Mapping PS2 Controller Buttons to LED and Buzzer Functions

This Jupyter Notebook guides you through mapping PS2 controller buttons to control LED and buzzer functions on a Raspberry Pi.

Step 1: Setting Up the Environment

Ensure you have connected your PS2 controller and the necessary libraries installed.

Run the following command in the terminal to start Jupyter Notebook:

```
cd RPI-Demos
code
```
Now, proceed with the notebook to test the mappings.

Step 2: Import Required Libraries

First, import the necessary modules for joystick control, LEDs, and buzzer functions.

In [ ]:
```python
# Import necessary libraries
import time
import random
from joystick_control import JoystickController  # Ensure this file exists a
import rclpy
from omni_robot_controller import OmniWheelControlNode  # Ensure this matche
from image_capture import ImageCaptureNode

# Initialize joystick, LED, and buzzer controllers
rclpy.init()
node = OmniWheelControlNode()  # Initialize the robot control node
image_node = ImageCaptureNode()
joystick = JoystickController()  # Initialize joystick control

print("Joystick, LED, and Buzzer initialized.")
```

Step 3: Define LED and Buzzer Functions

These functions will be mapped to the PS2 controller buttons.

```
In [ ]:
# Function to turn LED red
def turn_on_led():
    node.set_color(1, 255, 0, 0)  # LED 1 Red
    print("LED set to Red")

# Function to turn off LED
def turn_off_led():
    node.set_color(1, 0, 0, 0)  # Off
    print("LED turned off")

# Function to activate the buzzer
def activate_buzzer():
    node.play_buzzer(1000, 2.0, 1.0, 1)  # 1000Hz for 2 seconds
    print("Buzzer activated")

# Function to change buzzer tone
def change_buzzer_tone():
    node.play_buzzer(800, 1.5, 1.0, 1)  # 800Hz for 1.5 seconds
    print("Buzzer tone changed")

# Function to change LED to a random color
def random_led_color():
    r, g, b = random.randint(0, 255), random.randint(0, 255), random.randint
    node.set_color(1, r, g, b)
    print(f"LED set to random color: {r}, {g}, {b}")
```

Step 4: Map Buttons to Functions

Use the `map_button` method to associate buttons with specific functions.

```
In [ ]:
# Map controller buttons to functions
joystick.map_button("cross", turn_on_led)
joystick.map_button("triangle", turn_off_led)
joystick.map_button("square", activate_buzzer)
joystick.map_button("circle", change_buzzer_tone)
joystick.map_button("r3", random_led_color)

print("Button mappings set.")
```

Step 5: Run the Joystick Event Loop

Start listening for button presses and trigger the corresponding functions.

```
In [ ]:
print("Listening for button presses... Press Ctrl+C to stop.")
try:
    joystick.listen()  # This function should listen for button presses and
except KeyboardInterrupt:
    print("Joystick listening stopped.")
```

Step 6: Testing and Debugging

Press the following buttons to test the mappings:

- **Cross** → LED turns red.
- **Triangle** → LED turns off.
- **Square** → Buzzer activates.
- **Circle** → Buzzer changes tone.
- **R3** → LED changes to a random color.

If anything doesn't work, check for errors and restart the script.