RaptorHabGS - Python Ground Station

Cross-platform Python port of the macOS RaptorHabGS ground station application for receiving telemetry and images from high-altitude balloon payloads.

Features

- **Dual GUI Options**: Desktop (PyQt6) and Web-based (Flask) interfaces
- **Headless Mode**: Run on remote servers (Raspberry Pi, etc.) and access via web browser
- **Serial Communication**: Connects to Heltec LoRa radio modems (SX1262-based)
- **Telemetry Display**: Real-time position, altitude, speed, and system status
- **Interactive Map**: Leaflet-based map with track visualization
- **Image Reception**: RaptorQ forward error correction for reliable image transfer
- **GPS Integration**: Ground station position via external GPS receiver
- **SondeHub Upload**: Upload telemetry to the SondeHub Amateur network
- **Real-time Updates**: WebSocket-based live data streaming for web GUI

Requirements

- Python 3.10 or later
- Windows 10/11, macOS, or Linux

Installation

1. Clone or download the repository

```bash
cd PythonApp
```

2. Create a virtual environment (recommended)

```bash
```

```
# Windows
python -m venv venv
venv\Scripts\activate

# macOS/Linux
python3 -m venv venv
source venv/bin/activate
```

3. Install dependencies

```bash
pip install -r requirements.txt
```

4. Install RaptorQ library (for image decoding)

The `raptorq` Python package requires Rust to compile. Install Rust first:

Windows: Download and run the installer from https://rustup.rs/

macOS/Linux:

```bash
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
source $HOME/.cargo/env
```

Then install raptorq:

```bash
pip install raptorq
```

Running the Application

Option 1: Desktop GUI (PyQt6)

For local use with a native desktop interface:

```bash
python main.py
```

Option 2: Web GUI (Flask) - Headless Mode

For remote access or headless servers (e.g., Raspberry Pi):

For remote access of headless servers (e.g., Raspberry Pi):

```bash
# Run on all interfaces, port 5000 (default)
python web_server.py

# Specify host and port
python web_server.py --host 0.0.0.0 --port 8080

# Localhost only (more secure)
python web_server.py --host 127.0.0.1 --port 5000

# Enable debug mode
python web_server.py --debug
```

Then open a web browser and navigate to `http://<server-ip>:5000`

Web GUI Features

The web interface provides the same functionality as the desktop app:

- Real-time telemetry display with WebSocket updates
- Interactive map with payload tracking
- Serial port selection and control
- GPS connection for ground station position
- Bearing and distance calculations
- Image reception and display
- SondeHub configuration
- Settings management

Running on Raspberry Pi

For a headless Raspberry Pi setup:

```bash
# Install dependencies
pip install -r requirements.txt
```

```
# Run at startup (add to /etc/rc.local or create systemd service)
python3 /path/to/web_server.py --host 0.0.0.0 --port 5000 &

# Or create a systemd service file
sudo nano /etc/systemd/system/raptorhabgs.service
```

Example systemd service file:

```ini
[Unit]
Description=RaptorHabGS Ground Station
After=network.target

[Service]
Type=simple
User=pi
WorkingDirectory=/home/pi/RaptorHabGS
ExecStart=/usr/bin/python3 web_server.py --host 0.0.0.0 --port 5000
Restart=always

[Install]
WantedBy=multi-user.target
```

Usage

Connecting to the Radio Modem

1. Connect your Heltec LoRa modem to a USB port
2. Select the correct COM port (Windows) or /dev/tty* device (Linux/macOS) from the dropdown
3. Click "Start Receiving"

GPS Setup (Optional)

1. Connect a GPS receiver (NMEA output) to another USB port
2. In the GPS section, select the port and click "Connect"
3. Ground station position will be used for bearing calculations and SondeHub uploads

SondeHub Configuration

1. Go to File → Settings → SondeHub tab

1. Go to File → Settings → SondeHub tab
2. Enable SondeHub uploads
3. Enter your callsign and payload callsign
4. Telemetry will be uploaded to https://amateur.sondehub.org/

Directory Structure

```
PythonApp/
├── main.py                 # Desktop GUI entry point
├── web_server.py           # Web GUI entry point (headless)
├── requirements.txt        # Python dependencies
├── raptorhabgs/
│   ├── __init__.py
│   ├── core/               # Core functionality
│   │   ├── __init__.py
│   │   ├── config.py       # Configuration management
│   │   ├── telemetry.py    # Data structures
│   │   ├── protocol.py     # Packet parsing
│   │   ├── serial_manager.py  # Serial port communication
│   │   ├── gps_manager.py  # GPS/NMEA parsing
│   │   ├── ground_station.py  # Main controller
│   │   └── sondehub.py     # SondeHub API client
│   ├── ui/                 # Desktop GUI (PyQt6)
│   │   ├── __init__.py
│   │   ├── main_window.py  # Main application window
│   │   ├── sidebar.py      # Status sidebar
│   │   ├── map_widget.py   # Leaflet map display
│   │   └── settings_dialogs.py # Configuration dialogs
│   └── web/                # Web GUI (Flask)
│       ├── __init__.py
│       ├── server.py       # Flask server with WebSocket
│       ├── templates/
│       │   └── index.html  # Main web interface
│       └── static/         # CSS, JS, assets
```

Data Storage

Application data is stored in:

- **Windows**: `%USERPROFILE%\Documents\RaptorHabGS\`
- **macOS**: `~/Documents/RaptorHabGS/`
- **Linux**: `~/.local/share/RaptorHabGS/`

Subdirectories:

- `images/` - Decoded images
- `missions/` - Mission recordings
- `telemetry/` - Telemetry logs
- `logs/` - Application logs
- `config.json` - Settings

Troubleshooting

Serial port not appearing

Windows: Install the CH340 or CP210x USB driver for your modem

Linux: Add your user to the `dialout` group:

```bash
sudo usermod -a -G dialout $USER
# Log out and back in
```

Map not loading

Ensure you have an internet connection for OpenStreetMap tiles. The map requires PyQt6-WebEngine.

RaptorQ installation fails

Make sure Rust is installed and in your PATH:

```bash
rustc --version
```

Protocol Compatibility

This application is compatible with the RaptorHab balloon payload firmware using:

- FSK modulation at 100 kbps
- Custom packet format with sync word `0xAA55`
- RaptorQ fountain codes for image transfer

License