Lithium Runtime (hours)
Idle	**Raspberry Pi 4B**	540	5	2700			
	Total:	540		2700	**7**	**8**	**10**
Running	**Raspberry Pi 4B**	670	5	3350			
	LiDAR	180	5	900			
	Camera	150	5	750			
	USB	50	5	250			
	Total:	1050		5250	**3**	**4**	**5**
Moving	**W/Motors**	500	5	2500			
	Total:	1550		7750	**2.3**	**2.9**	**3.5**

Discussion

Challenges:

- Ensuring reliable data transmission over I2C- false encoder readings.
- Raspberry Pi camera V3 incompatibility with Ubuntu 20.04.

Limitations:

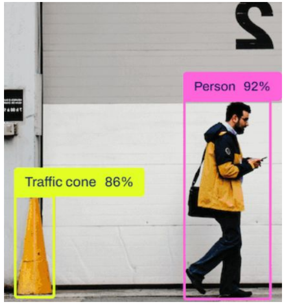
- **Processing:**
  - Raspberry Pi has limitations in handling high-computation tasks simultaneously.
  - Tasks may be offloaded to a PC for efficient operation.
- **Physical Constraints:**
  - Size and weight restrictions for integrating additional components with the Romi.
- **Power:**
  - Limited by 6x AA batteries, can use 40 pin extension header to power from USB-C.

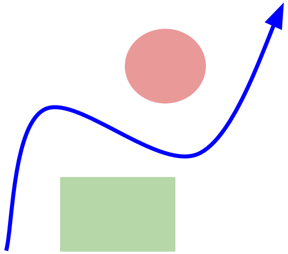
Conclusion

1. Successfully implemented a ROS platform on the Pololu Romi robot.
2. Developed a reliable and accurate ROS Control stack.
3. Achieved seamless integration of multiple sensors (LiDAR, camera) for sophisticated environmental perception.
4. Achieved reasonable power consumption, further extended if not all sensors used.
5. Comprehensive educational tool for in-depth learning and practical application of ROS.

Future work:

- Vision processing (YOLOv8)
- Machine learning techniques
- Path planning
- Lab exercises for project-based learning





Demo & Questions