Northeastern University

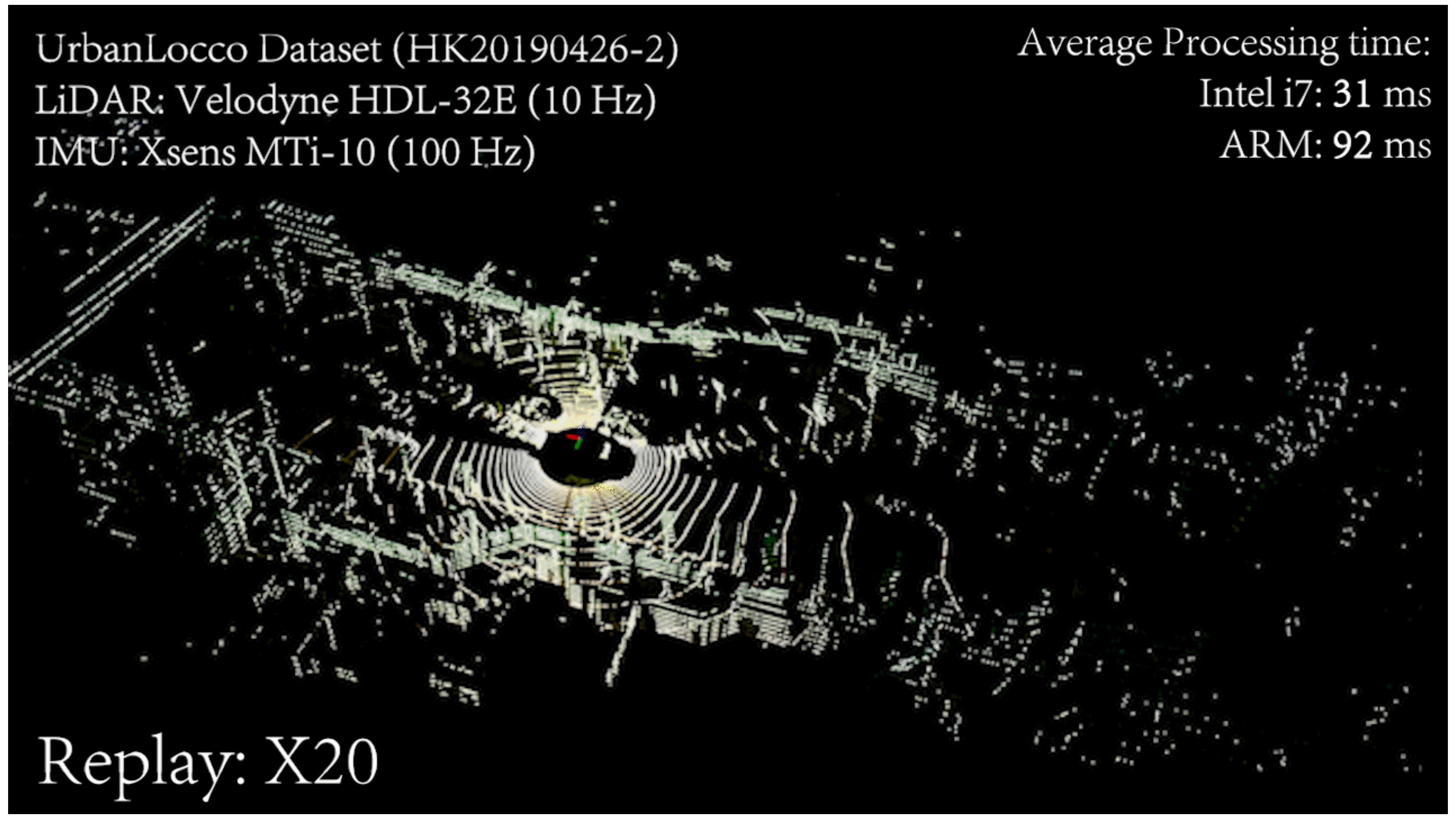
Final Project Presentation

EECE 5554 - Robot Sensing and Navigation

Group 14 : Jaykumar Goswami, Aryaman Shardul, Jake Ross, Rongxuan Zhou

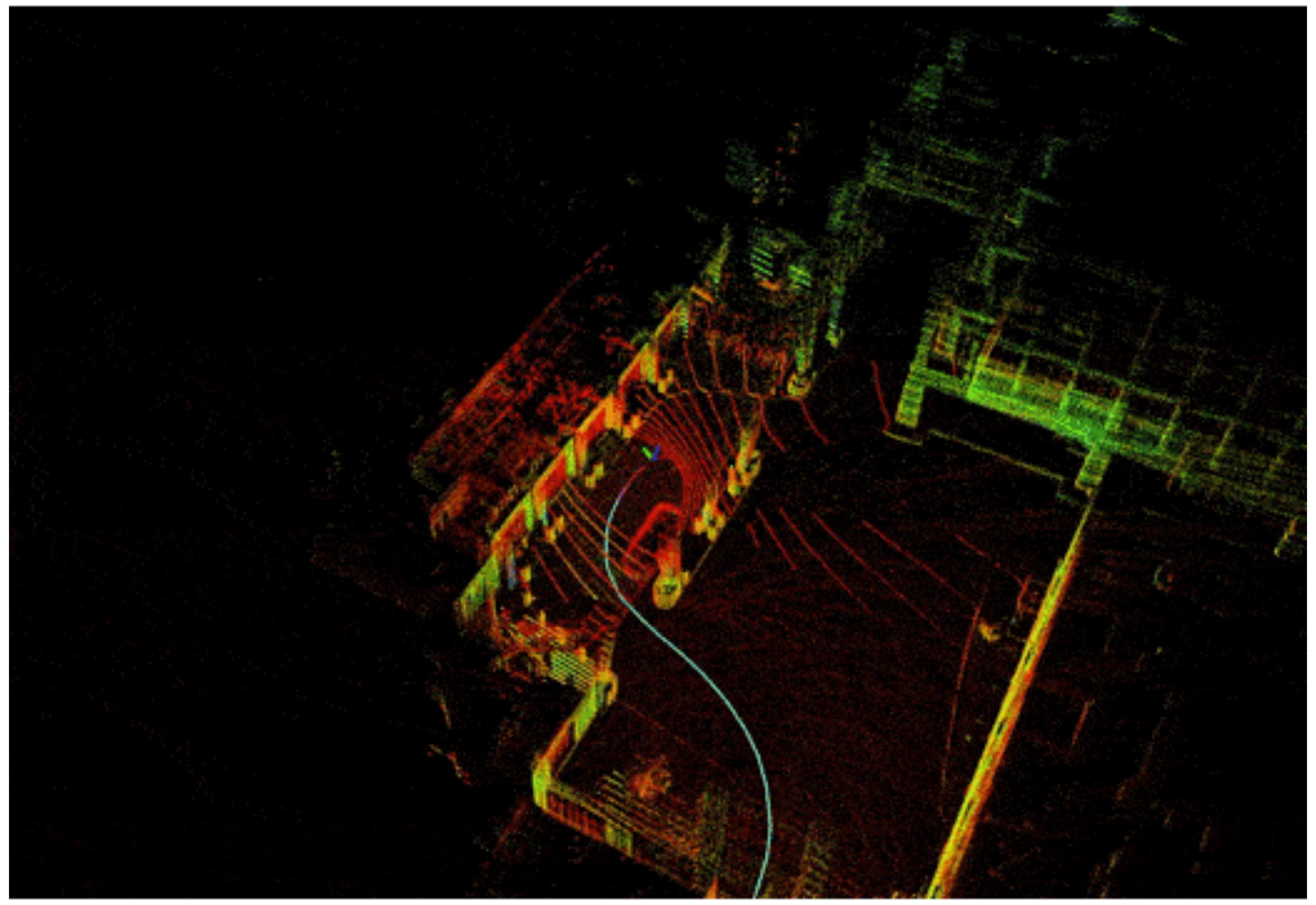
Objective

To evaluate and compare the performance of **FAST-LIO** and **LIO-SAM** in terms of Loop Closure and 3D Mapping accuracy and efficiency. The project will investigate how effectively each algorithm can detect and correct loops, as well as the quality and consistency of the 3D maps they produce in diverse environments.



FAST-LIO

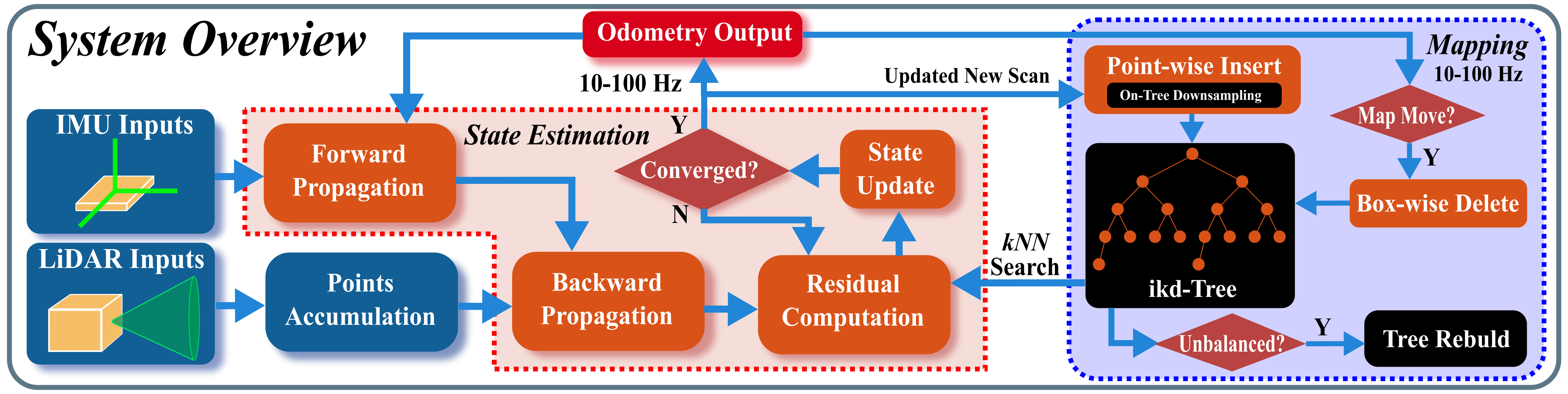
Vs

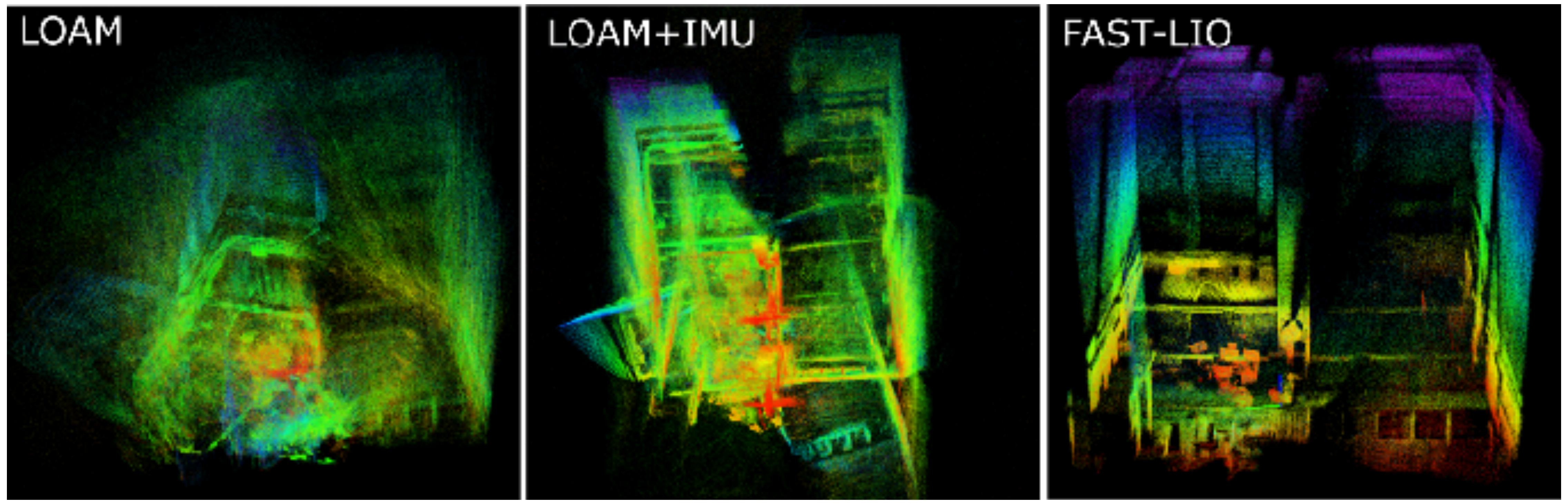


LIO-SAM

FAST-LIO (Overview)

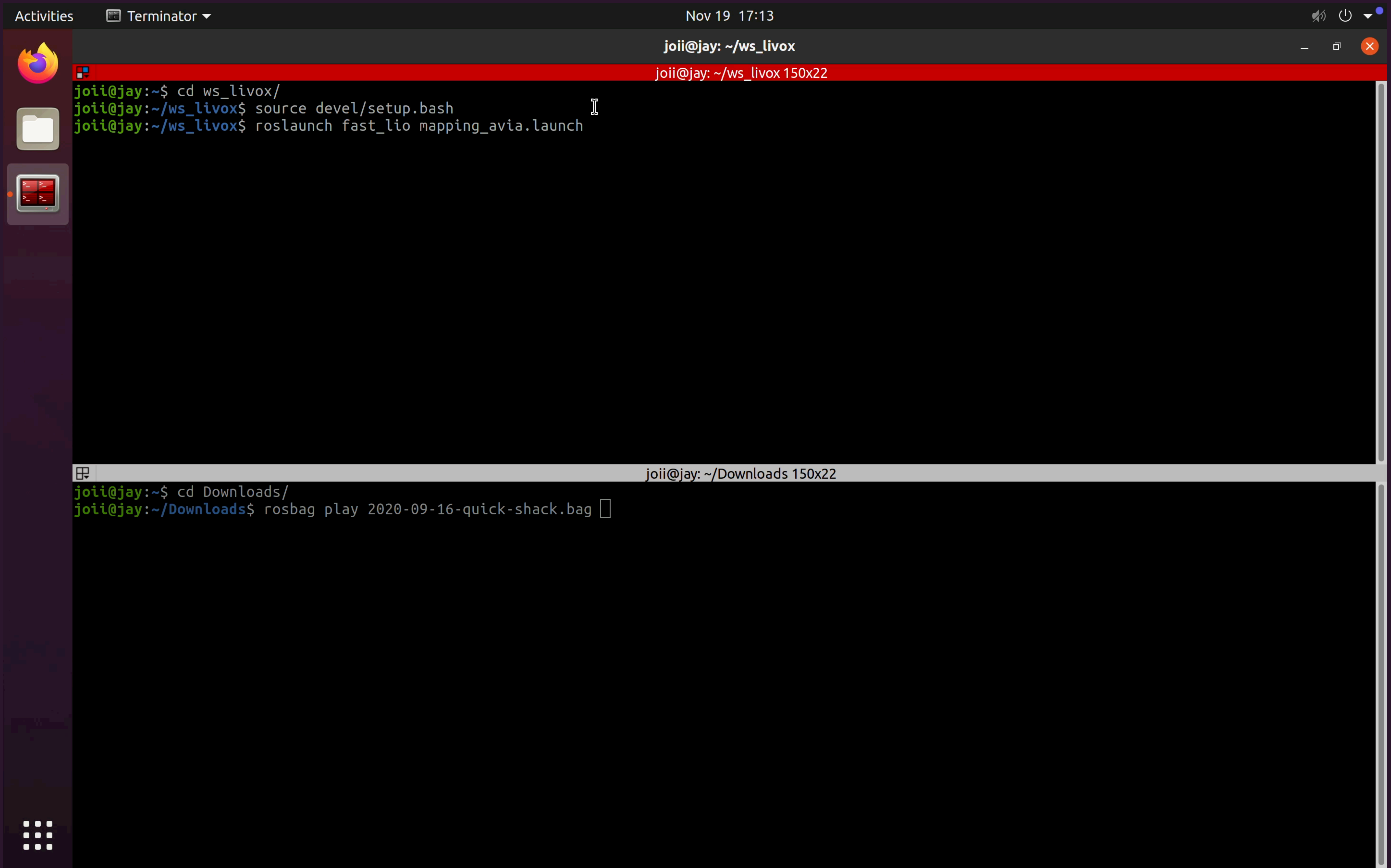
- A fast, robust LiDAR-Inertial Odometry framework designed for real-time navigation and mapping.
- **Tightly-Coupled**: Directly fuses LiDAR and IMU data, improving robustness and accuracy.
- **Filter-Based (iEKF)**: Iterated Extended Kalman Filter efficiently processes sensor data for real-time performance.
- Feature Extraction, Motion Compensation, Efficient Kalman Gain





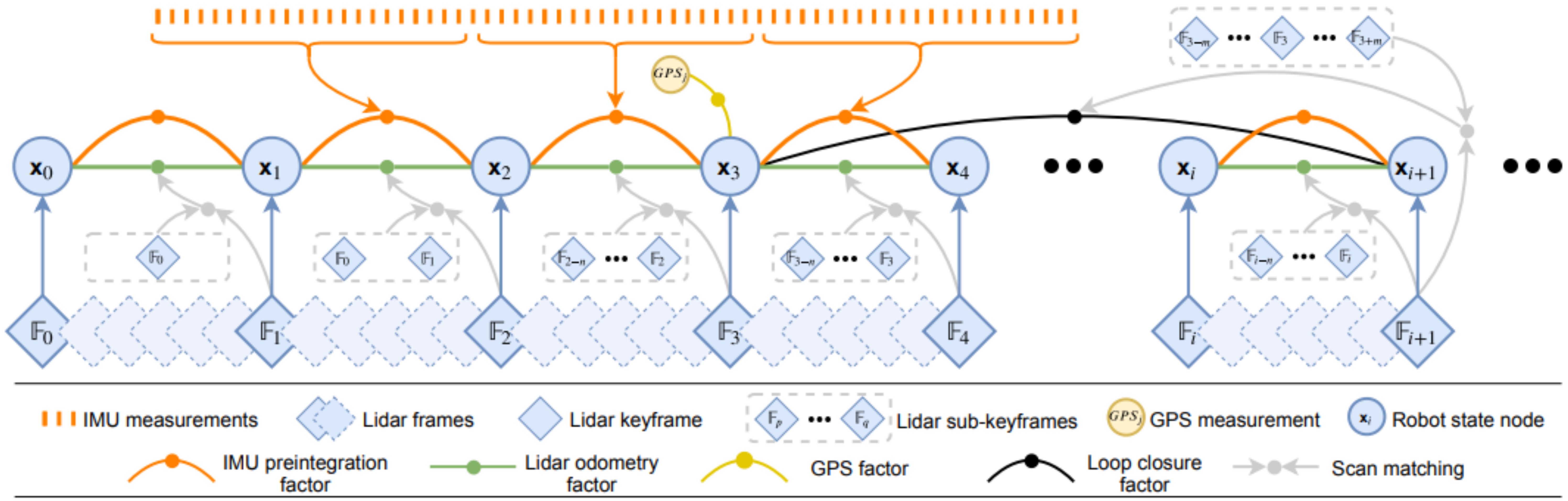
FAST-LIO Quick Test Dataset

Video Drive Link - https://drive.google.com/file/d/1Gui_EjZgYUo-86LwDyazdUjb3sYY0T4R/view?usp=sharing



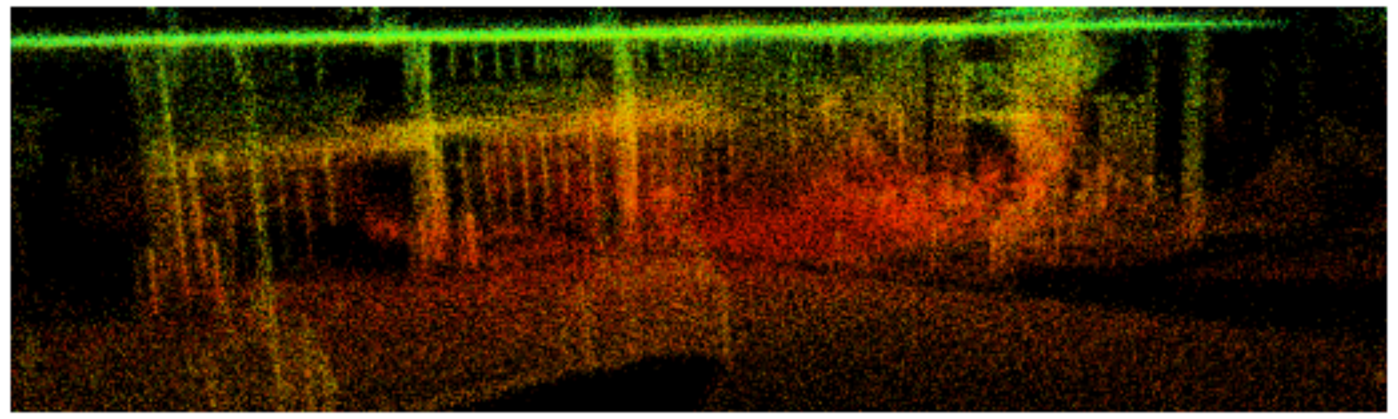
LIO-SAM (Overview)

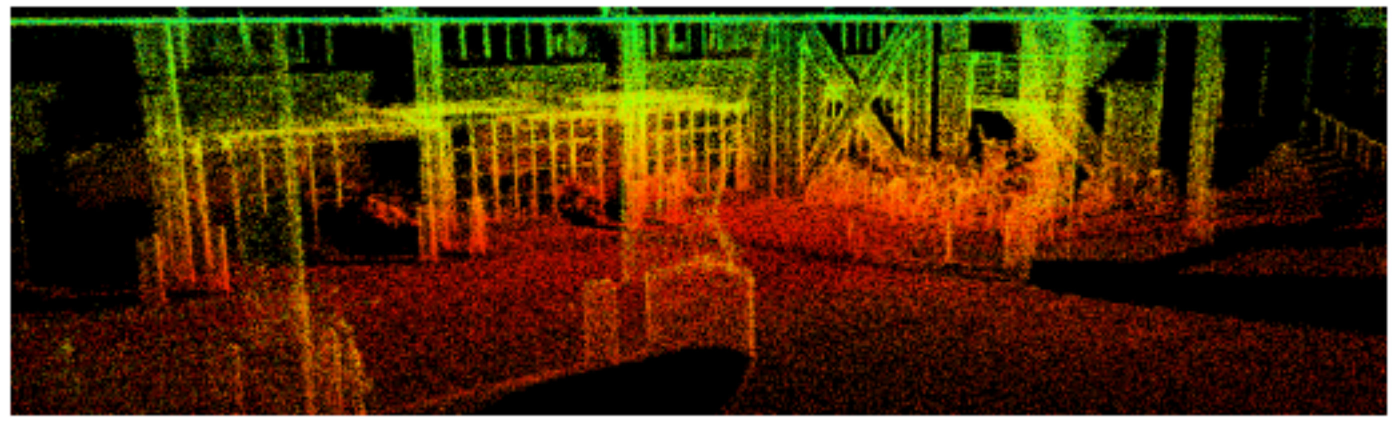
- LIO-SAM stands for tightly coupled Lidar Inertial Odometry via Smoothing and Mapping

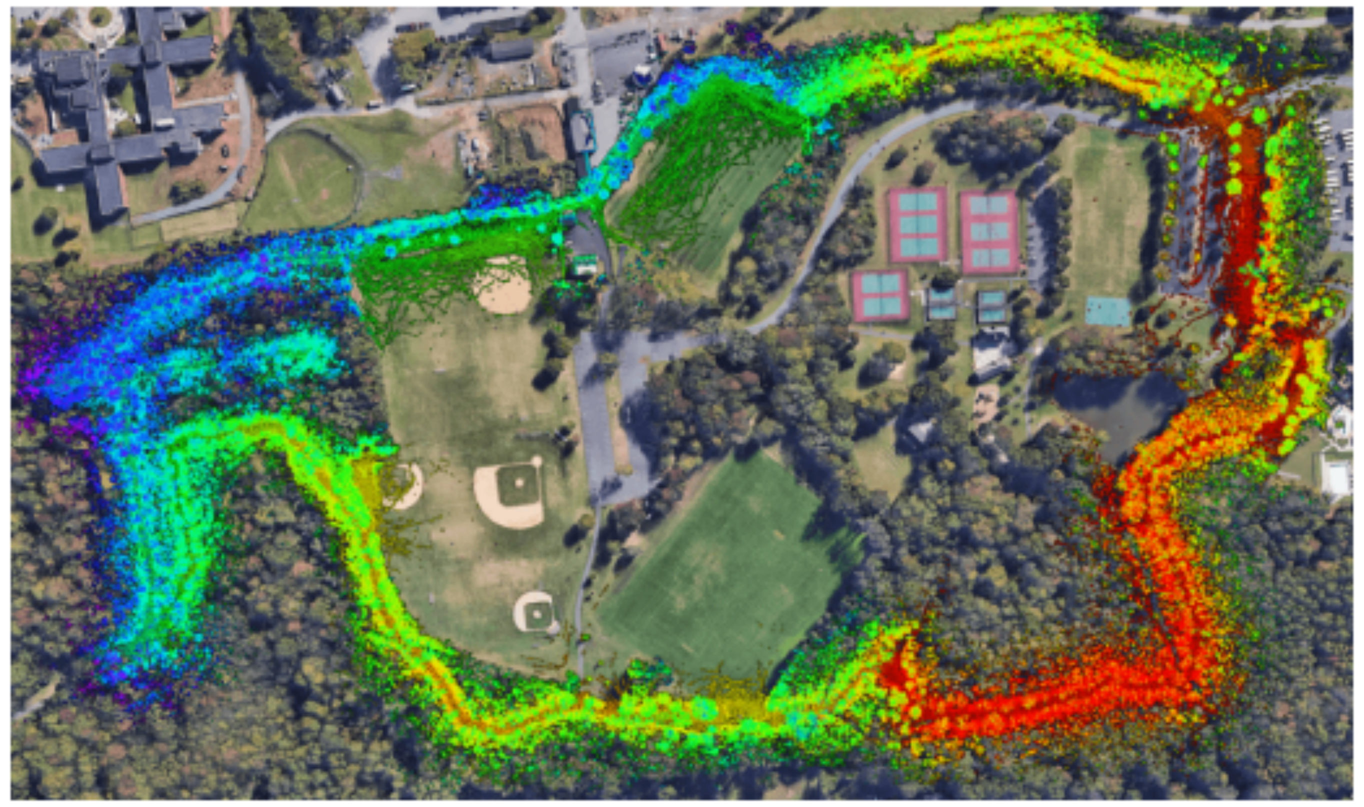


- The system receives sensor data from a 3D lidar, an IMU, and optionally a GPS. The aim is to estimate the state of the robot and its trajectory using the observations of these sensors. This state estimation problem can be formulated as a maximum a posteriori (MAP) problem. A factor graph is used to model this problem
- The factor graph consists of four types of factors (IMU preintegration factors, lidar odometry factors, GPS factors, and loop closure factors) along with one variable node type (that represents the robot's state at a specific time) for factor graph construction.


(a) Test environment

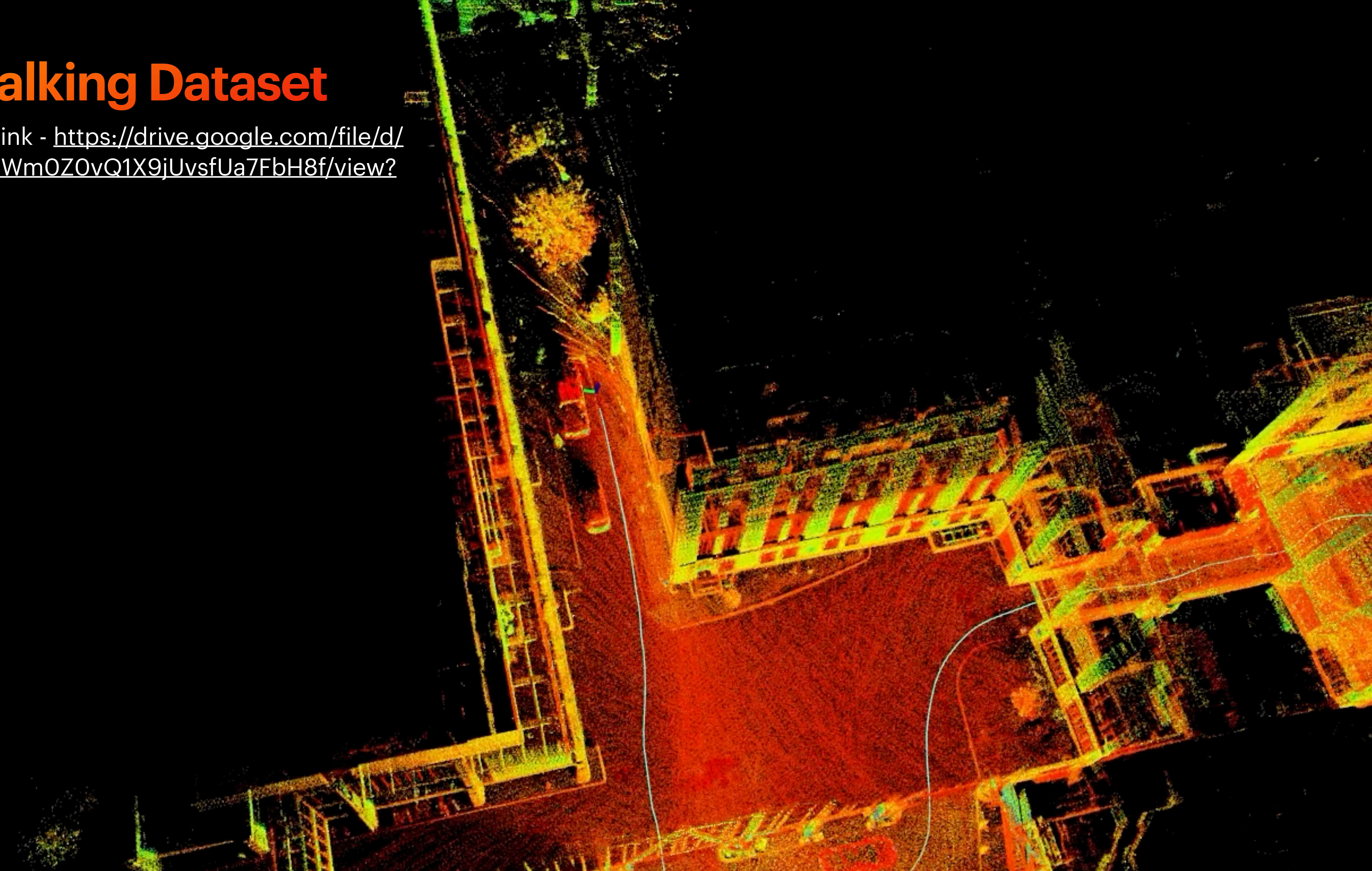

(b) LOAM

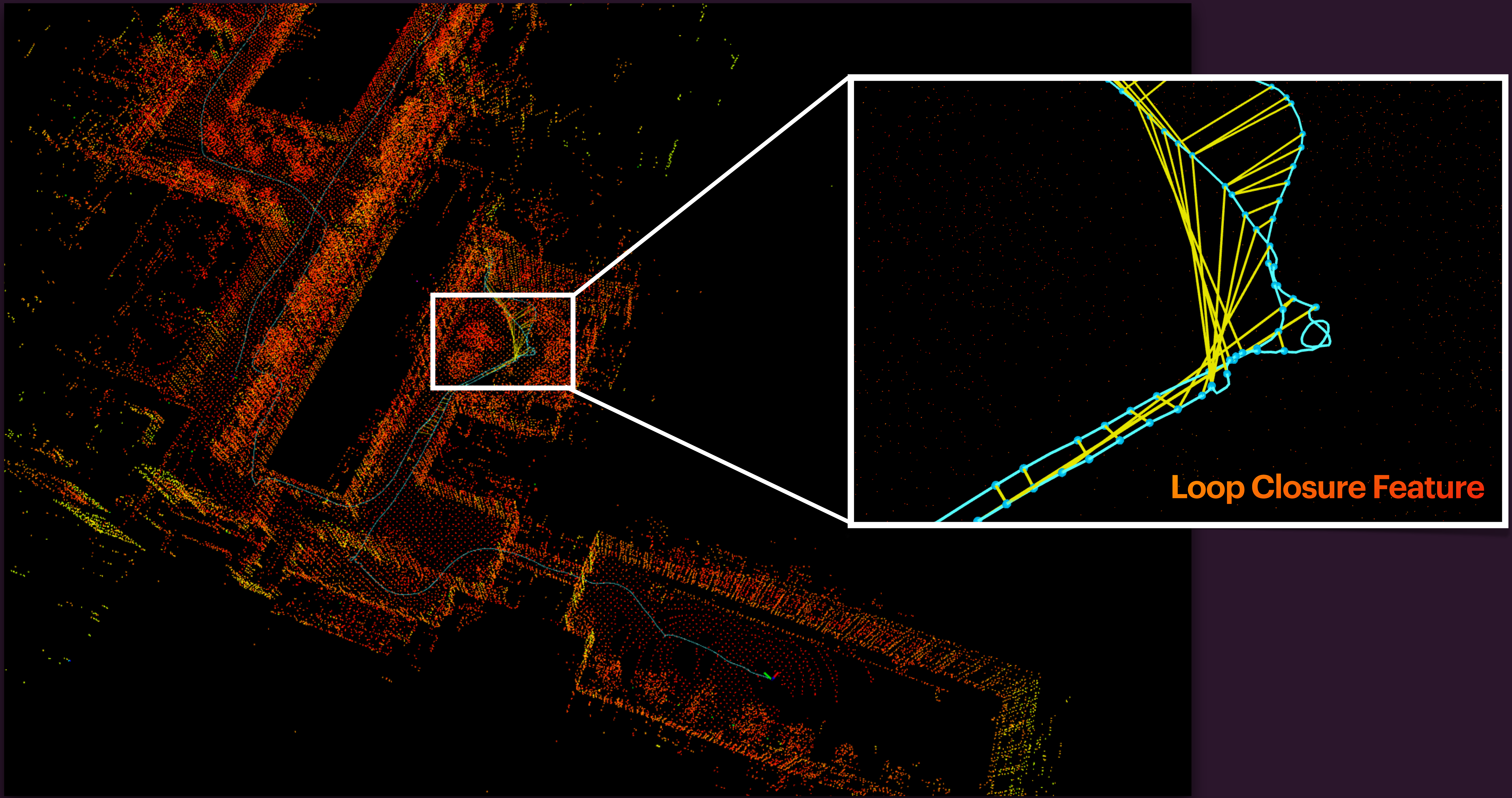

(c) LIO-SAM



MIT Walking Dataset

Video Drive Link - https://drive.google.com/file/d/1nyQVqnCHnWm0Z0vQ1X9jUvsfUa7FbH8f/view?usp=sharing



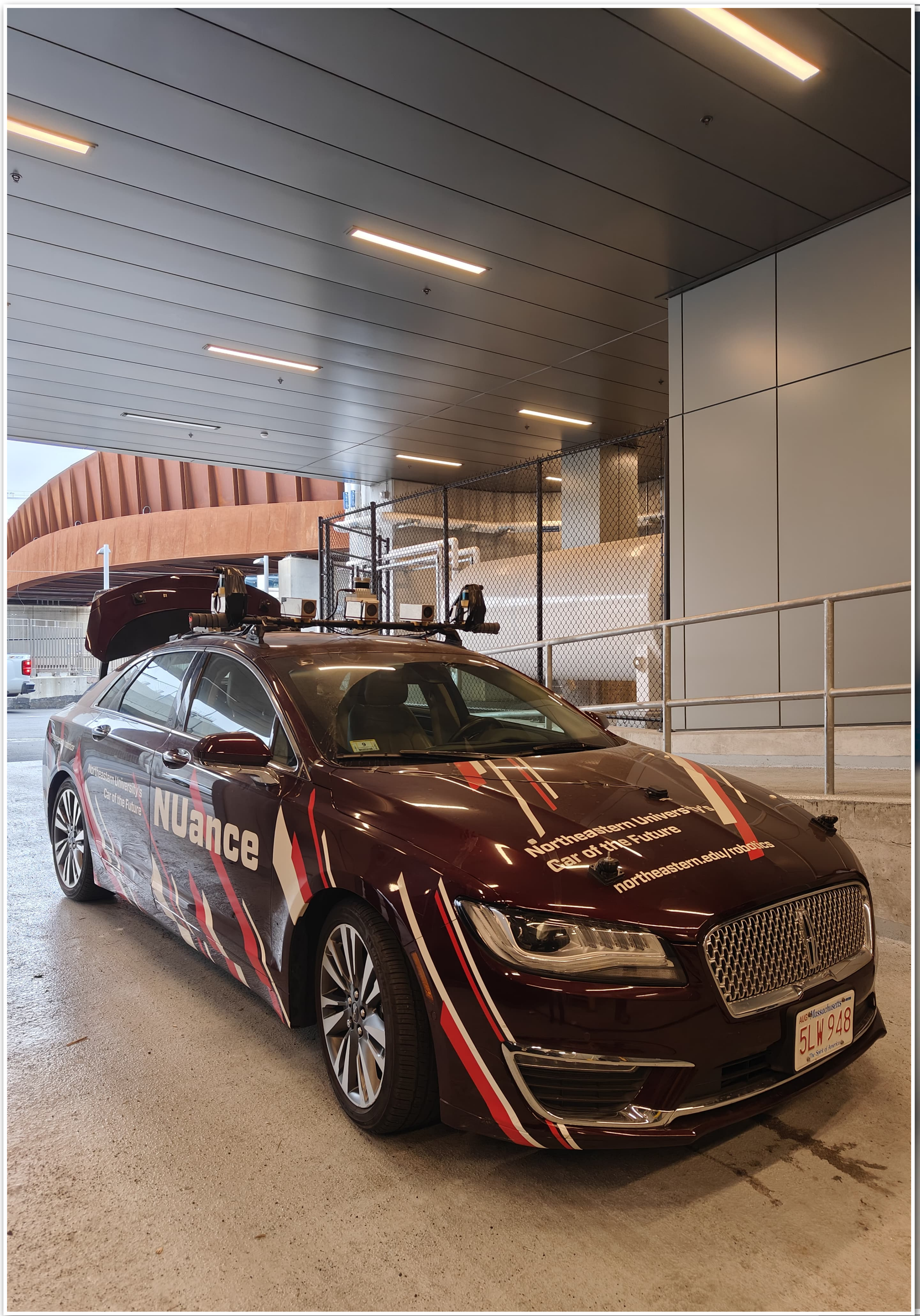

Loop Closure Feature

LIO-SAM (Challenges)

- First of all, since LIO-SAM doesn’t accept the Lidar’s 6-DOF IMU, we have to use the data from the external 9-DOF IMU
- To do that, we require the extrinsic translation and rotation matrices, as well as sync the Lidar and IMU time
- To obtain those matrices, we used the LiDAR_IMU_Init repository made by the FAST-LIO developers and got those matrices
- But we couldn’t sync the timestamps

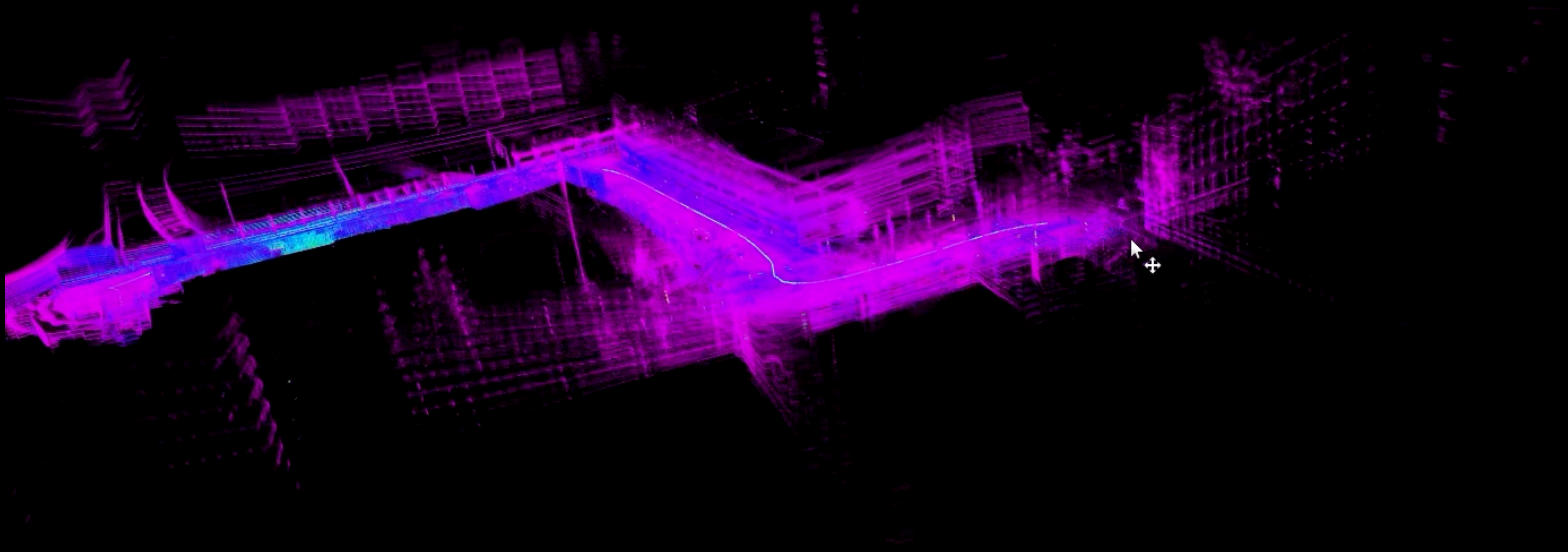
DATA Collection using NUance

Northeastern University



Video Drive Link -https://drive.google.com/file/d/1XWqXalCk5tTeW1ue5knqP4ECNQudW9GW/view?usp=sharing

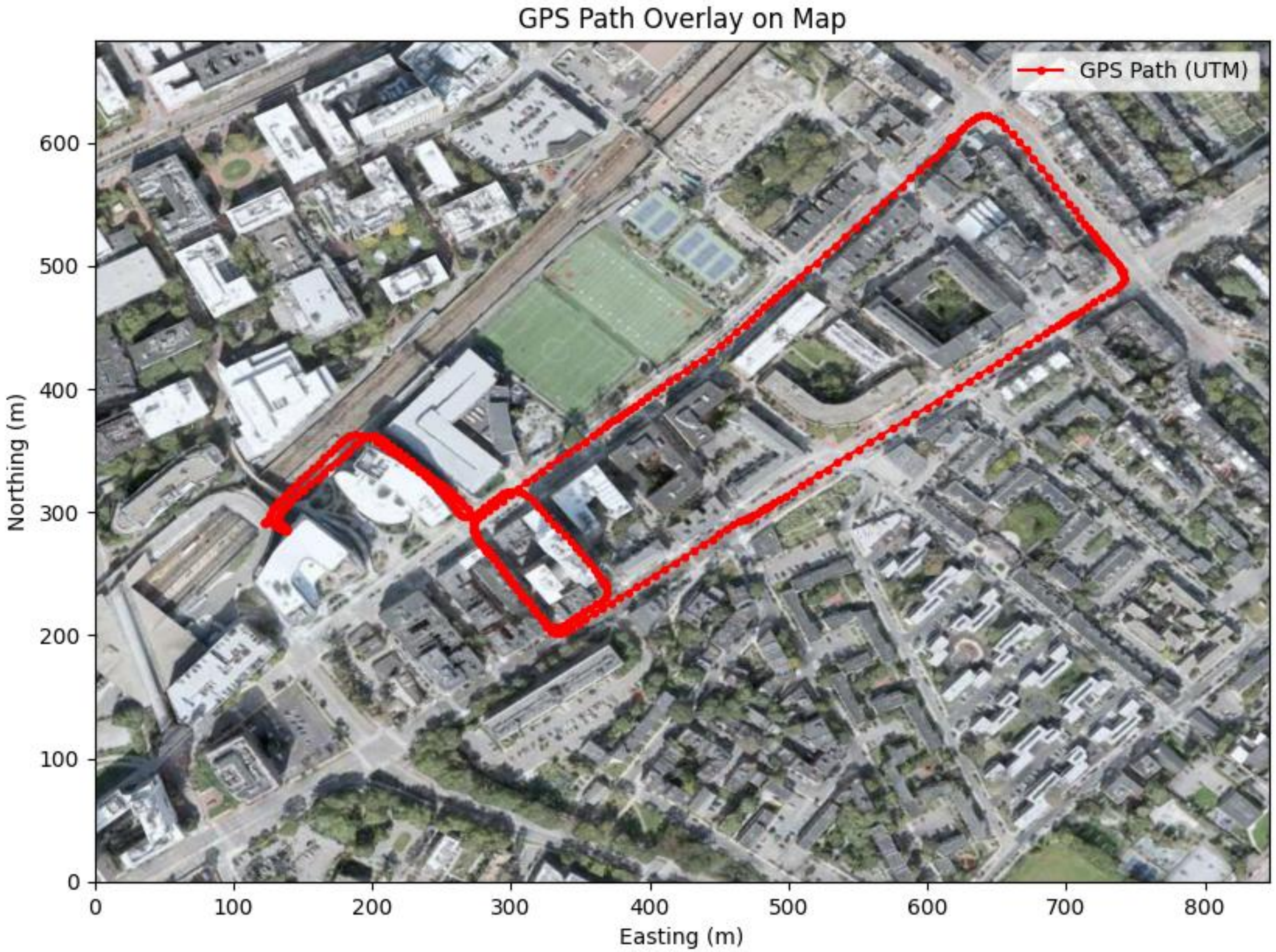
FAST-LIO running on our dataset



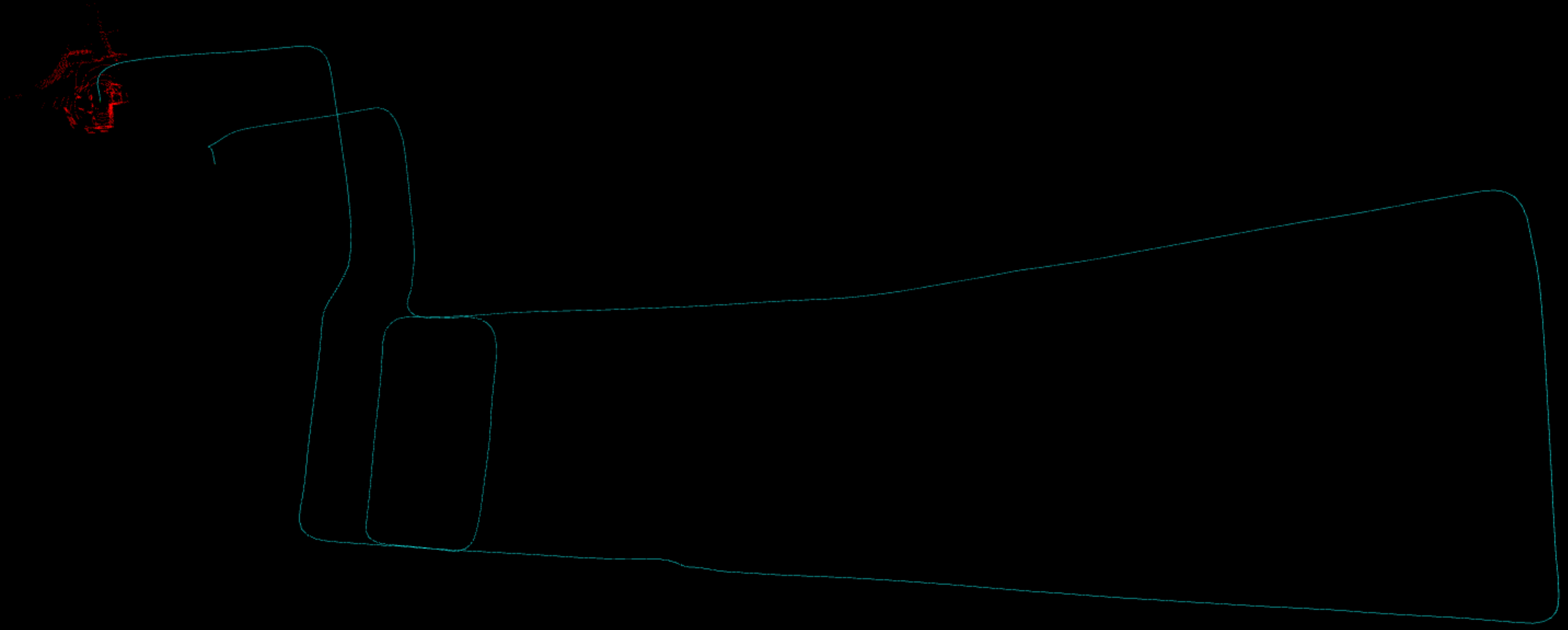
Video Drive Link - https://drive.google.com/file/d/1OU_M-pQtfSGL5jxChRSK6qdtqYfYjtls/view?usp=sharing

3D Map of the streets behind EXP Building

Actual Trajectory (GPS)



FAST-LIO (Path)

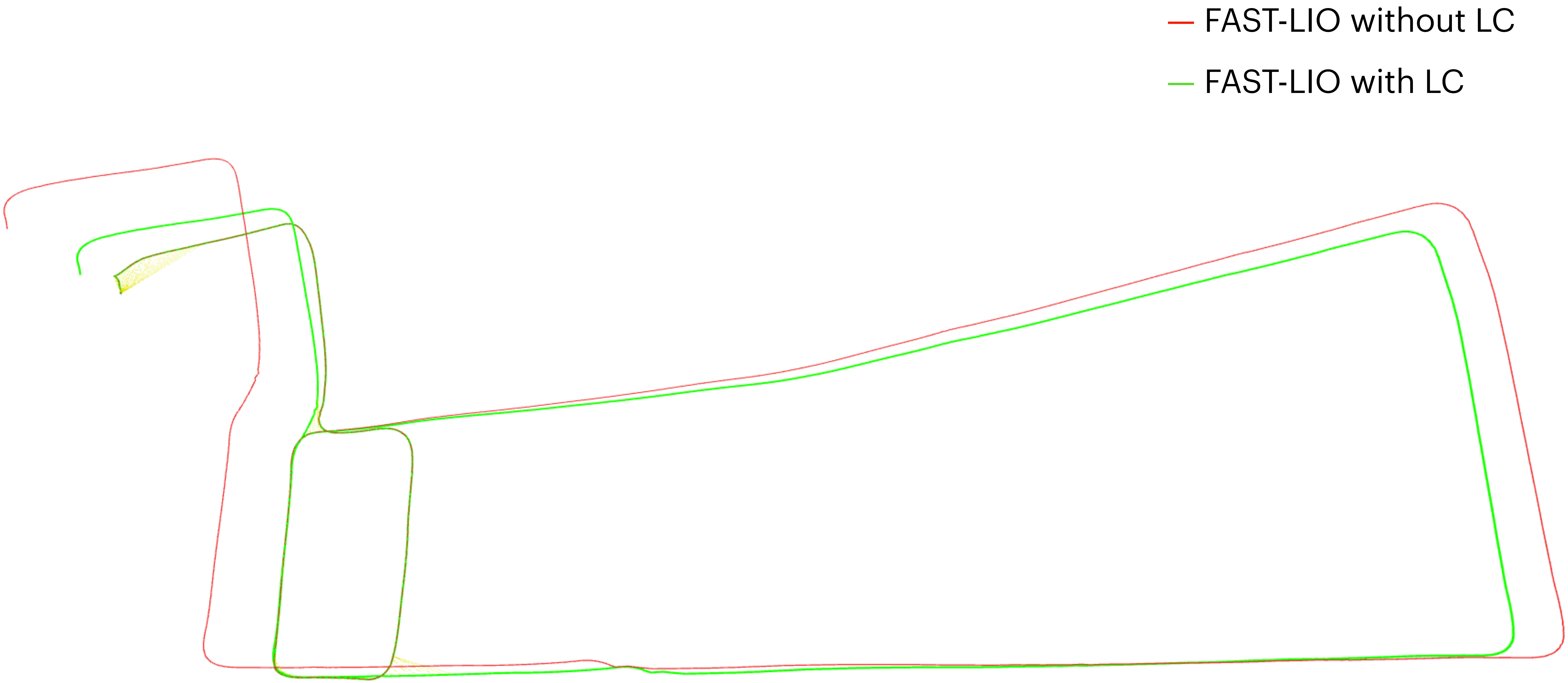


FAST-LIO (Path)

Shortcomings in FAST-LIO

- Since we couldn't proceed with LIO-SAM on our custom dataset, we decided to use FAST-LIO. But FAST-LIO doesn't have loop closure which when we travel using a longer path, the trajectory of the return journey doesn't align well with the trajectory of the initial part of the journey. This can be seen in slide 13 where the path obtained from FAST-LIO has been plotted by us. Slide 14 depicts the drift in the Z-axis of the trajectory as well.
- Due to this, we decided to look for options with loop closure. FAST-LIO 2 has loop-closure but its repository isn't open-source when doing this project. So, we chose to try out FAST-LIO-LC, an open-source repository that incorporated loop-closure into the original FAST-LIO repository.

FAST-LIO with Loop Closure



FAST-LIO with Loop Closure

- As discussed earlier, we used FAST-LIO-LC, an open-source repository that incorporated loop-closure into the original FAST-LIO repository.
- We can see from slide 16 that although the path obtained using FAST-LIO-LC isn't perfectly aligned, the results that it provides are much better than the one provided by FAST-LIO without loop-closure.
- We suspect that the perfect alignment with FAST-LIO-LC is not obtained either due to some extra drift/bias/noise in the sensors that is unaccounted for or it might be the case that the FAST-LIO-LC algorithm's loop-closure is not extremely efficient (We must remember that this code comes from an open source repository and not directly from FAST-LIO 2 that has loop-closure implemented by the original makers of FAST-LIO).

Conclusion

- So, to summarize, we initially set out to compare FAST-LIO and LIO-SAM algorithms. Due to time synchronization problems in the various sensors, we couldn't implement LIO-SAM on our data.
- We used FAST-LIO on our data and got good results. However, there was a certain misalignment in the trajectory due to the absence of loop closure from the algorithm. Using FAST-LIO-LC made the results much more efficient, but it too, couldn't give perfect alignment.
- But, we do observe that when you implement loop-closure, it has a good amount of positive influence on the trajectories obtained.

References

- FAST-LIO Repository - https://github.com/hku-mars/FAST_LIO
- FAST-LIO Paper - https://ieeexplore.ieee.org/document/9372856
- LIO-SAM Repository - https://github.com/TixiaoShan/LIO-SAM
- LIO-SAM Paper - https://ieeexplore.ieee.org/document/9341176
- Lidar IMU Init Repository - https://github.com/hku-mars/LiDAR_IMU_Init
- FAST-LIO-LC Repository - https://github.com/yanliang-wang/FAST_LIO_LC
- NUance Autonomous Car by Northeastern University
- Dataset that we used - https://drive.google.com/drive/folders/1GNqsowswBNSMr-d3rEnTFEqyhZRcBWLL?usp=sharing
- We have uploaded the dataset link and the custom scripts in the following gitlab account

  Gitlab user name : lnu.arya (Aryaman Shardul)

  Folder Name : Final_Project

RS&N Dream Team