Standard Operating Procedure (SOP) for Magnet Control System

Overview

This SOP outlines the procedure for controlling magnets on a drone platform attached to a WOLF vehicle using a Raspberry Pi. The system operates through ROS, with a publisher node (magnet_publisher.py) on the WOLF computer and a subscriber node (magnet_controller.py) on the Raspberry Pi. It's crucial to note that the publisher code only functions correctly when it is running concurrently with the subscriber code.

Prerequisites

Basic understanding of Raspberry Pi, ROS, and SSH (Secure Shell).
ROS installed on both the WOLF computer and Raspberry Pi.
Python 3 installed on both devices.
Raspberry Pi connected to the WOLF vehicle via an Ethernet port.
Relays on the Pi connected to the magnets.

Equipment Setup

Raspberry Pi Connection:

- Connect the Raspberry Pi to the WOLF vehicle using an Ethernet port.
- Connect the relays on the Pi to the magnets on the drone platform.

WOLF Computer Preparation:

- Ensure the WOLF computer is connected to the same network as the Raspberry Pi.

Software Configuration

ROS Environment:

- Confirm ROS is installed and configured on both the WOLF computer and Raspberry Pi.

Script Placement:

- Place `magnet_publisher.py` on the WOLF computer.
- Place `magnet_controller.py` on the Raspberry Pi.

Operational Procedure

SSH into Raspberry Pi:

- Open a terminal on the WOLF computer.
- SSH into the Raspberry Pi: `ssh [username]@[Raspberry Pi IP address]`.

Running the Controller Node on Raspberry Pi:

- Navigate to the directory with `magnet_controller.py`.
- Run the script: `python3 magnet_controller.py`.

Starting the Publisher Node on the WOLF Computer:

- Open a second terminal on the WOLF computer.
- Navigate to the directory with `magnet_publisher.py`.
- Run the script: `python3 magnet_publisher.py`.
- The script will prompt for user input ('y' for magnets off, 'n' for magnets on) and publish the command.
- Ensure this script runs simultaneously with the subscriber script on the Raspberry Pi for proper operation.

Script Details

Magnet Publisher

ROS Node Creation:

- Initializes a ROS node named `magnet_publisher`.
- This node is responsible for publishing messages to control the magnets.

Publisher Setup:

- Creates a publisher that can send messages of type `Bool` (Boolean) on the topic `toggle_magnets`.
- The `Bool` message type is used because the magnet state is binary: on or off.

Timer and Callback Function:

- Sets up a timer to trigger a callback function every 2 seconds.
- This regular interval is used to prompt the user for input and send commands.

User Input and Message Publishing:

- During each callback (every 2 seconds), the script prompts the user to enter 'y' (yes) or 'n' (no) to turn the magnets off or on, respectively.
- Converts the user input to lower case for consistency.
- Translates the input into a Boolean message: `True` for 'y' (magnets off) and `False` for 'n' (magnets on).
- Publishes this Boolean message on the `toggle_magnets` topic.

Logging Information:

- Logs the state being published ("True" or "False") for the user's confirmation.

Magnet Controller (magnet_controller.py):

- Initializes a ROS node and sets up GPIO for relay control.
- Subscribes to the `toggle_magnets` topic.

- Controls the magnets based on the received message.

Magnet Controller

- ROS Node Creation:
  - Initializes a ROS node named `magnet_controller`.
  - This node subscribes to messages and controls the magnets based on the received commands.
- GPIO Setup:
  - Configures the General Purpose Input/Output (GPIO) pins on the Raspberry Pi, which are connected to the relays controlling the magnets.
  - Sets the GPIO mode and initializes the pins to a default state (HIGH or LOW), initially turning the magnets off for safety.
- Parameter Declaration and Relay Pin Initialization:
  - Declares a ROS parameter for relay pins, allowing for configuration flexibility.
  - Retrieves the relay pins configuration from the declared parameter.
- Subscription to Topic:
  - Creates a subscription to the `toggle_magnets` topic.
  - Specifies that the node should listen for `Bool` messages on this topic.
- Listener Callback Function:
  - Defines a callback function that is invoked when a message is received on the `toggle_magnets` topic.
  - If the received message is `True`, the script turns the magnets off (sets GPIO pins to HIGH).
  - If the message is `False`, it turns the magnets on (sets GPIO pins to LOW).
- Resource Cleanup:
  - Includes a destructor method to clean up resources when the node is terminated.
  - Ensures that the GPIO pins are set to a safe state (turning the magnets off) and performs GPIO cleanup.
- Logging Information:
  - Logs messages indicating the actions being performed (e.g., turning magnets on or off) for user awareness and debugging purposes.