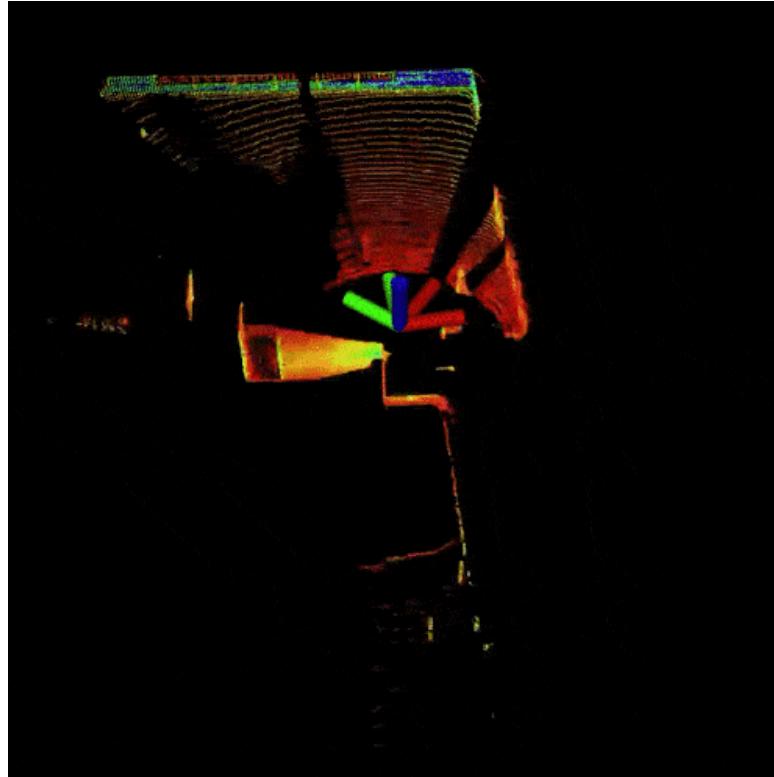
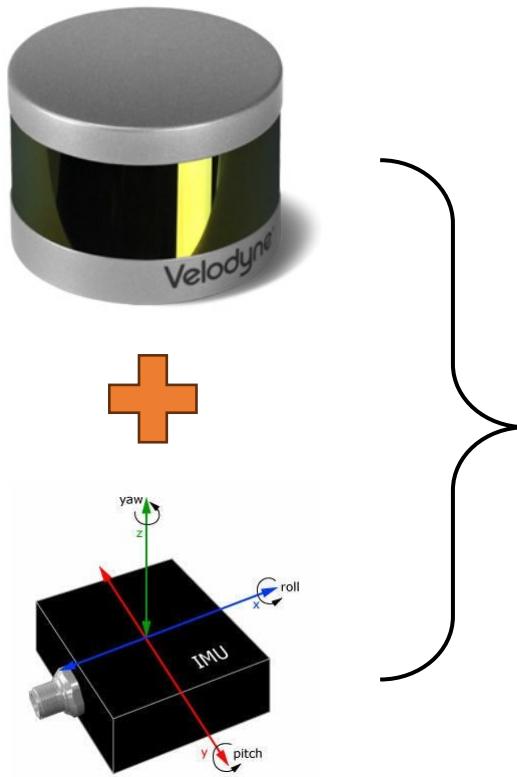


# LIO-SAM with SPOT

By Group - 7

- Dhyey Mistry
- Yash Wakde
- Adithya Rajendran
- Kevin Jason



Northeastern  
University

BostonDynamics

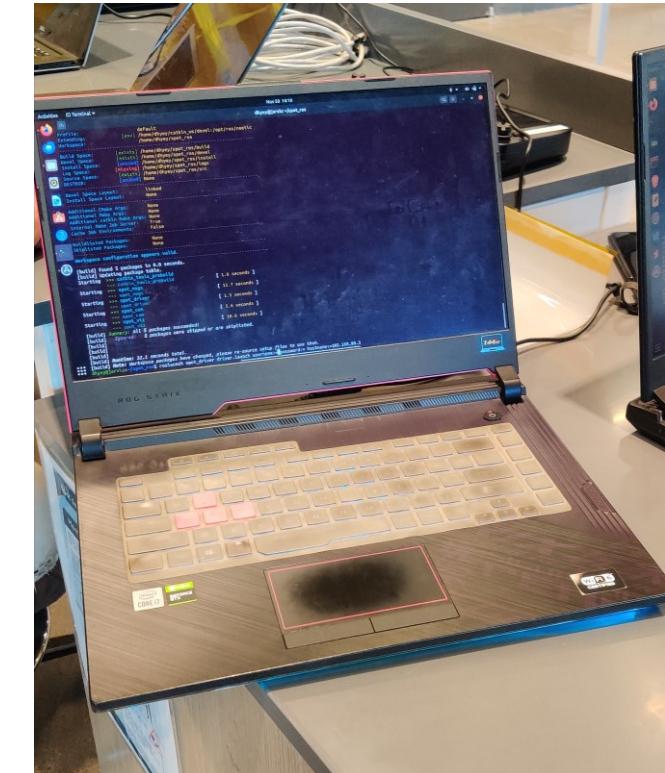


# LIO-SAM with SPOT

## A Journey of Challenges and Progress

We implemented the LIO-SAM (LiDAR-Inertial Odometry with Smoothing and Mapping) algorithm on data collected from the Spot robot, aiming to achieve high-precision 3D mapping and localization in a dynamic environment.

Although the project is still in progress, we have made substantial strides in understanding and implementing the algorithm and collecting the data.





# Motivation

- Spot is changing how organizations monitor and operate their sites, ensure the safety of their teams, and make data-driven decisions.
- We can put Spot's unprecedented agility and advanced autonomy to work and extend the reach of industries.

# What is LIO SAM

---

- **LIO-SAM (LiDAR-Inertial Odometry with Smoothing and Mapping)** is a state-of-the-art SLAM algorithm that combines data from LiDAR and IMU (Inertial Measurement Unit) sensors to achieve accurate and robust 3D mapping and localization. It is designed for real-time operation in dynamic and challenging environments.
- **Key Features:**
- **Tight Integration of LiDAR and IMU:**
  - Uses LiDAR for spatial accuracy and IMU for motion compensation, enabling precise localization and mapping in environments where GPS may not work.
- **Factor Graph Optimization:**
  - Builds a factor graph to optimize poses and trajectories, ensuring global consistency and minimizing drift.
- **Real-Time Performance:**
  - Capable of providing real-time odometry and mapping, suitable for mobile robotics applications.
- **Loop Closure:**
  - Detects when the robot revisits an area to correct drift and improve map consistency.

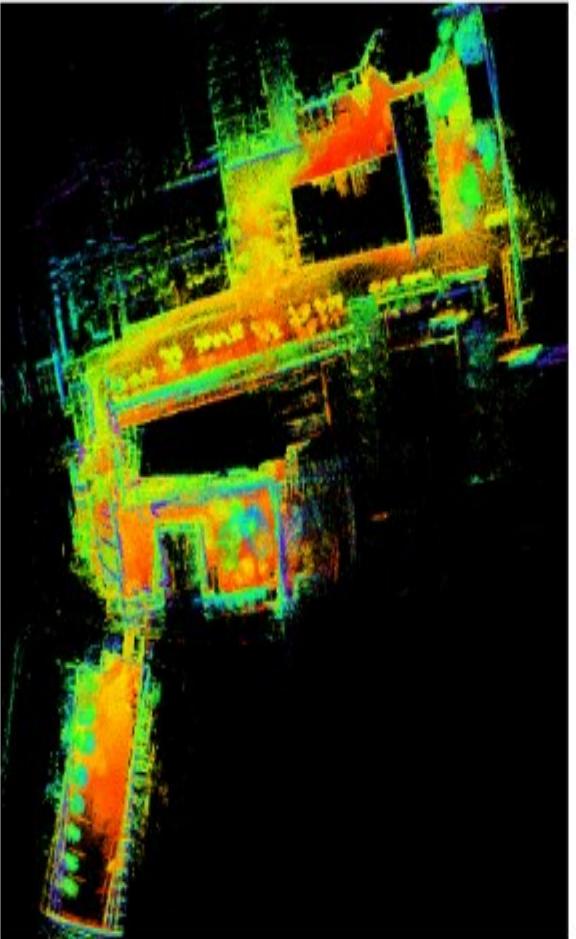
# Why LIO SAM ?

Feature	LOAM	LIOM	LIO-SAM
<b>Sensor Fusion</b>	Relies mostly on LiDAR	Loosely integrates LiDAR with IMU	Tightly integrates LiDAR and IMU
<b>Optimization</b>	No global smoothing (relies on ICP-like)	Loosely optimized over poses	Factor graph-based global smoothing
<b>Real-time Capability</b>	Efficient for real-time	Moderate real-time performance	High real-time performance
<b>Accuracy</b>	Good in LiDAR-dominant environments	Better than LOAM due to IMU use	Best, especially in dynamic motion
<b>Robustness to Dynamics</b>	Moderate	Better than LOAM	High robustness in fast/dynamic motions
<b>Loop Closure Handling</b>	Limited	Improved	Strong with backend optimization
<b>Implementation</b>	Computationally efficient but lacks smoothing	Computationally heavier	Balance of efficiency and accuracy

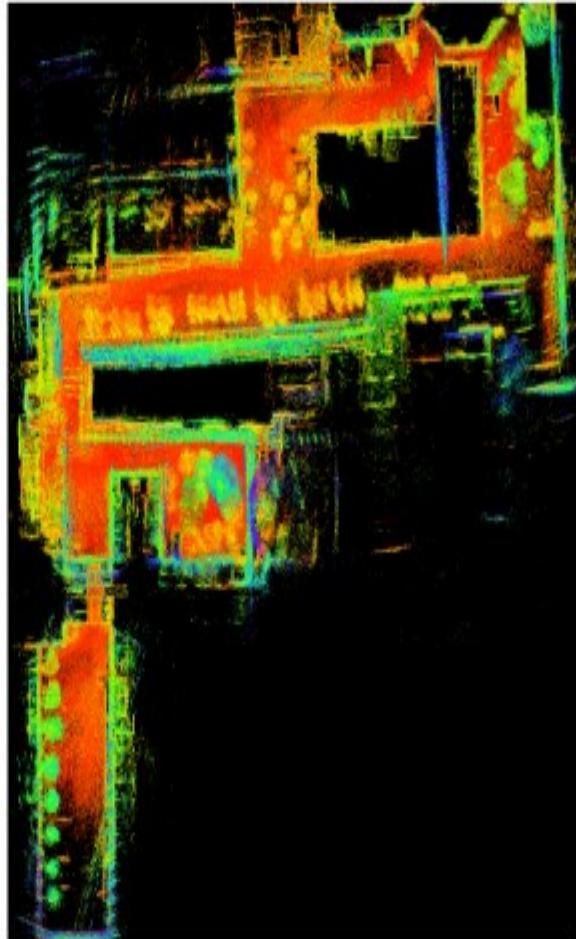
# Why LIO-SAM



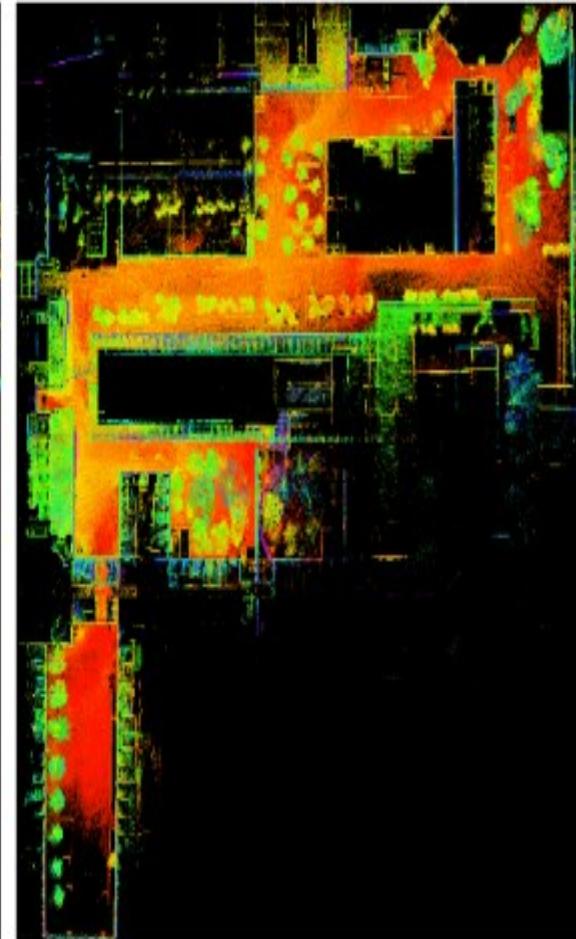
(a) Google Earth



(b) LOAM



(c) LIOM



(d) LIO-SAM

# Algorithm

- **LIO-SAM Algorithm (Simplified)**
- **Input Sensors:**
  - **IMU:** Captures motion.
  - **LiDAR:** Provides 3D point clouds.
  - **GPS (Optional):** Adds global positioning.
- **Data Processing:**
  - **IMU Pre-integration:** Estimates motion.
  - **LiDAR Feature Extraction:** Identifies edges/planes.
- **Factor Graph Construction:**
  - Combines constraints from IMU, LiDAR, GPS, and loop closure.
- **Loop Closure:**
  - Detects revisited areas to reduce drift.
- **Optimization:**
  - Refines poses using factor graph optimization.
- **Output:**
  - Builds a consistent 3D map.

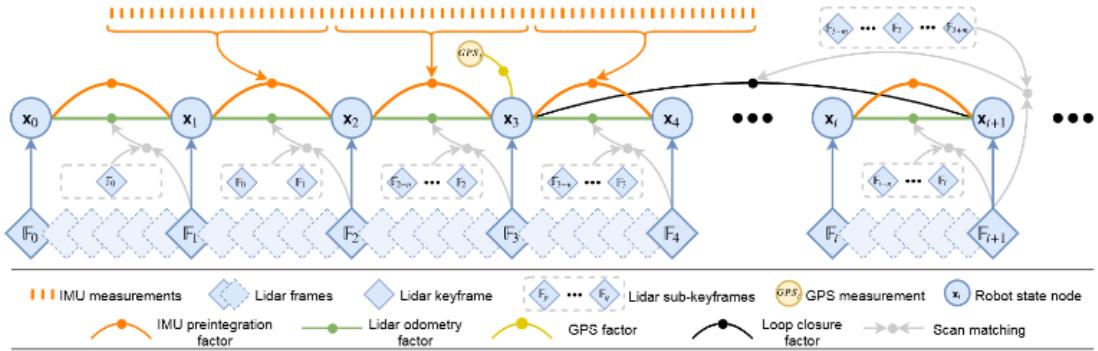
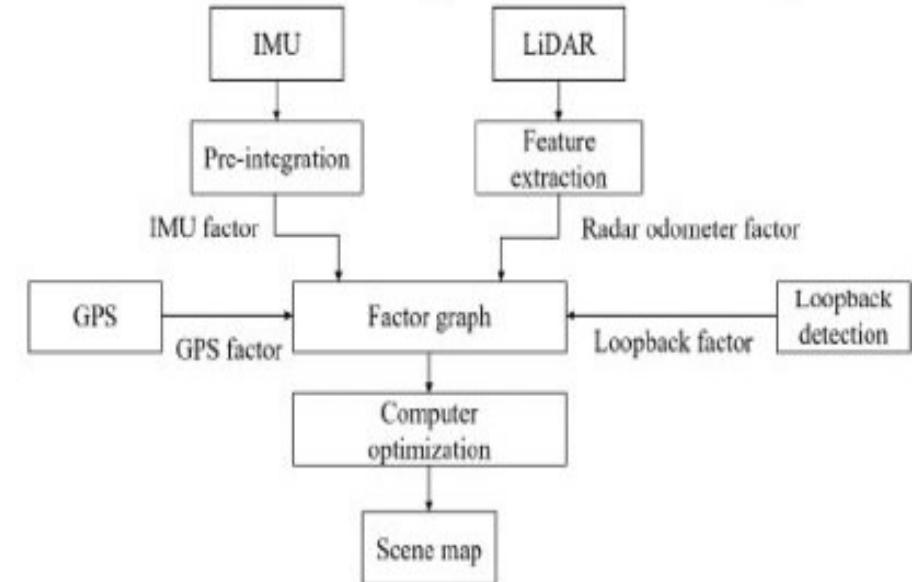


Fig. 1: The system structure of LIO-SAM. The system receives input from a 3D lidar, an IMU and optionally a GPS. Four types of factors are introduced to construct the factor graph.: (a) IMU preintegration factor, (b) lidar odometry factor, (c) GPS factor, and (d) loop closure factor. The generation of these factors is discussed in Section III.



# Algorithm

$$\mathbf{x} = [\mathbf{R}^T, \mathbf{p}^T, \mathbf{v}^T, \mathbf{b}^T]^T, \quad (1)$$

$$\hat{\omega}_t = \omega_t + \mathbf{b}_t^\omega + \mathbf{n}_t^\omega \quad (2)$$

$$\hat{\mathbf{a}}_t = \mathbf{R}_t^{\text{BW}}(\mathbf{a}_t - \mathbf{g}) + \mathbf{b}_t^a + \mathbf{n}_t^a, \quad (3)$$

We can now use the measurements from the IMU to infer the motion of the robot. The velocity, position and rotation of the robot at time  $t + \Delta t$  can be computed as follows:

$$\mathbf{v}_{t+\Delta t} = \mathbf{v}_t + \mathbf{g}\Delta t + \mathbf{R}_t(\hat{\mathbf{a}}_t - \mathbf{b}_t^a - \mathbf{n}_t^a)\Delta t \quad (4)$$

$$\begin{aligned} \mathbf{p}_{t+\Delta t} &= \mathbf{p}_t + \mathbf{v}_t\Delta t + \frac{1}{2}\mathbf{g}\Delta t^2 \\ &\quad + \frac{1}{2}\mathbf{R}_t(\hat{\mathbf{a}}_t - \mathbf{b}_t^a - \mathbf{n}_t^a)\Delta t^2 \end{aligned} \quad (5)$$

$$\mathbf{R}_{t+\Delta t} = \mathbf{R}_t \exp((\hat{\omega}_t - \mathbf{b}_t^\omega - \mathbf{n}_t^\omega)\Delta t), \quad (6)$$

$$\Delta \mathbf{v}_{ij} = \mathbf{R}_i^T(\mathbf{v}_j - \mathbf{v}_i - \mathbf{g}\Delta t_{ij}) \quad (7)$$

$$\Delta \mathbf{p}_{ij} = \mathbf{R}_i^T(\mathbf{p}_j - \mathbf{p}_i - \mathbf{v}_i\Delta t_{ij} - \frac{1}{2}\mathbf{g}\Delta t_{ij}^2) \quad (8)$$

$$\Delta \mathbf{R}_{ij} = \mathbf{R}_i^T \mathbf{R}_j. \quad (9)$$

$$\mathbf{d}_{e_k} = \frac{|(\mathbf{p}_{i+1,k}^e - \mathbf{p}_{i,u}^e) \times (\mathbf{p}_{i+1,k}^e - \mathbf{p}_{i,v}^e)|}{|\mathbf{p}_{i,u}^e - \mathbf{p}_{i,v}^e|} \quad (10)$$

$$\mathbf{d}_{p_k} = \frac{|(\mathbf{p}_{i+1,k}^p - \mathbf{p}_{i,u}^p) \times (\mathbf{p}_{i,u}^p - \mathbf{p}_{i,w}^p)|}{|(\mathbf{p}_{i,u}^p - \mathbf{p}_{i,v}^p) \times (\mathbf{p}_{i,u}^p - \mathbf{p}_{i,w}^p)|}, \quad (11)$$

where  $k$ ,  $u$ ,  $v$ , and  $w$  are the feature indices in their corresponding sets. For an edge feature  $\mathbf{p}_{i+1,k}^e$  in  $'F_{i+1}^e$ ,  $\mathbf{p}_{i,u}^e$  and  $\mathbf{p}_{i,v}^e$  are the points that form the corresponding edge line in  $M_i^e$ . For a planar feature  $\mathbf{p}_{i+1,k}^p$  in  $'F_{i+1}^p$ ,  $\mathbf{p}_{i,u}^p$ ,  $\mathbf{p}_{i,v}^p$ , and  $\mathbf{p}_{i,w}^p$  form the corresponding planar patch in  $M_i^p$ . The Gauss–Newton method is then used to solve for the optimal transformation by minimizing:

$$\min_{\mathbf{T}_{i+1}} \left\{ \sum_{\mathbf{p}_{i+1,k}^e \in 'F_{i+1}^e} \mathbf{d}_{e_k} + \sum_{\mathbf{p}_{i+1,k}^p \in 'F_{i+1}^p} \mathbf{d}_{p_k} \right\}.$$

- **2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) October 25-29, 2020, Las Vegas, NV, USA (Virtual)**
- Tixiao Shan, Brendan Englot, Drew Meyers, Wei Wang, Carlo Ratti, and Daniela Rus

# Progress made so far

## Research

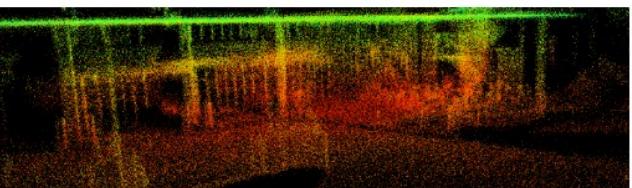
- Reviewed the paper:

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

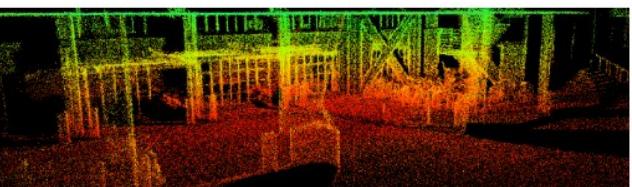
(Authors: Tixiao Shan, Brendan Englot, Drew Meyers, Wei Wang, Carlo Ratti, and Daniela Rus)



(a) Test environment



(b) LOAM



(c) LIO-SAM

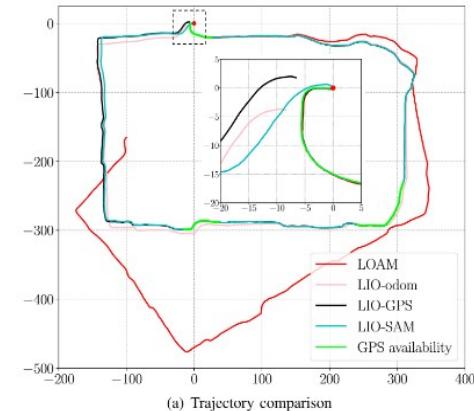


Fig. 5: Results of various methods using the *Campus* dataset that is gathered on the MIT campus. The red dot indicates the start and end location. The trajectory direction is clock-wise. LIOM is not shown because it fails to produce meaningful results.

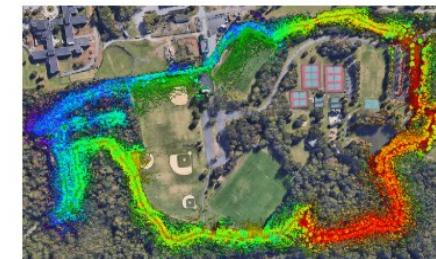
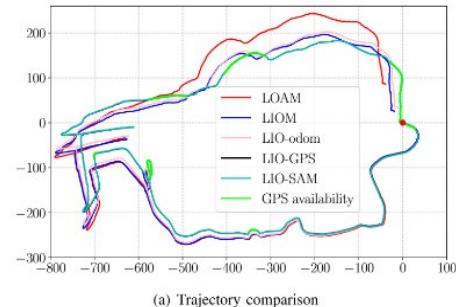


Fig. 6: Results of various methods using the *Park* dataset that is gathered in Pleasant Valley Park, New Jersey. The red dot indicate the start and end location. The trajectory direction is clock-wise. LIOM is not shown because it fails to produce meaningful results.

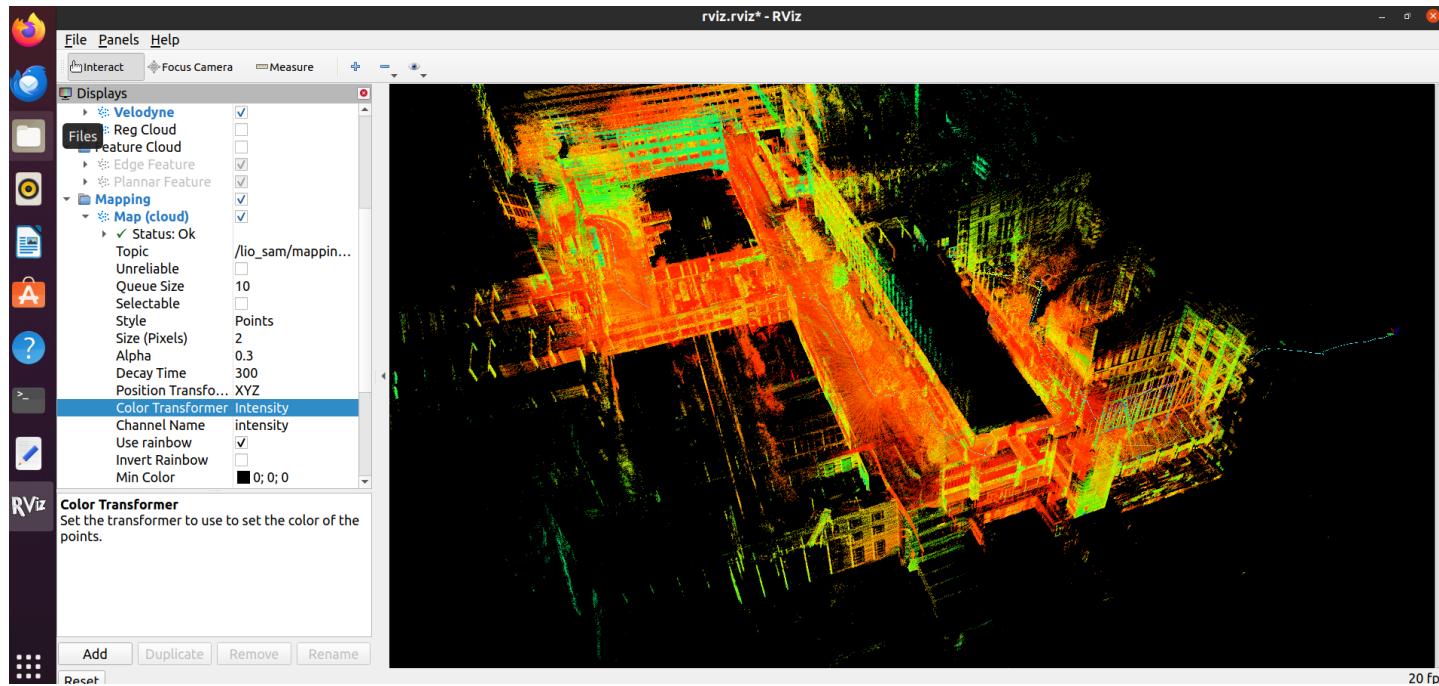
TABLE IV: Runtime of mapping for processing one scan (ms)

Dataset	LOAM	LIOM	LIO-SAM	Stress test
Rotation	83.6	Fail	41.9	13×
Walking	253.6	339.8	58.4	13×
Campus	244.9	Fail	97.8	10×
Park	266.4	245.2	100.5	9×
Amsterdam	Fail	Fail	79.3	11×

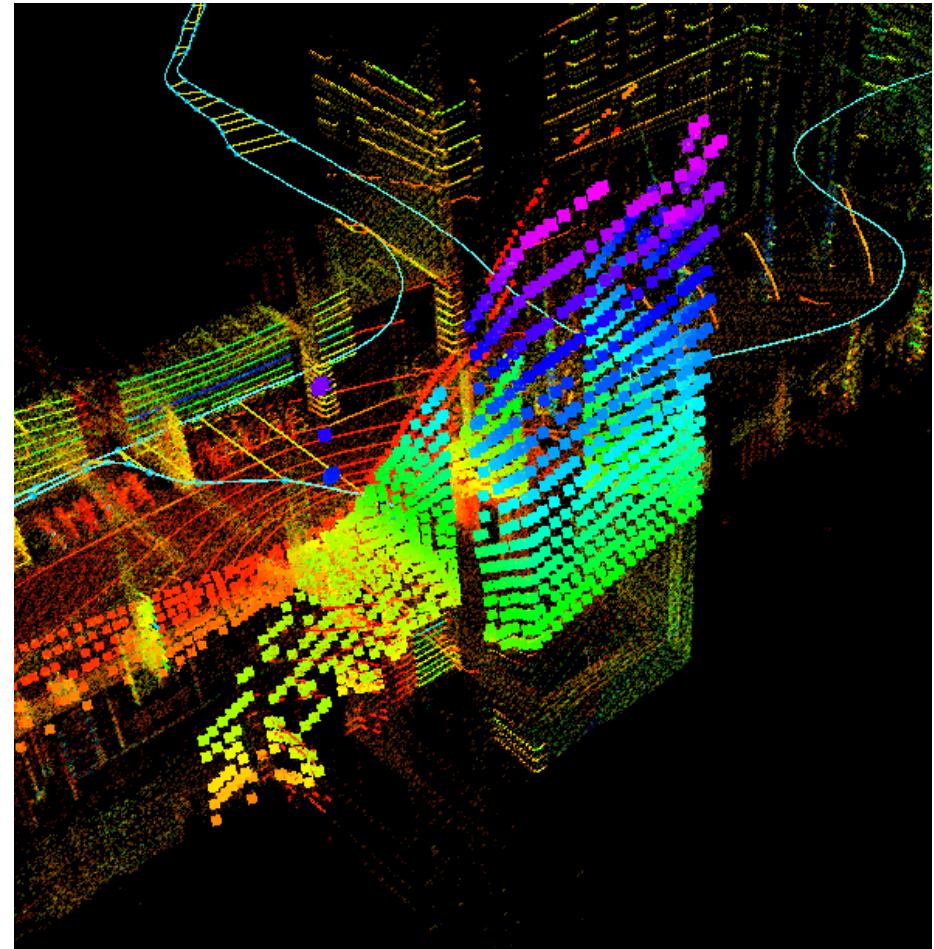
# Progress made so far

## Implementation

- Successfully implemented LIO-SAM for the KITTI dataset (*walking\_data.bag*).
- Applied the loop closure.



Map of the MIT campus



Loop closer and ICP working

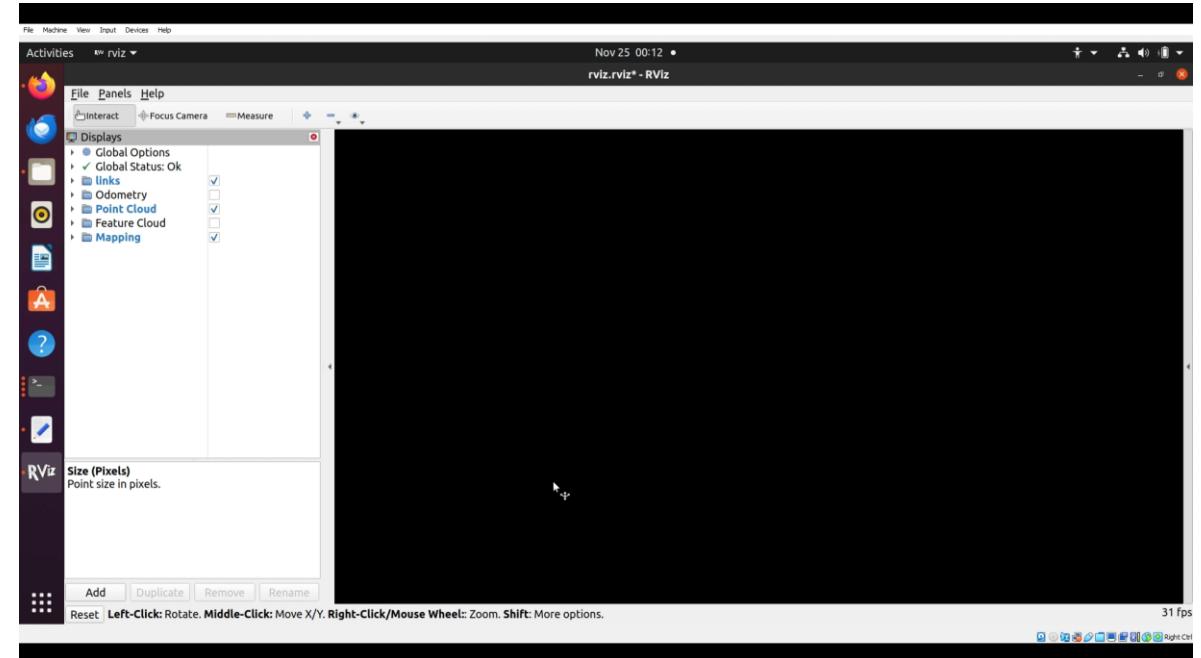
# Progress made so far

## Data Collection

- Collected a trial dataset in the High Bay environment for further testing and validation.
- Got nice and clean point cloud data.

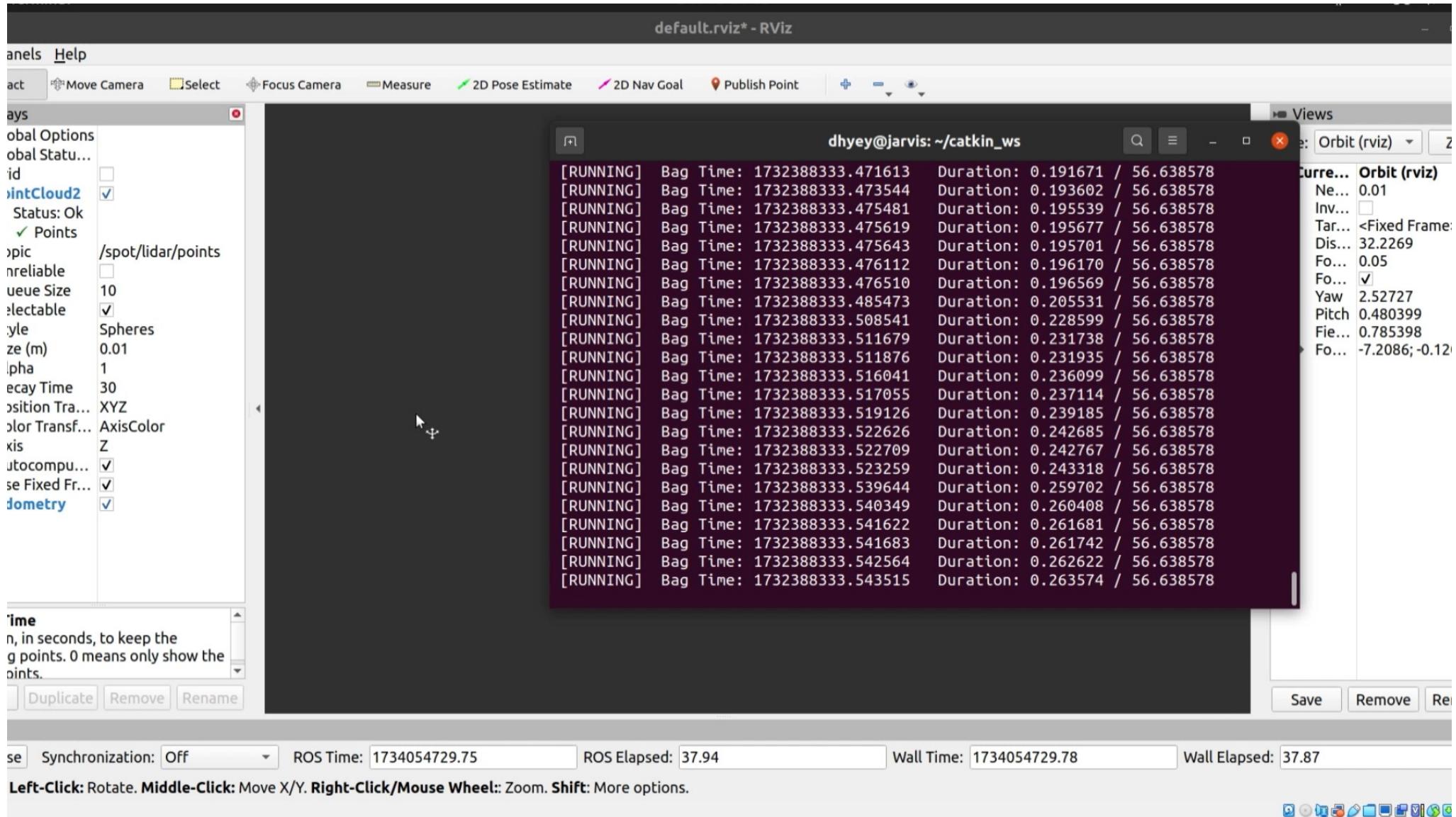


[Mapping the High-bay with spot](#)



[Visualizing the map in Rviz](#)

# Progress made so far

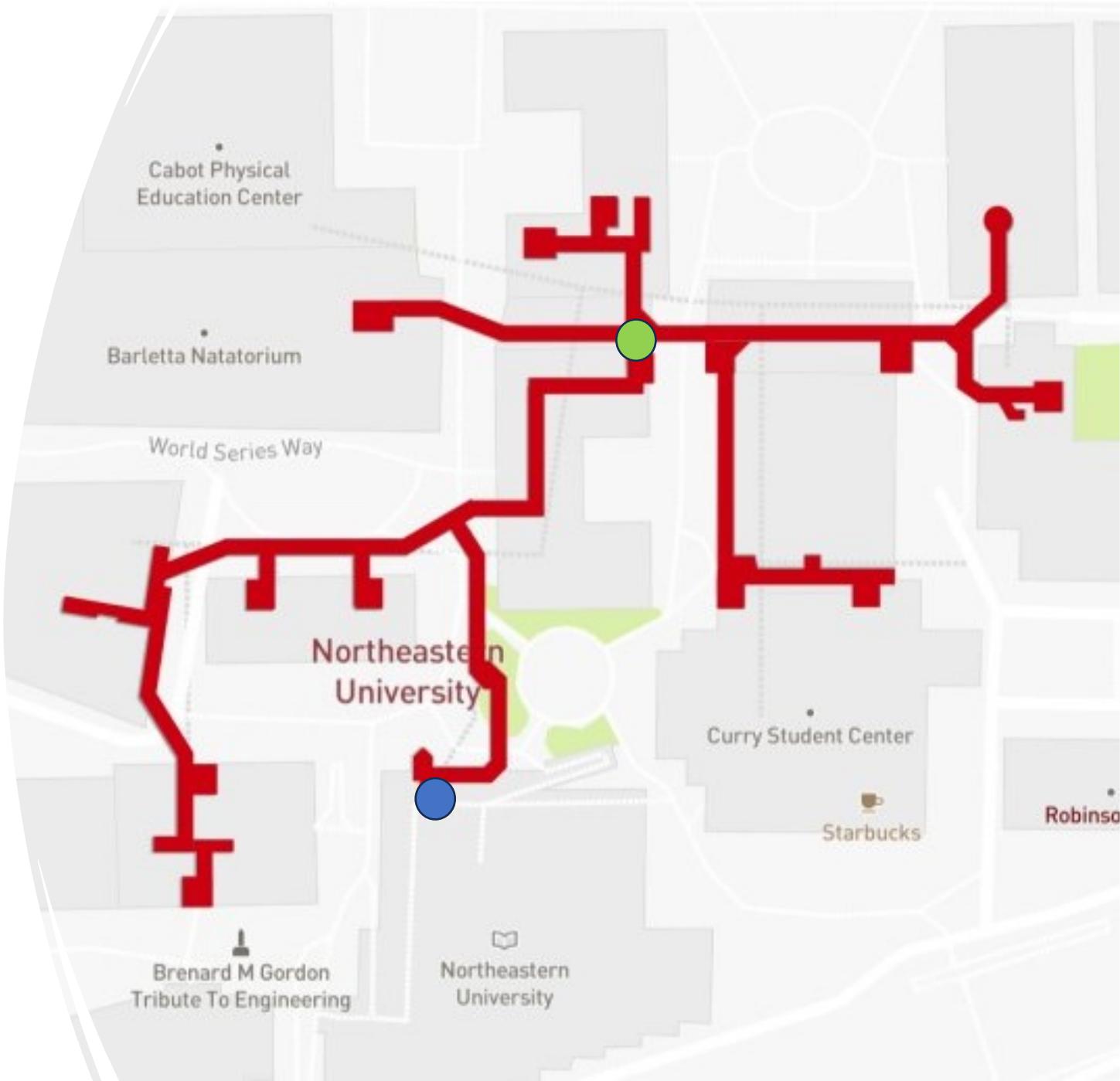


[High-Bay Rviz Spot](#)

# Future plan

---

- Mapping NEU Tunnels
- Applying LIO-SAM to the collected data
- Applying Loop closer
- Trying multi floor mapping



# Nothing went according to our Plan:

(It's part of learning)

- ~~Mapping NEU Tunnels~~ → Mapped Exp building ✓
- ~~Applying LIO-SAM to the collected data~~ → Tried to apply LIO-SAM to the collected data (IMU issue) ✓
- ~~Applying Loop closer~~ → Still trying LIO-SAM ✓
- ~~Trying multi floor mapping~~ → Let's just hope ✓



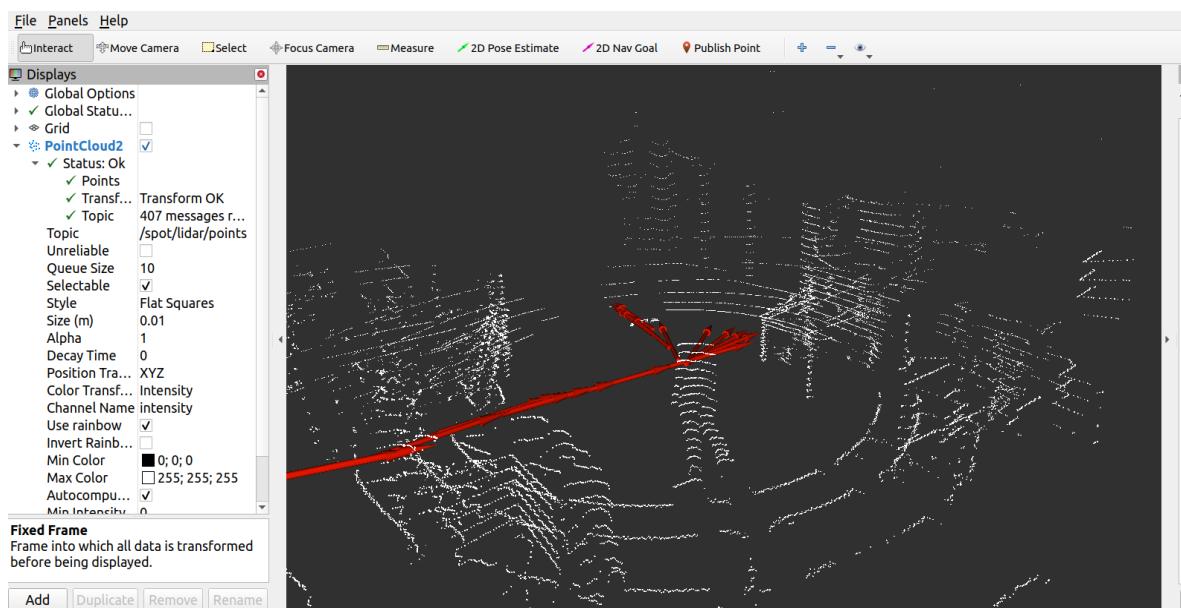
# Further progress, Challenges and Outcomes:

## 1. Improved Understanding of LIO-SAM:

- We tried to dig in the algorithm code to find a workaround of IMU.
- Also Tried to take the quaternions and modify it for our use.

```
/spot/depth/right/depth_in_visual/camera_info
/spot/dock/status
/spot/lidar/points
/spot/motion_or_idle_body_pose/status
/spot/navigate_route/status
/spot/navigate_to/status
/spot/odometry
/spot/odometry_twist
/spot/odometry_corrected
/spot/status/battery_states
/spot/status/behavior_faults
/spot/status/estop
/spot/status/feedback
/spot/status/feet
/spot/status/leases
/spot/status/metrics
/spot/status/mobility_params
/spot/status/motion_allowed
/spot/status/power_state
/spot/status/system_faults
/spot/status/wifi
/spot/trajectory/status
/spot/world_objects
/tf
/tf_static
dhvey@jarvis:~/catkin_ws$ S
```

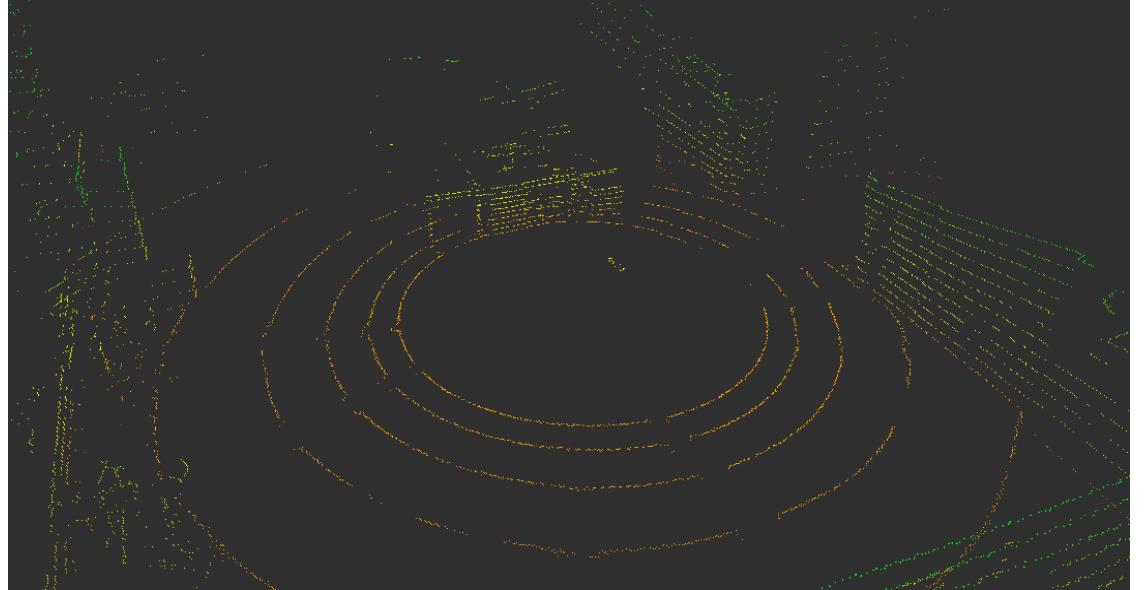
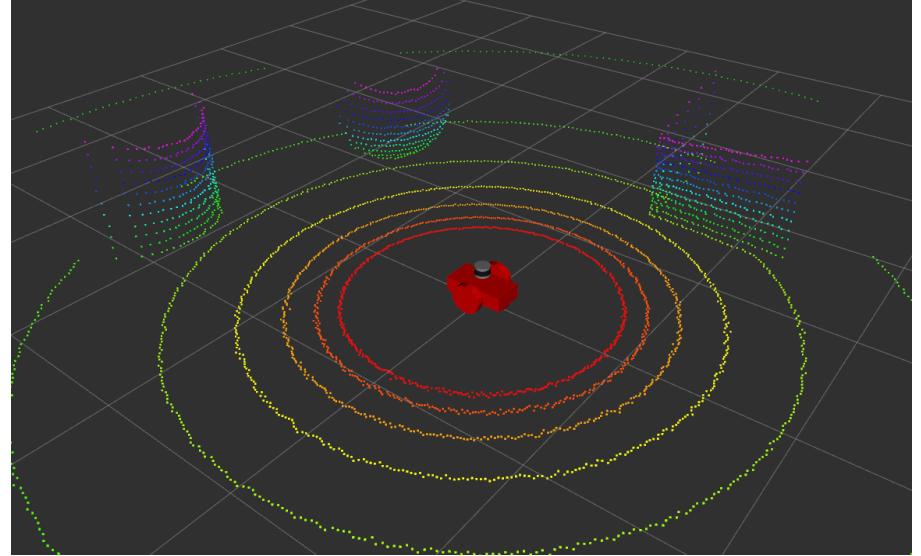
```
dhvey@jarvis:~/catkin_ws$ rosmsg info nav_msgs/Odometry
std_msgs/Header header
  uint32 seq
  time stamp
  string frame_id
  string child_frame_id
geometry_msgs/PoseWithCovariance pose
  geometry_msgs/Pose pose
    geometry_msgs/Point position
      float64 x
      float64 y
      float64 z
    geometry_msgs/Quaternion orientation
      float64 x
      float64 y
      float64 z
      float64 w
  float64[36] covariance
geometry_msgs/TwistWithCovariance twist
  geometry_msgs/Twist twist
    geometry_msgs/Vector3 linear
      float64 x
      float64 y
      float64 z
    geometry_msgs/Vector3 angular
      float64 x
      float64 y
      float64 z
  float64[36] covariance
```



# Challenges and Outcomes:

- Time\_stamp issue (in our own custom driver).
- The ring and intensity data not present in the SPOT\_IMU msg. (even though the rings were visible in rviz, Bottom RIGHT IMAGE)

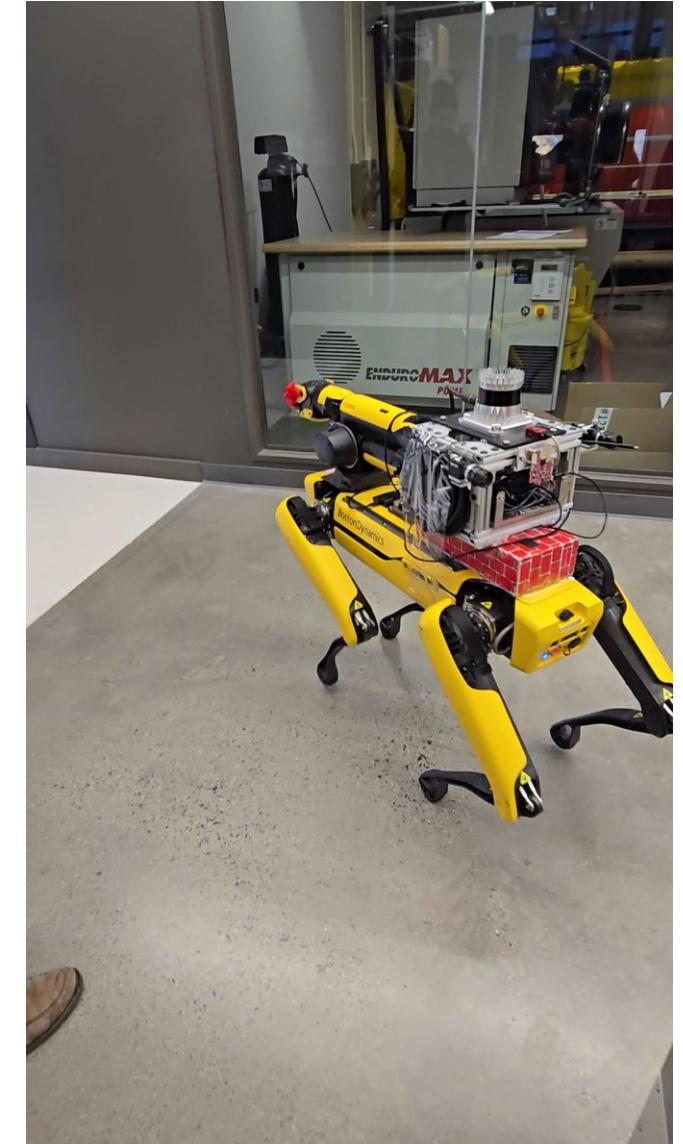
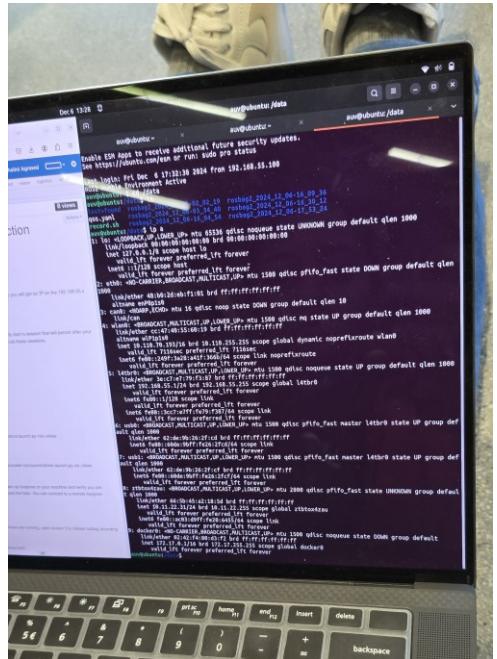
```
dhyey@jarvis:~/catkin_ws$ rosmsg info sensor_msgs/PointCloud2
[sensor_msgs/Header header
[  uint32 seq
[  time stamp
[  string frame_id
[  uint32 height
[  uint32 width
[  sensor_msgs/PointField[] fields
[    uint8 INT8=1
[    uint8 UINT8=2
[    uint8 INT16=3
[    uint8 UINT16=4
[    uint8 INT32=5
[    uint8 UINT32=6
[    uint8 FLOAT32=7
[    uint8 FLOAT64=8
[    string name
[    uint32 offset
[    uint8 datatype
[    uint32 count
[    bool is_bigendian
[    uint32 point_step
[    uint32 row_step
[    uint8[] data
[    bool is_dense
]
dhyey@jarvis:~/catkin_ws$
```



# Further progress, Challenges and Outcomes:

## 2. Collected new Data with spot: (with new rig)

- We used new rig attached on spot with a separate jetson nano, lidar and vector nav IMU.
- We then took spot for a walk around the EXP building from High-Bay to a left & right loop, around and back.

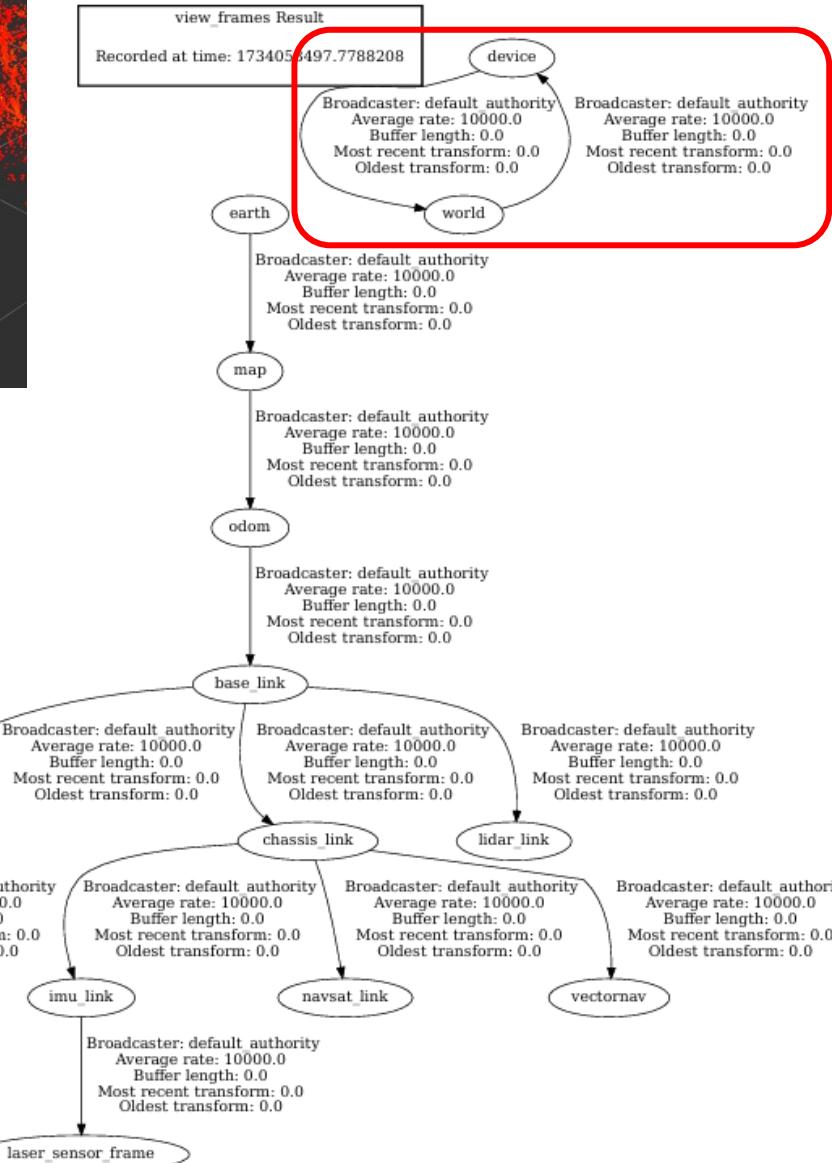
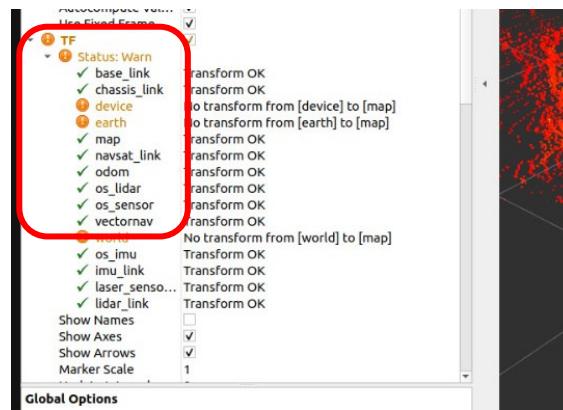
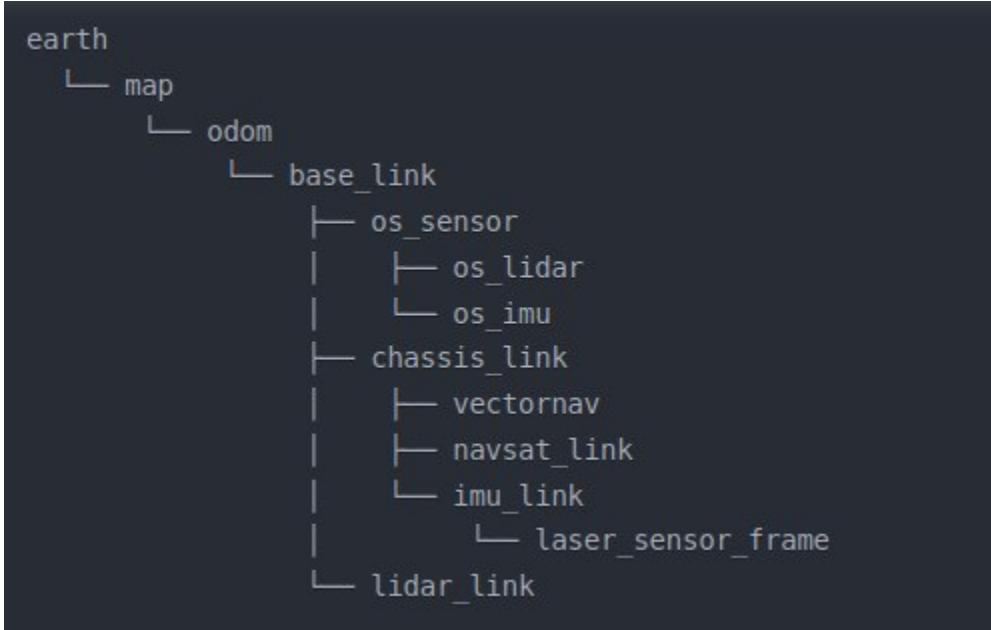


[EXP Spot Data Rec](#)

# Challenges and Outcomes:

-Initially we were not able to convert ros2bag that was recorded to ros1bag for our LIO-SAM algorithm, but later we found the ros2 support of the LIO-SAM for ubuntu 22.04 ROS2 – Humble.

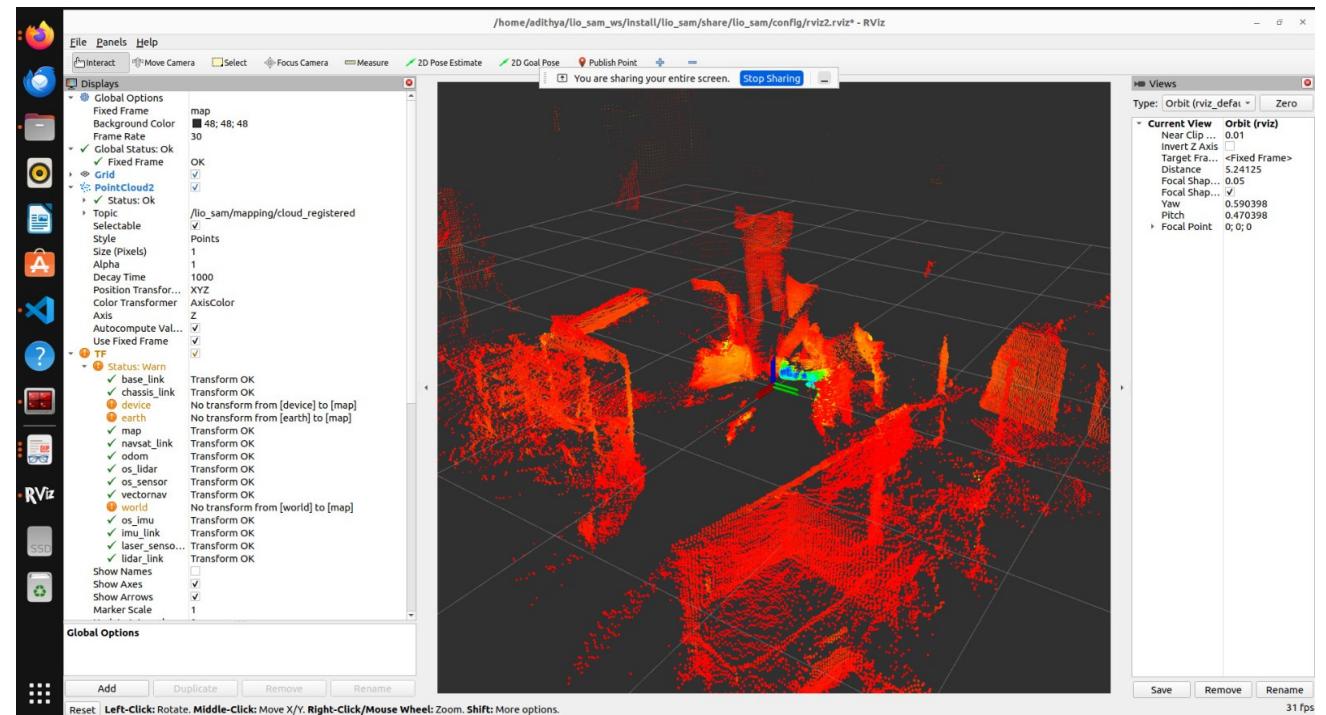
- We are getting errors in device and earth TF.
- The world and device TF is looping in itself.



# Challenges and Outcomes:

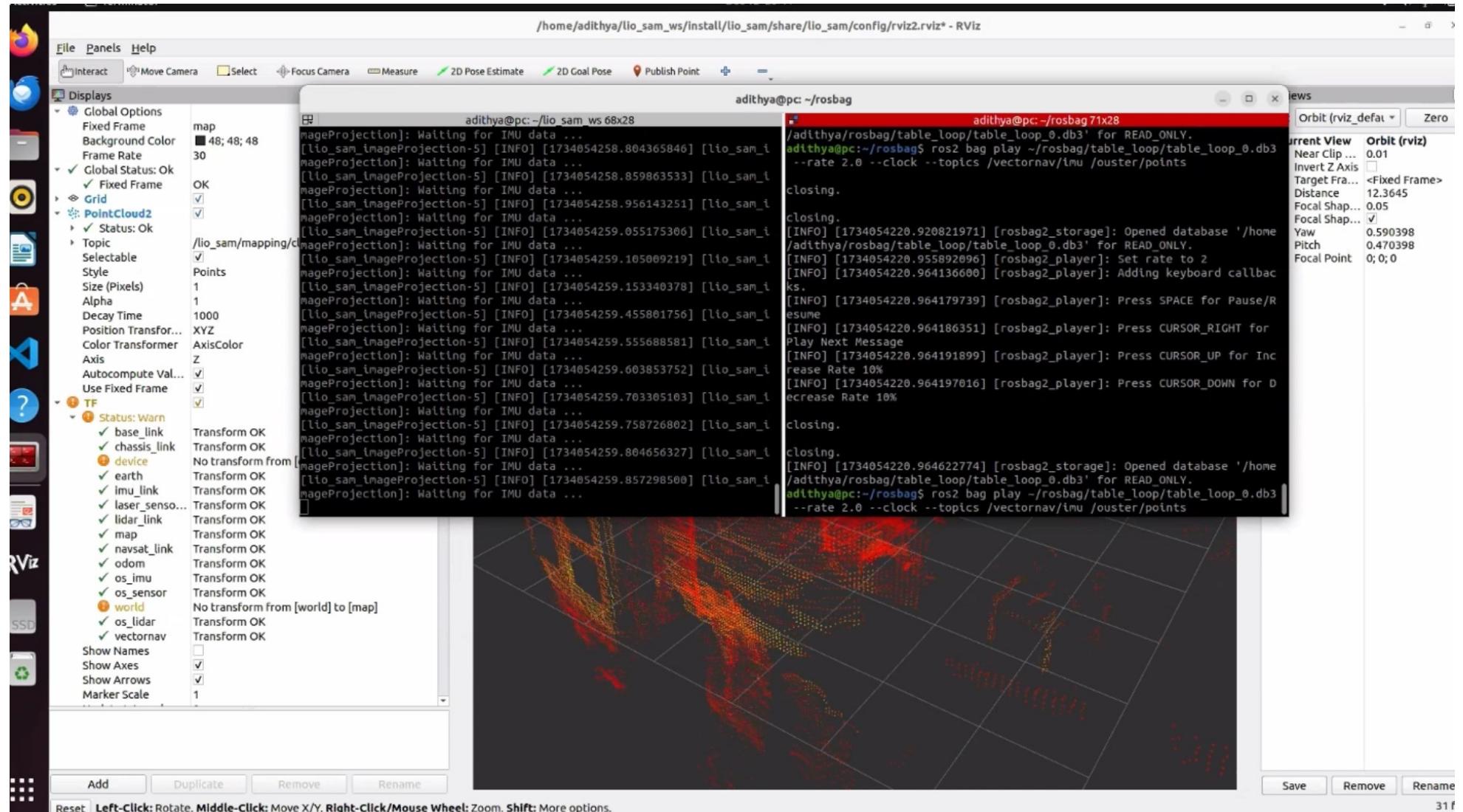
- Since the LIO-SAM is been tested in Microstrain IMUs since we are using vectornav the LIO-SAM package is not able to receive the data properly

```
adithya@pc:~/lio_sam_ws 68x28
[lio_sam_imageProjection-5] [INFO] [1734054316.884128189] [lio_sam_i
mageProjection]: Waiting for IMU data ...
[lio_sam_imageProjection-5] [INFO] [1734054317.134660124] [lio_sam_i
mageProjection]: Waiting for IMU data ...
[lio_sam_imageProjection-5] [INFO] [1734054317.185566834] [lio_sam_i
mageProjection]: Waiting for IMU data ...
[lio_sam_imageProjection-5] [INFO] [1734054317.233844918] [lio_sam_i
mageProjection]: Waiting for IMU data ...
[lio_sam_imageProjection-5] [INFO] [1734054317.284827350] [lio_sam_i
mageProjection]: Waiting for IMU data ...
[lio_sam_imageProjection-5] [INFO] [1734054317.386087962] [lio_sam_i
mageProjection]: Waiting for IMU data ...
[lio_sam_imageProjection-5] [INFO] [1734054317.435602705] [lio_sam_i
mageProjection]: Waiting for IMU data ...
[lio_sam_imageProjection-5] [INFO] [1734054317.46122520] [lio_sam_i
mageProjection]: Waiting for IMU data ...
[lio_sam_imageProjection-5] [INFO] [1734054317.533608407] [lio_sam_i
mageProjection]: Waiting for IMU data ...
^C[WARNING] [launch]: user interrupted with ctrl-c (SIGINT)
[lio_sam_featureExtraction-6] [INFO] [1734054326.861777787] [rclcpp]
: signal_handler(signum=2)
[static_transform_publisher-2] [INFO] [1734054326.861779951] [rclcpp]
: signal_handler(signum=2)
[lio_sam_imuPreintegration-4] [INFO] [1734054326.861805300] [rclcpp]
: signal_handler(signum=2)
[lio_sam_mapOptimization-7] [INFO] [1734054326.861811636] [rclcpp]:
signal_handler(signum=2)
[rviz-8] [INFO] [1734054326.861811721] [rclcpp]: signal_handler(signum=2)
```



- On the left image it can be seen that, there is no IMU data, and it shows "waiting for IMU data ... "

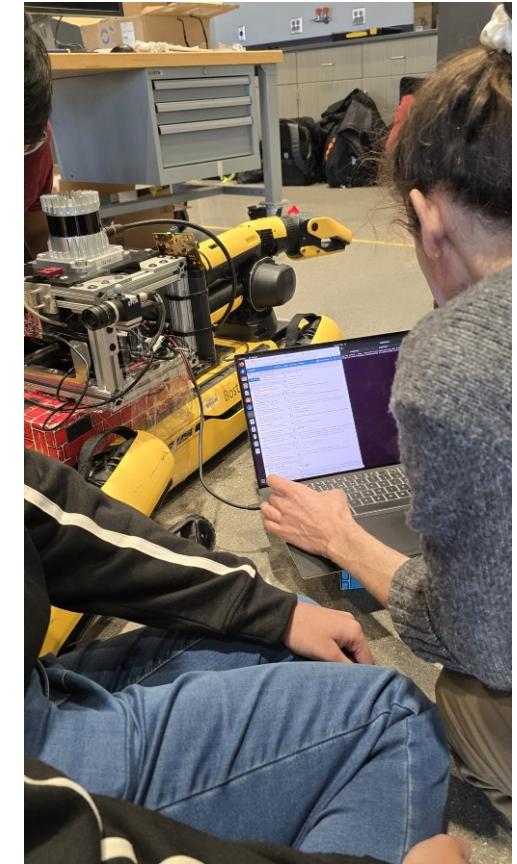
# Challenges and Outcomes:



[Spot new Test rig Data Rviz - ROS2](#)

# Results and Discussion:

- Successfully implemented the LIO-SAM algorithm on the KITTI dataset of MIT buildings, demonstrating functional loop closure and mapping capabilities.
- Conducted data collection in the High Bay environment, producing high-quality point cloud data visualized effectively in Rviz.
- Resolved initial issues with converting ROS2 bags to ROS1 compatibility by utilizing LIO-SAM support on Ubuntu 22.04 with ROS2-Humble.
- Integrated a new rig on the Spot robot featuring a separate Jetson Nano, LiDAR, and VectorNav IMU. The system enabled thorough data collection across the EXP building and dynamic loops.
- Exploring the algorithm's internals improved the team's grasp of LIO-SAM's intricacies, including factor graph optimization and IMU integration.
- Modifications to quaternions for custom use cases demonstrated adaptability in problem-solving.
- The reliance on specific sensor configurations limits LIO-SAM's flexibility across diverse platforms. Future work could explore making the algorithm sensor-agnostic.
- Expanding tests to multi-floor and tunnel environments is necessary to validate real-world applicability.
- Despite the challenges, this work highlights the robustness and potential of LIO-SAM for dynamic, GPS-denied environments, paving the way for refined 3D localization and mapping solutions.





Any questions?

## References:

- <https://github.com/TixiaoShan/LIO-SAM?tab=readme-ov-file>
- T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping"
- [https://drive.google.com/drive/folders/1ofWXLbhqu4m-7foQu\\_30O7dn41ErS4w?usp=sharing](https://drive.google.com/drive/folders/1ofWXLbhqu4m-7foQu_30O7dn41ErS4w?usp=sharing)

(The above link is our drive link for Spot data)



Northeastern  
University



BostonDynamics