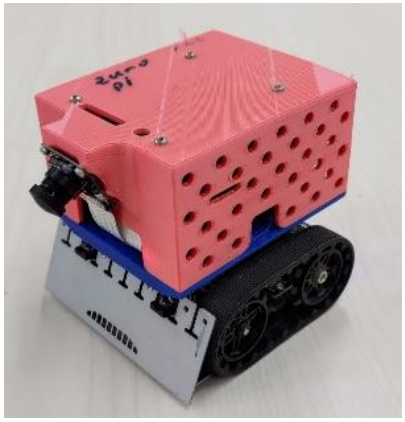
ZumoPi Telemetry System User Manual



Contents

System Activation 2
GUI User Guide 2
Sensor sampling settings 3
Camera 3
Battery 5
Charging 5
Buzzer 5
Encoders 6
Line Sensors 6
IMU 6
Proximity Sensors 7
Joystick 8
WIFI Stats 8
Applications Status 9
Logger 10
Simulator 12

System Activation

1) Arduino Code Upload

Open terminal on ZumoPi robot. Type **zardrun** and press enter.
Wait until compilation and uploading of Arduino code to ZumoPi is finished.

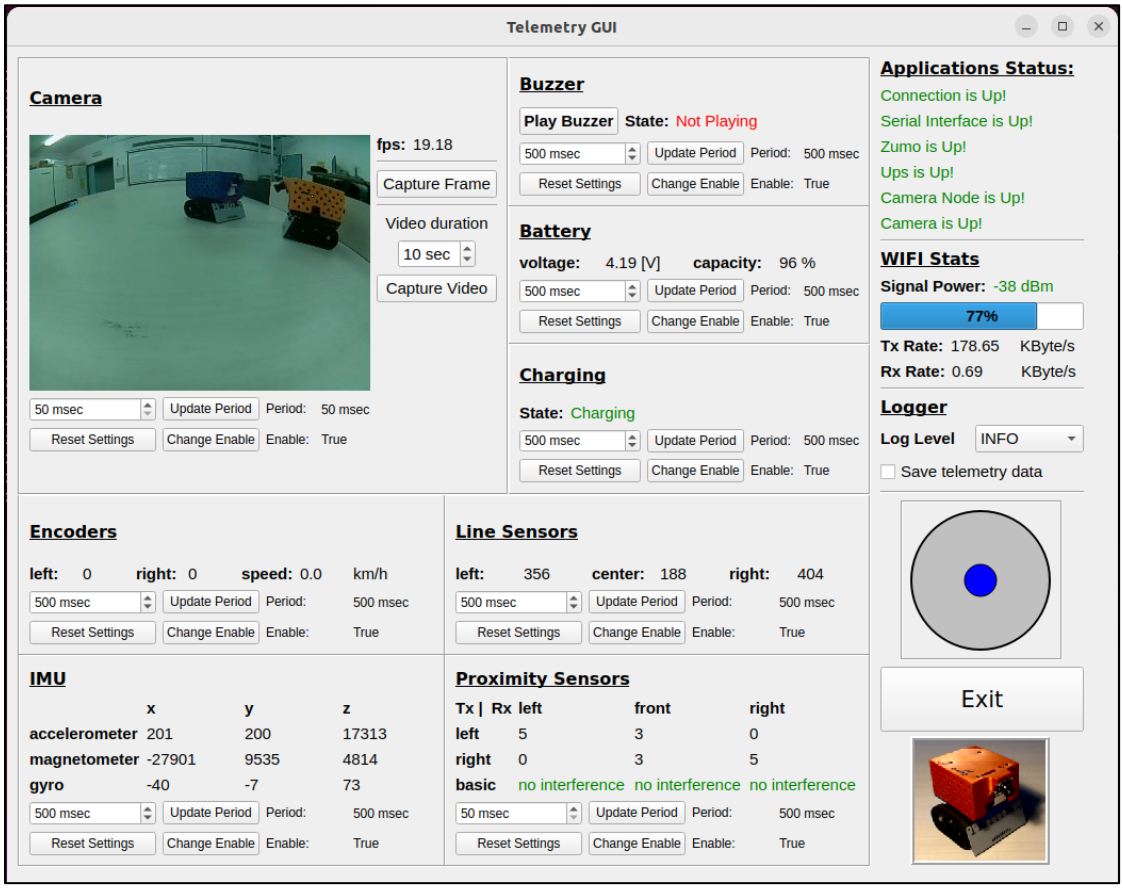
2) Launch ROS2 Nodes

On the ZumoPi's terminal, type **tel** and press enter.
This will start all the telemetry nodes on ZumoPi.

3) Start GUI Node

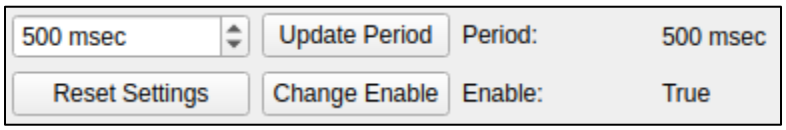
Open terminal on the remote computer. Type **gui** and press enter.
This will start the GUI of the telemetry system.

GUI User Guide



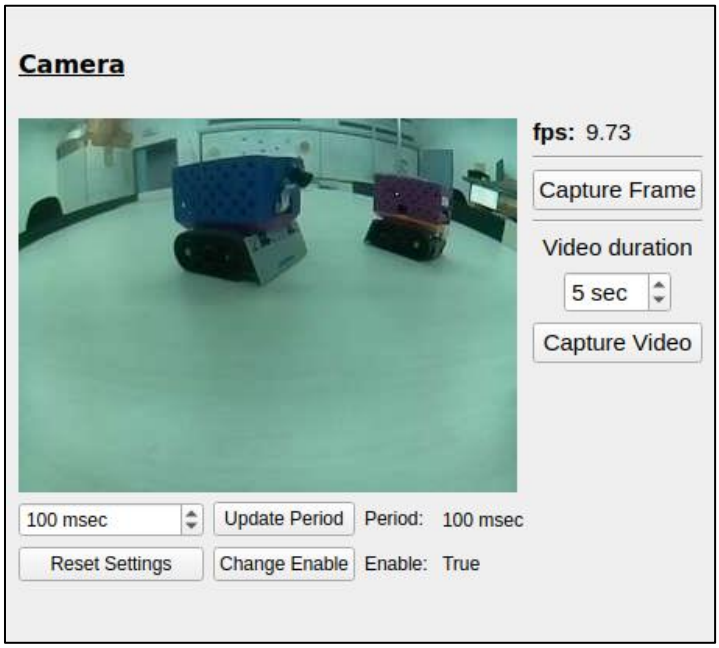
Sensor sampling settings

Every sensor has the following panel:



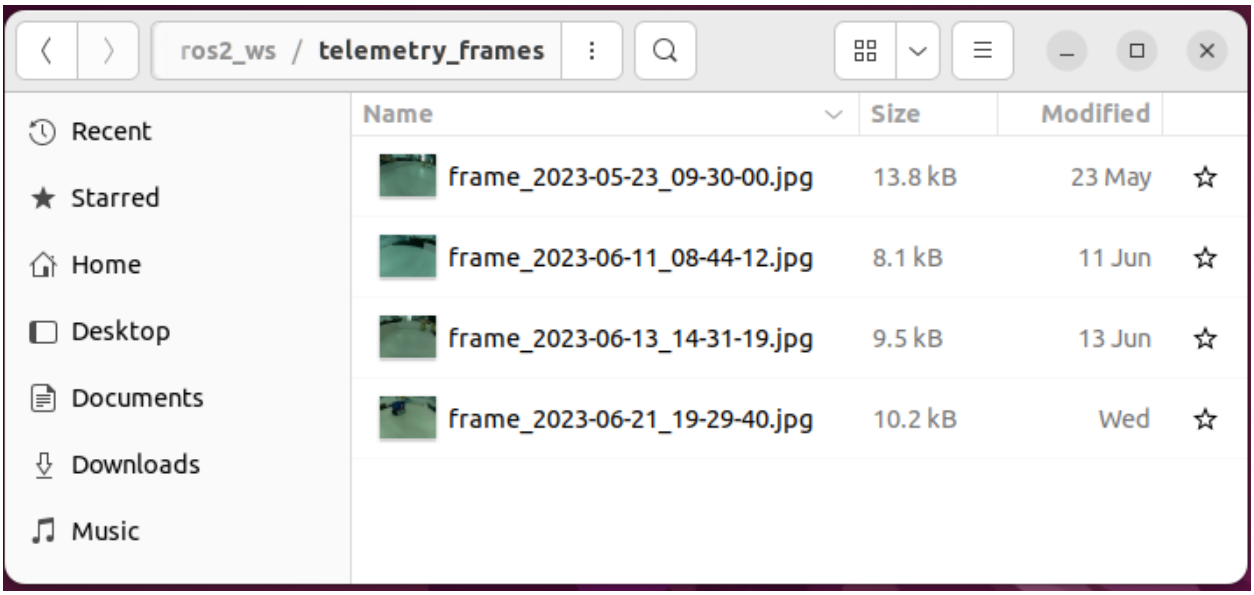
- On the right side of the panel, **Period** and **Enable** indicate the current settings on the ZumoPi robot
- Change the sampling period by entering a new value to the spin-box and pressing the **Update Period** button
- Enable/Disable the sampling from the sensor by pressing the **Change Enable** button
  Note: Disabled sensors will not show data.
- Reset the settings by pressing the **Reset Settings** button. This sets the sampling period to 500ms (except for the camera, which resets to 100ms) and enables the sampling

Camera

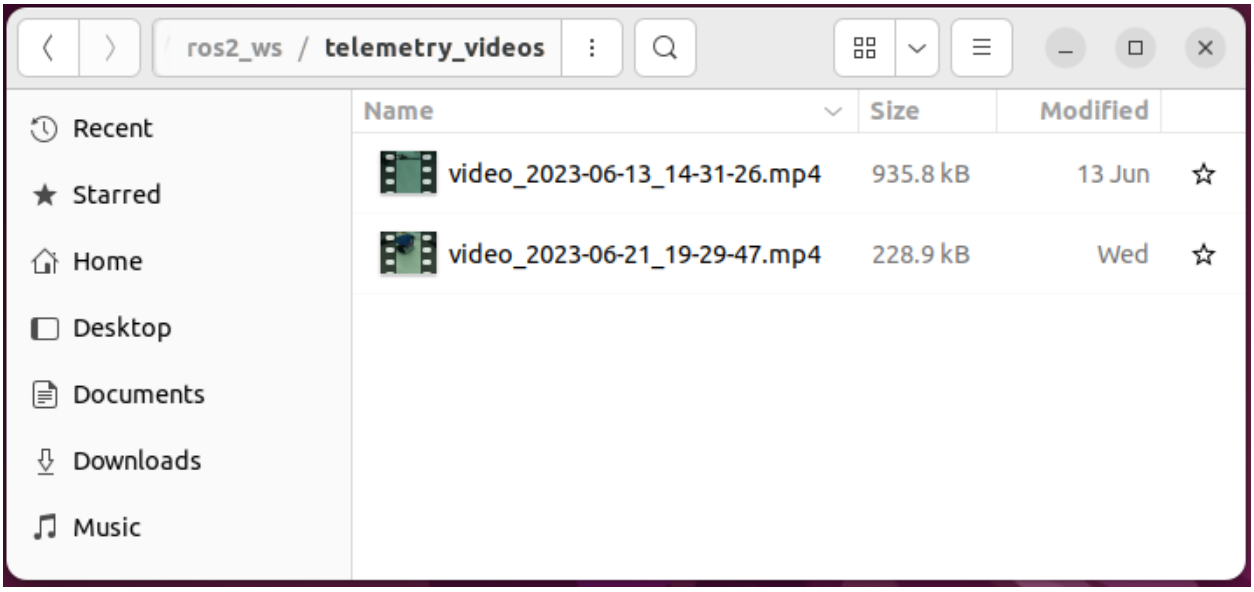


- **fps** indicates the frames per second received by the GUI node

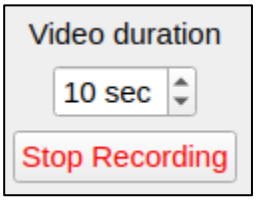
- **Capture Frame** button captures the current frame in the video and stores it as an image in ~/ros2_ws/telemetry_frames



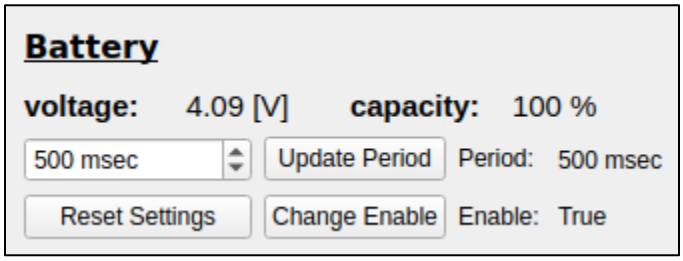
- **Capture Video** button captures a video for the specified duration and stores it in ~/ros2_ws/telemetry_videos



Change the duration of the video by using the spin-box.
Stop the recording at any time by pressing the **Stop Recording** button, which appears during the video capture process.

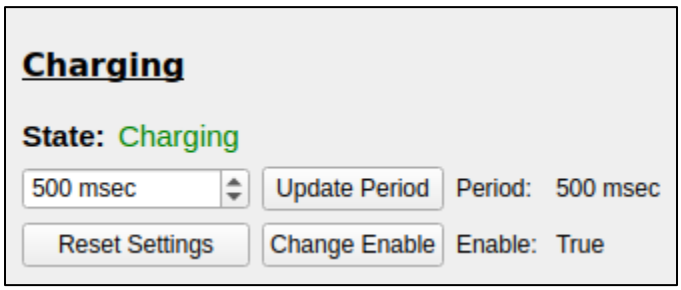


Battery



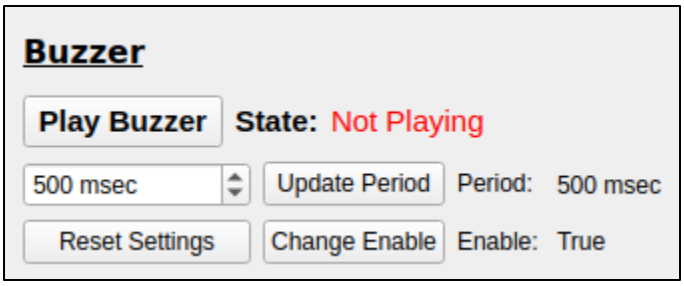
- **voltage** indicates the current voltage supplied by the battery
- **capacity** indicates the current battery capacity

Charging



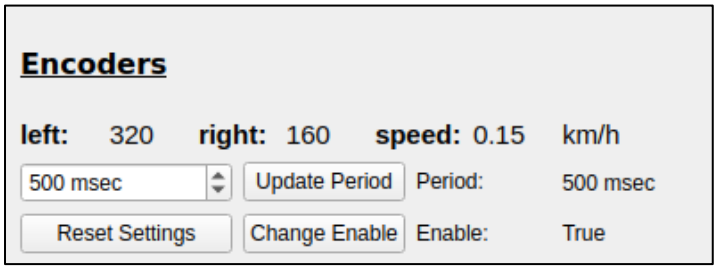
- **State** indicates whether the robot is currently charging or not

Buzzer



- **State** indicates whether the buzzer is currently playing or not
- **Play Buzzer** button plays the buzzer for 5 seconds
  Stop it at any time by pressing the **Stop Buzzer** button which appears when the buzzer plays.

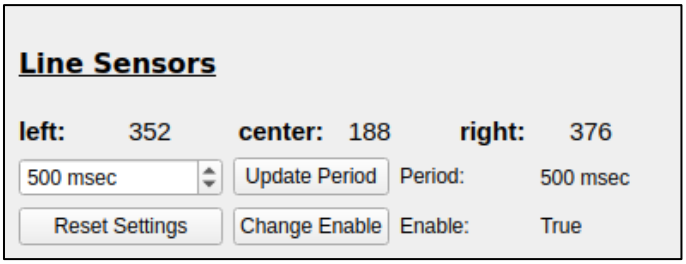
Encoders



- **left** shows the reading from the left encoder
- **right** shows the reading from the right encoder
- **speed** shows the robot's speed in km/h

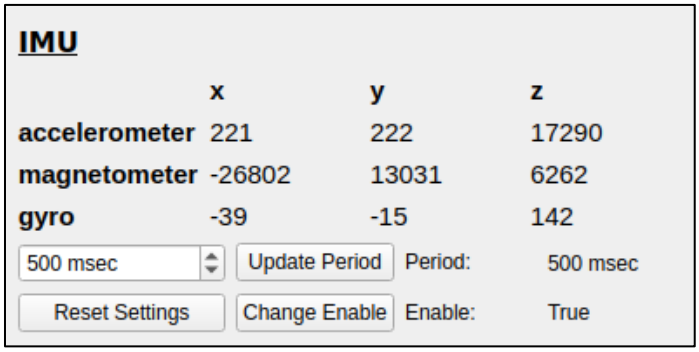
Note: errors reading from the encoders are indicated as follows:

Line Sensors



- **left** indicates the reading from the left line sensor
- **center** indicates the reading from the center line sensor
- **right** indicates the reading from the right line sensor

IMU



- **accelerometer** indicates the accelerometer readings in the **x, y, z** axes
- **magnetometer** indicates the magnetometer readings in the **x, y, z** axes
- **gyro** indicates the gyroscope readings in the **x, y, z** axes

Note: errors of IMU initialization are indicated as follows:

	x	y	z
accelerometer	ERROR INIT	ERROR INIT	ERROR INIT
magnetometer	ERROR INIT	ERROR INIT	ERROR INIT
gyro	ERROR INIT	ERROR INIT	ERROR INIT

Errors of IMU reading are indicated as follows

	x	y	z
accelerometer	ERROR	ERROR	ERROR
magnetometer	ERROR	ERROR	ERROR
gyro	ERROR	ERROR	ERROR

Proximity Sensors

Proximity Sensors

Tx | Rx	left	front	right
left	5	2	0
right	0	1	5
basic	no interference	no interference	no interference

500 msec | Update Period | Period: 500 msec

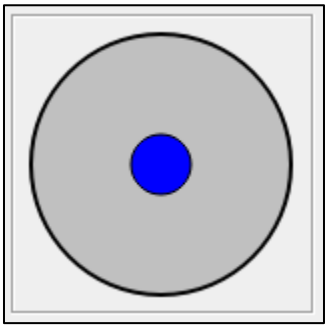
Reset Settings | Change Enable | Enable: True

- **Rx left** indicates the reading of the left proximity sensor, as a result of light emitted from the **Tx left** and **Tx right** LEDs
- **Rx front** indicates the reading of the front proximity sensor, as a result of light emitted from the **Tx left** and **Tx right** LEDs
- **Rx right** indicates the reading of the right proximity sensor, as a result of light emitted from the **Tx left** and **Tx right** LEDs
- **basic** indicates whether the proximity sensors sense an external interference (caused by external LEDs)

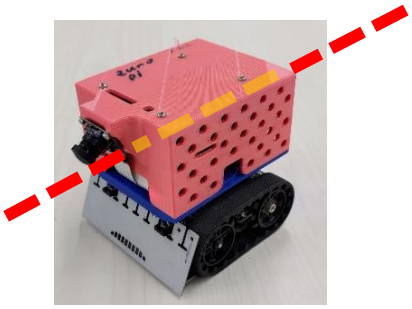
basic	no interference	no interference	interference!

Joystick

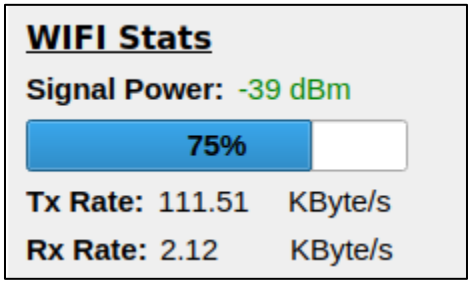
- The joystick allows to control the robot, by moving the blue circle



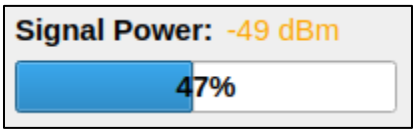
- The center of the gray circle is relative to the robot's center (from top view) and the location of the blue circle indicates the speed and direction of the robot's movement relative to its back-front axis:

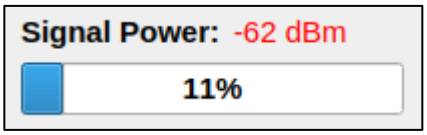


WIFI Stats



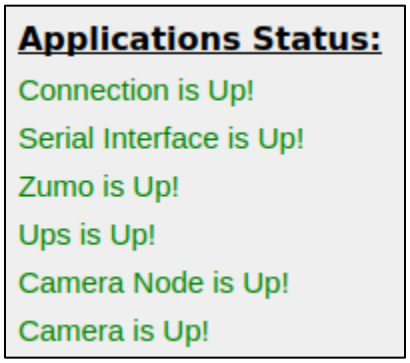
- **Signal Power** indicates the strength of the signal received by the robot, in dBm units. Its color changes to yellow for medium signal strength, and red for low signal strength
- The colored bar indicates the signal strength in a relative manner. 100% indicates a very good signal reception while 0% indicates a very bad signal reception.





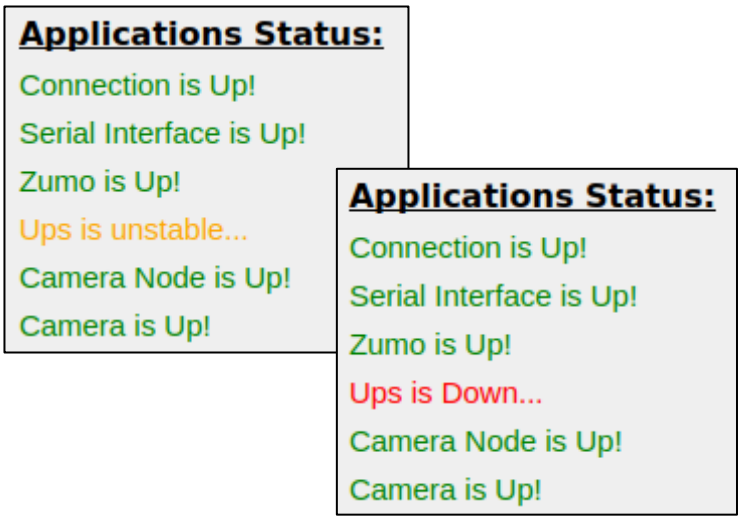
- **Tx Rate** indicates the data transmission rate of the robot in kBytes/s
- **Rx Rate** indicates the data reception rate of the robot in kBytes/s

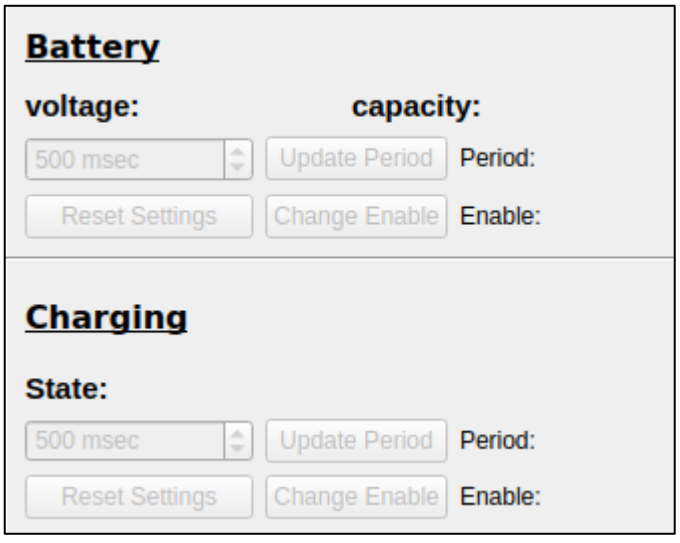
Applications Status



- **Connection** indicates that WIFI Stats node is running. If not, the GUI is blocked.
- **Serial Interface** indicates that Serial Interface node is running. If not, the following panels are blocked: Buzzer, Encoders, Line Sensors, IMU, Proximity Sensors, and Joystick.
- **Zumo** indicates that the telemetry Arduino code is running. If not, the following panels are blocked: Buzzer, Encoders, Line Sensors, IMU, Proximity Sensors, and Joystick.
- **UPS** indicates that UPS Reading node is running. If not, Battery and Charging panels are blocked.
- **Camera Node** indicates that Camera node is running. If not, camera panel is blocked.
- **Camera** indicates the connection with the robot's camera. If there's no connection, camera panel is blocked.

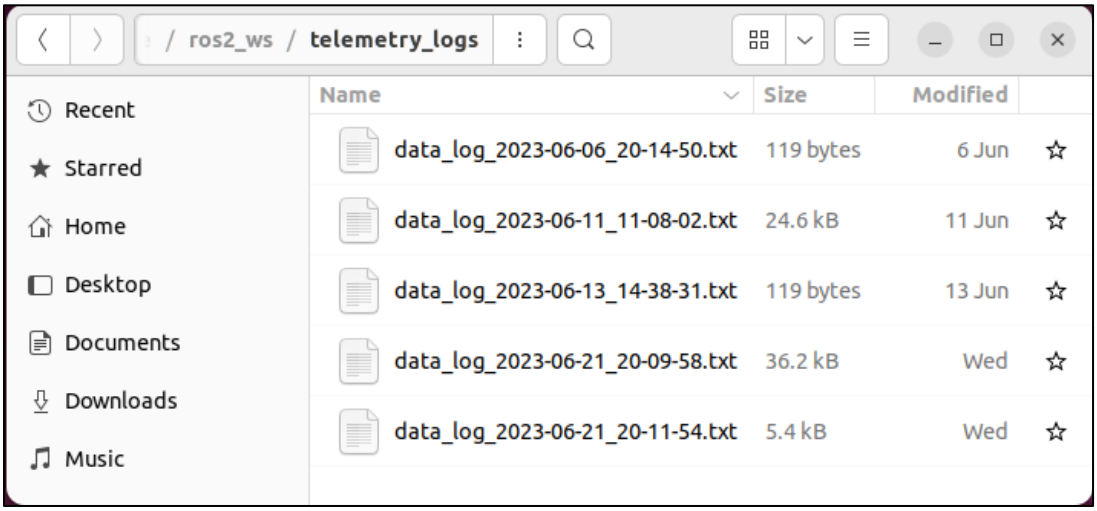
Note: When a problem is detected, the GUI declares a module as unstable. If it's not resolved after 6 seconds, the module is down and the GUI blocks the relevant panels.
Example: UPS Reading node stops running

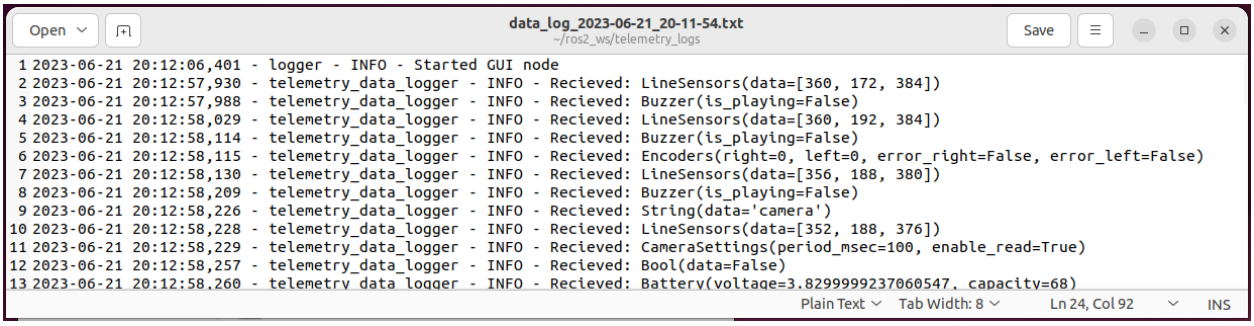




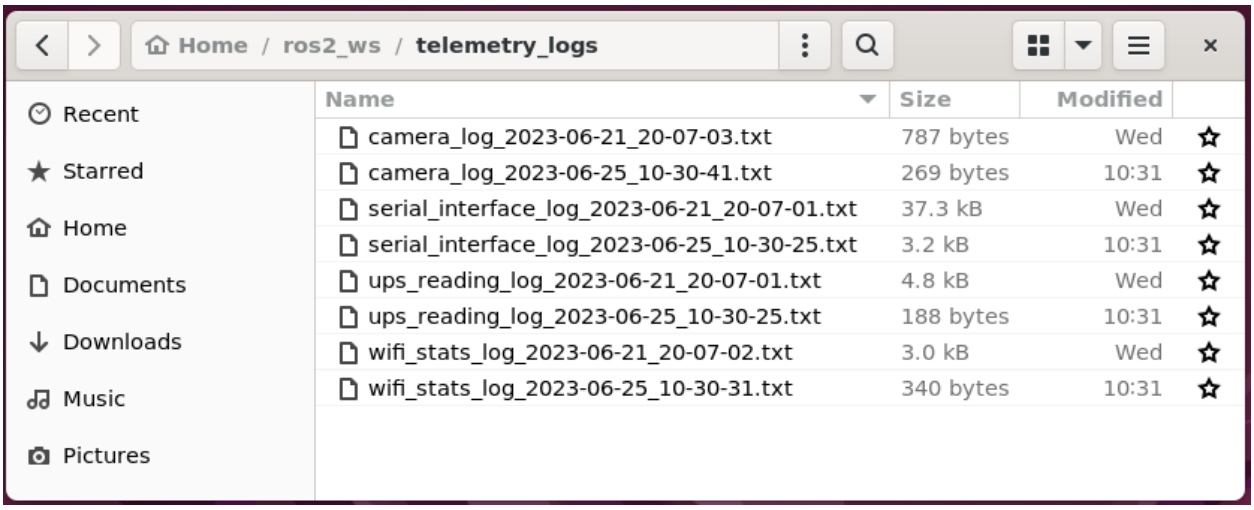
Logger

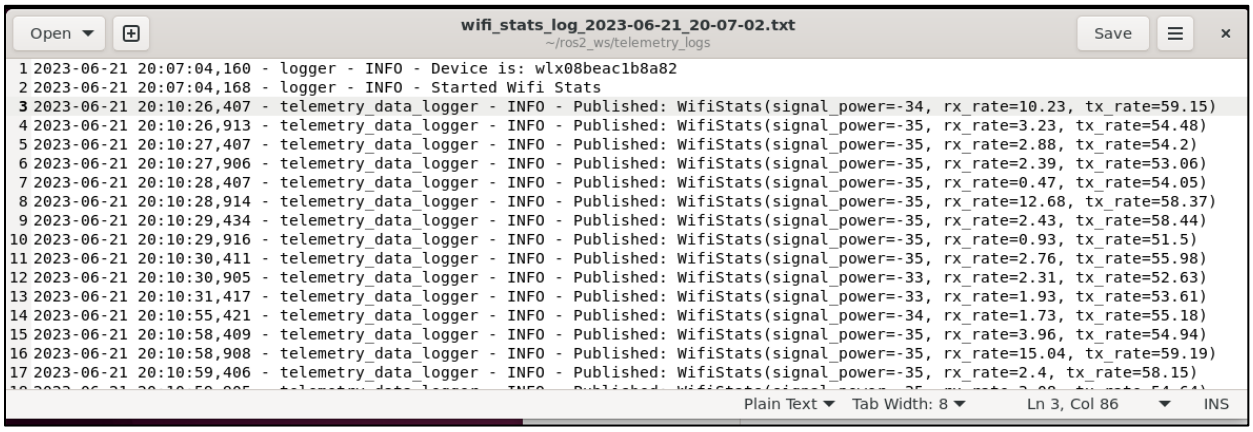
- GUI node creates a log file on the remote computer, in ~/ros2_ws/telemetry_logs, called data_log. It shows the logs triggered by the GUI code, as well as telemetry data received and published by the GUI



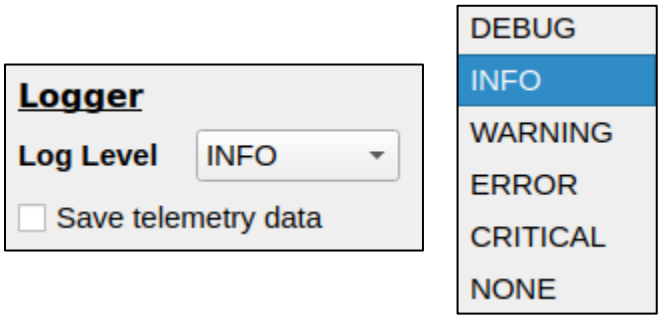


- Each node running on ZumoPi creates a log file locally on the robot, in ~/ros2_ws/telemetry_logs, according to node's name (e.g., wifi_stats_log). It shows the logs triggered by the node, as well as telemetry data received and published by the node.





- Set the **Log Level** using the dropdown list
  The logs can be viewed either in the terminal or in the log files, and indicated as "logger"



- Check **Save telemetry data** to store the telemetry data published and received by the nodes. This data doesn't appear in the console (to avoid spamming), and is always of INFO severity. Indicated as "telemetry_data_logger".
  Uncheck to stop saving the data.

Note: All the logs are initialized to INFO and don't store telemetry data

Simulator

The simulator simulates the ZumoPi robot's behavior. It generates telemetry data and can receive commands from the GUI just like the real robot.
It's useful when you want to test the GUI but have no access to the robot (e.g., it doesn't work, it doesn't exist).
In addition, it can generate possible edge case scenarios like errors. These don't usually happen on the real robot, so this feature can be used to test the GUI's operation when it receives such errors.

Run the simulator:

1) Make sure that **tel** is not running (see system activation)
2) On the remote computer (the same computer that runs the GUI) open a terminal, type **data**, and press enter