ROTRIX FIRMWARE MANUAL - ENTERPRISE VERSION

30/04/2025

V 2.0

REUDE Technologies Private Limited.

No.1, Nehru Street, Vasantham Nagar,
Avadi, Kamarajnagar, Poonamallee,
Tiruvallur- 600071, Tamil Nadu, India
CIN No: U26515TN2024PTC171846

Contact. No: +91 6384667822
Contact Email: business@reude.tech
Website: http://rotrix.reude.tech/

Description

This report provides guidance on designing the hardware and firmware necessary to integrate with the ROTRIX application. ROTRIX is an online tool that connects to your motor thrust test bed, helping you select the optimal powertrain for your drone. It enables you to characterize and evaluate the performance of motor, propeller, and ESC combinations under various test conditions, allowing you to choose the best configuration. To use this tool, specific hardware components and custom firmware are needed to transmit the necessary data to the software. This report outlines the hardware requirements and firmware development process for seamless integration with ROTRIX.

Contents

Tables

1. Introduction

The integration of a motor thrust test bed with the ROTRIX application requires a thorough understanding of both hardware and firmware. This manual guides users through selecting the appropriate components and developing firmware that ensures smooth data exchange between the test bed and the software. By following this guide, users can unlock ROTRIX's full potential to assess and optimize drone powertrain configurations. A complete documentation on the ROTRIX firmware can be found at this link.

2. Hardware

This section outlines the hardware requirements essential for integrating the thrust test bed with the ROTRIX application.

A list of potential hardware components that can be used to build the thrust bed and connect with the software can be viewed here.

2.1. Microcontroller board

An Arduino IDE-compatible microcontroller

2.2. Motor testbed

The testbed should be capable of mounting the motors and propellers, and should house the sensors for thrust, torque, speed, voltage, current, thermal and image data. It also should include provisions for connecting the batteries to the Electronic Speed Controllers (ESCs), the microcontroller to the sensors, and the controller to a computer system. A sample motor thrust test bed can be referred here.

2.3. Electronic Speed controller (ESC)

The ESC moderating the motor speed has to be controlled by PWM signals. The PWM limits for the ESC can be set in the software to ensure proper signal communication.

2.4. Thrust load cell

A load cell that can measure the thrust generated by the motor-propeller setup.

2.5. Torque load cell(s)

A torque transducer or a set of load cells that will measure the torque produced by the setup.

2.6. Load cell Amplifiers

Amplifiers that amplify the load cell signals for microcontroller readability

Ensure these load cells have higher load-measuring capacities than the motors chosen for testing/thrust stand load specifications.

2.7. Speed sensor

Any sensor that is able to measure and output the instantaneous speed (RPM) of the motor. Ensure compatibility with the motor's maximum RPM.

2.8. Voltage and current sensor

Any sensor that can measure the voltage and current drawn by the ESCs from the battery. Make sure the voltage and current specifications of the sensor exceed the motor and ESC requirements.

2.9. Temperature sensor

Temperature measuring sensors, which could either be a single-point temperature sensor to get temperature of a specific location on the motor and ESC or a thermal camera to get overall temperature values. If using a thermal camera, the accepted resolutions are 32x24, 320x240 or 640x480.

2.10. Image sensor

A camera that can produce RGB data of pixel resolutions 320x240 or 640x480

3. Software

The integration process requires two primary software tools:

3.1. Arduino IDE

Arduino IDE serves as the platform for developing and uploading the firmware to the microcontroller. It facilitates serial communication between the microcontroller and the ROTRIX application.

You can install the latest release of the Arduino IDE at this Arduino IDE Download link

3.2. Python 3

Python is essential for processing data on the software side, enabling interaction between the application and the hardware components.

You will need to install version 3.10 or higher for this software and can be downloaded from the link Python Download link

Both hardware and software components play a critical role in ensuring accurate data collection and seamless communication between the firmware and the application.

4. Firmware Development

The firmware acts as the bridge between the hardware and the software, processing sensor data and communicating with the ROTRIX application. This section outlines how to develop firmware that ensures reliable data transmission and compatibility with the application.

The firmware (Arduino code) could be developed according to the sensors used by the user. However, the firmware must process sensor outputs into predefined formats and units that the application can interpret. Data must adhere to specific formats, as detailed in the subsequent sections, to ensure smooth communication.

A reference source code for the firmware can be viewed here - Source code along with a circuit diagram illustrating the multi-sensor integration for sensors used in the source code in here.

4.1. Sensor data formats

Though any sensors of the description mentioned in section 2 can be used, the outputs from these sensors have to be processed to a specific datatype and format. The acceptable data formats are listed in Table 1 below. ROTRIX supports both metric and imperial units for user convenience.

Sensor	Final values needed for software	Metric Units	Imperial units	Datatype
Thrust load cell	Current thrust measured	kgf	lb	float
Torque load cell	Current torque measured	Nm	lb-in	float
Speed sensor	Current RPM of motor	RPM	RPM	float
Voltage sensor	Current battery voltage	V	V	float
Current sensor	Battery current drawn	A	A	float
Single-point Temperature sensor	Current temperature value measured. Should be an array of size 1	°C	°F	float
Thermal camera	Current temperature array. Can be an array of sizes 32x24 (or) 320x240 (or) 640x480	°C	°F	float
Electronic Speed Controller	PWM value			float

Table 1. Final sensor data format needed for software

4.2. Output data to software

The serial data to be sent from the firmware to software must be of the below order inorder for the software to receive and process the data properly. Ensure that the baud rate for the serial communication is 115200.

Parameter	Values to be sent	Datatype	Size	Sample data
Device ID	Given by REUDE	String	1	REU-MTB
Current PWM value	Latest PWM value sent to ESC	float	1	1100
RPM	Latest RPM value of the motor	float	1	1500
Current	Current drawn by motor, in Amps	float	1	40
Voltage	Latest voltage of the battery, in Volts	float	1	50
Thrust	Latest thrust value measured	float	1	15
Torque	Latest torque value measured	float	1	5
Thermocouple 1*	Temperature measured at any location on the setup (motor or ESC)	float	1	28.5
Thermocouple 2*	Temperature measured at any location on the setup (motor or ESC)	float	1	40.2
Thermocouple 3*	Temperature measured at any location on the setup (motor or ESC)	float	1	32.5

Thermocouple 4*	Temperature measured at any location on the setup (motor or ESC)	float	1	29.3
Thermocouple 5*	Temperature measured at any location on the setup (motor or ESC)	float	1	46.3
Thermocouple 6*	Temperature measured at any location on the setup (motor or ESC)	float	1	39.8
Ambient temperature	Ambient temperature at the setup area	float	1	30
Thermal data*	An array of temperature values (image[height][width]) from a thermal camera	float	32 x 24 (or) 320x240 (or) 640x480	img[32][24]

Table 2. Data order in which data needs to be sent by firmware

Note:

- Each data has to be delimited by ','
- Data has to be sent in the same order as in the table
- * - indicates that the specific sensor data will be displayed based on the subscription plan chosen
- If a particular sensor is not used, send '0' as its value

An instance of the final data to be sent by the firmware can be viewed here.

4.3. Input data from software

The following serial data will be sent to firmware from software via serial communication as well inorder to control the motor setup,

1. ESC inputs to control motor speeds
2. ESC calibration limits for safety purpose
3. No.of propeller blades used in current test incase required for RPM calculation

The input data to arduino from software will be the following data in the corresponding order

Parameter	Values to be sent	Datatype	Size	Sample data
No.of.propeller blades	No.of propeller blades value set by user during powertrain setup in software	String	1	2
Minimum PWM value	Minimum ESC calibration value set by user during powertrain setup in software		1	1000
Maximum PWM value	Maximum ESC calibration value set by user during powertrain setup in software		1	2000
User input PWM to ESC	The latest PWM value (throttle value) set by user in the software for testing		1	1300

Table 3. Data order in which data will be sent to firmware

Note:

- Each data has to be delimited by ','
- Data has to be sent in the same order as in the table

A sample of the data that will be sent to the firmware can also viewed here.

5. Safety considerations

When connecting firmware to software via serial communication, several safety measures should be implemented to ensure reliable operation and prevent damage to hardware components:

5.1. Voltage Compatibility

Ensure all connected devices operate within their voltage ranges to avoid damage due to overvoltage or undervoltage conditions.

5.2. Secure Connections

Verify that all connections between the microcontroller and sensors are secure to prevent accidental disconnections during operation.

5.3. Signal Noise Mitigation

Shield cables and avoid running power and signal wires in parallel to minimize electromagnetic interference (EMI).

5.4. Fail-safe mechanisms

Implement fail-safe mechanisms in the firmware to handle unexpected software or hardware failures.

5.5. Safety limits

Include safety limits in the firmware to ensure hardware safety during unexpected software or hardware failures.

6. Troubleshooting Tips

If issues arise during the integration or operation of the system, refer to the following troubleshooting guidelines:

6.1. No Data Received by Software

- Verify that the microcontroller is correctly connected to the computer via the serial interface.
- Ensure the correct COM port and baud rate are configured in the software.

6.2. Incorrect or Corrupted Data

- Check for loose or faulty connections between the sensors and the microcontroller.

6.3. Sensor Readings Not Updating

- Confirm that the firmware is properly polling the sensors and processing their outputs.
- Inspect the power supply to ensure all sensors are receiving adequate power.

6.4. Firmware Not Responding

- Restart the microcontroller and ensure the firmware is flashed correctly.
- Check for infinite loops or crashes in the firmware code.

6.5. Overheating Components

- Monitor the temperature of all critical components and ensure proper ventilation or cooling mechanisms are in place.

6.6. Software-Firmware Communication Issues

- Verify that the data format matches the protocol specified in this manual.
- Inspect for buffer overflows or memory allocation issues in the firmware.

If you encounter persistent issues or require further assistance, please reach out to the REUDE Technologies Pvt. Ltd. technical support team through the following channels:

- Phone : +91 6384667822
- Email : support.rotrix@reude.tech
- WhatsApp Community : Join RotriX India on WhatsApp to post queries, share feedback, and engage with the user community in real time. Alternatively, you can scan the below QR code to join our WhatsApp community.