Mars rover Presentation

By Group 2: Chenxi Zhao, Huimin Wang, Zheyu Jiang

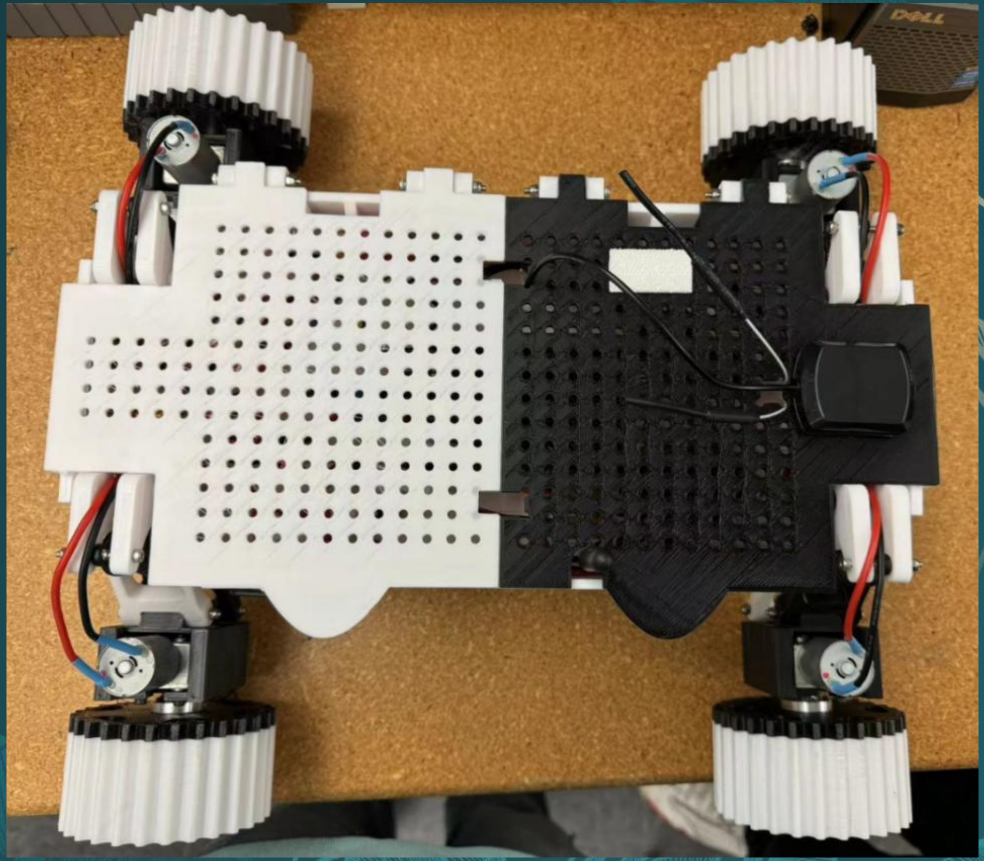
MRE 320: Sensors and Actuators
Instructor: Dr.Zhang

Introduction

The Mars Rover is designed to handle various tasks including environmental data collection and autonomous navigation. It incorporates advanced sensors and actuators to measure weather data and navigate through predefined waypoints autonomously.

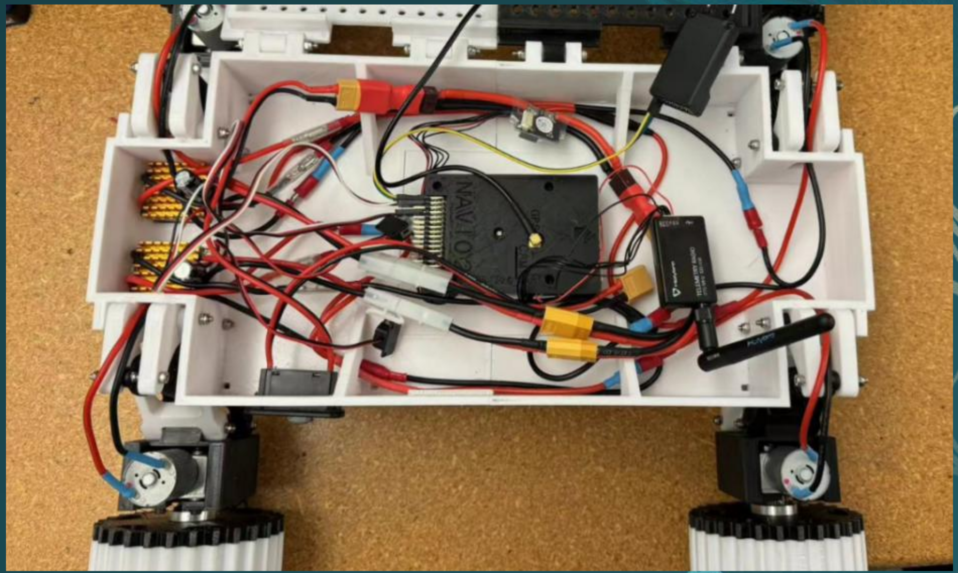
Hardware and Construction

The Rover's chassis is assembled from 3D-printed components, motors are installed within these frames, supported by shock absorbers and wheels attached to motor wheel hubs. The assembly process includes the connection of various bolts and lines and the installation of sensors.



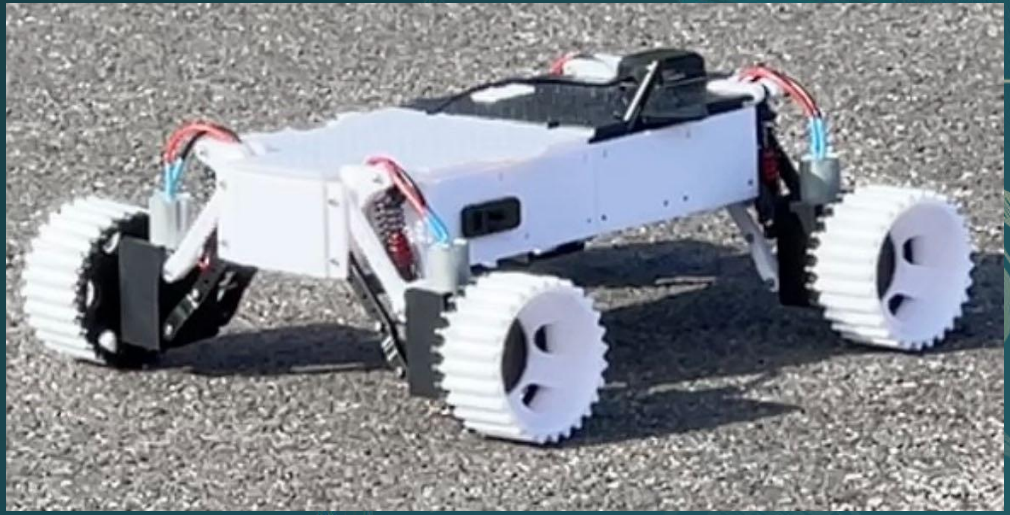
Avionics and Electronics

The Mars Rover's electronic systems are built around a Raspberry Pi and Navio2 flight controller. Setup includes software configuration on the Pi, linking to a mission planner, and activating telemetry for data communication. Key elements are motor drivers, a power module, GPS antenna, and sensors for efficient operation control.



Functional Capabilities

The Mars Rover employs Ardupilot for navigation, configured via a ground control interface. Data from missions are analyzed to evaluate sensor accuracy and environmental responses, offering insights into Rover performance and sensor reliability.



Task 1: Weather data measurement

The Mars Rover has the Onboard barometer.

The onboard barometer include pressure sensor (BARO.Press) and temperature sensor (BARO.Temp)

Principle: Atmospheric pressure decreases with an increase in altitude.

Barometers use a known pressure-altitude relationship, to calculate altitude.

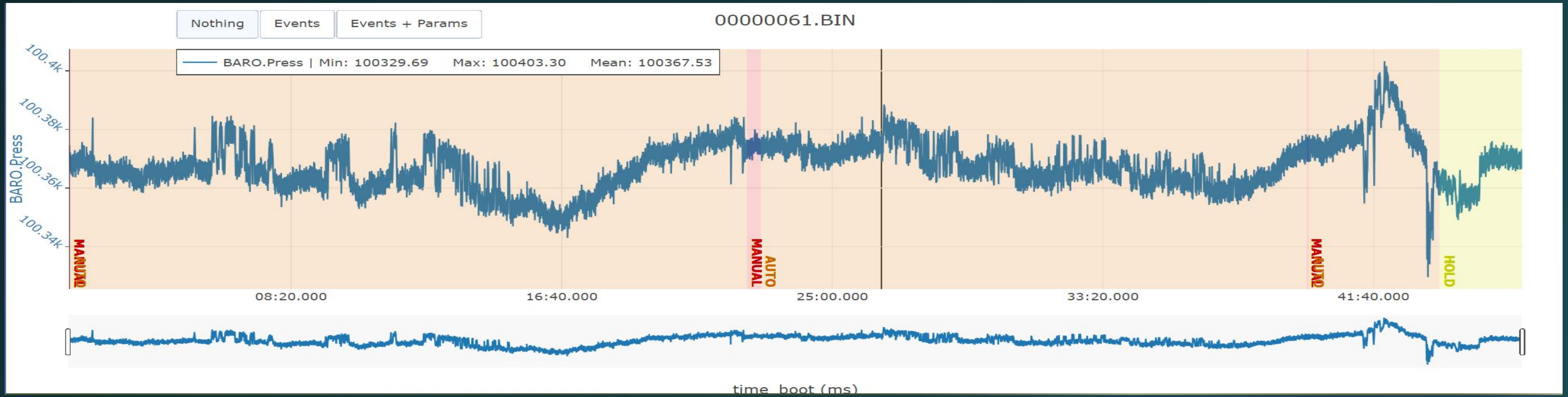


Figure 1: BARO.Press

Figure 2: BARO Temp

Figure 3: The current tempurature

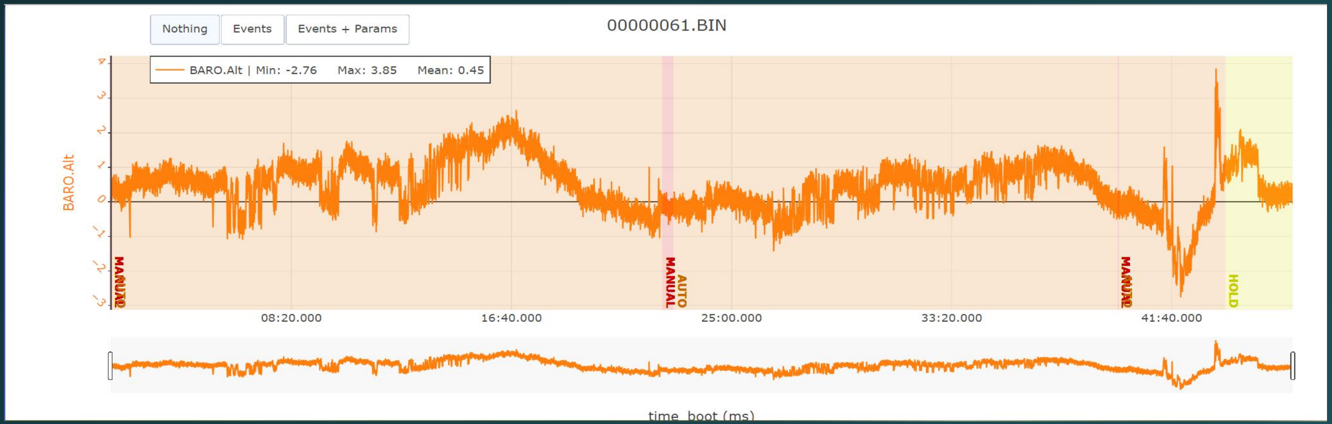


Figure 4: Baro. Alt

How Temperature Affects Pressure:

Temperature changes impact the density of air, which influences atmospheric pressure. Warm air is lighter and causes lower pressure readings, while cold air is heavier and leads to higher pressure readings.

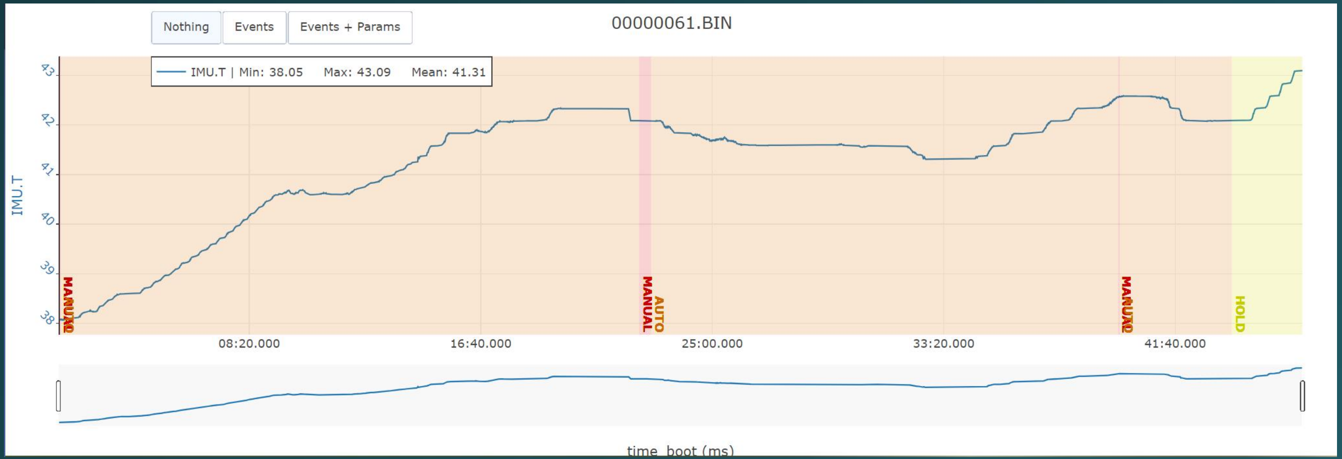


Figure 5: IMU. T

- 1.Obtain readings from both pressure and temperature sensors under different environmental conditions.
- 2.Use a recognized model like the International Standard Atmosphere (ISA) to adjust the pressure readings.
- 3.After implementing temperature compensation, calibrate the sensor at known altitude points such as sea level or a known mountain altitude. Check the precision of the altitude readings by comparing them with these established points or with other reliable measurements like GPS data.
- This ensures that the measurements from Mars Rover are dependable and accurate for any missions.

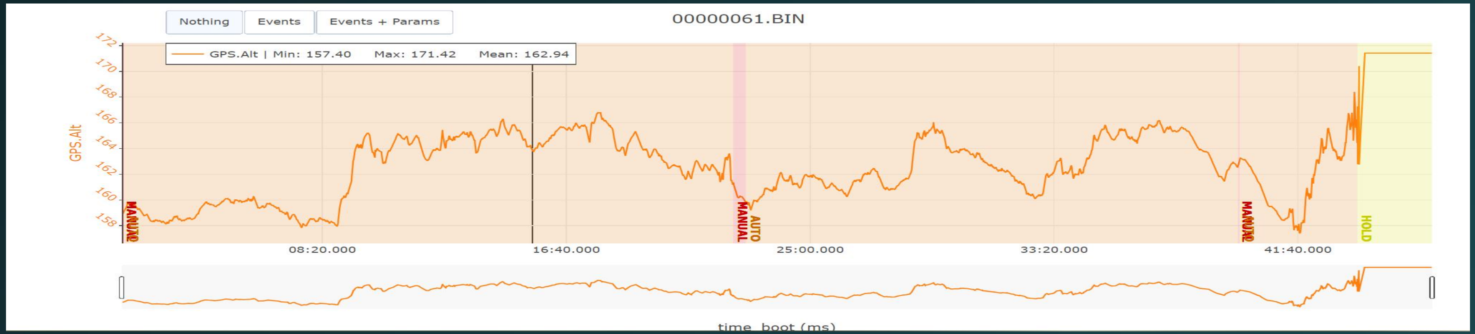


Figure 6: GPS. Alt

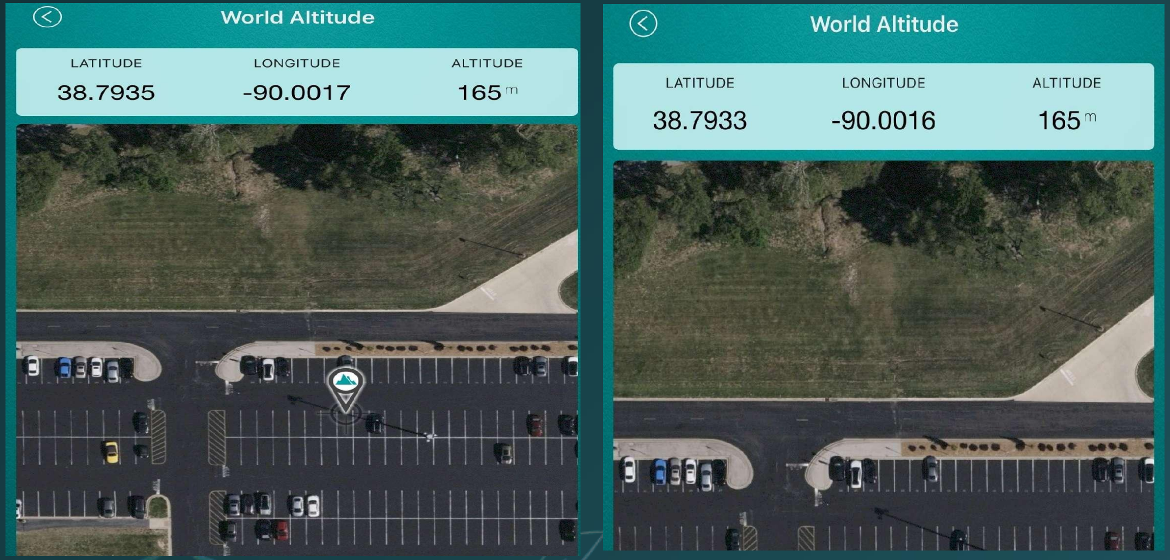


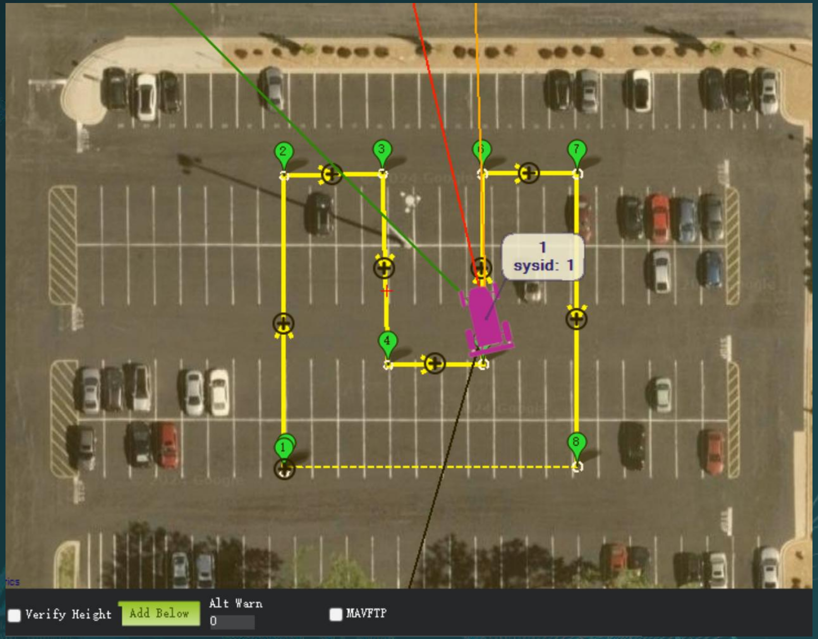
Figure 7: data from GPS track

Characteristics

The sensor exhibits high accuracy, with its measurements closely aligning with the actual values. This precision ensures reliable and consistent data collection.

Task 2: Autonomous navigation through waypoint

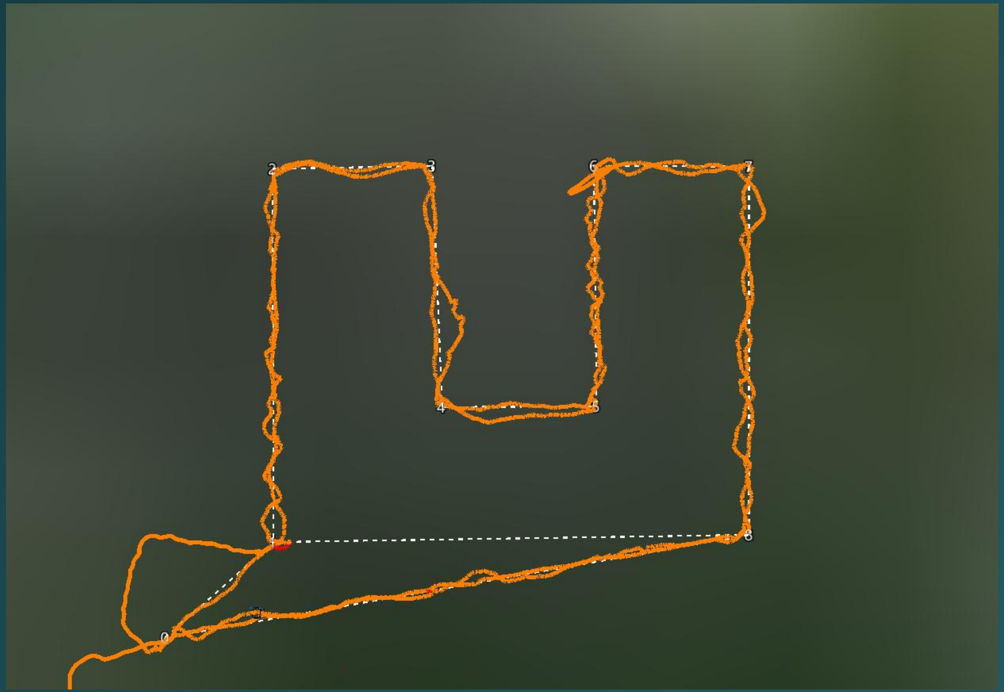
We set eight waypoints like this (in parking lot E):



The distance between each neighborhood dot is around 10 meters.

The distance from waypoint 1 to waypoint 2 is around 30 meters.
The distance from waypoint 2 to waypoint 3 is around 10 meters.
The distance from waypoint 3 to waypoint 4 is around 20 meters.
The distance from waypoint 4 to waypoint 5 is around 10 meters.
The distance from waypoint 5 to waypoint 6 is around 20 meters.
The distance from waypoint 6 to waypoint 7 is around 10 meters.
The distance from waypoint 7 to waypoint 8 is around 30 meters.
The distance from waypoint 8 to waypoint 1 is around 30 meters.

By opening .BIN file by using UAV Log Viewer, we can find the 2D trajectory diagram:



By proper PID tuning, the errors between the actual path and designed path as well as the waypoints were minimized. And by modifying the values of P,I and D for many times, and running on the site for many times, we gradually reduced the error value.

P - Proportional:

This is the proportional part that deals with the present value of the error, which is the difference between the desired setpoint and the current value.

I - Integral:

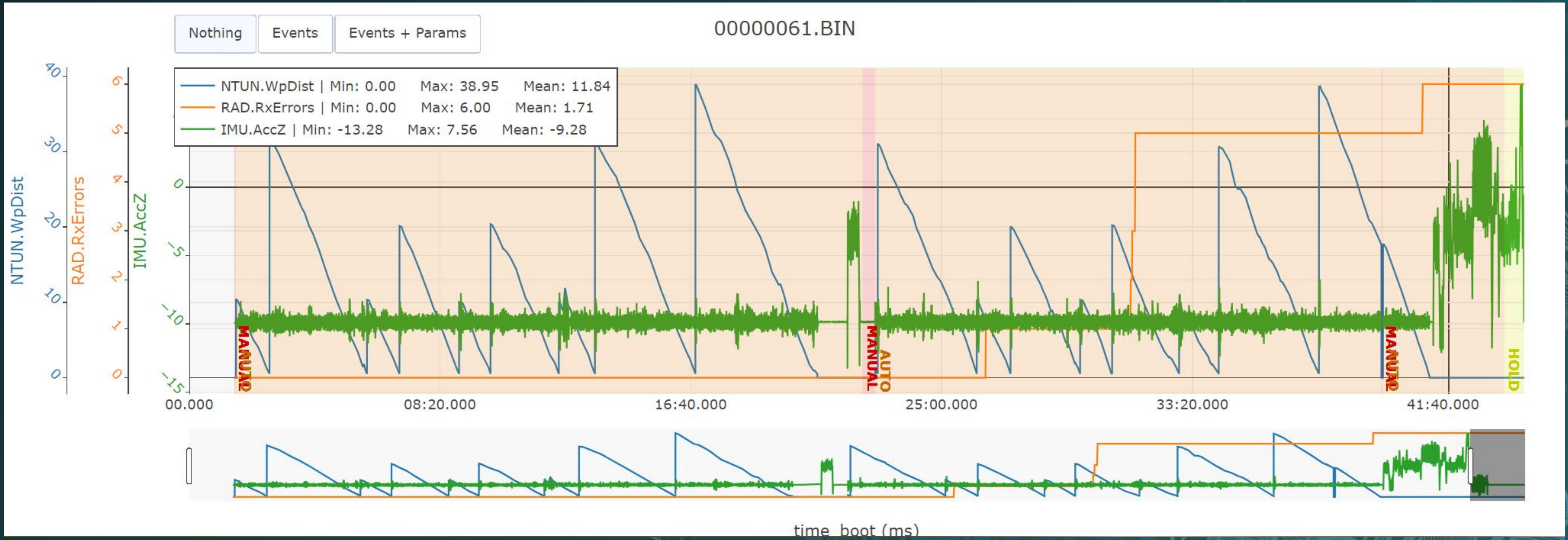
This part of the controller deals with the accumulation of past errors. If the error has been present for a prolonged period, the integral part builds up over time and attempts to eliminate the steady-state error.

D - Derivative:

The derivative part predicts the future trend of the error based on its current rate of change. It helps in reducing the overshoot and settling time.

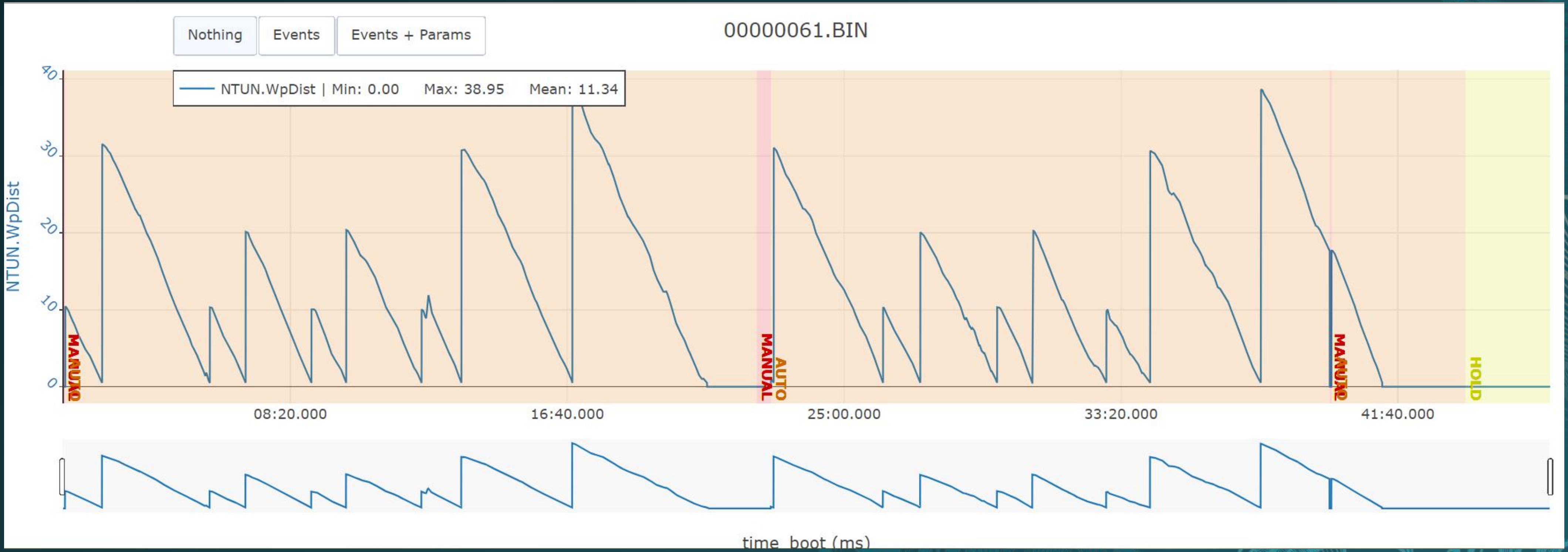
We can see from the picture below, the total time it takes to complete the path is around 41 mins.

41:40 - 00.000 = 41:40 min.



We can see Errors (distance) from the designed path, under NTUN.Xtrack. Like the picture below:

And we can see distance from Waypoints, which can be found in NTUN.WpDist, like the picture below:



Thank you for your attention!