PS2 Controller Camera Mapping

Overview

This notebook demonstrates how to use a PS2 controller to:

- Capture images using the `capture_photo()` function.
- Control an RGB LED using `set_color()`.
- Play buzzer sounds using `play_buzzer()`.
- Map controller buttons to trigger these functions.

By the end, you will implement additional features through exercises.

Step 1: Importing Necessary Libraries

We need to import the required libraries for controlling the joystick, robot, and image capture system.

In [ ]:
```
# Import necessary libraries
import time
import random


import rclpy
from joystick_control import JoystickController  # Ensure this file exists a
from omni_robot_controller import OmniWheelControlNode  # Ensure this matche
from image_capture import ImageCaptureNode  # Import image capture controlle
```

Step 2: Initializing the Controller and Nodes

We initialize the **joystick, LED, and camera controller** so they can be used for different tasks.

In [ ]:
```
# Initialize joystick, LED, and camera controller
rclpy.init()
node = OmniWheelControlNode()  # Initialize the robot control node
image_node = ImageCaptureNode()
joystick = JoystickController()  # Initialize joystick control
```

```
print("Joystick and Camera Controller initialized.")
```

Step 3: Controlling an RGB LED

We can control an LED using `set_color(led_id, r, g, b)`, where:

- `led_id = 1` (Refers to the first LED)
- `r, g, b` are values between 0 and 255 (for Red, Green, and Blue).

Example:

The code below sets the LED to **Red**, then **Green**, and then turns it off.

In [ ]:
```
# Set the LED to different colors
print("Setting LED to red...")
node.set_color(1, 255, 0, 0)  # Red
time.sleep(2)

print("Setting LED to green...")
node.set_color(1, 0, 255, 0)  # Green
time.sleep(2)

print("Turning off LED...")
node.set_color(1, 0, 0, 0)  # Turn off LED
```

Step 4: Activating a Buzzer

We can activate a buzzer using `play_buzzer(frequency, duration, amplitude, buzzer_id)`, where:

- `frequency` is in Hz (e.g., 1000 for 1 kHz).
- `duration` is in seconds.
- `amplitude` controls volume (0.0 to 1.0).
- `buzzer_id = 1` (Refers to the first buzzer).

Example:

The code below plays a **1000 Hz sound for 1 second** and an **800 Hz sound for 1.5 seconds**.

In [ ]:
```
# Play a buzzer sound with different tones
print("Playing buzzer at 1000Hz for 1 second...")
node.play_buzzer(1000, 1.0, 1.0, 1)  # 1000Hz for 1 second

print("Playing buzzer at 800Hz for 1.5 seconds...")
node.play_buzzer(800, 1.5, 1.0, 1)  # 800Hz for 1.5 seconds
```

Step 5: Capturing Photos

We capture an image using `capture_photo()`.
To save an image with a timestamp, we update `image_node.save_path` before calling the function.

Example:

The code below captures a photo and saves it with a timestamp.

In [ ]:
```python
# Function to capture a photo
def capture_photo():
    image_node.capture_photo()
    print("Photo captured")

# Function to capture a photo with a timestamp and store it in a specific di
def capture_photo_with_timestamp():
    timestamp = time.strftime("%Y%m%d-%H%M%S")
    file_path = f"/home/pi/captured_images/image_{timestamp}.jpg"
    image_node.save_path = file_path  # Update save path dynamically
    image_node.capture_photo()
    print(f"Photo saved as {file_path}")
```

Step 6: Combining Functions

Now, we can create functions that perform multiple actions, such as **turning on the LED when a photo is captured** or **playing a buzzer sound after taking a picture**.

Example:

The following functions will:

1. Capture a photo and turn the LED green.
2. Capture a photo and activate the buzzer.

In [ ]:
```python
# Function to capture a photo and turn on an RGB LED
def capture_photo_and_led():

# Function to capture a photo and activate the buzzer
def capture_photo_and_buzzer():
```

Step 7: Mapping Controller Buttons

We can map controller buttons to call specific functions.
The following mappings allow us to trigger functions using button presses.

Button Mappings:

- `r3` → Capture a Photo
- `triangle` → Capture a Photo when an Obstacle is Detected
- `circle` → Capture a Photo and Turn on an LED
- `square` → Capture a Photo and Activate the Buzzer

In [ ]:
```
# Map controller buttons to camera functions
joystick.map_button("r3", capture_photo)
joystick.map_button("circle", capture_photo_and_led)  # Assign LED activatic
joystick.map_button("square", capture_photo_and_buzzer)  # Assign buzzer act

print("Button mappings set.")

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 8: Coding Exercises

Try modifying the following functions based on these exercises:

1. **Capture a Series of Photos & Blink the LED**
   - Write a function that takes 5 images at 2-second intervals and blinks the LED after each capture.
2. **Create a Flashing LED Effect**
   - Modify the LED function to blink 3 times after capturing a photo.
3. **Play a Melody on the Buzzer**
   - Write a function that plays a sequence of buzzer tones (e.g., 500Hz, 700Hz, 1000Hz) when a photo is captured.

Challenge:

Test your modifications by using the controller to trigger different events.

In [ ]:
```
#Challenge 1    - Write a function that takes 5 images at 2-second intervals
def photo_series_with_led_blink():
    """Takes 5 photos with a 2-second pause and blinks the LED after each.""
    for i in range(5):
        capture_photo()
        led.blink(1, 0, 255, 0, times=1, delay=0.3)  # Blink green once
        time.sleep(2)  # Wait before next photo
    print("5 photos taken with LED blink after each.")
```

In [ ]:
```
# Challenge 2    - Modify the LED function to blink 3 times after capturing
def capture_and_blink_led_three_times():
    """Captures a photo, then blinks the LED 3 times."""
    capture_photo()
    led.blink(1, 0, 0, 255, times=3, delay=0.2)  # Blink blue 3 times
    print("Photo captured and LED blinked 3 times.")
```

In [ ]:
```
# Challenge 3       - Write a function that plays a sequence of buzzer tones
def capture_photo_and_play_melody():
    """Captures a photo and plays a buzzer melody."""
    capture_photo()
    melody = [(500, 0.3), (700, 0.3), (1000, 0.4)]

    for freq, dur in melody:
        play_buzzer(freq, dur, 0.8, 1)
        time.sleep(0.1)
```