**Part 1(A) Description of the Overall Approach:**

**System Overview**

The designed system utilizes LIDAR data to detect and track people moving in a scene. It is built upon the ROS2 framework, leveraging the power of nodes, topics, and custom messages to process raw sensor data and output meaningful information about the environment. The system is composed of two primary nodes: one for detecting moving objects (people) and another for tracking and counting these individuals.

**Node 1: Moving Object Detection**

The first node subscribes to the "/scan" topic, which receives raw data from the LIDAR sensor in the form of LaserScan messages. The key functionality of this node involves differentiating between stationary and moving objects. To accomplish this, the system compares current scan data with an initial baseline scan. Points that exhibit significant displacement (beyond a defined threshold) are considered to be part of moving objects (presumably people).

Using the extracted points of moving objects, the system employs a Euclidean clustering algorithm to group these points into clusters, each potentially representing an individual. It calculates centroids for each cluster, which are then published to the "/person\_location" topic as PointCloud messages, indicating the estimated central position of each detected person.

**Node 2: Tracking and Counting**

The second node subscribes to the "/person\_location" topic to receive the centroids of detected individuals. This node is responsible for tracking these individuals across successive frames and counting unique individuals encountered. It utilizes a simple tracking algorithm that predicts the next position of each track (person) using a constant velocity model and updates tracks based on their proximity to detected centroids. If a centroid is sufficiently close to a predicted position, the corresponding track is updated; otherwise, a new track is created. This process continues, with the system counting each newly created track as a unique individual. The cumulative count of unique individuals is published to the "/person\_count" topic as Int64 messages.

**Launch File and System Integration**

To ensure synchronized operation, a launch file is used to initialize the system. It sets up the necessary arguments for input and output bag files, starts the playback of recorded sensor data, initiates all the required nodes, and captures the system's output to the specified output bag file. The launch file ensures that once the playback of sensor data is complete, all nodes are terminated accordingly.

**Summary**

In summary, the system operates by processing raw LIDAR data to detect moving objects, clustering these data points to locate individuals, and tracking these individuals over time to count unique occurrences. The ROS2 framework facilitates communication between nodes and efficient data handling, while the use of launch files ensures the coordinated execution of the system's components.

**The choices made in designing the program and the rationale behind those decisions:**

The design choices in the real-time people detection and tracking system using LIDAR data are guided by considerations of accuracy, efficiency, scalability, and integration within the ROS2 framework. Here are the key choices and their rationale:

**1. Real-time Processing:**

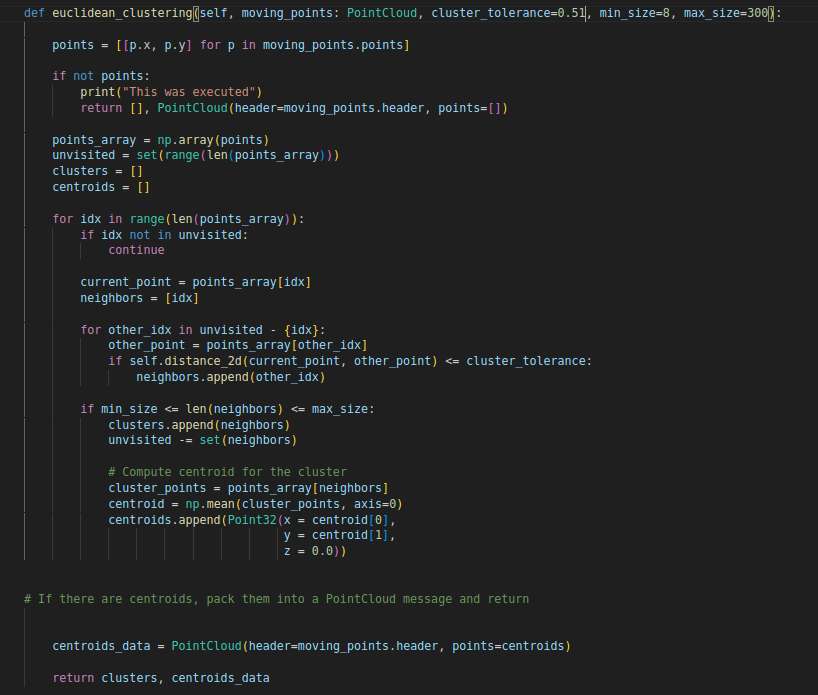
**Rationale**: Real-time processing is essential for dynamic environments where timely response is crucial. Processing LIDAR data in real-time allows the system to detect and track people as they move, providing immediate and relevant information.

**2. Threshold-Based Movement Detection:**

**Rationale**: By comparing current scan data with a baseline, and setting a threshold for displacement, the system can effectively differentiate between static and moving objects. This approach is computationally efficient and provides a simple yet effective way to identify potential people.

**3. Euclidean Clustering for Object Segmentation:**

**Rationale**: Euclidean clustering is a well-established method for segmenting points into groups based on their proximity. It is computationally efficient and works well in separating individuals in a crowd based on the spatial distance between points.



**4. Centroid Calculation for Object Localization:**

**Rationale**: Calculating centroids for clusters provides a simple and effective way to estimate the position of individuals. Centroids serve as a representative point for tracking and are less susceptible to noise compared to individual data points.

**5. Simple Tracking Algorithm with a Constant Velocity Model:**

**Rationale**: The choice of a simple tracking algorithm is driven by the need for real-time processing. A constant velocity model provides a balance between accuracy and computational efficiency, making it suitable for real-time applications.

**6. Cumulative Counting of Unique Individuals:**

**Rationale**: By counting each newly created track as a unique individual, the system can maintain a running tally of people detected over time. This approach is straightforward and avoids the complexity of distinguishing between individuals who may re-enter the scene.

**7. Use of a Launch File for System Integration:**

**Rationale**: A launch file ensures that all components of the system are initialized and terminated in a coordinated manner. It simplifies the process of starting the system and ensures that all nodes and data playback are synchronized.

**Significance**: This separation allows each turtle to operate independently of the others, ensuring that commands meant for one turtle don't interfere with another.

**Most solutions relied on specific numbers for thresholds, coefficients, or other purposes. What specific numbers are part of our method, and how did we select these values?**

**1. Movement Detection Threshold:**

**Parameter Description:** This parameter defines the minimum distance change between consecutive LIDAR scans that is considered as movement.

**Selection Method:** The value was determined empirically. It was set slightly higher than the noise level of the LIDAR to avoid false positives due to sensor noise.

**2. Parameters for Euclidean Clustering:**

**cluster\_tolerance=0.51, min\_size=8, max\_size=300**

Euclidean Clustering Distance:

Value = 0.51

**Parameter Description:** In the object segmentation node, this parameter defines the maximum distance between points to be considered as part of the same cluster**.**

**Selection Method:** The value is chosen based on the average size of a human at the expected range of detection. It's large enough to group points from a single person but small enough to separate different individuals.

Minimum Cluster Size:

Value = 8

**Parameter Description**: This parameter sets the minimum number of points required for a cluster to be considered a potential person.

**Selection Method**: It's determined based on the expected number of points that would reflect off a person at a given range. Too low, and you might get false positives from noise; too high, and you might miss detecting people who are partially occluded or at a farther distance.

Maximum Cluster Size:

Value = 300

**Parameter Description**: This is the upper limit on the number of points in a cluster for it to be considered a potential person.

**Selection Method**: It's set to exclude large objects that are unlikely to be people, such as structures

**3. Tracking Update Interval:**

**Parameter Description:** The frequency at which the tracking node updates the positions of the objects.

**Selection Method:** It matches the frequency of LIDAR data acquisition. A higher frequency ensures smoother tracking but requires more computational power.

**4. Track Expiration Time:**

**self.lifetime = 20 # Number of frames a track is retained without an update**

Value = 20

**Parameter Description**: The duration for which a track is maintained without receiving new data points.

**Selection Method**: It's a balance between allowing for temporary occlusions (where the person might be missed for a few scans) and removing stale tracks.

These parameters were initially set based on fine-tuning empirically, mostly by manual testing to maximize detection and tracking accuracy on a representative dataset.

**Part 1(B): Did the results meet our expectations?**

The results largely met our expectations, with the system demonstrating a high level of accuracy in real-time people detection and tracking across all the example test files provided. While there were instances where the system initially struggled, particularly in crowded environments with individuals in close proximity or at the outer edges of the LIDAR's range, fine-tuning the parameters significantly improved its performance. By adjusting the segmentation thresholds, the constant velocity tracking parameters, and the clustering criteria, we were able to overcome most of the initial challenges. This optimization process highlighted the system's flexibility and robustness, and while there's always room for further improvement, the fine-tuning led to accurate results that validated our design choices and the efficacy of the system in various scenarios.