Pole Detection and Midpoint Calculation for Autonomous Robot

1. Problem Statement

In this scenario, a Robot is designed to operate underneath solar panels to trim grass or plants that have grown in the area. The robot is equipped with a LiDAR sensor and relies on the solar panel support poles as landmarks or reference points to determine equidistant midpoints. These calculated midpoints are then used to generate paths for the robot to follow, enabling autonomous mowing.

2. Challenges Encountered

A significant challenge arises when the grass or plants beneath the solar panels grow taller than expected. The LiDAR system mistakenly identifies these tall plants or grass as additional poles, treating them as false landmarks. As a result, the robot's software calculates midpoints using this incorrect data, which leads to inaccuracies in path generation. Consequently, the generated paths deviate from the intended straight, narrow lines, causing the robot to cross paths irregularly instead of maintaining consistent and precise paths.

3. Approaches to Mitigate the Issue

To address the problem of the LiDAR system mistakenly identifying tall grass or plants as poles, I propose a systematic approach using advanced point cloud processing techniques. The following steps outline how to refine pole detection using effective point cloud libraries and filtering strategies.

3.1 Identifying True Poles using Effective Point Cloud Techniques

This approach focuses on enhancing the accuracy of pole detection by employing a series of filtering and clustering techniques, allowing the system to differentiate between genuine pole structures and other vertical elements like tall grass or plants.

Step 1: Ground Plane and Noise Filtering

- **Objective:** Remove the ground plane and noisy data points to focus on the vertical structures.
- **Method:** Apply the RANSAC plane segmentation technique to identify and exclude points that belong to the ground surface. This step ensures that the point cloud data only contains elements standing above the ground.

Step 2: Pass-Through Filter for Height-Based Removal

- **Objective:** Filter out low-height objects to prevent small grass or plants from being considered as poles.
- **Method:** Use a pass-through filter on the Z-axis (height) to exclude points below a specific height threshold (e.g., 0.5 meters) from the ground. This helps in eliminating minor vertical elements that do not represent actual poles.

Step 3: Height Filtering for Vertical Structures

- **Objective:** Focus only on vertical structures that fall within a specified height range.
- **Method:** Define a height range(e.g., 0.5m to 2.0m) to filter the point cloud, targeting structures that have a significant vertical extent. This ensures that only taller objects (like poles) are considered for further analysis, reducing the influence of non-pole-like objects.

Step 4: Euclidean Cluster Extraction for Pole Identification

- **Objective:** Segment the point cloud into clusters to identify individual poles.
- **Method:** Perform Euclidean Cluster Extraction to group points into clusters. Set the "**minimum and maximum number of points per cluster"** to define what constitutes a valid pole-like structure. This step is crucial as it helps eliminate small grass clusters or large non-pole obstacles by only considering clusters with a size that matches the expected characteristics of poles.

Step 5: Cluster Analysis to Confirm Pole-Like Structures

- **Objective:** Identify genuine poles by analyzing the vertical extent of the clusters.
- **Method:** Calculate the height of each cluster and evaluate whether it matches the expected dimensions of a pole. If the cluster has a significant height within the defined range, classify it as a pole. Use these identified poles to calculate centroids for both the right and left poles, then determine the midpoint between them for path generation relative to the robot's current position.

Note: Please refer to the attached pseudocode file that was followed to implement this approach.

3.2 Path Generation with Midpoint Calculation

To ensure accurate navigation, midpoints are calculated only when both left and right pole centroids are detected within the LiDAR's field of view and in front of the robot:

1. **Midpoint Calculation:** Calculate the midpoint only if both left and right poles are present in the immediate vicinity ahead of the robot.
2. **Midpoint Alignment:** The calculated midpoint's Y-coordinate is adjusted to ensure it remains constant or zero to keep the robot's path straight, which is essential for mowing in clean, narrow lines.

3.3 Straight-Line Path Tracking using Bezier Line

In this step, a straight-line path is generated to ensure that the robot follows an accurate trajectory from its current position to the calculated midpoint:

1. **Straight-Line Path Calculation:** A Bezier line or straight-line formula is used to generate the path from the robot's current position to the calculated midpoint. This helps in maintaining a direct and predictable path for the robot to follow.
2. **Visualization:** The straight-line path is published as a marker in RViz for real-time monitoring, aiding in verifying the robot's planned trajectory.

This structured approach, incorporating point cloud processing and targeted filtering, will significantly improve the robot's ability to distinguish true poles from false landmarks, ensuring more reliable and precise path generation and tracking.

Note: Out of curiosity, I developed a code based on this approach and tested it using tools like Gazebo Sim, ROS 2, C++, and Rviz. I created a small Gazebo world model using simple gazebo elements and utilized a small robot equipped with a 3D LiDAR to test the approach. Please refer to the **attached video** for a better visual understanding for this approach. And also refer to the attached **pseudo code** file to analyze the implementation strategy followed to implement this approach.

Another simplest approach:

Using Layout Map Data for Pole Detection

An alternative and more straightforward approach to enhancing pole detection is by utilizing the layout map of the solar plant.

Approach Outline

Step 1: Utilize the Layout Map Information

- **Objective:** Use the pre-existing layout map data of the solar plant to precisely identify the poles' positions.
- **Method:** Reference the known coordinates of poles from the layout map during the detection process. By comparing the LiDAR data with the mapped pole locations, the system can verify if a detected structure aligns with the expected pole position.

Step 2: Midpoint Calculation Between Known Poles

- **Objective:** Simplify the midpoint calculation by using the fixed positions of poles from the map.
- **Method:** Calculate the midpoint between poles using their known distances from the layout map, ensuring more precise and consistent midpoint estimation.