Project Requirement Document

Project Title: RTK-GPS Based Rover Navigation for Precision Agriculture

Version: 1.0

Date: April 11, 2025

Prepared By: MG / Kanan Robotics

---

1. Project Overview

This project involves the development of a precision navigation system for an autonomous agricultural rover using RTK-GPS technology. The system will utilize the Emlid Reach M2 GNSS receiver, online NTRIP CORS correction, and a Python-based control stack. QGIS will be used for mapping and route planning.

2. Project Scope

The primary goal is to enable the rover to navigate within farm boundaries with high positional accuracy (less than 5 cm deviation). The rover should autonomously follow predefined paths generated in QGIS, respond to real-time positional updates from the Emlid Reach M2, and log its actual trajectory.

3. System Components

Hardware:

- Emlid Reach M2 GNSS Receiver
- Multi-band GNSS antenna
- Embedded system (e.g., Raspberry Pi or Jetson Nano)
- Motor-controlled rover chassis
- Internet connectivity module (4G LTE dongle or WiFi)

Software:

- QGIS for field mapping and route planning
- Python for rover navigation logic and control
- Libraries: PySerial, Pandas, Numpy, Matplotlib, Shapely, MQTT (optional)
- Emlid ReachView 3 app for configuration

Correction Services:

- NTRIP Client using credentials to access regional/state-provided CORS network

---

4. Functional Requirements

4.1 GPS Data Ingestion

- Connect to Reach M2 over TCP or serial (default TCP port 9001).
- Parse NMEA or LLH data (latitude, longitude, height, fix quality, number of satellites).

- Confirm and maintain RTK FIX or FLOAT status.

4.2 QGIS Map Planning

- Import or draw field boundaries.
- Define row geometry and direction.
- Use QGIS plugins (e.g., 'Points along geometry') to generate waypoints.
- Export paths as CSV, GeoJSON, or Shapefile for use in the navigation system.

4.3 Python Navigation Stack

- Read path coordinates generated from QGIS.
- Continuously compute real-time distance and heading to next waypoint.
- Implement PID or rule-based motor control logic.
- Handle edge cases: boundary reached, GPS fix lost, obstacle (future enhancement).

4.4 RTK and CORS Integration

- Configure Emlid Reach M2 to receive NTRIP correction over mobile network.
- Validate RTK status via Emlid ReachView or GPS logs.
- Automatic reconnection if NTRIP stream is interrupted.

4.5 Data Logging and Diagnostics

- Log real-time rover GPS coordinates and timestamps.
- Log fix quality, satellite count, and deviation from planned path.
- Optional: Visualize trajectory using Matplotlib or save for post-processing.

4.6 RTK-GPS Based Automation Tasks and Features

- Compute and maintain real-time heading of the rover using GPS coordinates.
- Navigate from current location to the start of the first sowing row autonomously.
- Align the rover’s orientation to match the heading angle of the row before entry.
- Execute turns at headlands to align with the next row using geometric constraints.
- Detect and flag unreachable paths or waypoints based on minimum turning radius and physical constraints of the rover.
- Realign or re-plan path based on GPS drift, obstacle detection (future), or system calibration.
- Enable return-to-home (RTH) mode using GPS coordinates in case of emergency stop or signal loss.

---

5. Non-Functional Requirements

- System accuracy: < 5 cm in RTK Fix mode.
- Runtime: 2+ hours of continuous autonomous navigation.
- Environmental tolerance: Daylight operation, low-dust tolerance.
- Modular and maintainable Python codebase.
- Open format for QGIS-based planning (GeoJSON, CSV).

6. Communication Interfaces

Interface	Protocol	Direction	Description
Reach M2 GNSS	TCP/Serial	Input	GNSS position and fix status
Rover Motors	GPIO/I2C	Output	Movement control
QGIS Path Data	File/CSV	Input	Planned routes and waypoints
Monitoring UI	MQTT/Web UI	Bi-directional	Optional remote monitoring and control

---

7. Architecture Diagram (To be created)

- Logical flow: QGIS Path -> Python Navigator -> Motor Commands
- RTK correction loop: CORS Server -> NTRIP Client (Reach M2)
- Data Logging: CSV + Optional visual logs

---

8. Future Enhancements

- Add obstacle avoidance using camera or LiDAR
- Integrate AI-based path correction for curved fields
- Offline RTK using base + rover setup (no internet dependency)
- Add touchscreen control interface
- Enable multi-rover coordination via MQTT

---

9. Glossary

- **RTK**: Real-Time Kinematic (high-precision GNSS correction)
- **NTRIP**: Networked Transport of RTCM via Internet Protocol
- **CORS**: Continuously Operating Reference Station
- **QGIS**: Open-source geographic information system
- **PID**: Proportional-Integral-Derivative (control algorithm)

10. Approval

Name	Role	Signature	Date
[Your Name]	Project Owner		
[Reviewer Name]	Technical Reviewer		

11. Field Testing Plan

To ensure the reliability and performance of the RTK-GPS based rover navigation system, a structured field testing plan will be executed. The objective is to evaluate system accuracy, behavior under real farm conditions, and responsiveness to control logic.

11.1 Test Environment:

- Selected farmland area with known boundaries (mapped in QGIS).
- Clear sky conditions for optimal satellite visibility.
- Pre-defined test path generated using QGIS plugins.

11.2 Pre-Test Checklist:

- Validate Emlid Reach M2 RTK FIX status.
- Confirm NTRIP credentials and correction stream.
- Verify CSV path data ingestion in Python stack.
- Ensure rover motor and control interfaces are responsive.

11.3 Test Procedures:

- **Straight Row Test:** Rover follows a straight path of at least 20 meters. Log actual vs. planned GPS data.
- **Headland Turn Test:** Rover performs a 180° turn at row end and aligns with the next row.
- **Return-to-Home Test:** Rover is commanded to return to initial GPS position.
- **GPS Drift Tolerance Test:** Temporarily disrupt correction stream and observe deviation handling.

11.4 Data Collection:

- Log GPS trajectory at 1 Hz frequency.
- Record timestamped RTK fix status, satellite count.
- Note all deviations, delays, or interruptions during navigation.

11.5 Post-Test Analysis:

- Overlay actual vs. planned paths in QGIS or Matplotlib.
- Generate deviation plots and error distribution.
- Assess smoothness of turns and precision of alignments.

12. Validation Metrics

Success of the system will be measured based on the following quantitative and qualitative indicators:

Metric	Target Value / Criteria
Path Deviation (RTK FIX mode)	≤ 5 cm RMSE
Heading Alignment Accuracy	≤ 5° deviation at row entry
Headland Turn Completion Accuracy	≥ 95% successful alignments
RTK Fix Time after Cold Start	≤ 2 minutes
GPS Update Frequency	≥ 1 Hz
Motor Command Response Time	≤ 500 ms
Uptime per Test Run	≥ 2 hours without failure
Emergency RTH Trigger Success	100% successful return within 30 cm radius
Visual Match with Planned Path	Minimal overlap error in plotted trajectory