EECE 5554: Robotics Sensing and Navigation

Final Project

LIO-SAM: Tightly coupled Lidar Inertial Odometry via Smoothing and Mapping

Group Members: Adithya Ramesh, Chirag Dhoka Jain, Narayanan Ramasamy, Shayak Bhadraray

1. Introduction:

Lidar-based simultaneous localization and mapping (SLAM) methods have become popular for accurate 6DOF state estimation. However, methods like lidar odometry and mapping (LOAM) suffer from drift over long durations and lack flexibility for sensor fusion. We thus investigate a lidar inertial odometry framework via smoothing and mapping (LIO-SAM) based on a factor graph optimization approach to address these challenges. Here are some of the **key claims** made in the LIO-SAM paper:

- Proposes a tightly coupled framework that achieves highly accurate, real-time mobile robot trajectory estimation and mapping by smoothing lidar inertial measurements in a factor graph formulation.
- The local sliding window matching approach enables real-time performance by avoiding costly global map matching, instead registering new keyframes to only a fixed-size set of recent sub-keyframes.
- The factor graph formulation allows straightforward incorporation of various sensor constraints like IMU, lidar, GPS, and loop closure for joint optimization and robustness.
- Extensive real-world evaluation showing state-of-the-art accuracy compared to prior lidar-inertial methods like LOAM and LIOM across datasets from different platforms and scales.
- Capable of real-time performance up to 13x faster than real-time rates on the tested datasets, enabling applicability to computationally constrained robotic platforms.

The key innovations of this algorithm are an efficient sliding window-based matching approach and a factor graph formulation that enables smoothing poses via robust sensor fusion and constraints. In summary, the key claims are around the accuracy, efficiency, versatility, and robustness enabled by the proposed tight integration and factor graph optimization approach for lidar inertial odometry and mapping. Hence, in this project report, we aim to investigate the key claims made by the paper and test the effectiveness of the LIO-SAM algorithm for mapping on dataset collected on the autonomous car (NUance) and the dataset uploaded by the publisher of the algorithm.

2. Experiments:

LIO-SAM was thoroughly evaluated using datasets from a ground vehicle (NUance) and the other dataset provided by the author. We tested the robustness of the algorithm using different scenarios and there were some issues faced while running it on the NUance dataset. These are discussed below:

2.1. Issues Encountered:

Set-up and configuration issues:

The original LIO SAM algorithm was developed for ROS '*Melodic*'. To run the algorithm on ROS '*Noetic*' we had to make some changes,

- OpenCV version compatibility: We updated the openCV version in the utility.h file from ***'#include <opencv/cv.h>'*** to ***'#include <opencv2/opencv.hpp>'.***
- CMake version compatibility: The ***'set(CMAKE_CXX_FLAGS"-std=c++11")'*** was changed to ***'set(CMAKE_CXX_FLAGS "-std=c++14")'.***
- Gtsam version compatibility: We observed occasional crashes during the map optimization process. Ensuring the correct version of GTSAM helped mitigate this issue to some extent.

Debugging and runtime issues:

Remapping the rostopics: Rostopics were remapped as required by the above-described convention while executing *roslaunch* commands and playing the data bags.

Real-Time Performance on our systems: One of the central claims of LIO-SAM is its ability to run up to 10 times faster than real-time. However, in our experiments, we observed instances where the algorithm struggled to meet real-time processing requirements, especially in scenarios with dense point clouds or rapid changes in the environment. Hence, the claimed real-time performance was challenging to achieve. Reducing the playback rate of a ROS bag file during experimentation can have an impact on the performance and results of algorithms like LIO-SAM for several reasons:

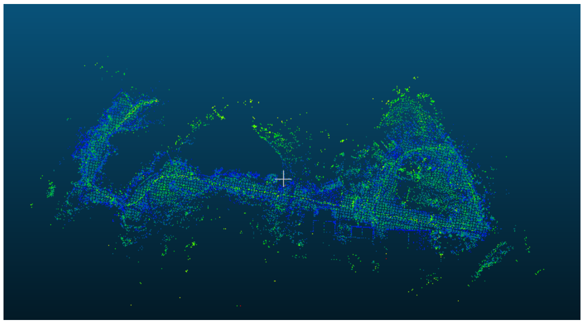
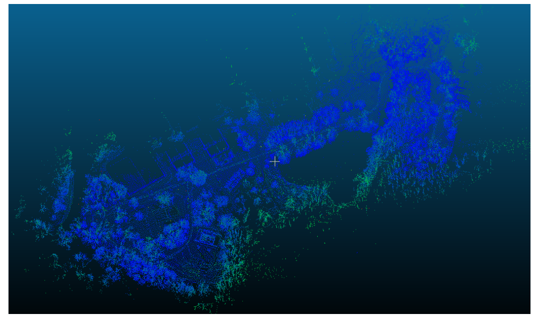
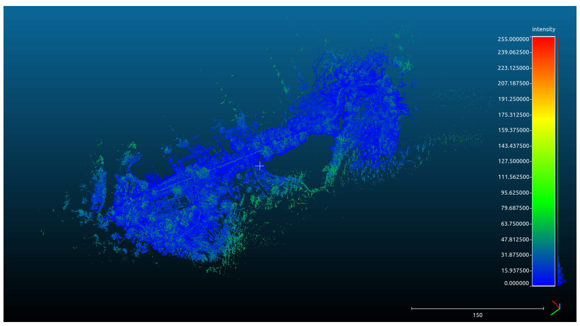
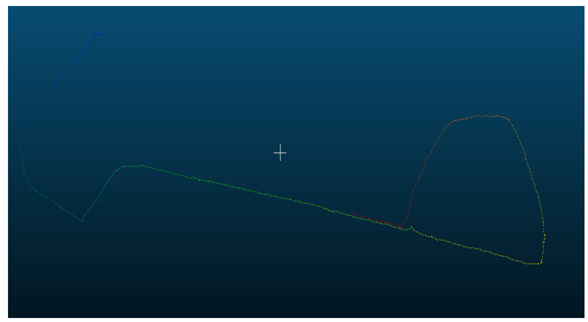
- **Random offset behavior in Odometry Estimates:** We observed a randomized offset in the odometry estimates, suggesting intermittent misalignments. Certain tweaks to the synchronization algorithm were made, and the playback speed during rosbag playback (rosbag play) was adjusted to improve temporal consistency.
- **Synchronization of Sensor Data:** ROS bags contain recorded sensor data with timestamps. Lowering the playback rate allows more time for the system to process and synchronize lidar and IMU data accurately. This can be crucial for algorithms that rely on precise temporal alignment between different sensor modalities.
- **Real-Time Constraints:** Some algorithms, including LIO-SAM, may have real-time processing constraints. Lowering the playback rate provides the algorithm with more time to process each time step, potentially reducing the computational load and allowing the algorithm to meet real-time requirements more effectively.
- **Algorithmic Stability:** Lowering the playback rate can enhance algorithmic stability. Some algorithms, especially those involving optimization and mapping, may benefit from a slower processing pace to maintain stability during complex computations. This is particularly relevant when dealing with resource-intensive tasks like map optimization.
- **Debugging and Visualization:** Slowing down the playback rate facilitates easier debugging and visualization. Users can observe the algorithm's intermediate states, inspect visualizations in real time, and identify potential issues more effectively at a slower pace.
- **Simulation of Real-Time Conditions:** In a real-time system, sensor data arrives at a fixed rate. Lowering the playback rate simulates this real-time condition more accurately, allowing users to evaluate how well the algorithm performs under the constraints it would face in an operational setting.
- **Parameter Tuning:** Some algorithms might have parameters that need fine-tuning. Lowering the playback rate provides more time for parameter adjustments and allows users to observe the impact of these changes on the algorithm's performance.

While decreasing the playback rate can help with debugging and stability, it may not always increase accuracy. Although LIO-SAM exhibited significant advantages in managing lidar and IMU data for odometry estimation, our investigations revealed computational and algorithmic stability issues. A careful balance between parameter adjustment, downsampling techniques, and hardware consideration was necessary to address these challenges.

2.2 Experimental Execution:

- **Running the Package:** We launched the LIO-SAM package using the provided run. launch file and play our custom dataset and the different datasets provided by the author.
- **Observations:** The LIO-SAM algorithm successfully processed the lidar and IMU data, providing real-time odometry estimates for the dataset provided by the author, but as discussed above, while running the algorithm on the NUance car dataset we had to play it at a lower rate.

The below visualization using CloudCompare software is taken from the saved point cloud maps produced after mapping the given datasets by the algorithm.

 Fig 1.a	 Fig 1.b
Here the points correspond to the corner/distinctive features of the environment.	Here the points correspond to the surface/continuous regions in the environment.
 Fig 1.c	 Fig 1.d
Visualization of the resulting trajectory.	Global map: Provides a high-level overview of the spatial layout, structure, and key features of the environment.

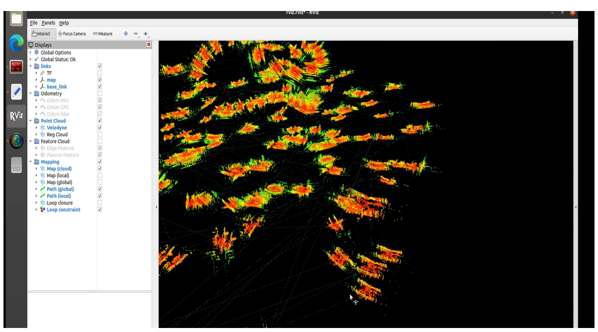


Fig 2.a

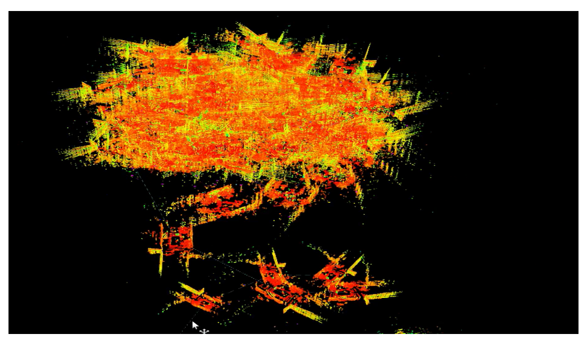


Fig 2.b

Our initial attempts at visualizing the mapping results of the NUANCE car dataset using Rviz. The resulting visualization is skewed over time. Further, problems in the IMU preintegration factor can also be inferred by observing the skewed point cloud maps.

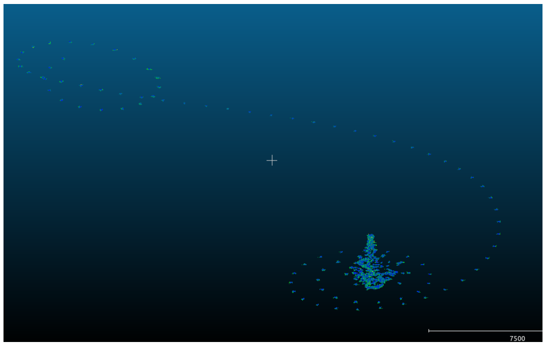


Fig 2.c

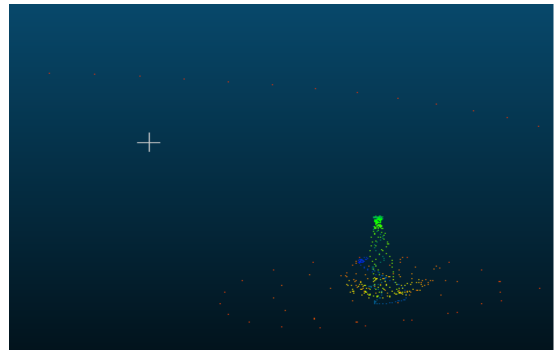


Fig 2.d

The above images represent the global map and the trajectory taken. They are quite random. We cannot make much sense of them.

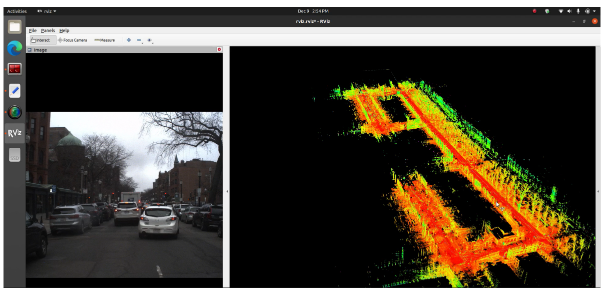


Fig 3.a

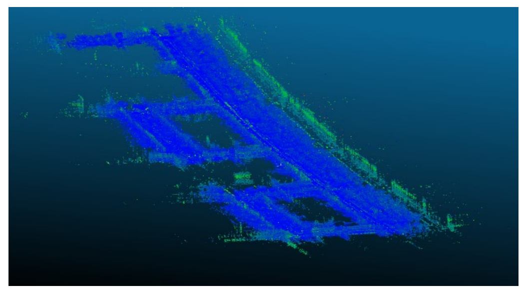


Fig 3.b

To overcome the issues we faced, we systematically tweaked the playback rate and other important parameters of the rosbag being played. These images show the corrected Rviz visualization

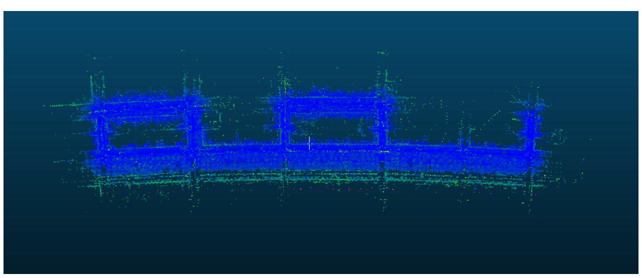


Fig 3.c

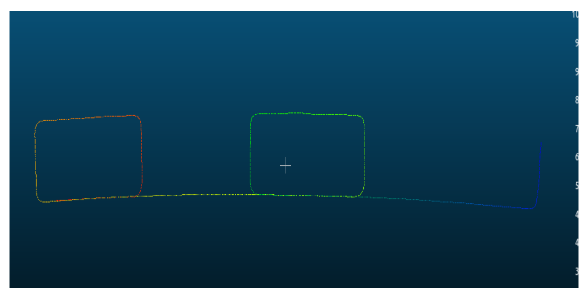


Fig 3.d

These images represent the global map and the trajectory taken saved as a PCD map

3. Conclusion:

We explored the LIO-SAM (Lidar Inertial Odometry via Smoothing and Mapping) paper, delved into the intricacies of its system architecture, package dependencies, installation procedures, and data analysis. Our tests have shown that to get the best results, careful parameter tuning, downsampling techniques, and hardware considerations are essential. Moreover, we emphasized the importance of a corner map in lidar-based mapping, where unique characteristics are essential for precise odometry calculation. To sum up, LIO-SAM showed encouraging results in lidar-based odometry and mapping, with an emphasis on precise feature representation and real-time performance.