MSD: Assignment 2

Sudhanva Joshi - 2023102022

15-03-2025

Note: Files are hosted on both Drive & GitHub. To run simulations, all file paths must be changed.

1 CAD Design

The following dimensions were considered for our links:

- Ground Link (L1) = **150 cm**
- Input Link/Crank (L2) = **75 cm**
- Coupler Link (L3) = **150 cm**
- Output Link/Rocker (L4) = **200 cm**

These parameters ensure that **Grashof's Condition** is satisfied:

$$S + L \leq P + Q$$

$$L2 + L4 \leq L1 + L3$$
$$75 + 200 \leq 150 + 150$$

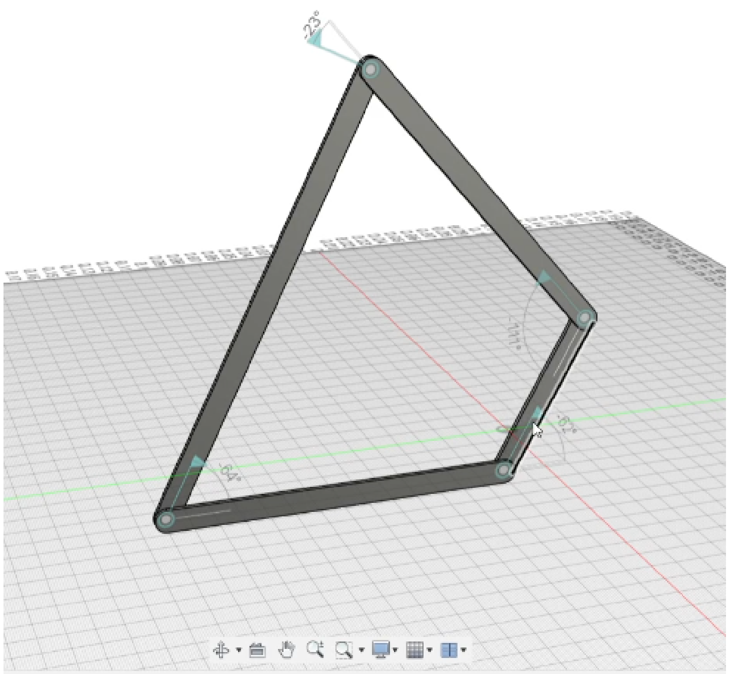


Figure 1: CAD Model of 4-Bar Mechanism

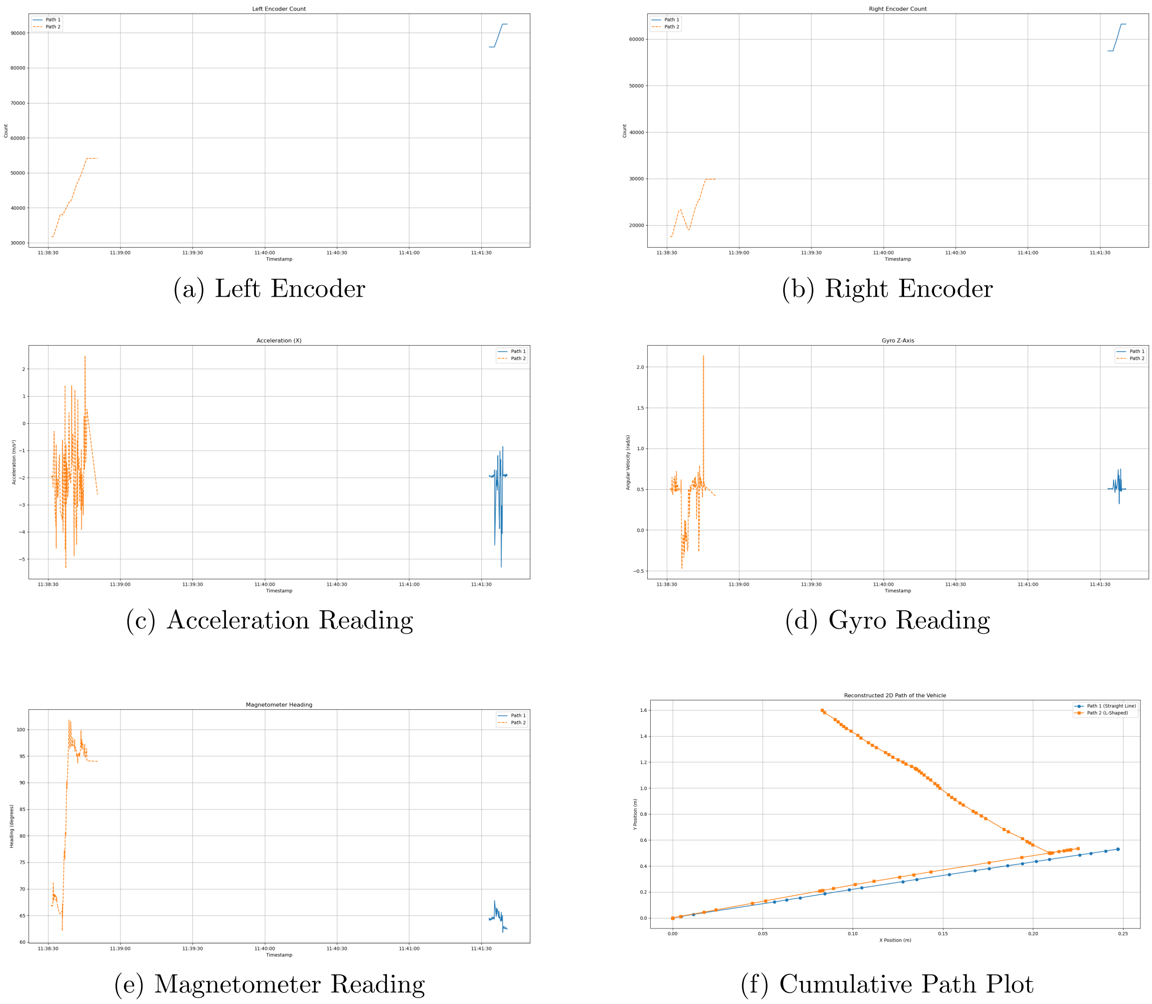
Refer to the video attached for the hitchless $360°$ *demonstration.*

2 Ground Vehicle Data Reconstruction

CSV Data Cleaning

- **Timestamp Conversion:**
  - The 'timestamp' column is converted to a datetime format using `pd.to_datetime`, ensuring compatibility with time-based operations.
  - Invalid or unparseable timestamps are handled gracefully with `errors='coerce'`, converting them to `NaT` (Not a Time) to maintain data integrity.
- **Missing Value Handling:**
  - Forward fill (`ffill`) is applied to propagate the last valid observation forward, ensuring continuity in the dataset.
  - Backward fill (`bfill`) is used to fill any remaining missing values with the next valid observation, further enhancing data completeness.
- **Result:** The dataset is cleaned to ensure no missing values, and timestamps are standardized into a consistent, usable format. This process also addresses uneven time intervals, making the data suitable for analysis and reconstruction tasks.

Given below are the comparative estimate graphs for both various sensor outputs of both paths.



(a) Left Encoder

(b) Right Encoder

(c) Acceleration Reading

(d) Gyro Reading

(e) Magnetometer Reading

(f) Cumulative Path Plot

Figure 2: Sensors & Reconstructed Data

Path Reconstruction Process

The trajectory is reconstructed using encoder and magnetometer data through the following steps:

- **Time Step Calculation:** The time difference between consecutive timestamps is computed as:

$$dt = t_i - t_{i-1}$$

  where t_i is the current timestamp and t_{i-1} is the previous timestamp.

- **Displacement Calculation:** The displacement for the left and right tracks is derived from encoder counts:

$$\Delta L = \Delta\text{left\_encoder\_count} \cdot \text{PULSE\_TO\_MM}/1000$$

$$\Delta R = \Delta\text{right\_encoder\_count} \cdot \text{PULSE\_TO\_MM}/1000$$

  where ΔL and ΔR are the displacements in meters.

- **Orientation Change:** The change in orientation ($\Delta\theta$) is calculated using the track width (W):

$$\Delta\theta = \frac{\Delta R - \Delta L}{W}$$

- **Heading Integration:** The magnetometer heading (θ) is used as the primary orientation:

$$\theta = \text{heading} \quad \text{(in radians)}$$

- **Average Displacement:** The average displacement (ΔD) is computed as:

$$\Delta D = \frac{\Delta L + \Delta R}{2}$$

- **Position Changes:** The changes in x and y positions are calculated using trigonometry:

$$\Delta x = \Delta D \cdot \cos(\theta)$$

$$\Delta y = \Delta D \cdot \sin(\theta)$$

- **Cumulative Position:** The cumulative x and y positions are obtained by integrating the changes:

$$x = \sum \Delta x, \quad y = \sum \Delta y$$

This process synthesizes encoder and magnetometer data into an approximate 2D trajectory, enabling accurate path reconstruction.

3 LIDAR, ToF & Camera Analysis

3.1 LIDAR

Note: To confirm the 2D Point Cloud, the *.pcd* file was visualized using *Blender* as well.

Figure 3: 2D Point Cloud Form, Open3D

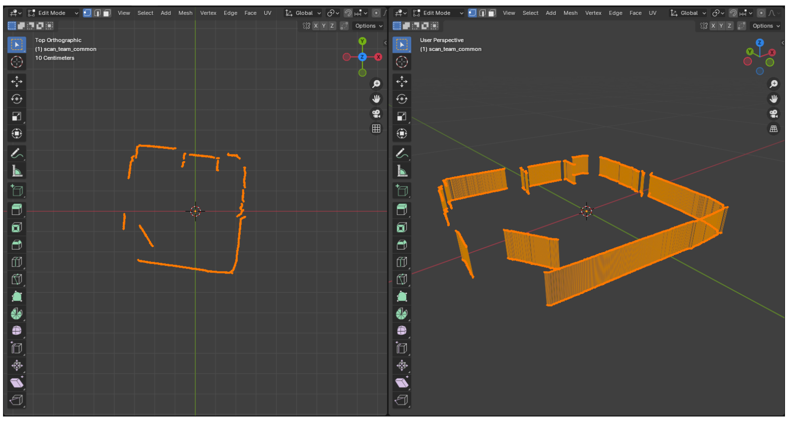


Figure 4: Blender Visualization, pcd files imported using this plugin

Detection of Black Objects in 2D LIDAR Point Cloud

- **Issue:** Black objects are not detected in the plotted LIDAR data.
- **Reason:** Black surfaces have low reflectivity, absorbing most incident light, especially at LIDAR's near-infrared wavelengths (e.g., 905 nm or 1550 nm).
- **Result:** Insufficient reflected signal for detection, leading to missing data points.
- **Contributing Factors:**
  - Low reflectivity of black surfaces.
  - Limited sensitivity of LIDAR sensors to weak or absent reflections.
  - Potential interference from ambient light or noise.

3.2 ZED Camera

3D Reconstruction from Depth Heatmap

- **Input:** Depth heatmap (HSV format) and RGB color image.
- **Depth Extraction:** Hue channel of HSV heatmap normalized to depth range $[0.1\,\mathrm{m}, 10\,\mathrm{m}]$:

$$d = \left(1 - \frac{H}{180}\right) \cdot (d_{\max} - d_{\min}) + d_{\min}$$

- **3D Point Cloud Generation:**
  - Camera intrinsics (f_x, f_y, c_x, c_y) map pixel (u, v) to 3D coordinates (x, y, z):

$$x = \frac{(u - c_x) \cdot z}{f_x}, \quad y = \frac{(v - c_y) \cdot z}{f_y}$$

  - RGB color assigned to each 3D point.
- **Output:** .ply file containing 3D point cloud with color.

The following results are observed from the 3D reconstruction using both methods:

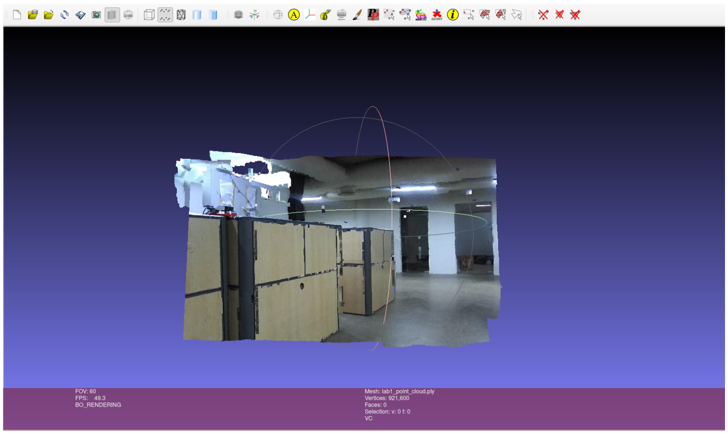


Figure 5: Lab 1, Method 1

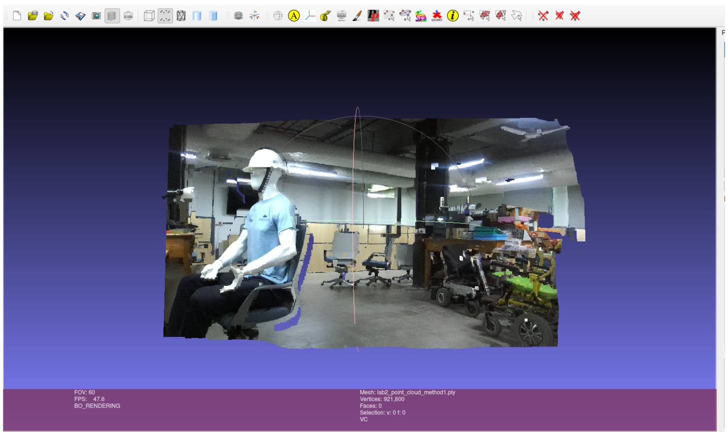


Figure 6: Lab 2, Method 1

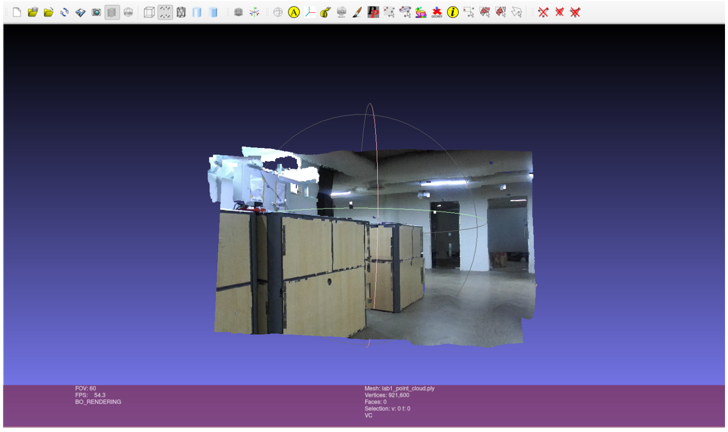


Figure 7: Lab 1, Method 2

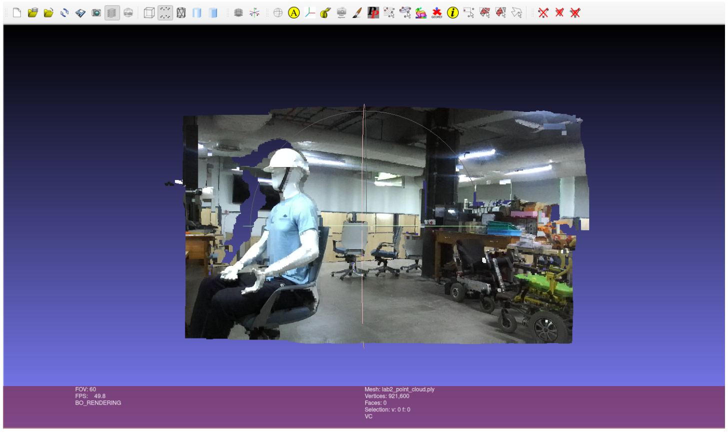


Figure 8: Lab 2, Method 2

Differences Between Depth Matrix and Heatmap-Based Reconstructions

- **Method 1 (Depth Matrix):**
  - Directly uses precomputed depth values from a depth matrix.
  - Higher accuracy as depth values are explicitly provided.
  - No additional processing or conversion steps required.
- **Method 2 (Heatmap):**
  - Converts a heatmap (HSV) into a depth matrix using hue values.
  - Introduces potential inaccuracies due to heatmap color-to-depth mapping.
  - Additional processing step (HSV conversion and normalization) required.
- **Observed Differences:**
  - Method 1: Cleaner, more precise point cloud with fewer artifacts.
  - Method 2: May exhibit noise depending on heatmap interpolation and color-depth mapping.

Reconstruction Complexity

- **Lab 1:** Simple scene with fewer objects, uniform lighting, and minimal occlusion, leading to cleaner and more accurate reconstructions.
- **Lab 2:** Complex scene with more objects, varied lighting, and increased occlusion, resulting in noisier and less precise reconstructions.

3.3 Time-of-Flight

Working Principle

Emits light (e.g., infrared laser) to target. Measures time t for light to reflect back. Distance d calculated as:

$$d = \frac{c \cdot t_{flight}}{2}$$

where c = speed of light. High accuracy, unaffected by target color/reflectivity. Ideal for short-medium range applications.

Possible Reasons for Discrepancies

- **Sensor Limitations:** Accuracy, resolution, calibration errors, or electronic/thermal noise.
- **Target Properties:** Low reflectivity or uneven surfaces reduce signal strength.
- **Environmental Interference:** Ambient light, temperature, or multipath reflections.

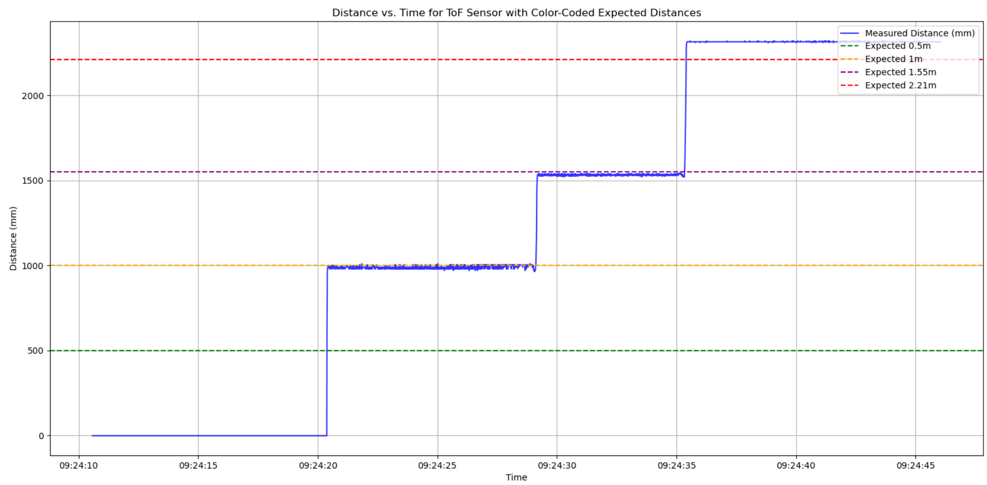


Figure 9: Estimated Distance vs Time, ToF Sensor