THRUSTER CONTROL SYSTEM

Autonomous Catamaran – IIT KGP

OVERVIEW

This module provides a robust, real-time keyboard interface for controlling the left and right thrusters of an autonomous surface vehicle (catamaran) using a Raspberry Pi. It leverages the pigpio library for precise PWM (Pulse Width Modulation) signal generation and curses for terminal-based user interaction. All thruster commands are logged for analysis and debugging.

SYSTEM ARCHITECTURE

- **Hardware**: Raspberry Pi (with pigpio daemon running), ESCs (Electronic Speed Controllers), dual thrusters
- **Software Dependencies**:
    - Python 3.x
    - pigpio (for GPIO PWM control)
    - curses (for keyboard-driven UI)
    - csv (for logging)

FEATURES

- **Real-time keyboard control** of left and right thrusters (individually or together)
- **Safety mechanisms**:
    - Emergency stop (toggle)
    - Pause/resume
    - ESC arming sequence
- **Comprehensive logging** of all command events (timestamped)
- **User feedback**: Live status display in terminal

CODE WALKTHROUGH

1. **Configuration & Initialization**

- **GPIO Pin Assignments:**
    - ESC_LEFT_PIN = 17

    - ESC_RIGHT_PIN = 18
- **PWM Parameters:**
    - Neutral: 1500 µs
    - Range: 1000–2000 µs
    - Step size: 10 µs
- **pigpio Initialization:**
    - Connects to the pigpio daemon.
    - Exits if the daemon is not running, ensuring no undefined hardware behavior.

2. **PWM Sending Function**

```
python
def send_pwm(l, r):
pi.set_servo_pulsewidth(ESC_LEFT_PIN, l)
pi.set_servo_pulsewidth(ESC_RIGHT_PIN, r)
now = time.strftime('%Y-%m-%d %H:%M:%S')
writer.writerow([now, l, r])
csvfile.flush()
```

- Sets the PWM signal on both thruster ESCs.
- Logs the command with a timestamp for traceability.

3. **User Interface (curses-based)**

- **Startup Sequence:**
    - Sends neutral PWM to both thrusters (arms ESCs safely).
    - Waits for user confirmation (Enter key) before enabling control.
- **Main Control Loop:**
    - **Arrow Keys / WASD:** Incrementally adjust left/right thruster PWM.
    - **Space:** Pause/resume output (sends neutral while paused).
    - **r:** Reset both thrusters to neutral.
    - **x:** Emergency stop toggle (locks both thrusters in neutral until released).
    - **q:** Quit and save log.

- **Live Status Display:**
    - Shows current PWM values, pause/emergency status, and control hints.

4. **Safety & Error Handling**

- **Emergency Stop:**
    - Immediate neutral signal to both thrusters.
    - Must be toggled off to resume control.
- **Graceful Shutdown:**
    - On exit (including Ctrl+C), sets both thrusters to neutral, stops pigpio, and closes the log file.

5. **Logging**

- All PWM commands are logged to thruster_log.csv with timestamps.
- Ensures reproducibility and supports post-mission analysis.

USAGE INSTRUCTIONS

1. **Prerequisites:**
    - Ensure the pigpio daemon is running:

```
bash

sudo pigpiod
```

    - Connect ESCs and thrusters to GPIO 17 and 18.

2. **Run the Script:**

```
bash

python thruster_control.py
```

3. **Follow On-Screen Instructions:**
    - Arm ESCs by pressing Enter.
    - Use arrow keys or WASD for control.
    - Refer to the on-screen guide for all commands.
4. **Shutdown:**
    - Press q to quit and save the log.
    - On exit, all GPIOs are cleaned up and thrusters are set to neutral.

KEY DESIGN DECISIONS

- **Terminal UI (curses):**
  Enables real-time, responsive control without the complexity of a GUI.
- **Comprehensive Logging:**
  Essential for debugging, safety audits, and performance analysis.
- **Safety First:**
  Emergency stop and pause features are prioritized for safe field operation.

EXTENSIBILITY

- **Modular Design:**
  - send_pwm() and control logic can be extended for more thrusters or different vehicle configurations.
- **Integration Ready:**
  - Can be integrated with higher-level autonomy modules or remote control interfaces.

KNOWN LIMITATIONS

- **Requires pigpio daemon**
- **Terminal-based UI only** (no GUI)
- **No built-in input validation for extreme/faulty hardware states**

CONCLUSION

This thruster control script is a **mission-critical tool** for safe, precise, and logged manual operation of the Autonomous Catamaran's propulsion system. It is designed for reliability, operator safety, and ease of use in both lab and field environments.

For any issues or feature requests, please open an issue on GitHub or contact the maintainers.

Prepared by:
Team, Autonomous Catamaran, IIT Kharagpur
Date: 2025-06-07