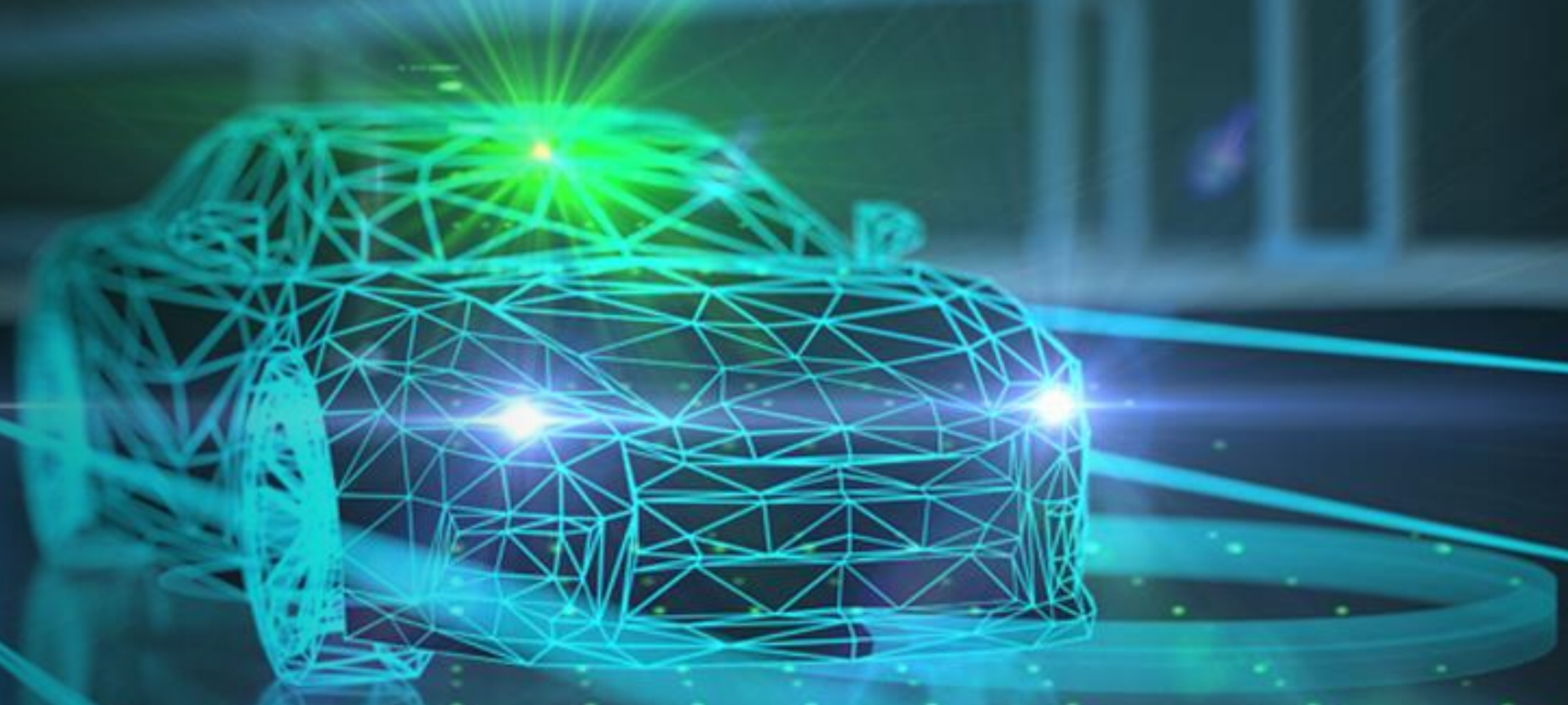
VISUAL SLAM

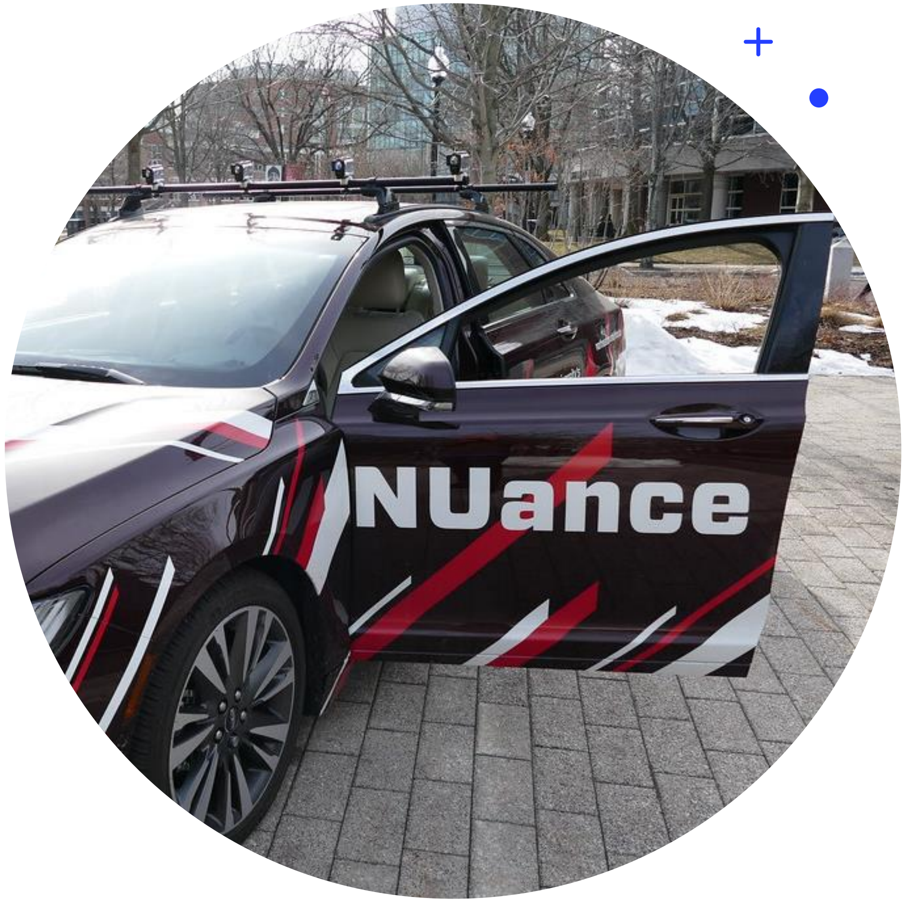


EECE 5554 Final Project

Becky Johnston
Dhruvil Parikh
Samruddhi Raut
Claire Romanczyk

Project Description

- Use Visual SLAM techniques to reconstruct a 3D map of an outdoor environment
  - VSLAM pipeline: matching extracted features with known features, estimating camera poses, optimizing poses and map points, and performing loop closure
- Test ORB SLAM 3 algorithm with provided EuRoC, TUM-VI, and KITTI datasets to compare
- Use the ORB SLAM 3 algorithm on provided ‘car_IR_RGB_lidar’, create a map, and compare with GPS data



What We Used

Hardware

- NEU data used RGB Cameras (synced to form a stereo pair), IR Camera, Lidar, IMU, GPS
- NVIDIA GPU for processing the data

Software

- ORB SLAM 3 with dependencies
  - Pangolin
  - OpenCV
  - Eigen3
  - DB0W2
  - G20
  - Python and C++ compilers
- With ROS Noetic on Ubuntu 20.04 for NEU dataset
- Without ROS for EuRoC, KITTI, and TUM VI datasets

Analysis

- Visual observation of map generation and loop closures while running the algorithm
- Analysis in Python to graph results against ground truth
- Visual comparison of map generation/start and end points

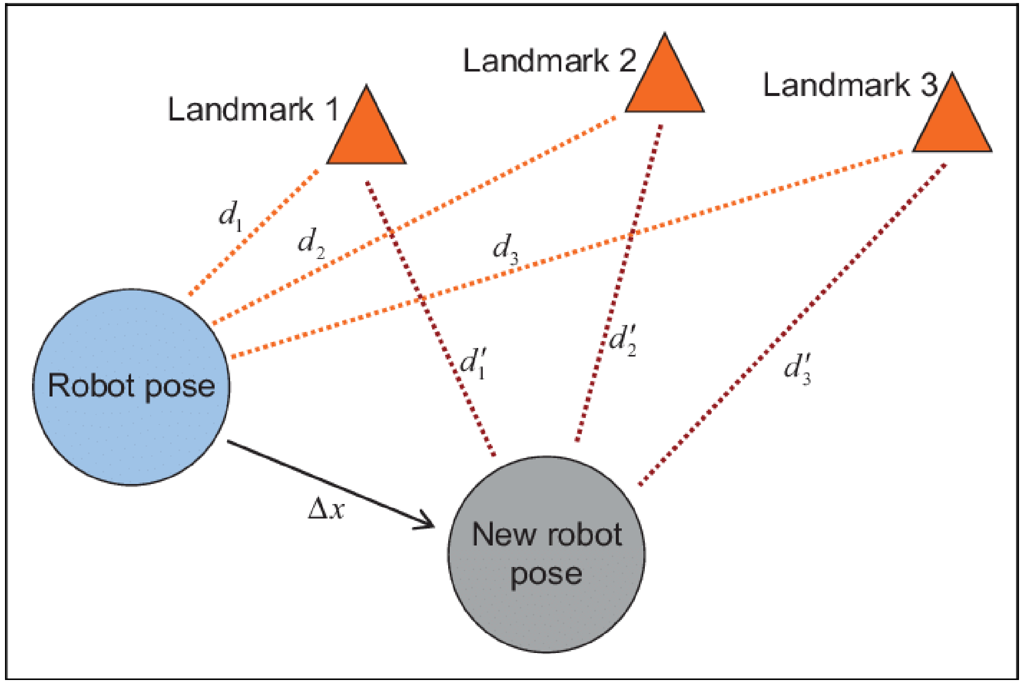
VISUAL SLAM

SLAM

SLAM is a technique used to build up a map within an unknown environment or a known environment while at the same time keeping track of the current location.

Mapping: ”What does the world look like?” Integration of the information gathered with sensors into a given representation

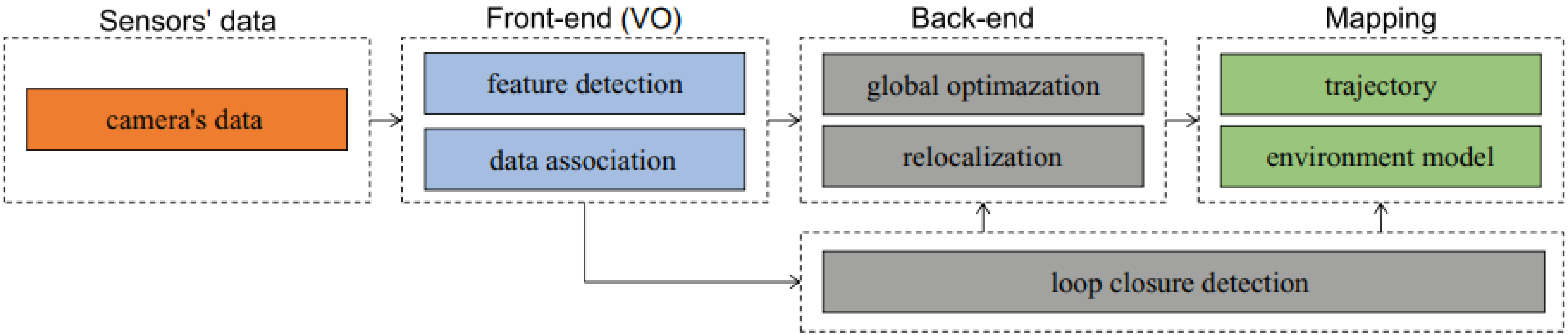
Localization: ”Where am I?” Estimation of the robot pose relative to a map.



The paradox:
In order to build a map, we must know our position
To determine our position, we need a map!

VSLAM

SLAM by using visual sensors such as monocular cameras, stereo cameras, RGB-D cameras, DVS



ORB SLAM 3

ORB SLAM

O riented
FAST
and
R otated
B RIEF

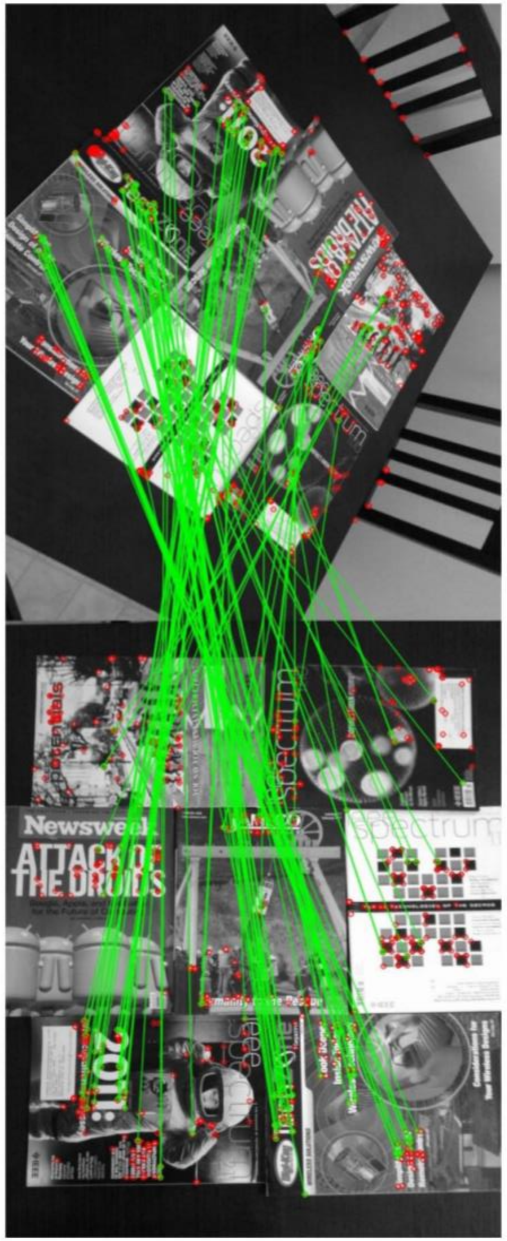
S imultaneous
L ocalization
A nd
M apping

Image processing:

- Corner detection
- Feature identification
- Feature alignment
- Image stitching

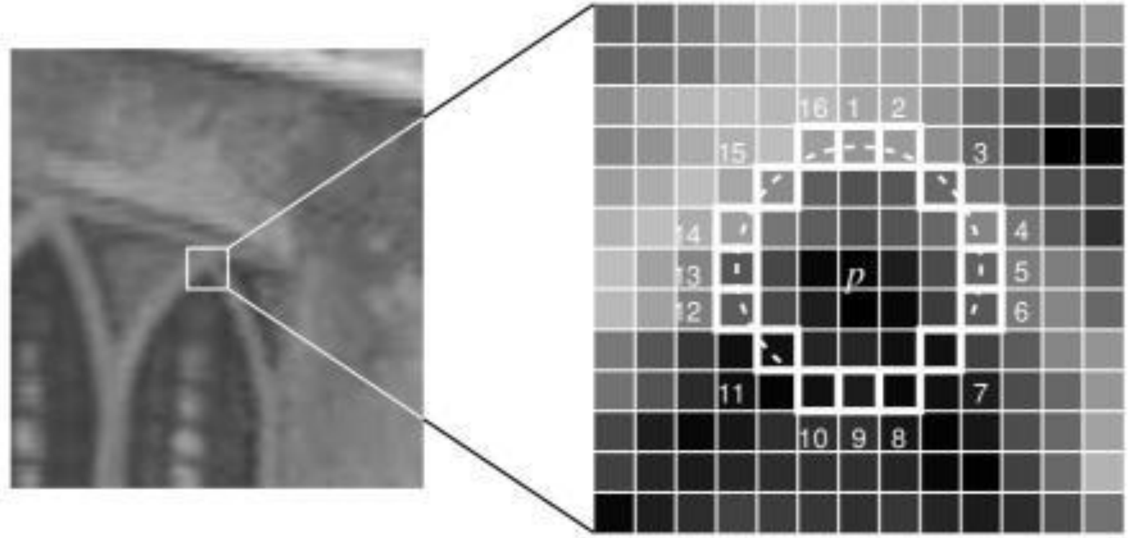
Real-time navigation (VSLAM):

- Tracking
- Mapping
- Relocalization



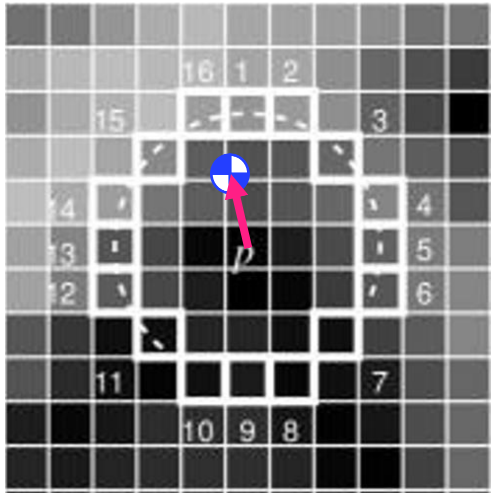
FAST

F eatures
from
A ccelerated
S egment
T est



Accelerated segment test

Corner ✓



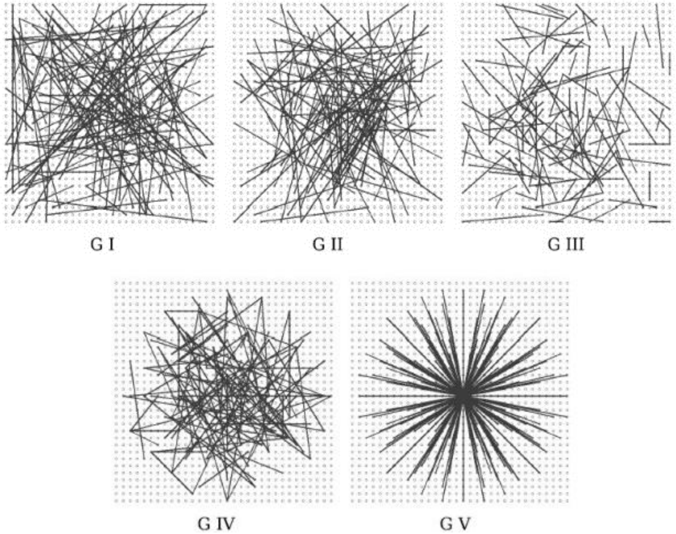
Now add orientation!

BRIEF

B inary
R obust
I ndependent
E lementary
F eatures

Step 1: smooth image

Step 2: choose points to compare



Step 3: Intensity comparisons

$$\tau(p;x,y) = \begin{cases} 1 & : p(x) < p(y) \\ 0 & : p(x) \geq p(y) \end{cases}$$

Binary feature descriptor!

$$V_1 = [1001101001\ldots]$$
$$V_2 = [1010010011\ldots]$$
$$V_3 = [0011010010\ldots]$$

Putting It Together

ORB SLAM

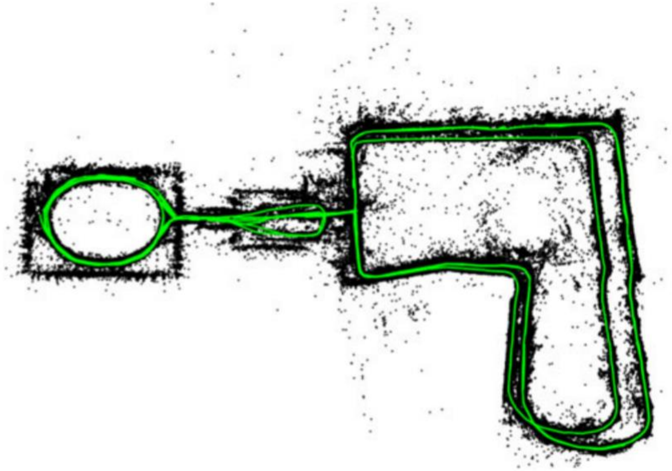
- tracking, mapping, relocalization, and loop closing uses the same features
- Real-time loop closure
- Camera relocalization is resilient to changes in viewpoint or lighting

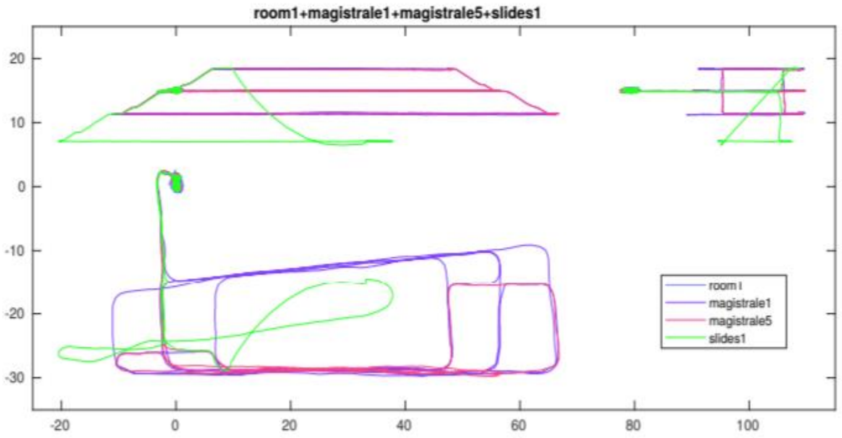
ORB SLAM2

- Added stereo and RGB-D camera capabilities
  - Depth perception
  - Knows scale of map and estimates trajectory

ORB SLAM3

- Added visual-inertial SLAM (integrates IMU)
- Added multi-map capabilities
  - Starts a new map if it gets lost
  - Compares that map to old maps to line them up





ANALYSIS

Testing and Evaluation on Various data sets

Along with the **NEU dataset** (**car_IR_RGB_lidar)** we used the following data set for analysis

- **EuRoC dataset (Stereo/Monocular, With/Without IMU).**
- **TUM-VI dataset (Fisheye Stereo/Monocular, With/Without IMU).**
- **Kitti dataset**

Associated Packages

- Pangolin: For Visualization and User Interface
- OpenCV: To Manipulate Images and Features
- Eigen3: For Linear Algebra
- DB0W2: Indexing and Converting Images into a bag-of-words representation
- G20: For optimizing graph-based non-linear error functions
- Python and C++ compilers
- ROS

Parameter Tuning

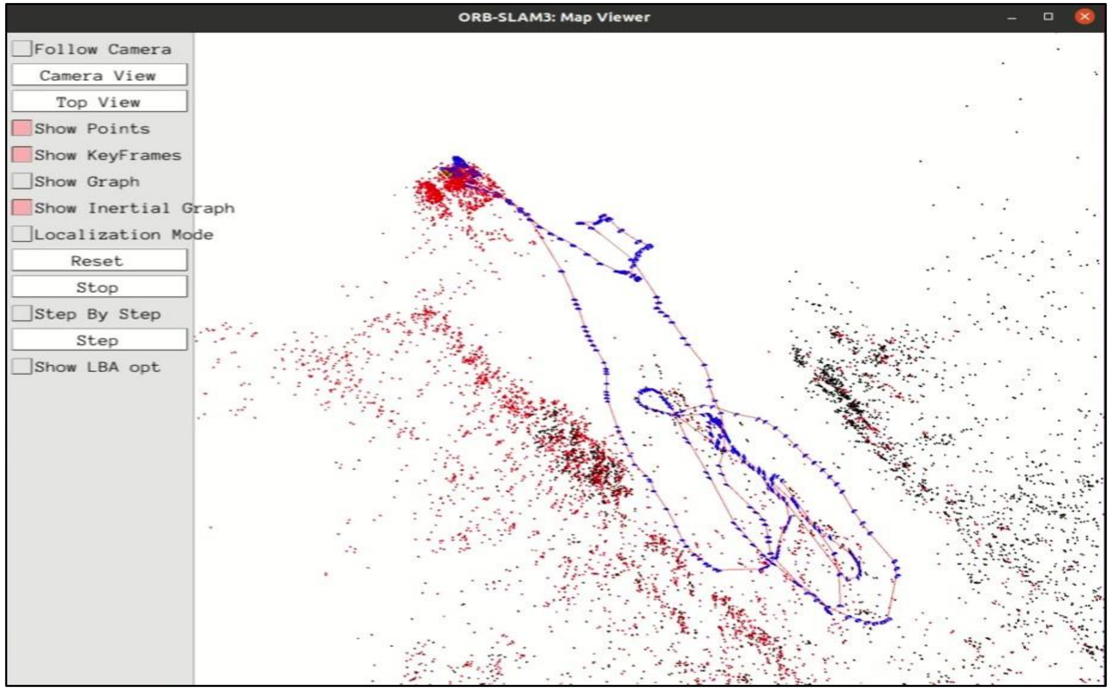
```
# Camera calibration and distortion parameters (OpenCV)
Camera.fx: 5.24888150200348832e+02
Camera.fy: 5.21776791343664968e+02
Camera.cx: 3.25596989785447420e+02
Camera.cy: 2.42392342491041603e+02

Camera.k1: -4.70302508060718438e-01
Camera.k2: 3.01057860458473880e-01
Camera.p1: 4.68835914496582538e-03
Camera.p2: 1.65573977268025185e-03

Camera.width: 640
Camera.height: 512
# Camera frames per second
Camera.fps: 60.0
```

- Parameters of the EuRoC dataset were adjusted and fine-tuned to get a good performance on the NUance car dataset
  - Camera Calibration parameters corresponding to the data provided with the bag file
  - Transformations from one camera to another
  - Stereo depth parameter
  - IMU noise parameters
  - Transformation from IMU to the left camera
  - Number of features and frames per second

EuRoC Dataset

EuRoC MAV is a visual-inertial datasets collected on-board a Micro Aerial Vehicle (MAV). The dataset contains stereo images, synchronized IMU measurements, and accurate motion and structure ground-truth. The datasets facilitates the design and evaluation of visual-inertial localization algorithms on real flight data.

Feature detection

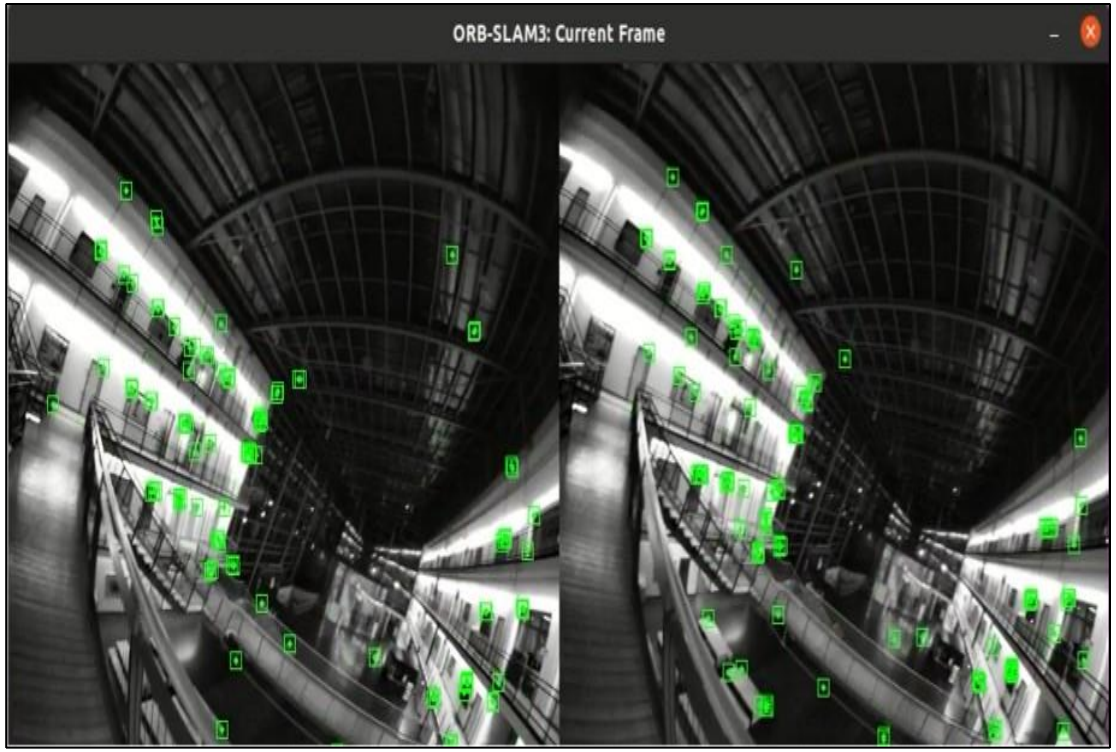


Map generation

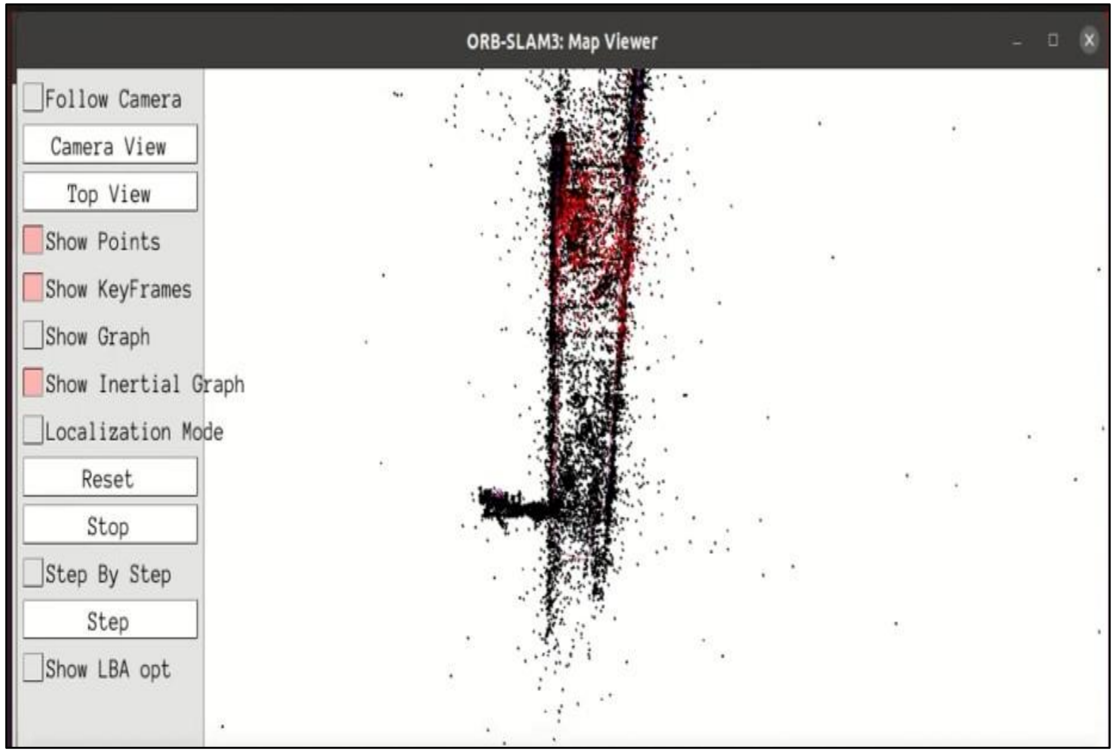
SLAM Results with EuRoC Dataset

TUM-VI Dataset

TUM VI is a novel dataset with a diverse set of sequences in different scenes for evaluating VI odometry. It provides camera images with 1024x1024 resolution at 20 Hz, high dynamic range and photometric calibration

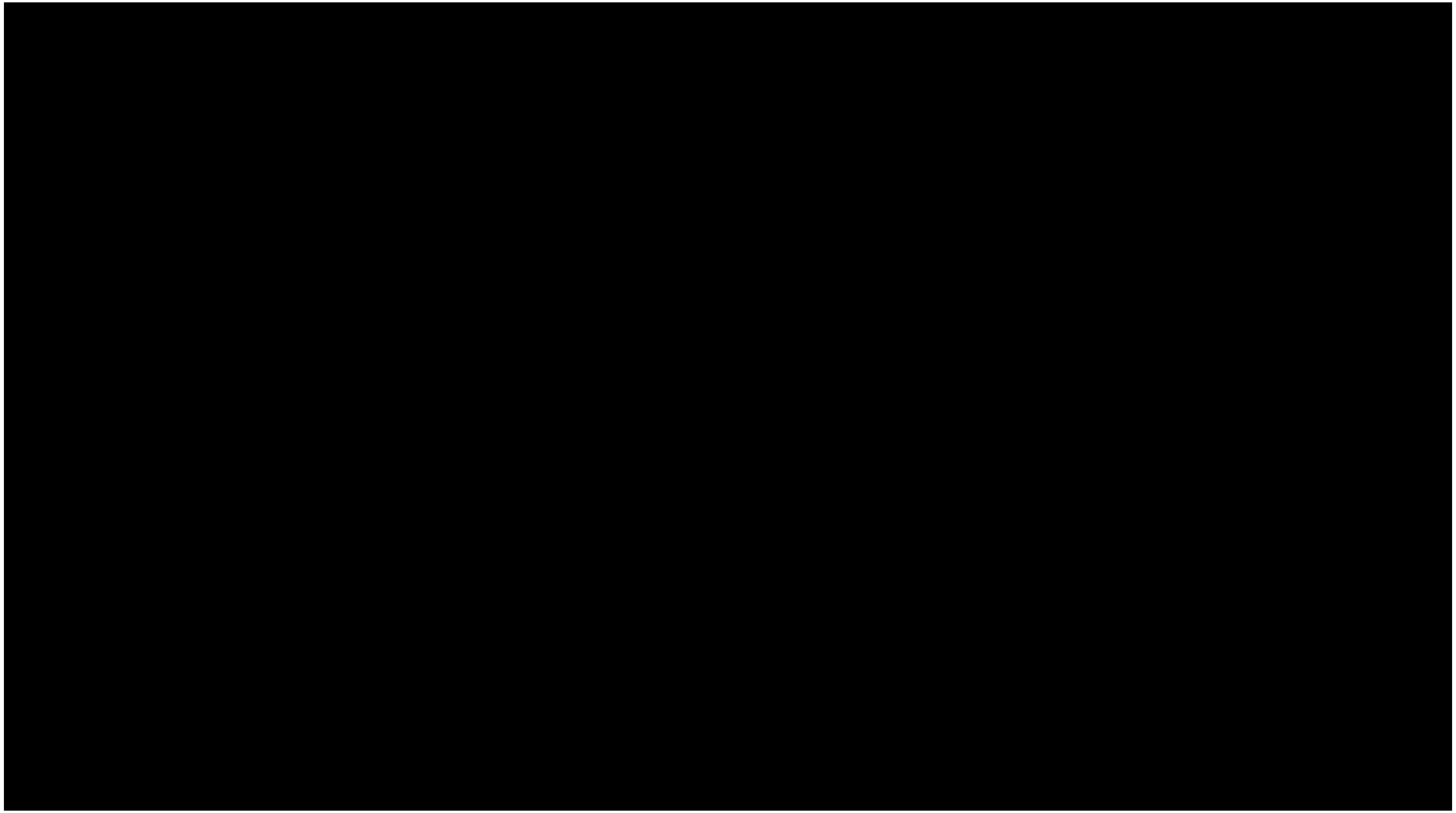


Feature detection



Map generation

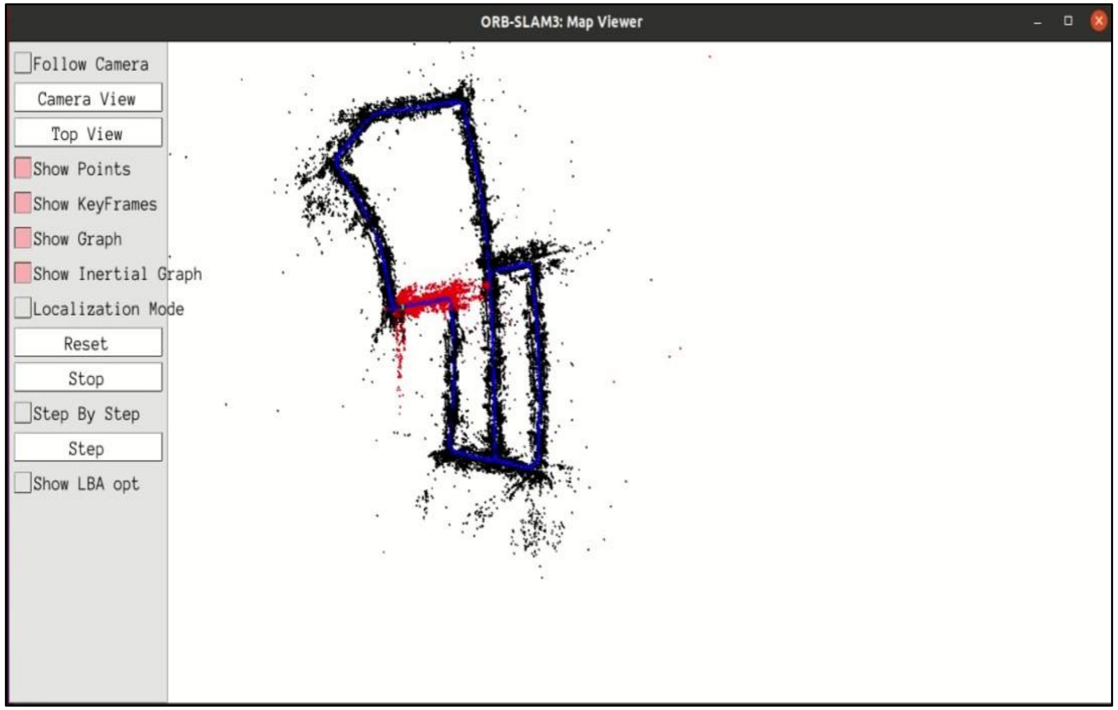
SLAM Results with TUM_VI Dataset



KITTI Dataset

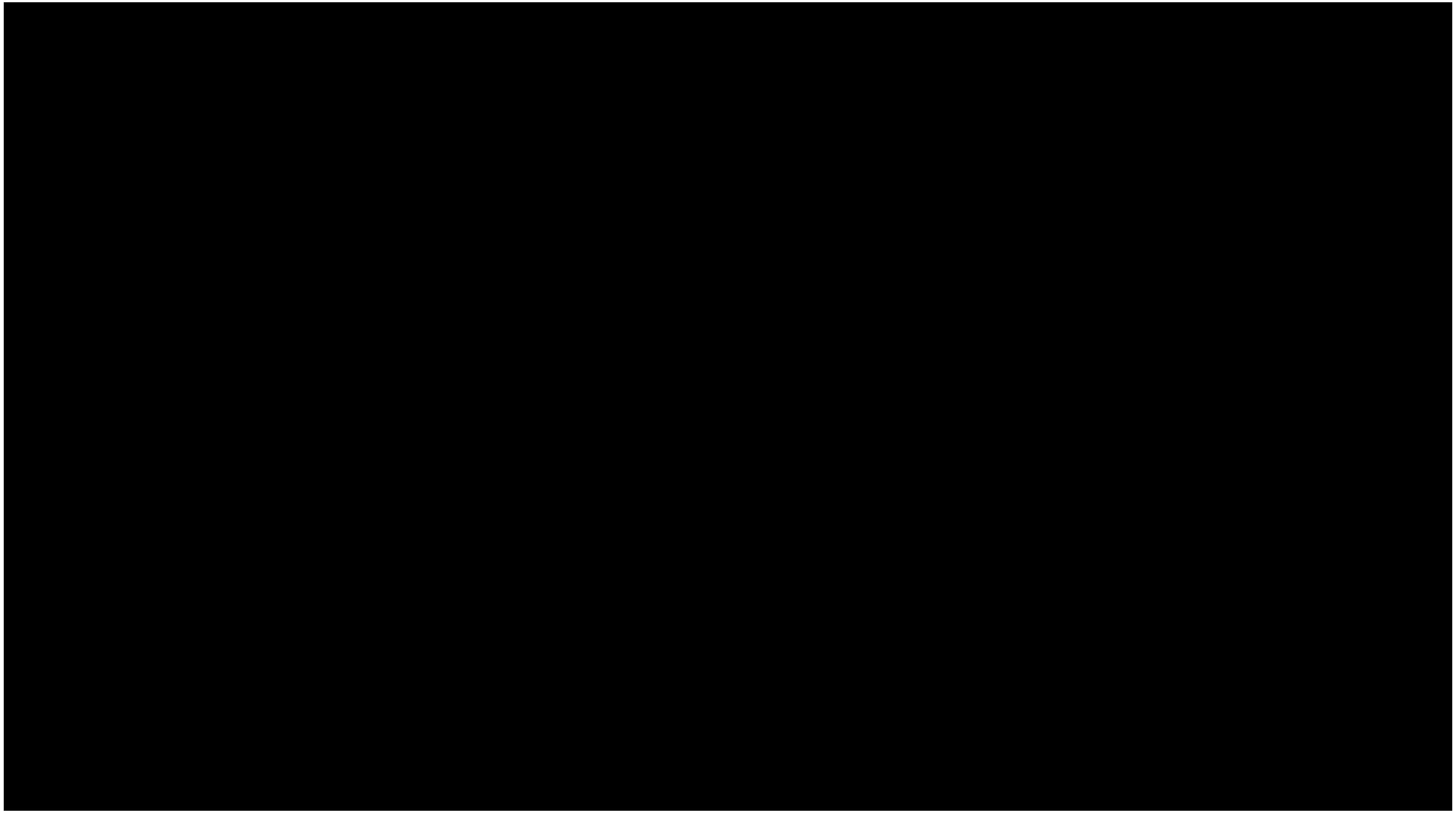
KITTI (Karlsruhe Institute of Technology and Toyota Technological Institute) is one of the most popular datasets for use in mobile robotics and autonomous driving. It consists of hours of traffic scenarios recorded with a variety of sensor modalities, including high-resolution RGB, grayscale stereo cameras, and a 3D laser scanner.

Feature detection



Map generation

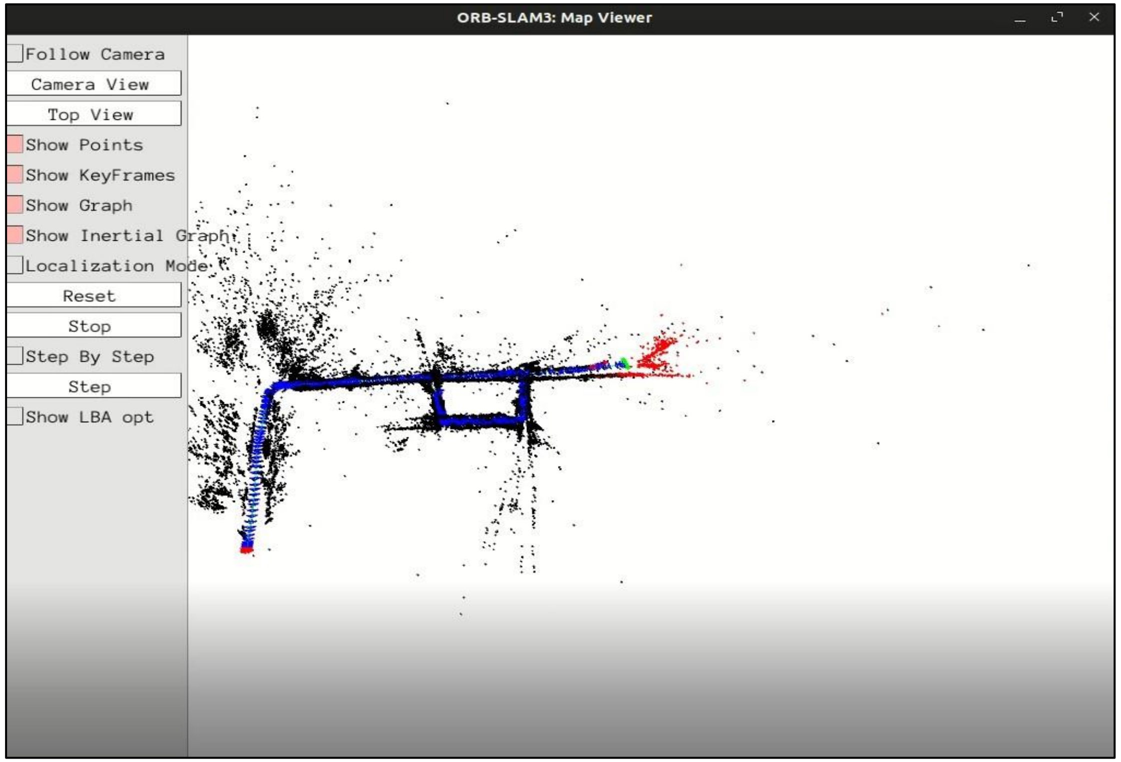
SLAM Results with KITTI Dataset



NUance Dataset

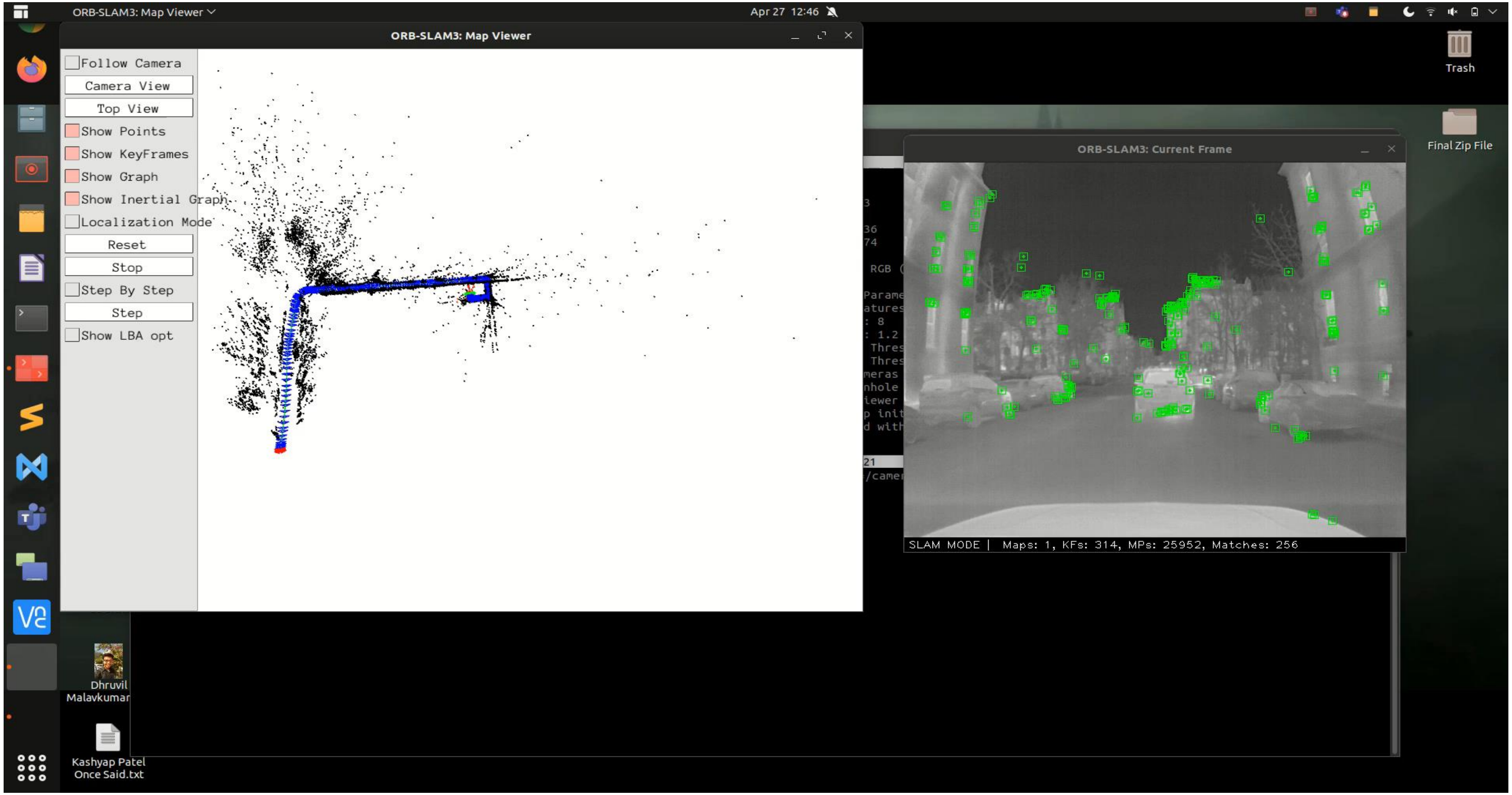
In this dataset, Northeastern's autonomous car is driven manually around Newbury Street in Boston. The dataset has stereo RGB cameras looking forward, an IR camera looking forward, 2 Velodyne VLP-16 lidar mounted on top of the car, IMU, and GPS. The main focus in this dataset was to collect camera data with at least one loop closure. The sensors like lidars, GPS, IMU in combination can serve as ground truth for visual slam algorithms.

Feature detection



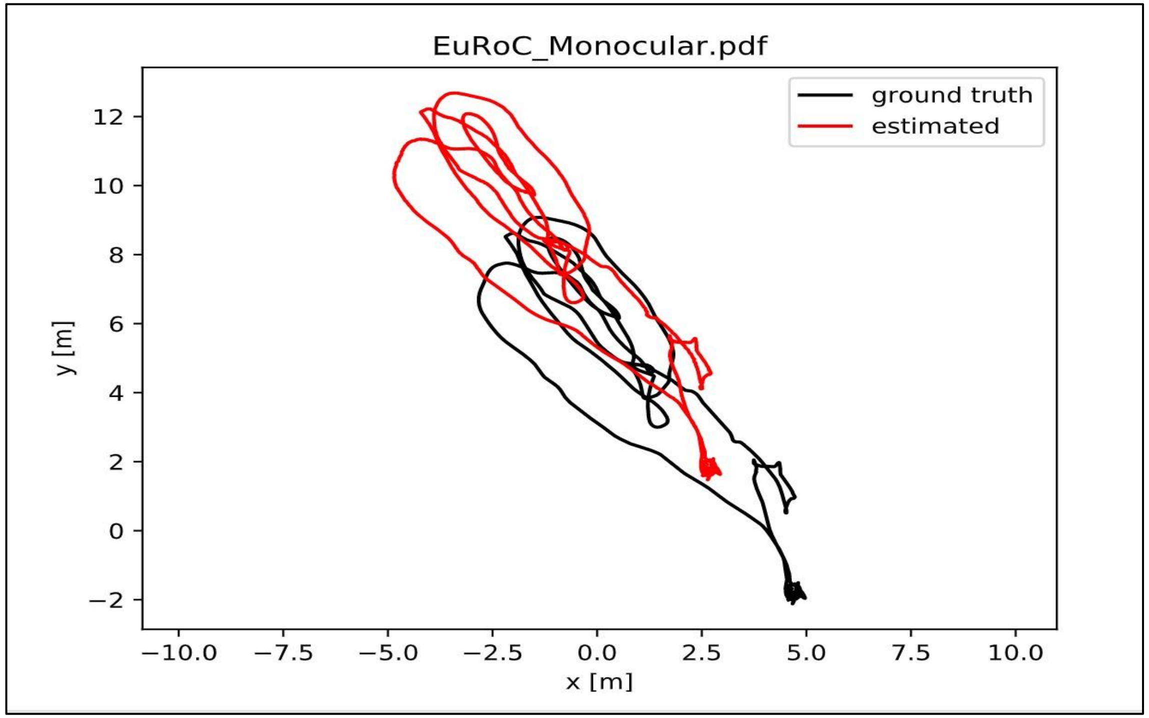
Map generation

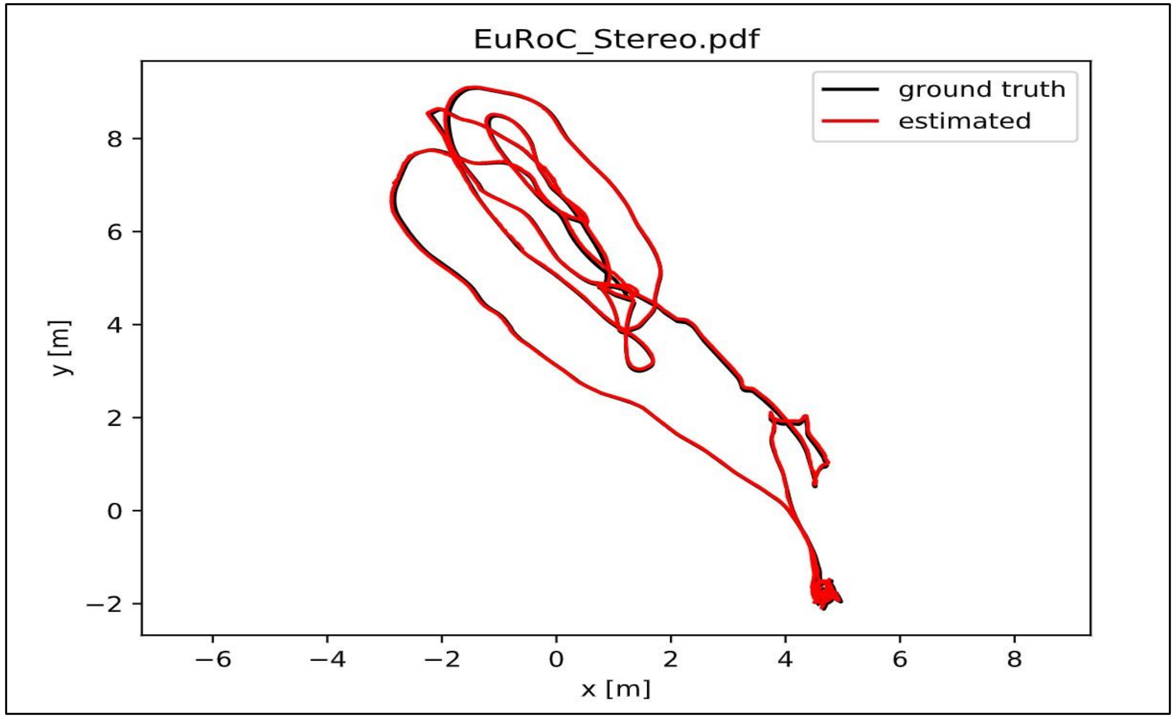
SLAM Results with NUance Dataset



Results and Comparison

EuRoC dataset : Monocular VS Stereo without IMU

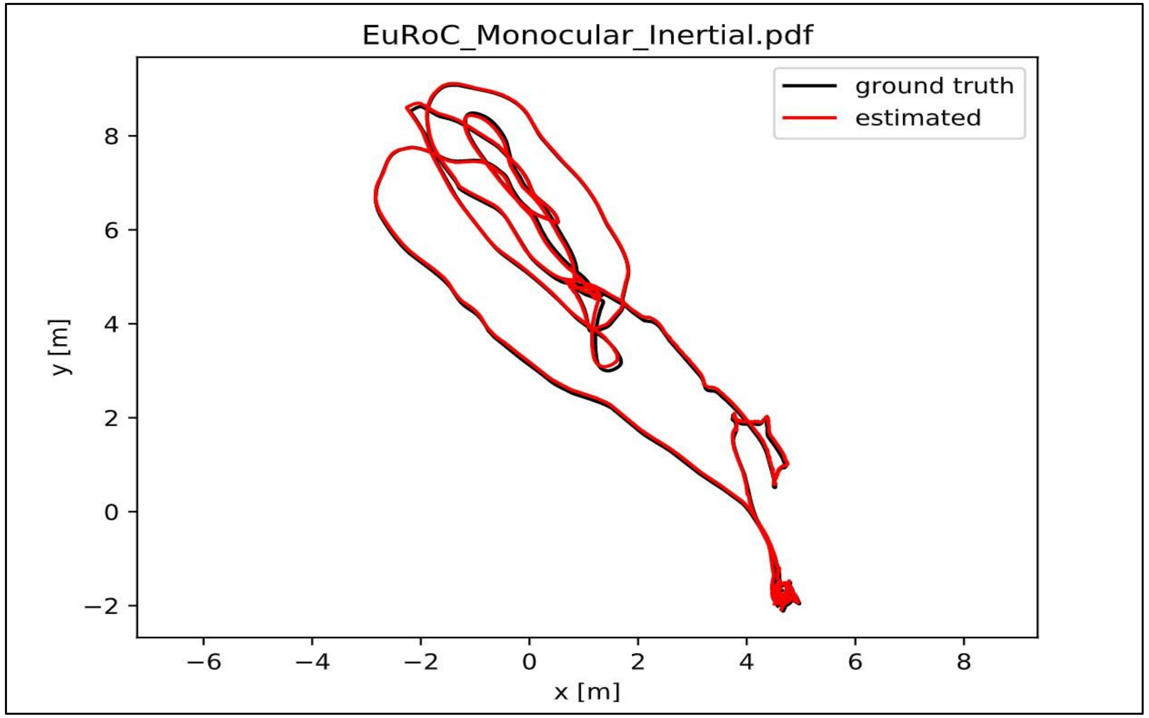


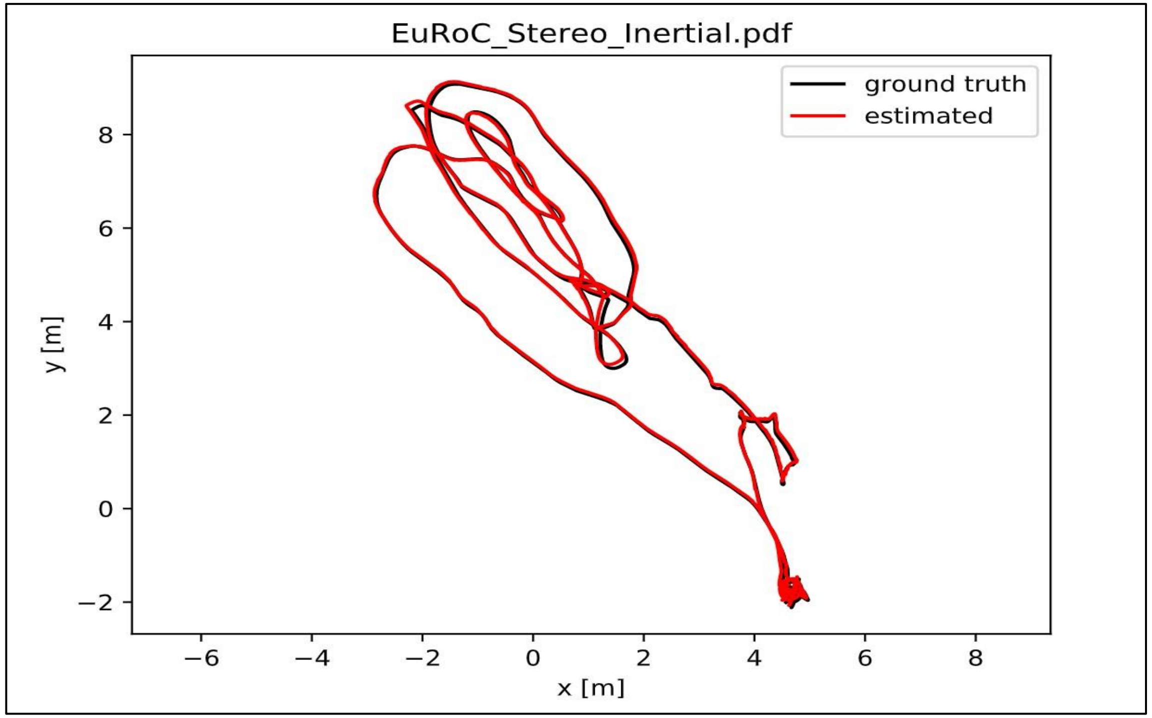


- Ground truth provided for entire segment
- Similar map generated and similar scaling for both Monocular and Stereo without IMU
- Localization is poor for Monocular, good for Stereo
- Start & end segments line up with each other for both, but only match ground truth for Stereo
- Conclusion: Stereo is necessary for good localization without IMU

Results and Comparison

EuRoC dataset : Monocular Inertial VS Stereo Inertial



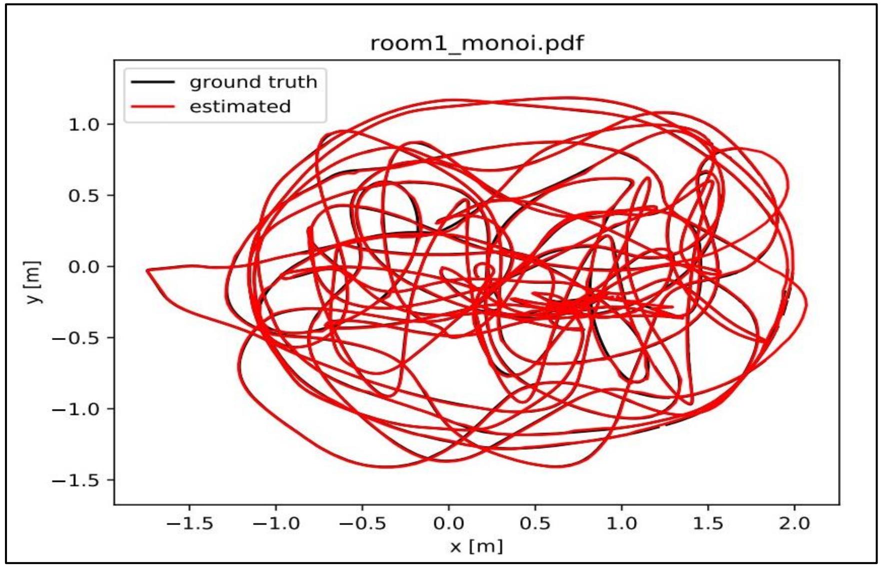


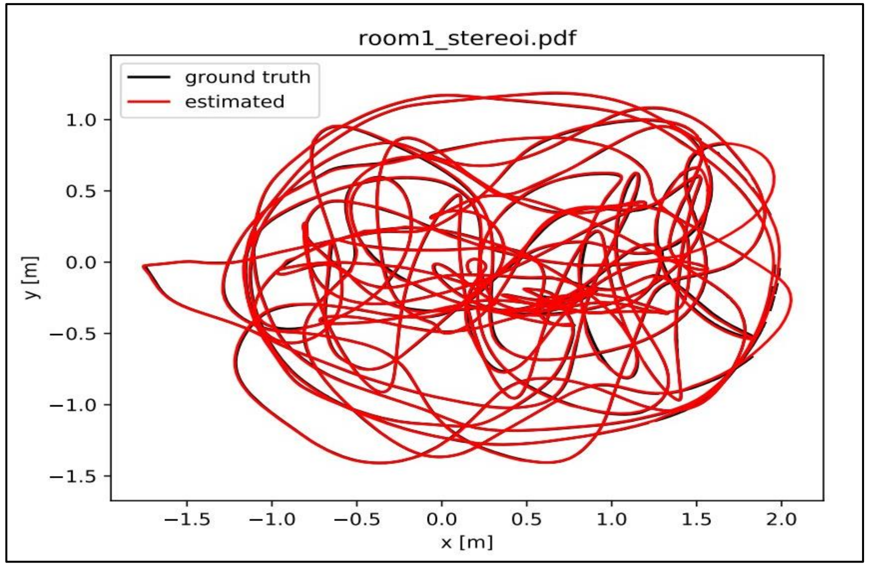
- Ground truth provided for entire dataset
- Similar map results and good localization for Monocular and Stereo with IMU
- Monocular slightly better around sharp turns, Stereo better for straighter segments
- Conclusion: no significant difference between Monocular and Stereo with IMU, Stereo just slightly better

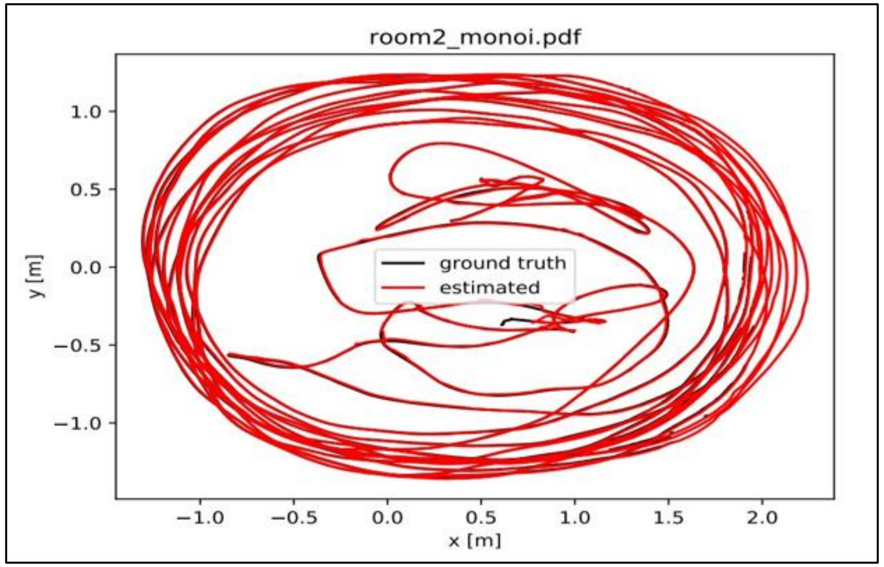
Results and Comparison

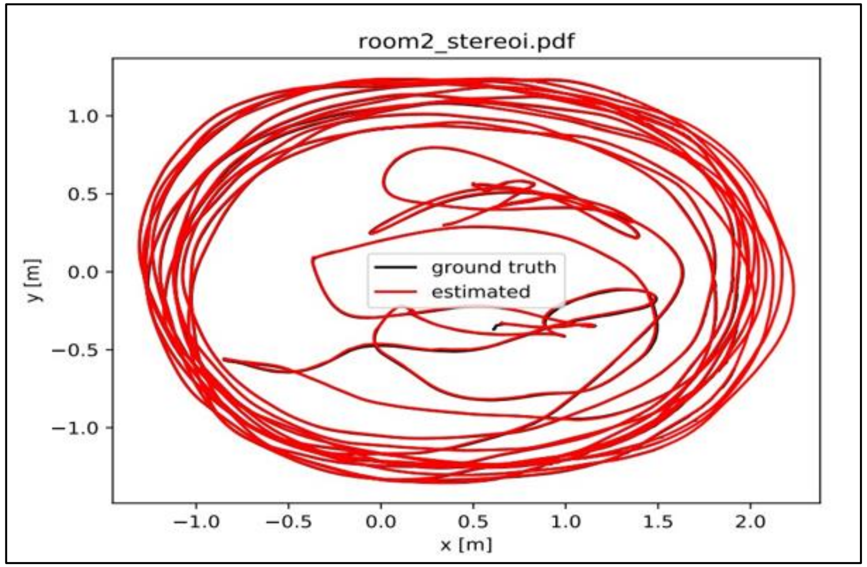
TUM-VI dataset : Monocular Inertial VS Stereo Inertial for Room1 and Room2

- Ground truth provided for entire dataset
- Many loop closures for both Room1 and Room2 dataset
- Both Monocular & Stereo with IMU performed well with localization and mapping; stereo perhaps marginally better (similar to EuRoC dataset results)
- Conclusion: Many loop closures = good mapping & little difference between Stereo & Monocular



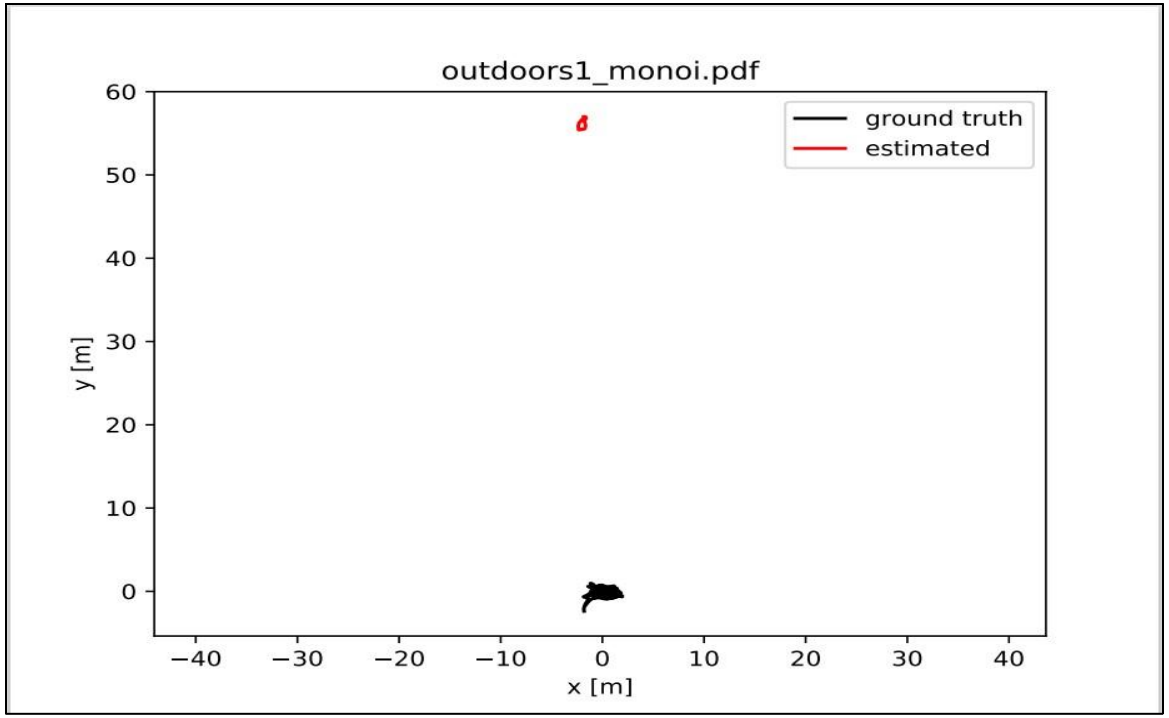


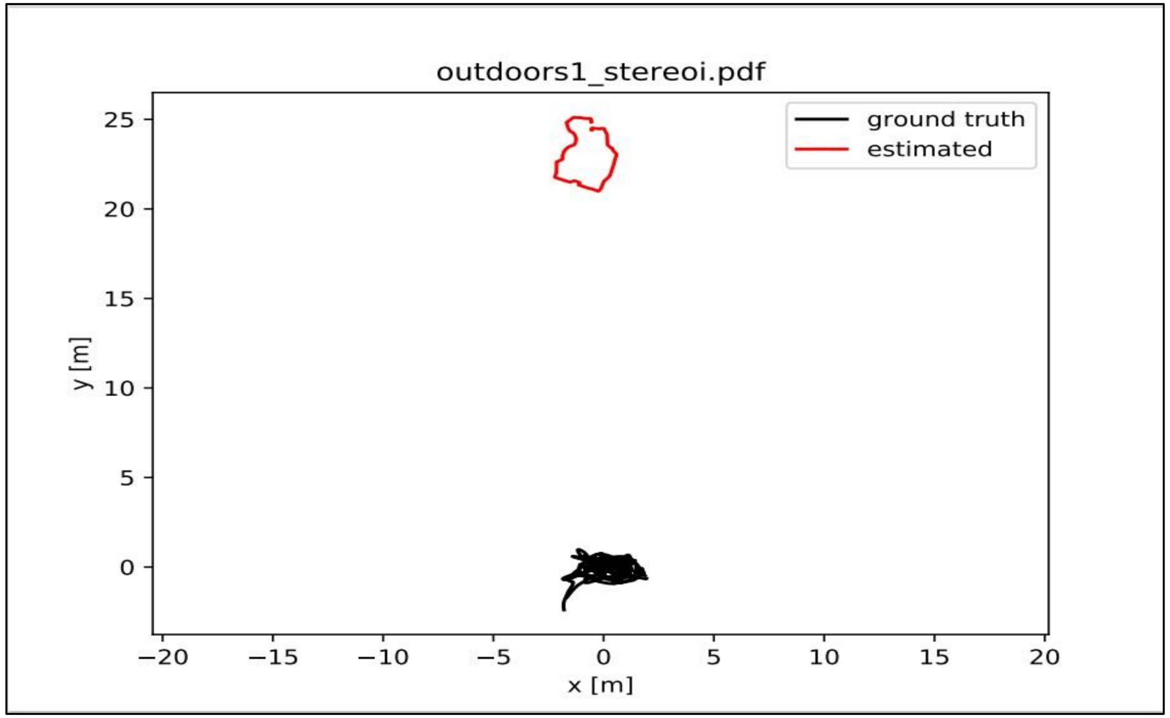




Results and Comparison

TUM-VI dataset : Outdoor Monocular Inertial VS Outdoor Stereo Inertial

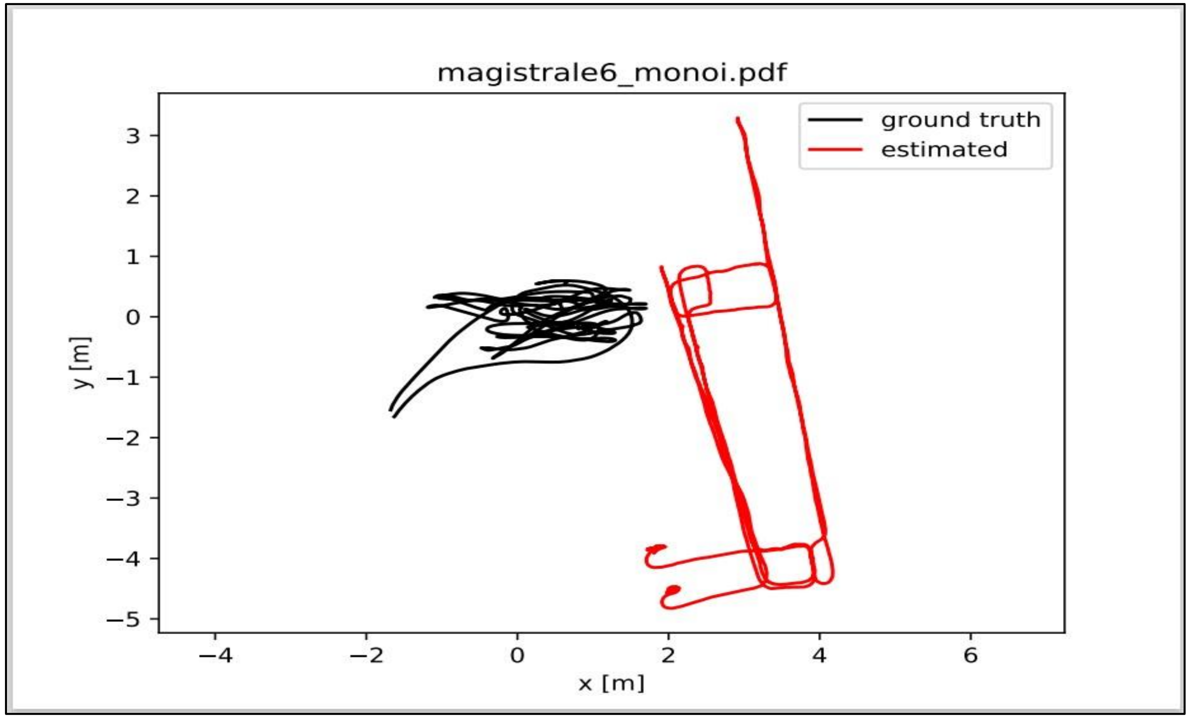


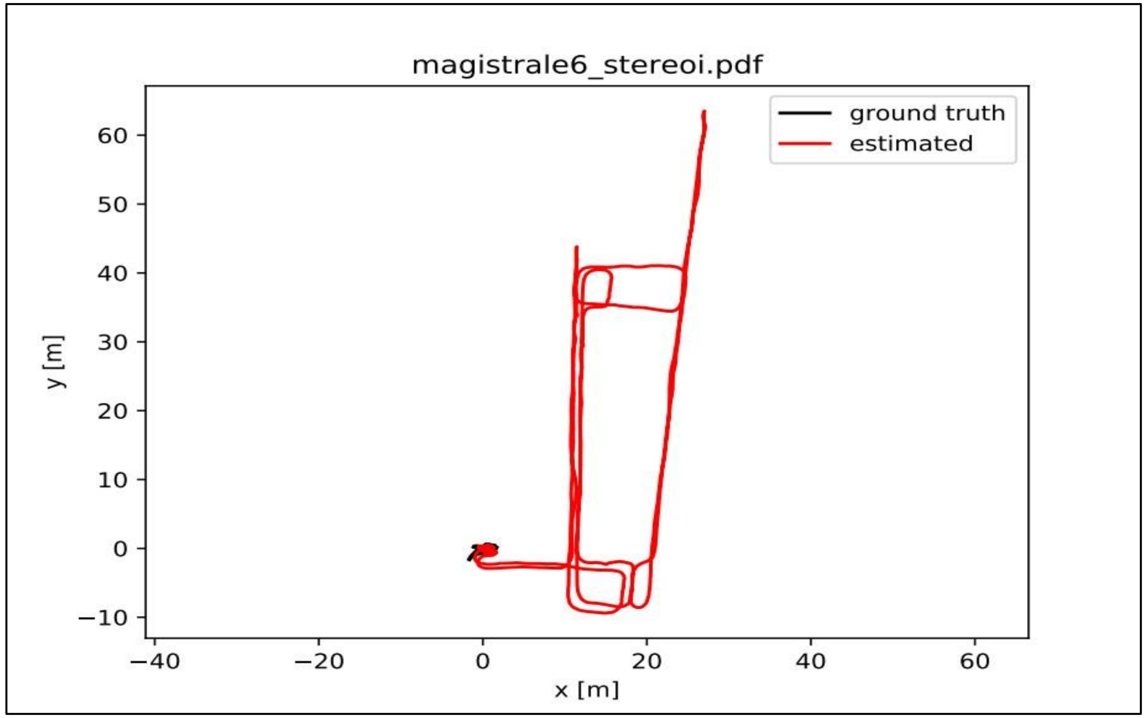


- Ground truth poses only provided for start and end segments
- No loop closure performed for either Monocular or Stereo with IMU
- Localization slightly better for Stereo (25m away vs 55m away), but scale is off for both
- Conclusion: Loop closure necessary to complete an accurate map & Stereo has only a slight advantage over Monocular in low loop closure cases

Results and Comparison

TUM-VI dataset : Magistrale Monocular Inertial VS Magistrale Stereo Inertial





- Ground truth provided for only start and end segments
- Stereo with IMU completed a loop closure, but Monocular with IMU did not
- Very similar paths between the two, but straightened out better and with correct scaling for stereo after loop closure
- Stereo in this case worked far better to pinpoint exact start and end location

Issues when working with NEU dataset

- Loss of features on sharp turns leading to failure in tracking local map resulting in the resetting of map
- Features concentrated on distant objects leading to distorted odometry data
- Features located on moving objects leading to distortions in map

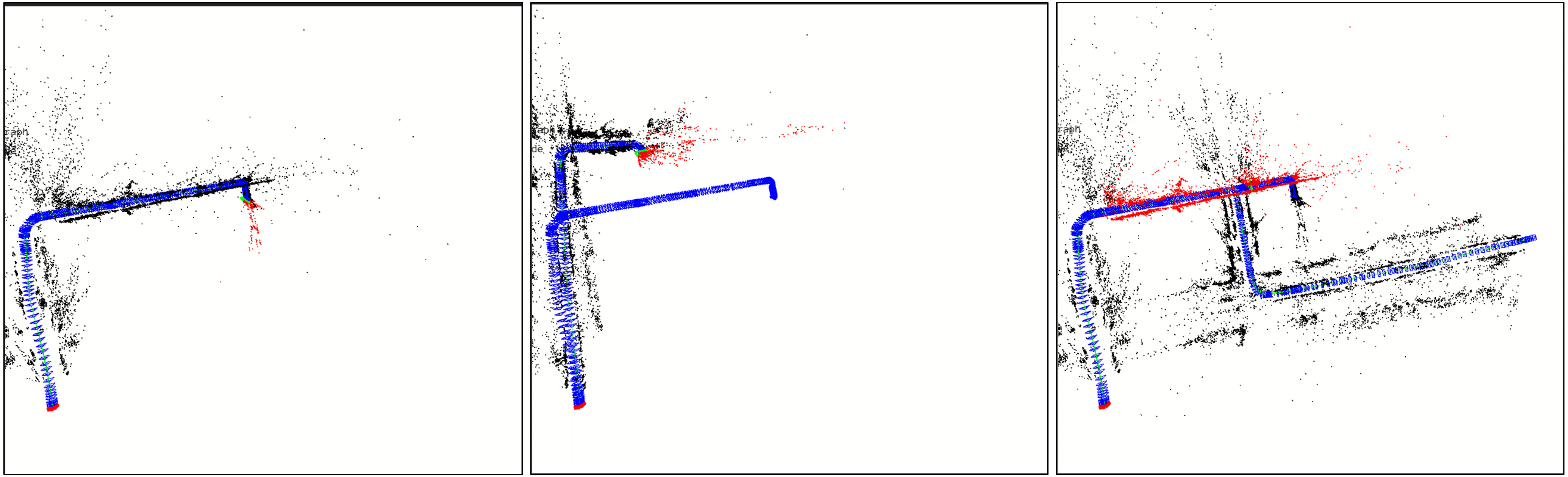
Corrective measures attempted

- Utilization of IR camera data which highly compensated for motion blur and provided accurate feature tracking even on sharp turns
- Examined working on various datasets to understand issues and how to compensate them
- Configuration changes

Remaining issues

- Getting the inertial data integrated with the monocular version that we have working right now
- Getting transformation parameters from IMU to camera just right

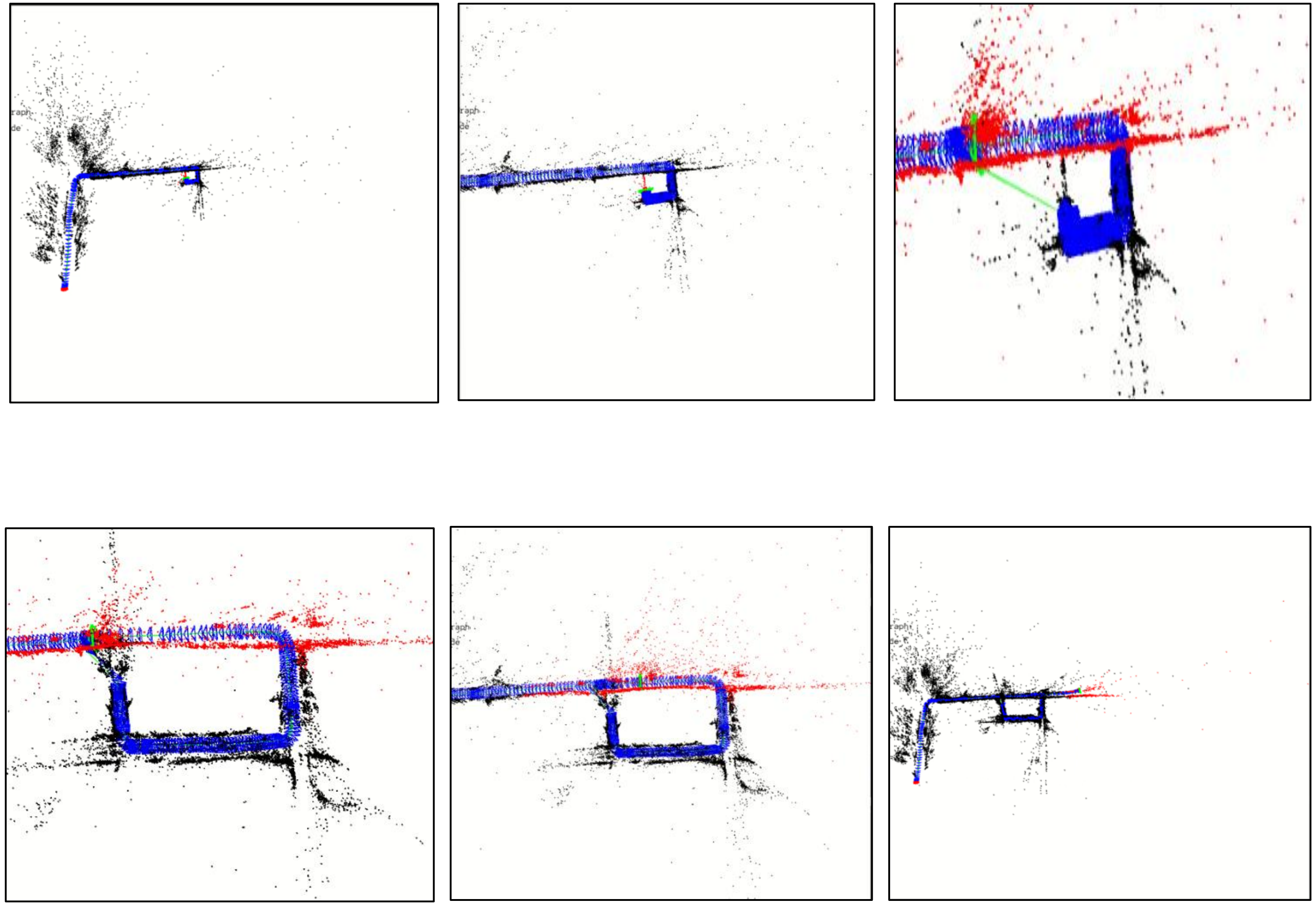
Failed Loop Closure (NUance Dataset)



Why it Failed:

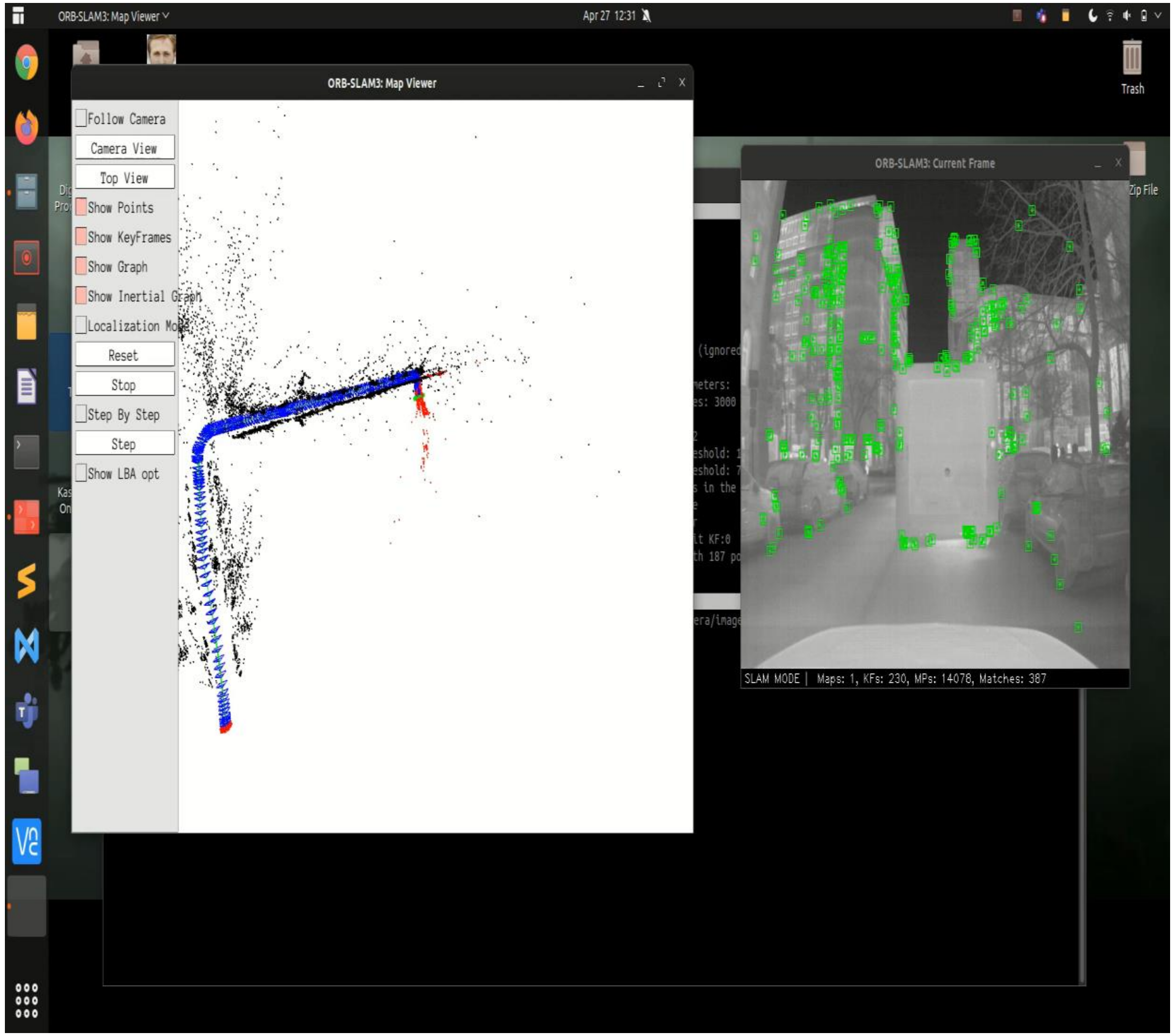
- ORB SLAM 3 stops tracking features after the second turn in the first image
- Error in tracking local map results in generation of a second map, shown in the second image
- Loop closure detected but map merged incorrectly due to tracking failure, shown in the third image

Successful Loop Closure(NUance Dataset)

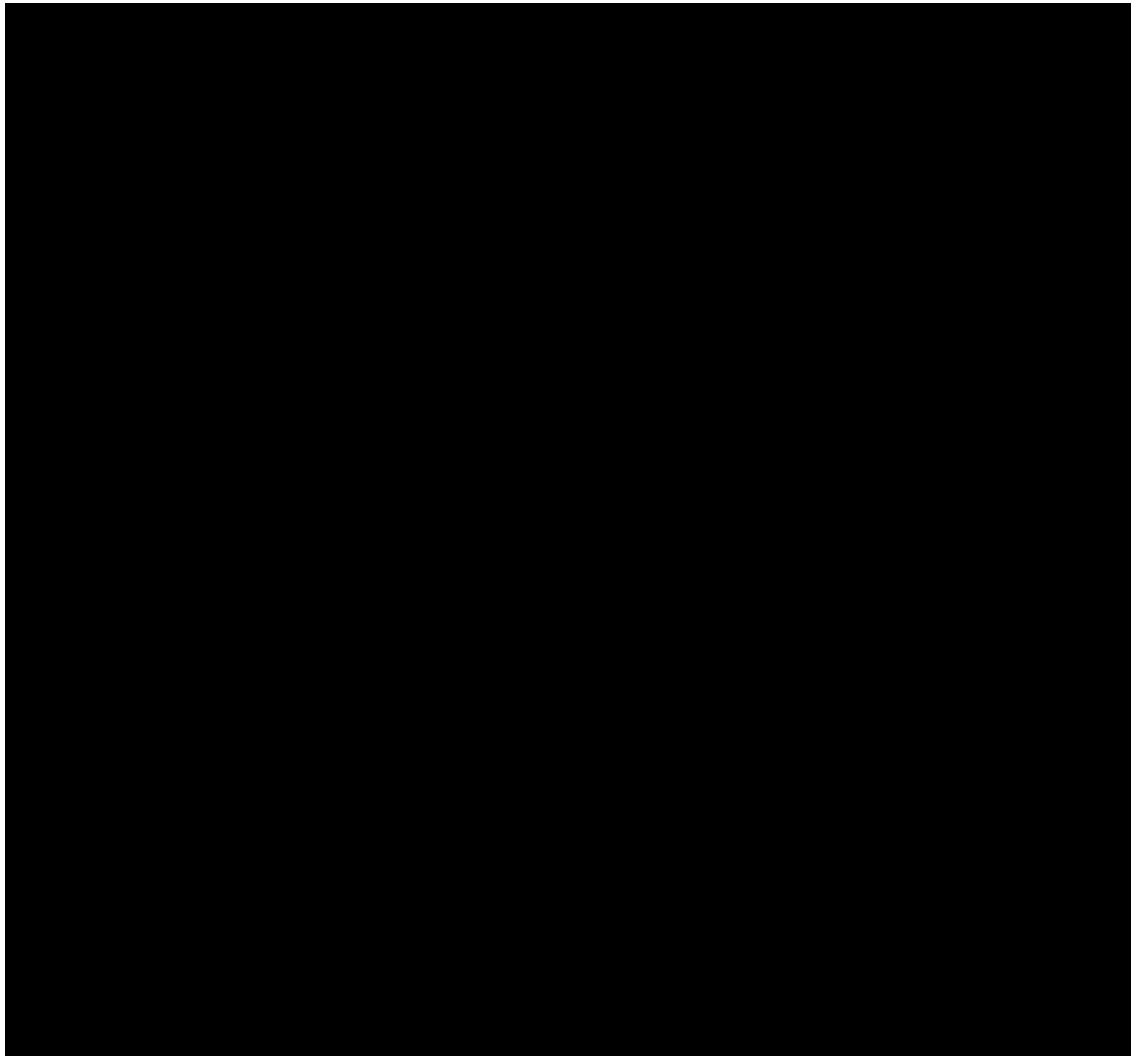


The first two images show that because of some errors in feature tracking, the mapping is not perfect and not according to scale.
After the last turn when the car is at a point it has previously mapped, you can observe a green line in the third image that corresponds to the two points that are found to be matching.
We can see in the later images how the keyframes are corrected taking the loop closure into account and adjusts the map accordingly.
It can be observed more clearly in the video clip on the next slide.

NUance Dataset



Failed loop closure



Successful loop closure

CONCLUSION

Conclusion

ORBSLAM3 was successfully implemented for performing VSLAM on EuRoC, TUM_VI, KITTI and NUance datasets.

EuRoC dataset:
- Stereo is necessary for good localization without IMU
- No significant difference between Monocular and Stereo with IMU, Stereo just slightly better.

TUM-VI dataset:
- For room1 and room2 we observe that there are many loop closures which means we got the good mapping with both Monocular and Stereo datasets.
- For outdoor mapping Stereo has a slight advantage over Monocular in low loop closure cases.
- In case of magistrale data set Stereo worked far better to pinpoint exact start and end location.

Kitti dataset:
- ORB SLAM 3 has worked the best on KITTI dataset so far. It could be because of the environment the data was collected in with good features to track and not so many dynamic objects nearby.
- While it does make mistakes while mapping initially, it compensates for it by doing an excellent job with the loop closures and bundle adjustment to generate an accurate map of the environment.

NUance Dataset:
- The NUance Dataset was by far the trickiest as it was our custom dataset. Various parameters needed adjusting and we solved issues one step at a time as seen in the previous slides.
- We were able to successfully run the algorithm on monocular version and create a map of the environment.
- Next steps are integrating the inertial measurement data to further improve the accuracy and precision

THANK YOU

GitLab Repository

https://gitlab.com/johnston.re/eece5554/-/tree/main/Final_Project