Connecting 2 raspberry Pis choice of LAN

Connecting 2 raspberry Pis choice of LAN

* Distance and bandwidth issue

The first issue we noticed when looking for a means of communication was the lack of range on most radio communications. Also, the modules that could communicate long range (LoRa for example) did not have a very high bandwidth and could not communicate large enough data reliably for a live FPV camera to work. Upon further research we found that LAN communication via a network would work for the longest range, as well as a higher bandwidth(this is also what the Rover Capstone team does).

Connecting 2 Raspberry Pi’s via LAN:

* Manually setting ipv4

The pis were able to communicate a short ping to each other and nothing else. We discovered that we must first

* Turning DHCP off
* Firewall

Installing ROS 2

* No problems encountered, here is a link to the installation process that worked for us, (First you must have the Ubuntu (24.04.2 lts OS installed) (we installed the Jazzy version of

ROS 2 Camera feed

ROS 2 controller communicating python

To start the joystick controller package on command pi

* source /opt/ros/jazzy/setup.bash
* cd ros2\_ws
* ./run\_joystick.sh

To start rviz2

* Note: The second pi needs to be running to see /image\_raw, so do the ssh first
* Open a new terminal (Just a new tab located in the top left corner of the terminal screen)
* source /opt/ros/jazzy/setup.bash
* rviz2
* When on you want to add an image and change the topic to /image\_raw
* To add the image, the bottom left corner has a “add” button, then find the image in by display type and select ok.
* After this find “image” in the left box called display.
* Hit the down arrow on image and select the right side of the display box next to “topic”
* Then type /image\_raw and select
* Make the image box bigger to see more :)

To ssh into the second pi and start the package

* Open a new terminal
* ssh spacecraft@192.168.0.2
* Enter in “spacecraft” for password
* source /opt/ros/jazzy/setup.bash
* When it says v4l2 below the l is a lowercase L
* export GSCAM\_CONFIG="v4l2src device=/dev/video0 ! video/x-raw, format=YUY2 ! videoconvert ! video/x-raw, format=BGR ! appsink"
* cd ros2\_ws
* colcon build --packages-select multi\_launch
* source install/setup.bash
* ros2 launch multi\_launch multi\_launch.py

Tab 2

# **Connecting 2 Raspberry Pis via LAN**

## **Initial Setup**

### **Software**

* Installed Ubuntu (24.04.2 LTS) on both Raspberry Pis
* Chosen for compatibility with ROS 2 (Robot Operating System)

### **Hardware**

* Connected both Raspberry Pis using a LAN cable
* LAN cable simulates planned long-distance communication with Ubiquiti Rocket radio modules

## **Initial Communications**

### **Network Configuration**

1. Access network settings on both Raspberry Pis:
   * Settings > Network > (gear on top right of wired) > IPv4
2. Configure IPv4 settings:
   * Set to manual
   * Pi 1 IP address: 192.168.1.10 (example)
   * Gateway: 192.168.1.10 (example)
   * Netmask: 255.255.255.0
3. Note: Manual configuration automatically disables DHCP (desired behavior)

### **Testing Connection**

* Ping command: ping 192.168.1.11 (use the IP address of the other Pi)
* Successful ping confirms basic connectivity

### **Firewall Adjustment**

* Issue: Unable to send data beyond ping
* Cause: Firewall enabled on both Pis
* Solution: Disable firewall
  + Command: sudo ufw disable
  + Restart both Pis for changes to take effect

Tab 4

## **Wired Controller Communication via ROS**

### **Creating ROS Package**

* Command: ros2 pkg create <package\_name> --build-type ament\_python
* Navigate to package directory (e.g., joystick\_drive/joystick\_drive)

### **Python Script for Joystick Node**

* Initial setup: Configured to receive keyboard inputs (generated via ChatGPT)
* Upgrade: Implemented Xbox controller for easier control
* Note: Ensure subscriber uses "joy" package (ChatGPT may suggest incorrect package)
* Verify setup.py (e.g., ros2\_ws/src/motor\_controller/setup.py):
  + Confirm entry\_points includes motor\_node for both ends

### **Running the Code**

1. Navigate to workspace: cd ~/ros2\_ws
2. Build package: colcon build --packages-select joystick\_drive
3. Source setup file: source install/setup.bash
4. Run node: ros2 run joystick\_drive joystick\_node

### **Bash Script Automation**

* Created run\_joystick.sh containing the above 4 commands
* Execute: ./run\_joystick.sh

### **Testing Controller Inputs**

* Command: jstest /dev/input/js0

## **Setting Up a GUI (Tkinter)**

* Place main GUI execution function within main() function
* Benefit: GUI simplifies testing with keyboard inputs