Robot Intention Communication
(Software Development Project)

Presented by:
Eduardo Cervantes
Priteshkumar Gohil

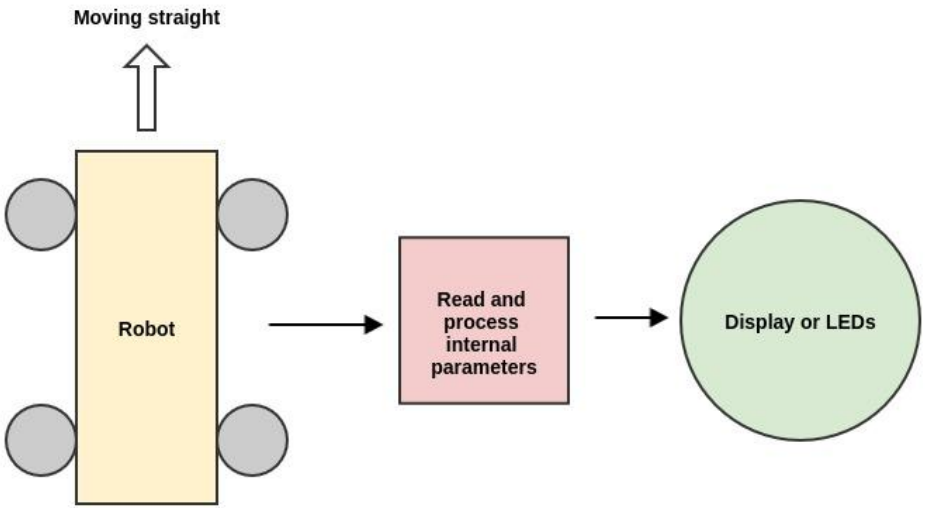
Guided by:
Santosh Thoduka

Objective

- To visualize the movement and special actions of the ROPOD (RObotic POD).

Introduction

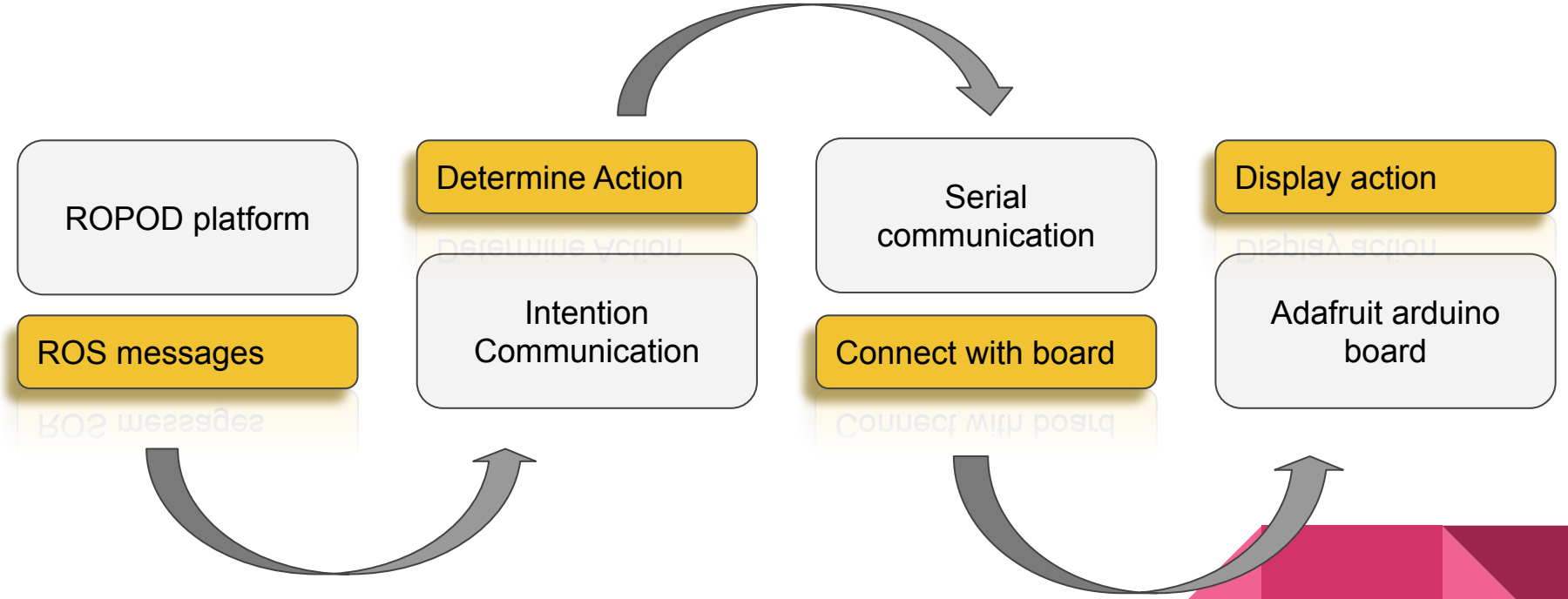
- Intention can be defined as ‘a determination to act in a certain way’ [1].
- The aim of this project is to inform the pedestrians about the next action performed by the Ropod platform [2].
- As a practical example, Ropod is moving straight and will show his direction through the use of leds from the Circuit Playground.



Problem

- Robot knows where to go now and the next, but it is not expressed to those interacting with it.
- Without communicating intention of the robot, It might cause conflict if robot and human approaching each other from the opposite side.
- It is safer for pedestrians if they know the next trajectory of the robot, avoiding in advance a location conflict between them.
- Users are not informed when the ROPOD is executing complex maneuvers. For example, while docking to the cart it is necessary not to be around.

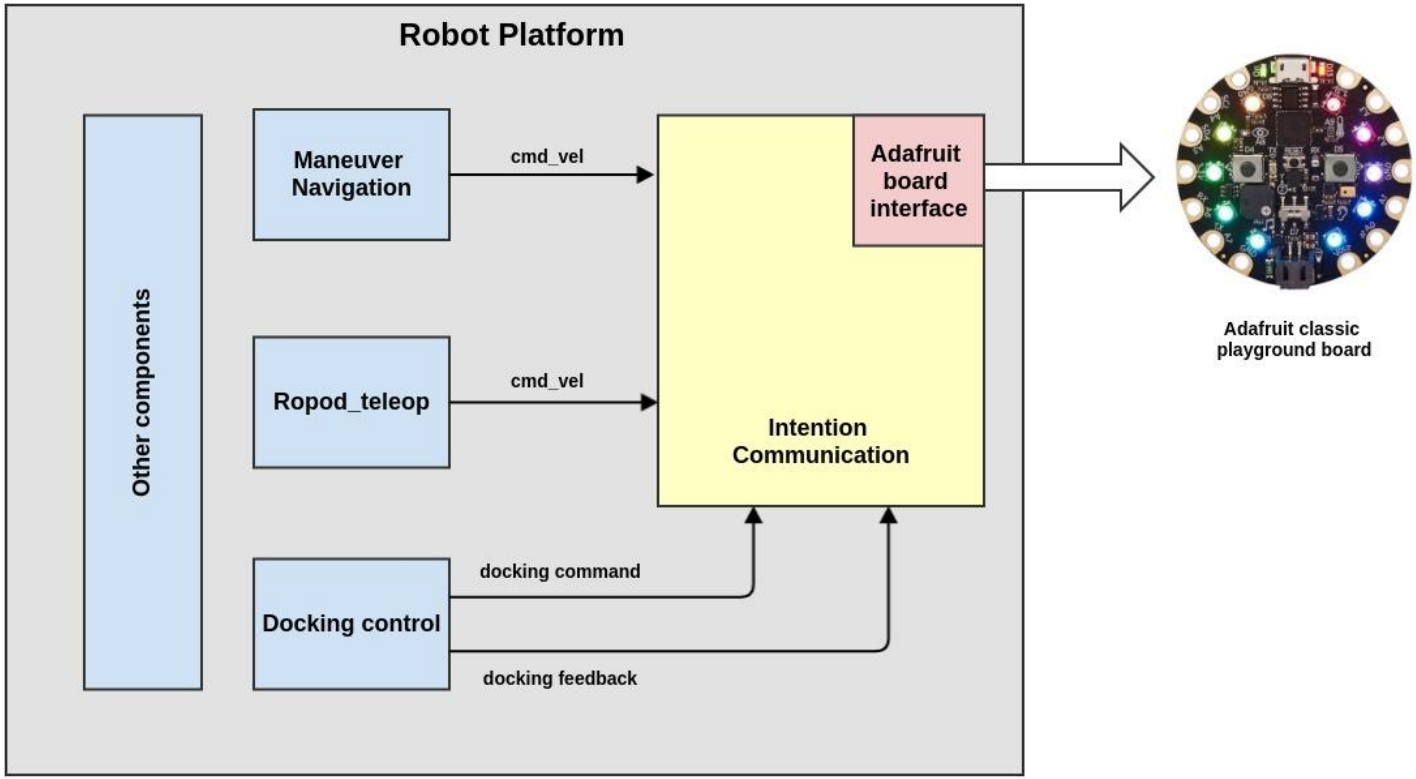
Solution Approach: Process diagram



Solution Approach: Description

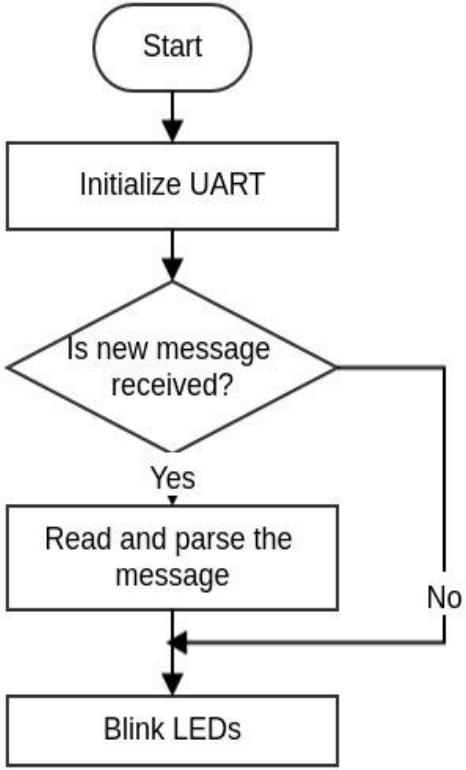
- Accessing published messages:
  - cmd_vel
  - Docking command
  - Docking feedback.
- Our module will determine the desired action of the robot and create a message framework for serial communication.
- Message framework is sent to serial communication module.
- Adafruit board interface will decode the information received from the robot and send it to our external interface through serial port.
- Finally, arduino board will parse received string message, unpack into the data structure and display robot’s action.

Solution Approach: Block diagram

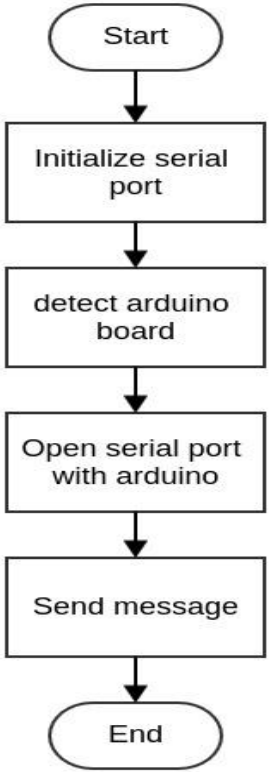


Adafruit classic playground board

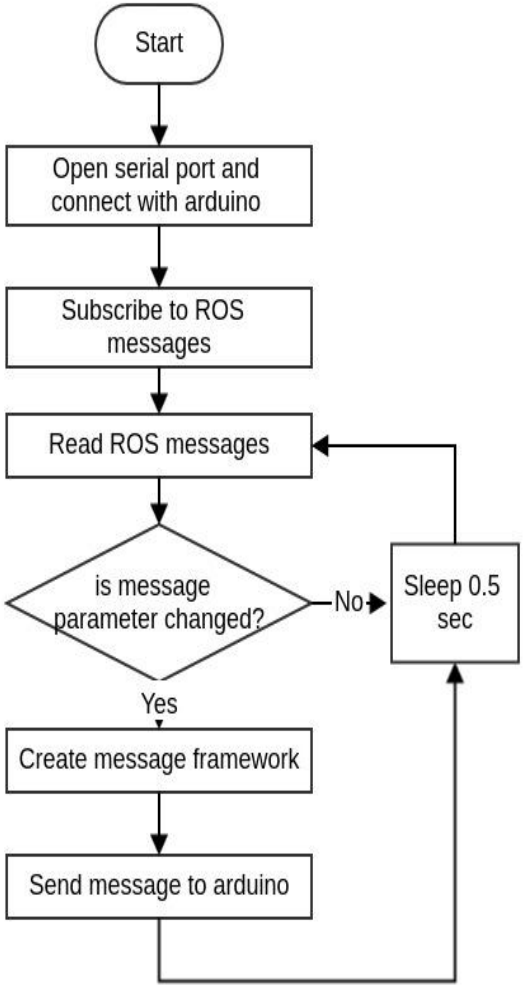
Solution Approach: Flowchart



Arduino board

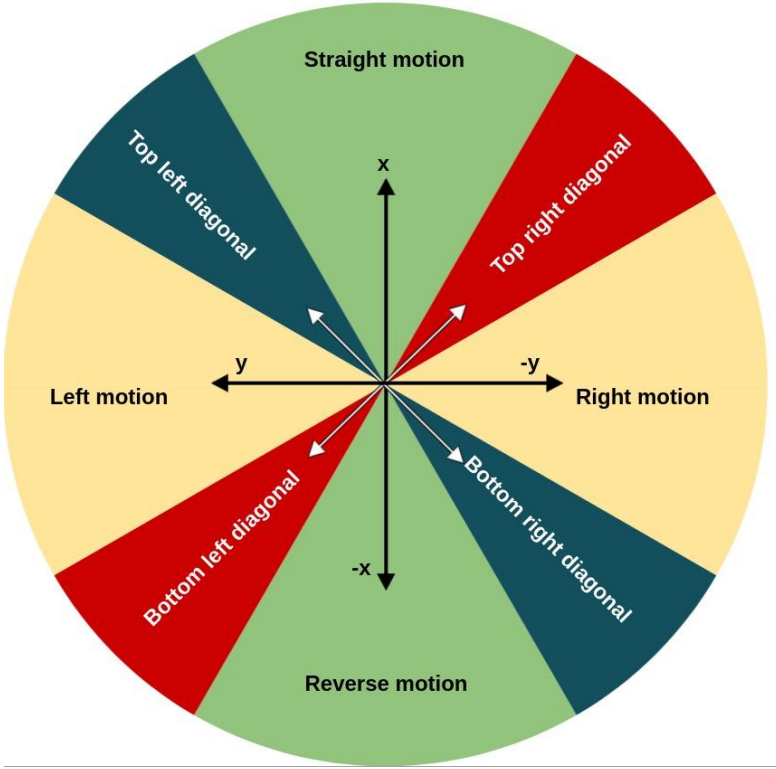


Adafruit interface

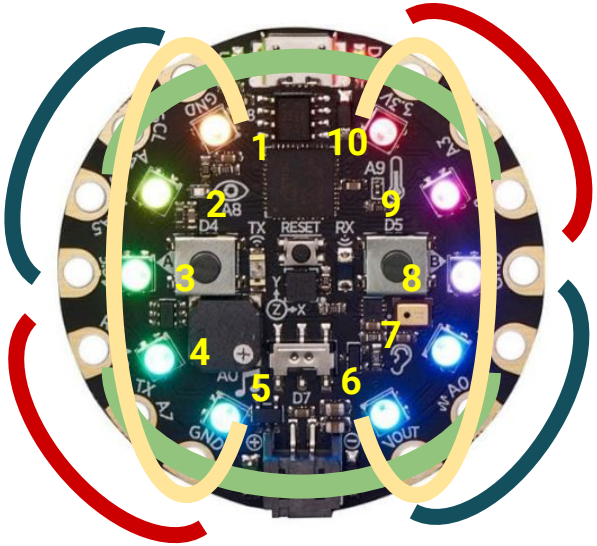


Intention Communication

Implementation: Visual indicators logic.

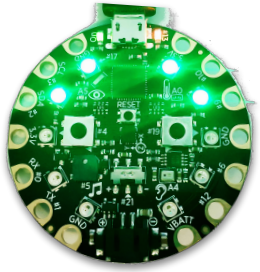
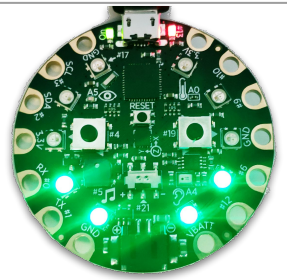
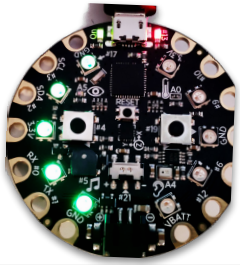
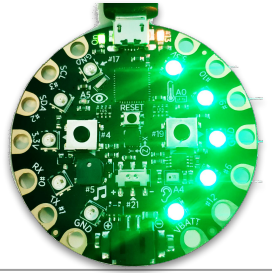
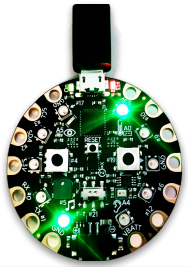
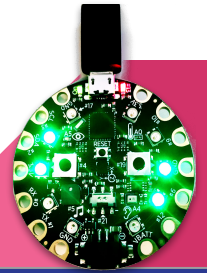


Logic to identify robot's motion

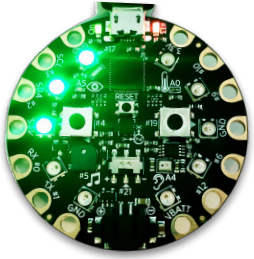
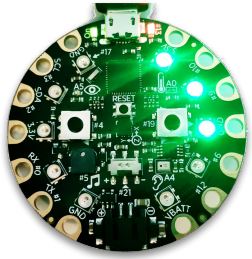
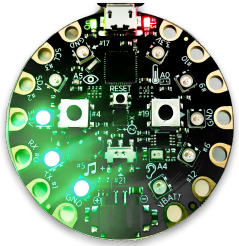
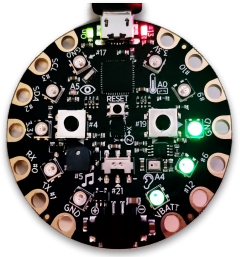


LED group to display motion

Implementation: Visual indications

Straight		**Reverse**	
Left		**Right**	
Clockwise Rotation		**Anticlockwise Rotation**	

Implementation: Visual indications

Top left diagonal		**Top right diagonal**	
Bottom left diagonal		**Bottom right diagonal**	

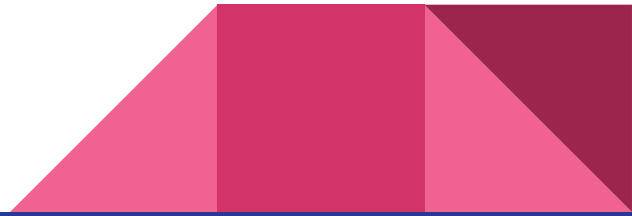
Implementation: Message framework

<RIC##LED:led_number:led_colour;PAT:blinking_pattern;BUZ:timein_ms:frequency:tone_pattern#RIC>

- Three Groups: LED, PAT, BUZ
- Group 1: Word to identify parameters related to LEDs
  - Led_number (2 byte) - Hexadecimal representation of the desired leds to be turned on
  - led_colour (3 byte)- Hexadecimal representation of the desired colour with format RRGGBB Red, Green, Blue
- Group 2: Word to identify parameters related to Pattern
  - blinking_pattern (2 byte) - Desired pattern to be executed by the Circuit Playground

Implementation: Message framework

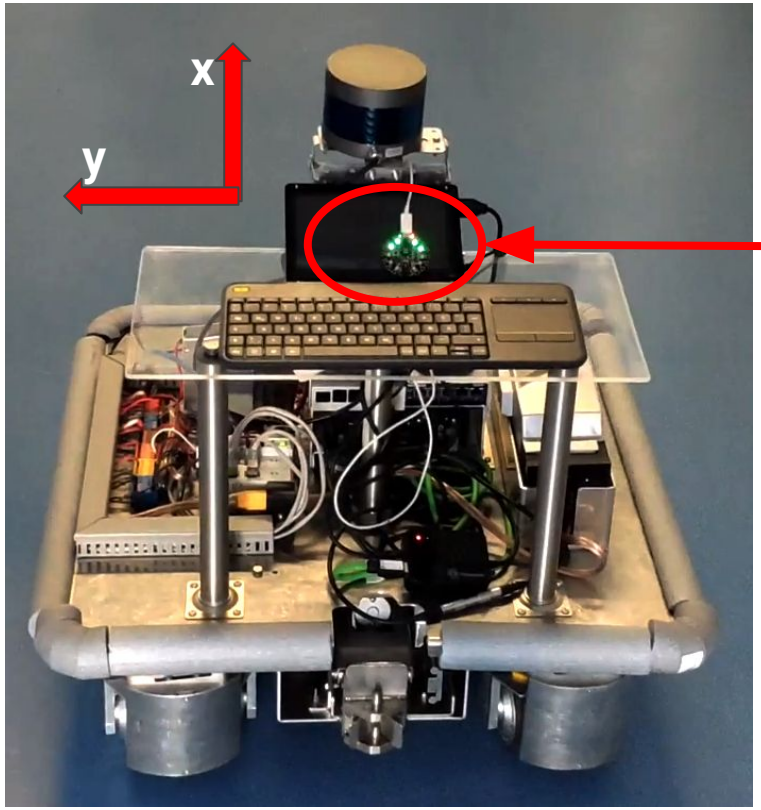
- Group 3: Word to identify parameters related to Buzzer
  - timein_ms (2 byte) : time period to beep buzzer.
  - frequency (1 byte) : frequency of the tone
  - tone_pattern (2 byte): pattern for sound tone



Approach 1 vs Approach 2

Approach 1 (Fixed LED pattern)	Approach 2 (Dynamic LED pattern)
LEDs to blink is decided in arduino board	LEDs to blink is decided from Ropod platform
+ Many different pattern choice	**-** Need to use only limited pattern
- Need to remember pattern code associated with each action.	**+** No need to remember pattern
- Change in requirement leads to change in Arduino board and files in ropod platform	**+** Minimal or almost no change in arduino board

Integration with the Ropod platform



Adafruit board

Problems occurred and solved. (1/ 2)

1. Connection to Circuit Playground was causing problems with the serial port used for communicate.
    a. Solved creating udev rule that gave us the specific port that should be used to connect.
2. Circuit Playground serial communication was crashing when too many messages were sent over serial port.
    a. Solved flushing the stored data in the buffer before send a new command.
3. First implementation was based entirely on the addition of predefined functions on the core algorithm.
    a. Solved implementing a dynamic version that only looks for the parameters being received through published topics.

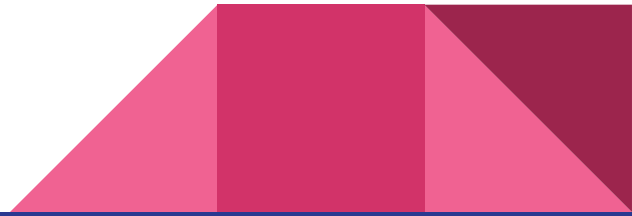
Problems occurred and solved. (2/2)

4. Ropod integrations weeks limited our working hours.
    a. A simulation was developed to run locally in any computer and stop dependency of the robot platform.
    b. This helped us to test and debug our implementation much faster.
5. Ropod_teleop works for specific joypads.
    a. Original code was modified to accept commands send through a ps4 controller.
6. PS4 controller, has limitations over Ubuntu.
    a. Limitations were solved and a script was provided for outstanding end-user experience.
7. IdProduct and Id Vendor are the same for all Circuit Playground boards.
    a. Use of udev rules to fix the communication to a specific port.

Summary

Happy moments / frustrations

1. The first time we were able to send commands from a joypad to remotely control our Circuit Playground.
2. As soon as we finished our debugging process.
3. After proving that our implementation could be integrated in any robot
4. When we tried understanding ropod navigation code.
5. Arduino IDE is not like other IDEs. It is not providing suggestions or autofill to developer.

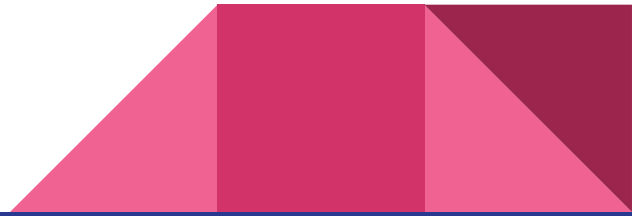


Demo (Teleoperated control)

Demo (Autonomous drive)

Future work

1. Implement additional sound indication.
2. Access small navigation plans to decide on robot's motion during autonomous navigation.



Software used

- ROS (Robot Operating system) is used to communicate with the robot. ROS Kinetic version was used in this project.
  Online link: http://wiki.ros.org/kinetic
- PySerial module was used for the serial communication between the Classic Playground board and the robot PC.
  Online link: https://pythonhosted.org/pyserial/
- Arduino IDE is used for development of code. Code was written in C language.
  Online link: https://www.arduino.cc/en/main/software

Repository link and dependencies

https://github.com/HBRS-SDP/sdp_ss2019_P6_IntentionComm

https://git.ropod.org/eduardocd/ropod_teleop

https://git.ropod.org/ropod/communication/ropod_ros_msgs

References

[1] https://www.merriam-webster.com/dictionary/intention

[2] https://www.h-brs.de/en/ropod

[3] http://wiki.ros.org/

[4] https://learn.adafruit.com/introducing-circuit-playground?view=all

[5] https://cdn-learn.adafruit.com/downloads/pdf/introducing-circuit-playground.pdf

Thank You