Calibrating the Cosmic Ray Tracking Detector

Zack Hu (Benjamin.N Cardozo High School) Research Mentor: Raul Armendariz Undergraduate research Mentor: Shorn Grant

Special Thanks to KOKA who composed this poster

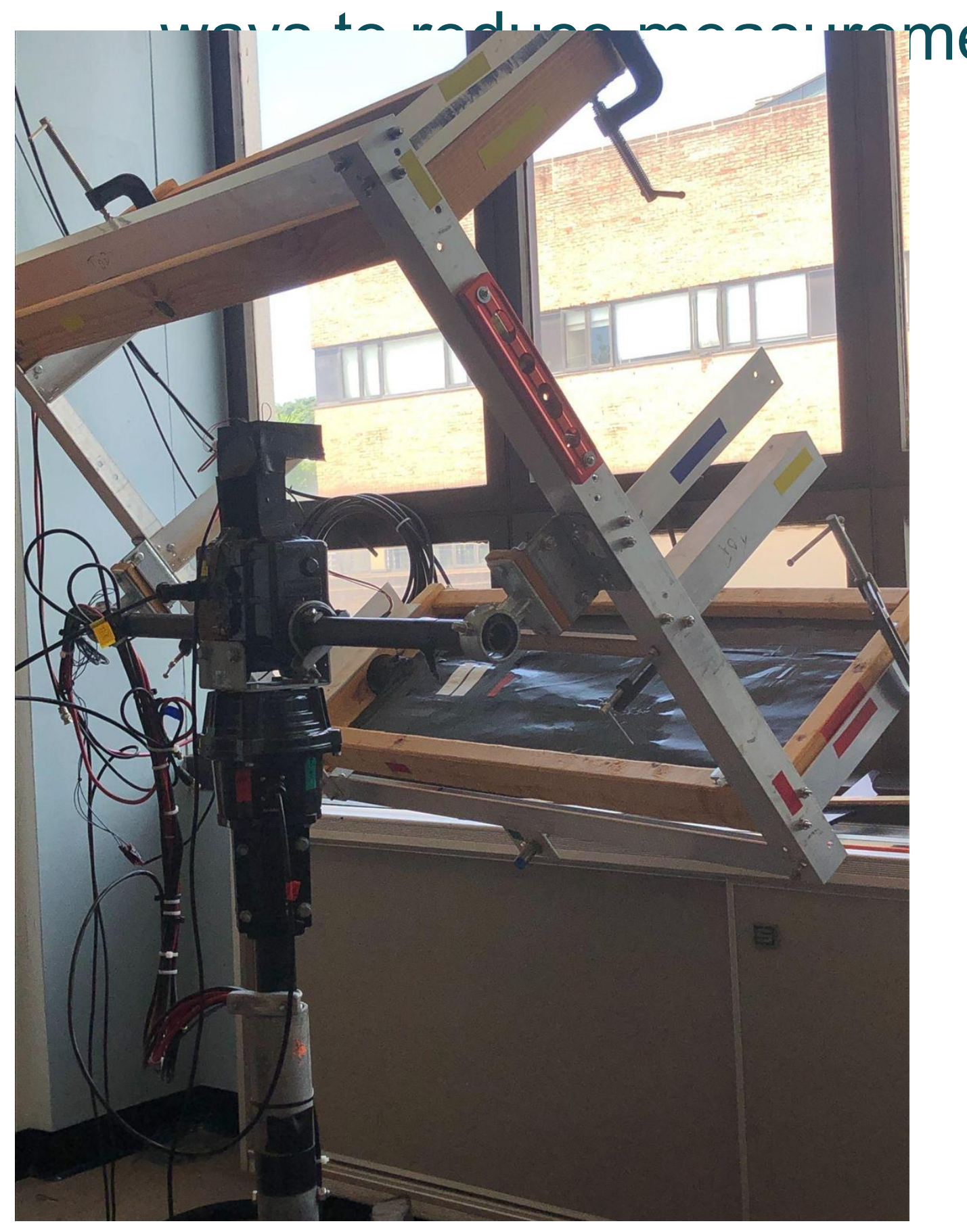
Abstract

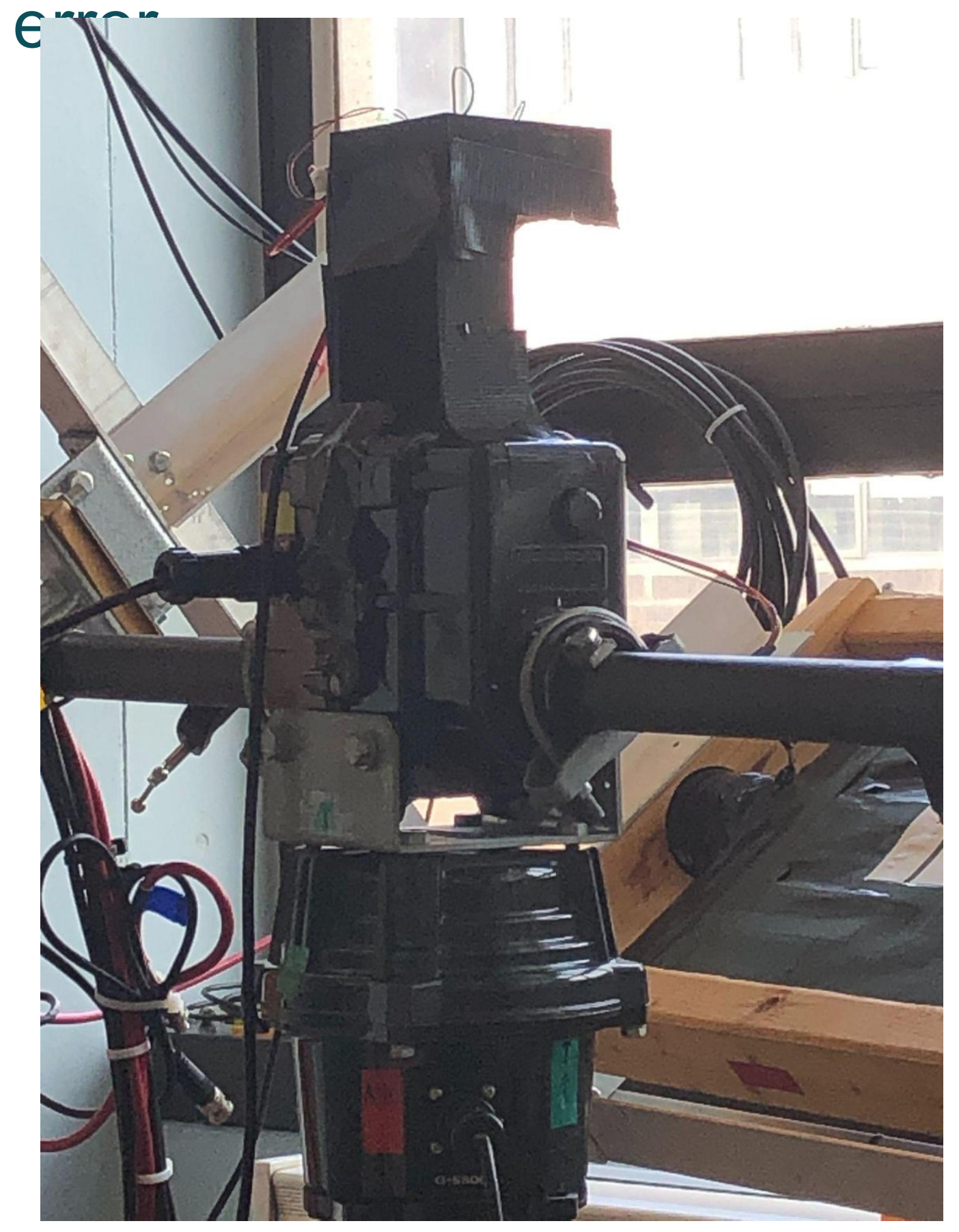
This project focuses on calibrating and improving the accuracy of azimuth and elevation measurements in a cosmic ray muon tracking telescope. Utilizing IMU sensors, Arduino microcontrollers, ARS-USB antenna controlling interface and real-time tracking software, we designed a system that can log positional data while automatically tracking the Sun. We identified and analyzed deviations in sensor outputs compared to calculated solar positions and proposed improvements to increase measurement reliability. Our ultimate goal is to develop a sensor system that can operate independently of magnetometers, which are

Introduction

The cosmic ray muon tracking detector is designed to observe the directional path of muons and track astronomical objects using elevation and azimuth coordinates. Prior implementations used IMU sensors that were prone to drift and electromagnetic interference, especially from surrounding electronics. Key components include:

- **IMU Sensors**: ISM330DHC (Gyroscope + Magnetometer) and Adafruit BNO055 (Absolute Orientation)
- **Rotator System**: Yaesu G5500 Antenna rotator paired with its own manual controller and ARS-USB antenna controlling interface
- **Tracking Software**: Nova for Windows provides real-time astronomical coordinates, Written python script calculating real time coordinate of the astrological object to perform auto tracking This project aims to quantify the accuracy of the sensor system and propose ways to reduce measurement error


The cosmic muon tracking telescope

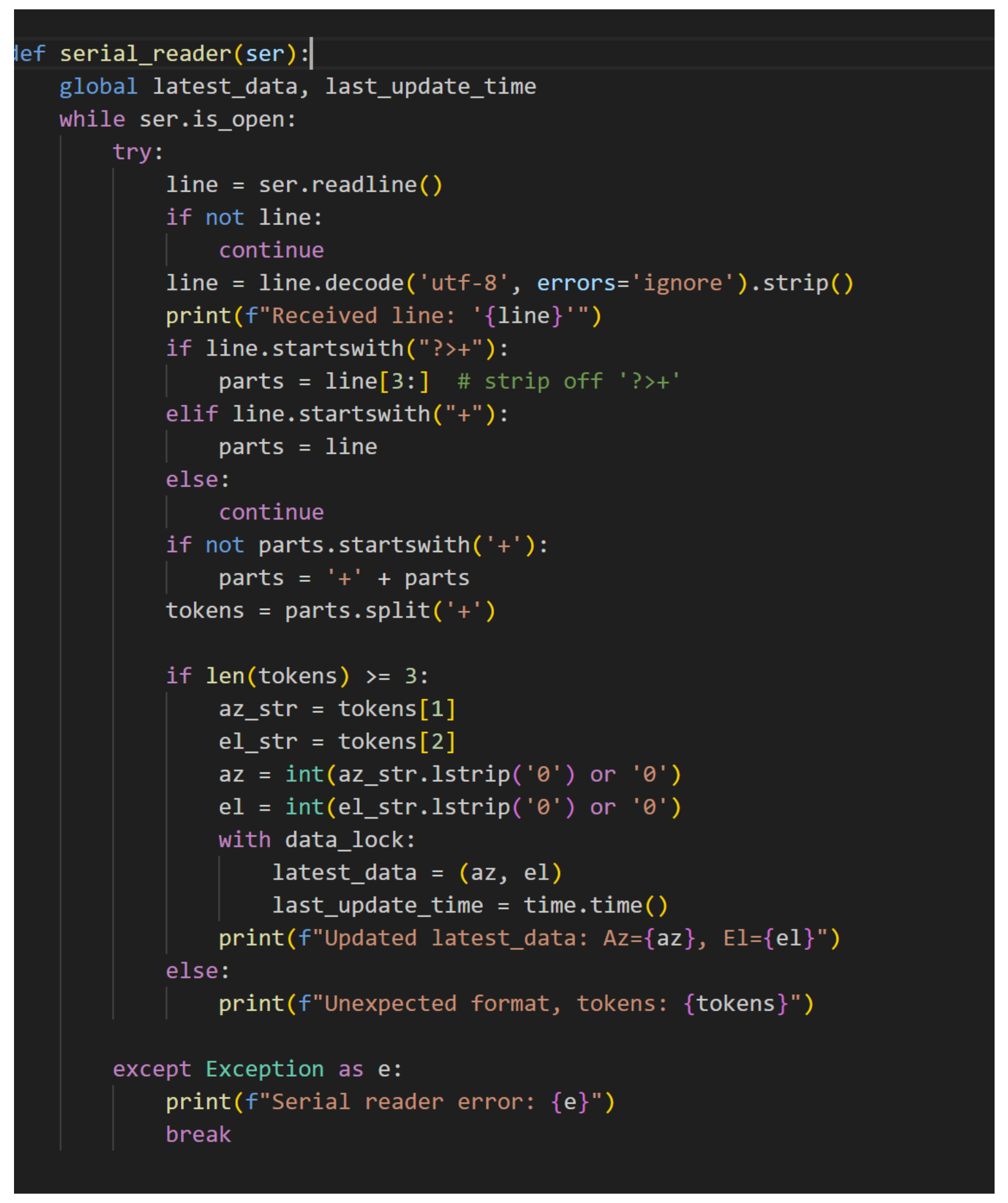

The Imu Sensors mounted upon horizontal Rotator

Methods and materials

- **Hardware**:
  - Arduino Uno R3
  - ISM330DHC IMU sensor (Azimuth measurement)
  - Adafruit BNO055 (Elevation and orientation)
  - Yaesu G5500 rotator
  - EATX ARS-USB controller
- **Software**:
  - Nova for Windows (for astronomical coordinates)
  - Python logger script (retrieves and logs C2 command output from ARS-USB)
  - Hyperterminal emulator for command testing
- **Test Environment**:
  - Sensors placed several inches above the rotator to minimize interference
  - Manual and auto tracking tests using Nova (1-min and 6-min intervals respectively)
  - Data collected over 3-hour or 24-hours sessions for analysis

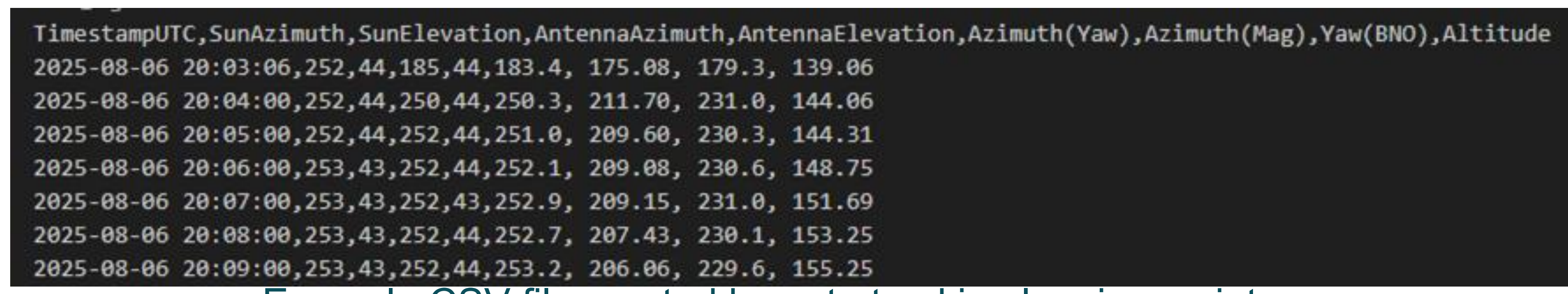
I later developed a Python script that removed the use of Nova and instead used the computer to actively calculate the sun's position. At the same time, I used serial commands to retrieve data from the ARS-USB to achieve a one-click recording and tracking effect in one script, greatly reducing the complexity of the operation and the possibility of errors.

Nova used to report astronomical coordinates of the selected astronomical object. The EATX ARS-USB rotator controller will stream the astronomical coordinate from nova to Yaesu controller which controls the antenna azimuth-elevation rotators, allowing detector to keep its track and direct itself facing the astronomical objects. While tracking record the azimuth and elevation from sensors with loggers, then comparing these data with calculated values to quantify the error sensors made.



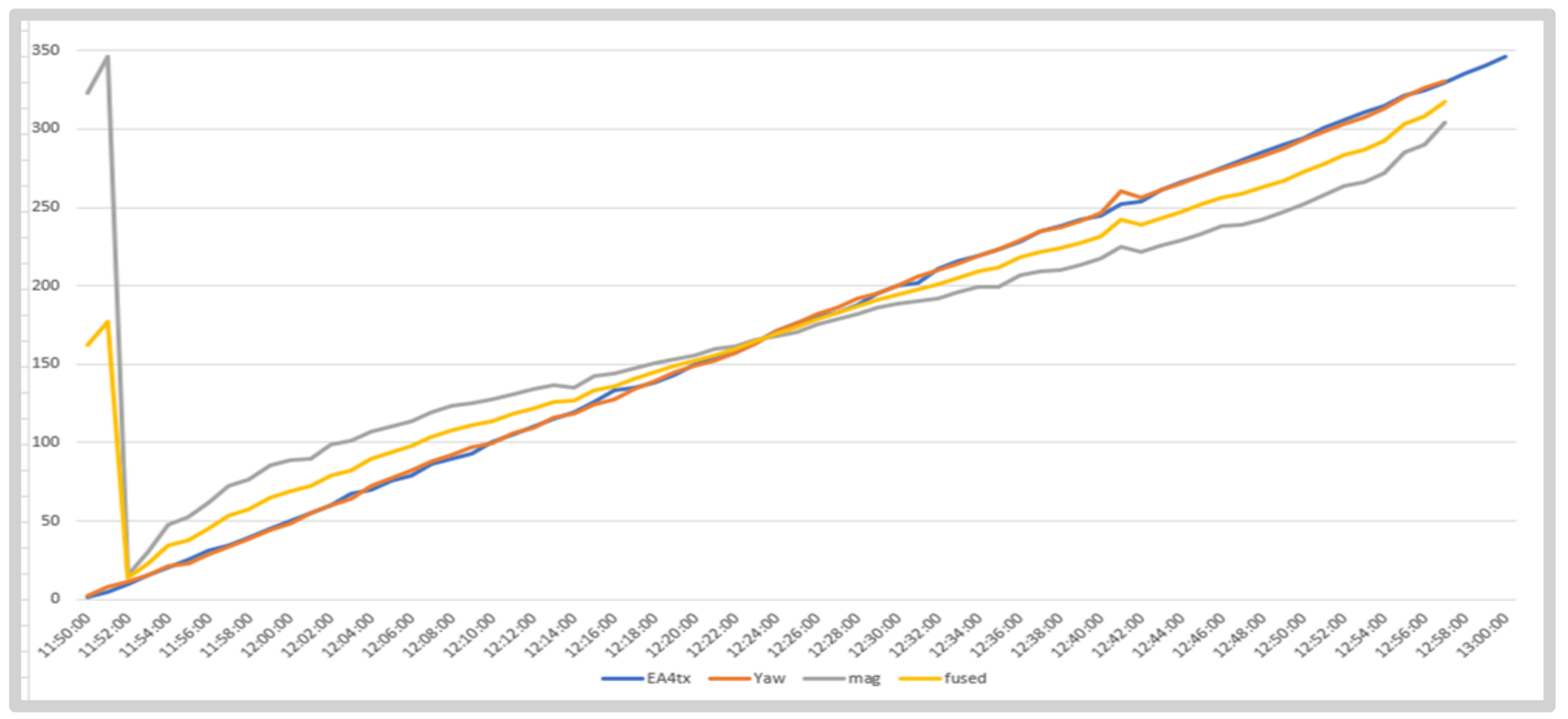
```python
def serial_reader(ser):
    global latest_data, last_update_time
    while ser.is_open:
        try:
            line = ser.readline()
            if not line:
                continue
            line = line.decode('utf-8', errors='ignore').strip()
            print(f"Received line: '{line}'")
            if line.startswith("?>+"):
                parts = line[3:]  # strip off '?>+'
            elif line.startswith("+"):
                parts = line
            else:
                continue
            if not parts.startswith('+'):
                parts = '+' + parts
            tokens = parts.split('+')

            if len(tokens) >= 3:
                az_str = tokens[1]
                el_str = tokens[2]
                az = int(az_str.lstrip('0') or '0')
                el = int(el_str.lstrip('0') or '0')
                with data_lock:
                    latest_data = (az, el)
                    last_update_time = time.time()
                print(f"Updated latest_data: Az={az}, El={el}")
            else:
                print(f"Unexpected format, tokens: {tokens}")

        except Exception as e:
            print(f"Serial reader error: {e}")
            break
```

The python script: reading and parse coordinate from the ARS-USB

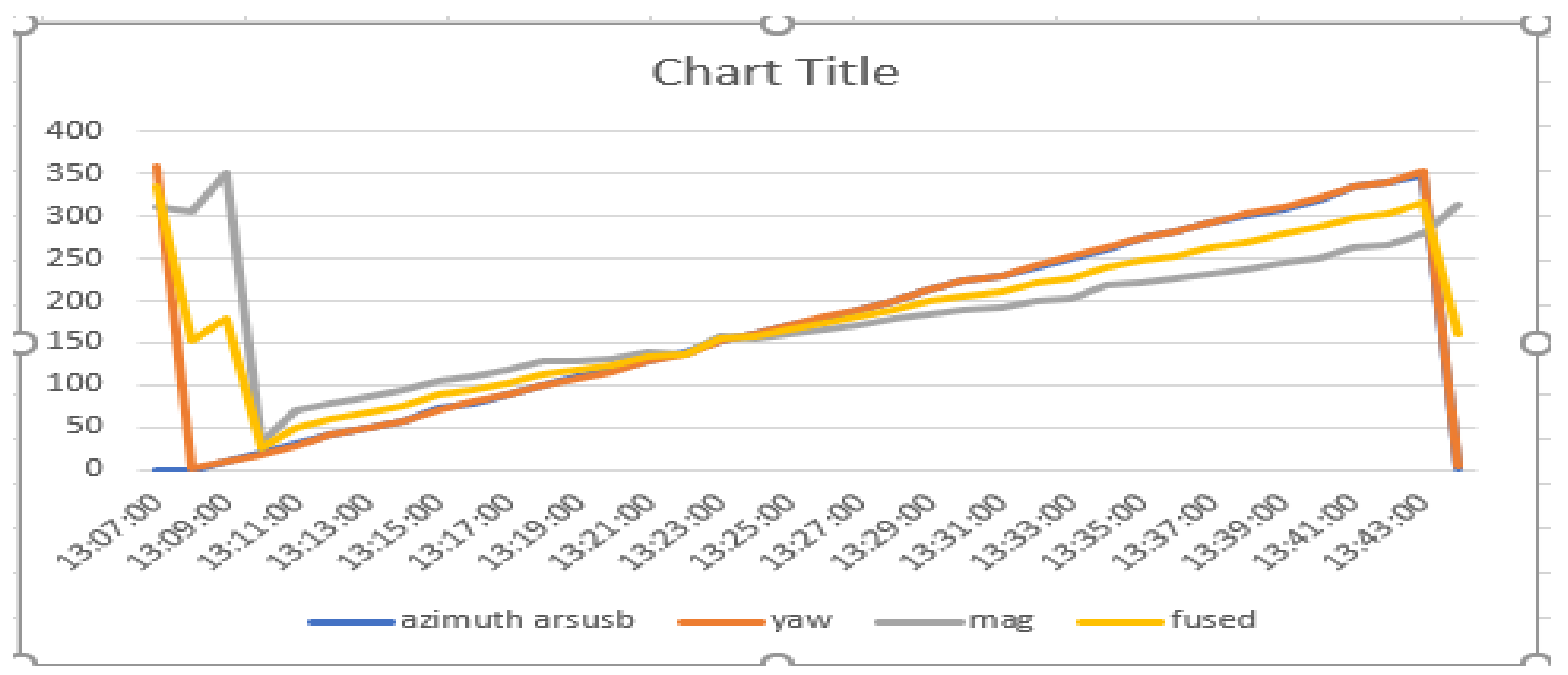


```
TimestampUTC,SunAzimuth,SunElevation,AntennaAzimuth,AntennaElevation,Azimuth(Yaw),Azimuth(Mag),Yaw(BNO),Altitude
2025-08-06 20:03:06,252,44,185,44,183.4, 175.08, 179.3, 139.06
2025-08-06 20:04:00,252,44,250,44,250.3, 211.70, 231.0, 144.06
2025-08-06 20:05:00,252,44,252,44,251.0, 209.60, 230.3, 144.31
2025-08-06 20:06:00,253,43,252,44,252.1, 209.08, 230.6, 148.75
2025-08-06 20:07:00,253,43,252,43,252.9, 209.15, 231.0, 151.69
2025-08-06 20:08:00,253,43,252,44,252.7, 207.43, 230.1, 153.25
2025-08-06 20:09:00,253,43,252,44,253.2, 206.06, 229.6, 155.25
```

Example CSV file created by auto tracking logging script



The logged data of Azimuth reading from ARS-USB(Blue), Gyroscope(Orange), Magnetometer(Gray), Fused Azimuth(Yellow) with auto tracking



The logged data of Azimuth reading from ARS-USB(Blue), Gyroscope(Orange), Magnetometer(Gray), Fused Azimuth(Yellow) with auto tracking with Manual Tracking

Results

- BNO055 Sensor Provided consistent and accurate elevation measurements
- ISM330DHC Gyroscope: Initial accuracy within 6 minutes, but significant drift observed beyond 18 minutes
- Magnetometer: Initial error higher than gyroscope but demonstrated better long-term consistency, a constant error in specific Azimuth has been found
- Fused Azimuth: A combination of both sensors (with tunable alpha parameter) was tested, showing a drifting trend as Gyroscope drifts

Discussion and Conclusion

The gyroscope, though accurate at the start, suffers from drift and is unreliable for long-duration tracking. The magnetometer is affected by environmental interference especially form azimuth from 60 degrees to 145 degrees, 225 degrees and 300 degrees, but is more stable over time by placing sensors higher off the rotator mitigated some electromagnetic interference. The fused algorithm approach offers a customizable balance but needs further calibration. The coordinate returned from ARS-USB is most accurate since its the reading from rotator itself, error oftens stays within 2-4 degrees.