Northeastern

Robotics Sensing and Navigation EECE 5554

GROUP-7

VISUAL SLAM

Real-Time Appearance-Based Mapping

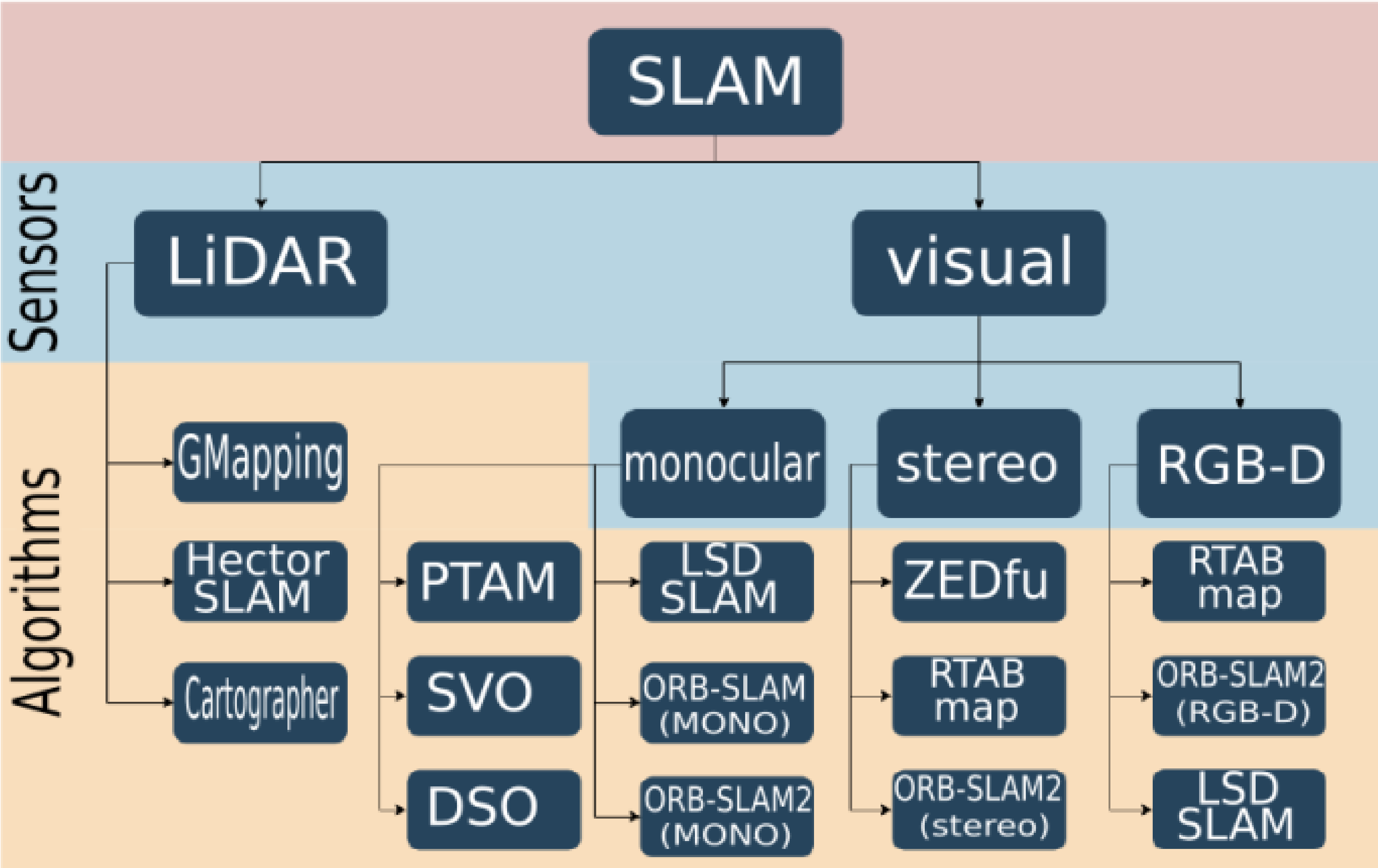
- Jose Thandapral
- Siddharth Maheshwari
- Smruti Suresh
- Kshama Dhaduti

SCOPE OF THE PROJECT

1. Implementing RTAB Mapping using ZED Mini camera.
2. Implementing April Tag detection.
3. Data collection.
4. Qualitative Analysis
5. RTAB Mapping using iPhone application.

VISUAL SLAM

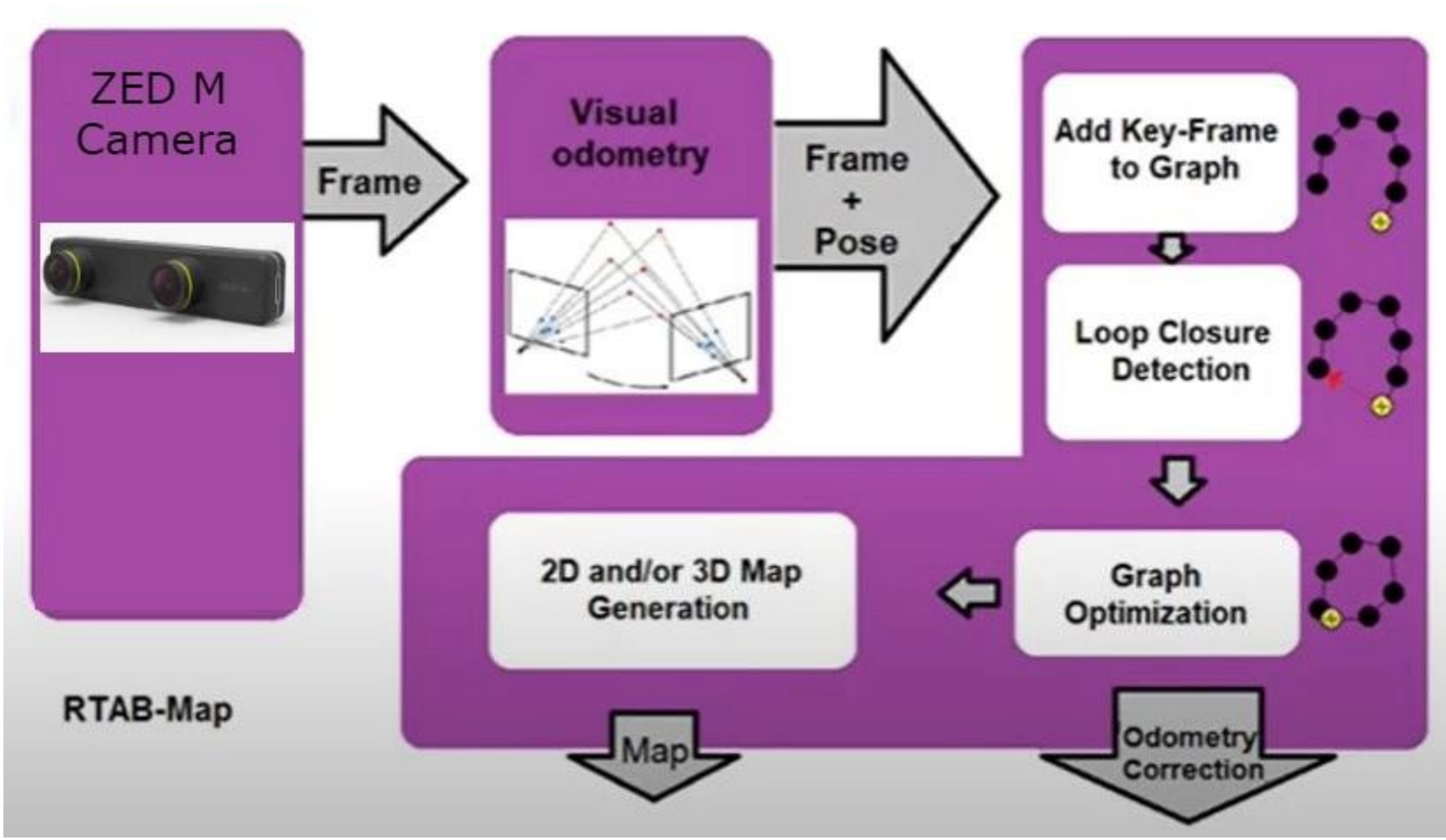
- **Visual SLAM (Simultaneous Localization And Mapping)** is a process of calculating the position and orientation of the camera with respect to its surrounding while simultaneously mapping the environment .
- It uses images from camera to map and localize, here we are using a stereo camera-ZED Mini.
- Visual SLAM tracks the points through each successive camera frame and triangulates to form 2D/3D map.
- Sufficient point detection assist with rapid environment mapping and better orientation of the sensor.
- All visual SLAM systems are constantly working to minimize reprojection error, or the difference between the projected and actual points, usually through an algorithmic solution called bundle adjustment.
- Visual SLAM systems need to operate in real-time, so often location data and mapping data undergo bundle adjustment separately, but simultaneously, to facilitate faster processing speeds before they’re ultimately merged.

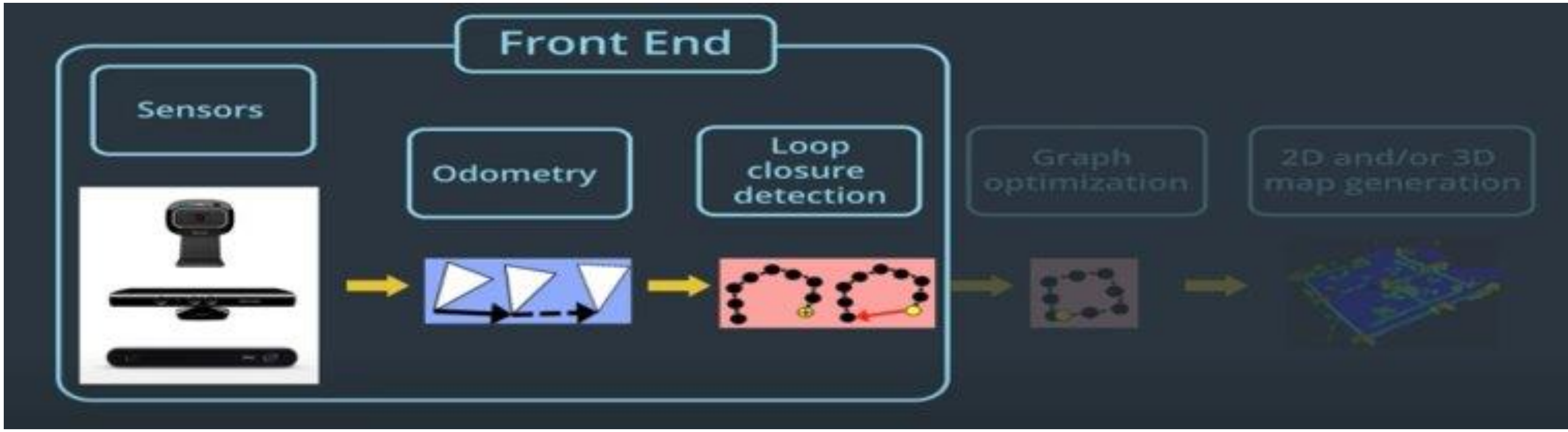

SLAM
Sensors
LiDAR
visual
Algorithms
monocular
stereo
RGB-D
GMapping
Hector SLAM
Cartographer
PTAM
SVO
DSO
LSD SLAM
ORB-SLAM (MONO)
ORB-SLAM2 (MONO)
ZEDfu
RTAB map
ORB-SLAM2 (stereo)
RTAB map
ORB-SLAM2 (RGB-D)
LSD SLAM

RTAB MAPPING

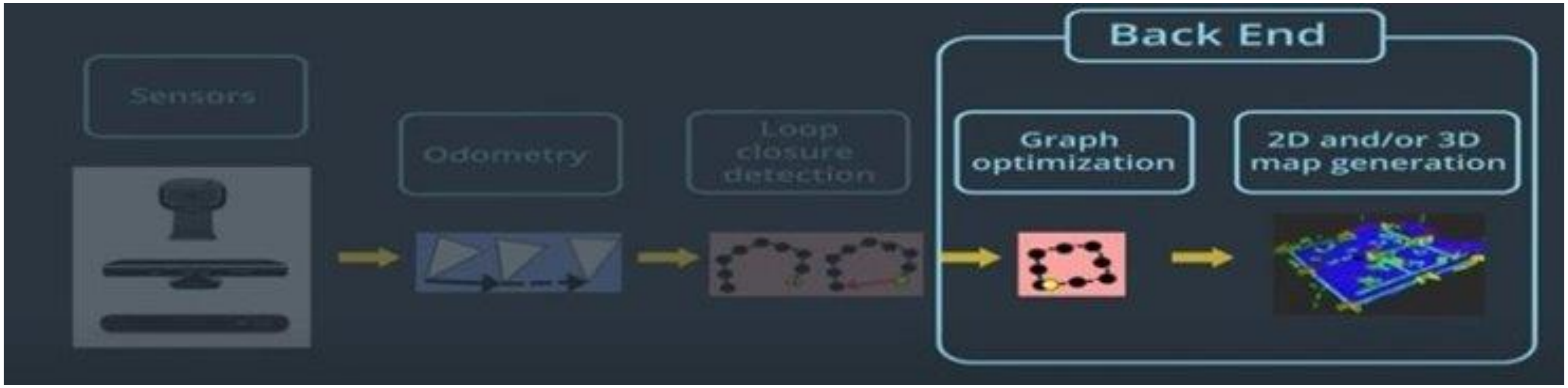
- **RTAB-Map** (Real-Time Appearance-Based Mapping) is a RGB-D, Stereo and Lidar Graph-Based SLAM approach based on an incremental appearance-based loop closure detector.
- The loop closure detector uses a bag-of-words approach to determinate how likely a new image comes from a previous location or a new location.
- When a loop closure hypothesis is accepted, a new constraint is added to the map's graph, then a graph optimizer minimizes the errors in the map.
- There are two types of loop closure detections: local and global.

System Architecture





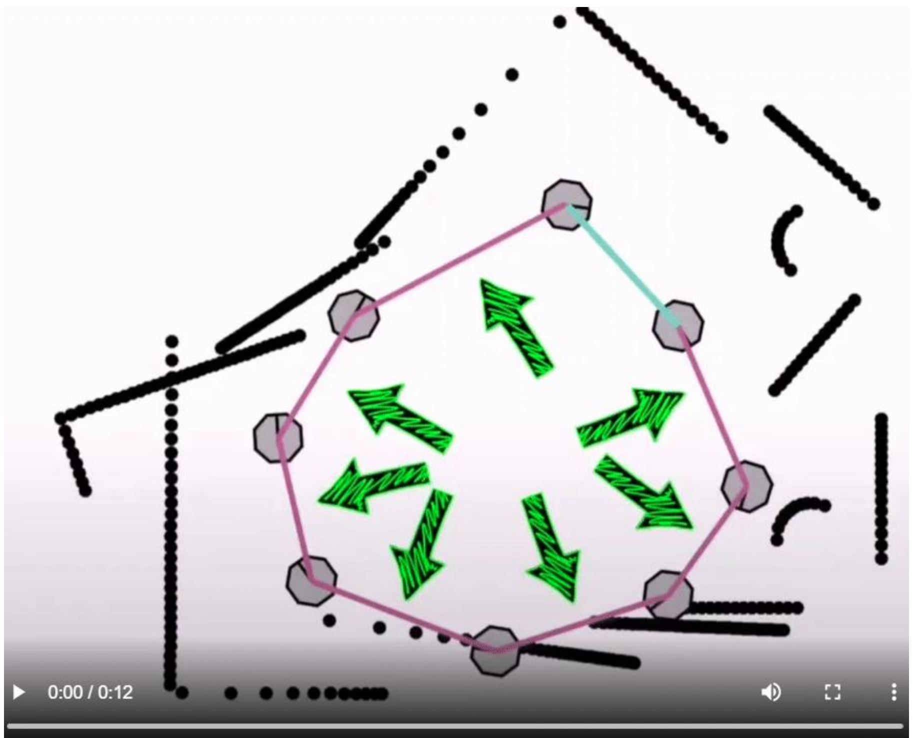
- Focuses on the sensor data used to determine the constraints used in feature optimization methods.
- Odometry constraints and loop closure constraints are considered.
- For metric GraphSLAM, RTAB-Map requires an RGB-D camera or a stereo camera to compute the geometric constraint between the images of loop closure.
- The front end also involves graph management, which includes node creation and loop closure detection using bag-of-words

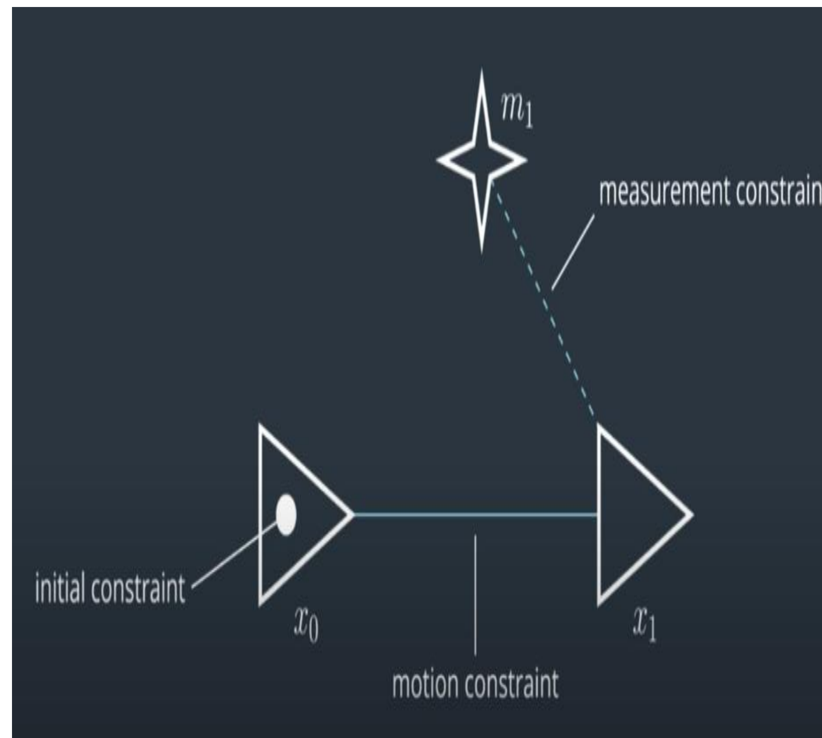


- Includes the graph optimization and an assembly of an occupancy grid from the data of the graph.
- When a loop closure hypothesis is accepted, a new constraint is added to the map's graph, then a graph optimizer minimizes the errors in the map.
- All of these optimizations use node poses and link transformations as constraints. When a loop closure is detected, errors introduced by the odometry can be propagated to all links, correcting the map.
- Only odometry constraints and loop closure constraints are optimized.

Graph SLAM

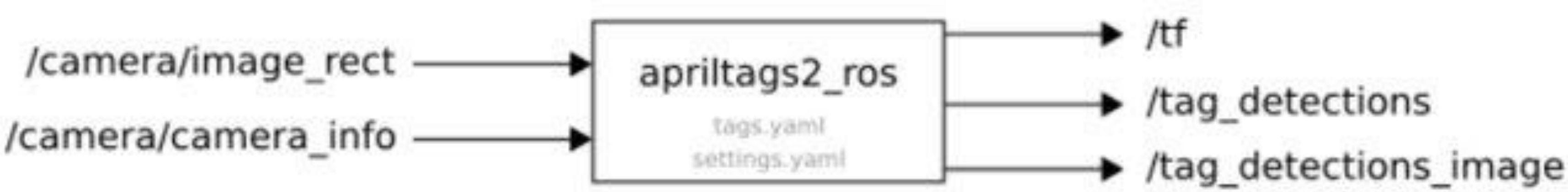
- The algorithm recovers the entire path and map instead of just the recent pose and map. Creates constrains between nodes.





APRIL TAGS

- We are implementing April tags to detect landmarks.
- Detector for continuous image (i.e. video) stream.



tag_family (string, default: tag36h11)

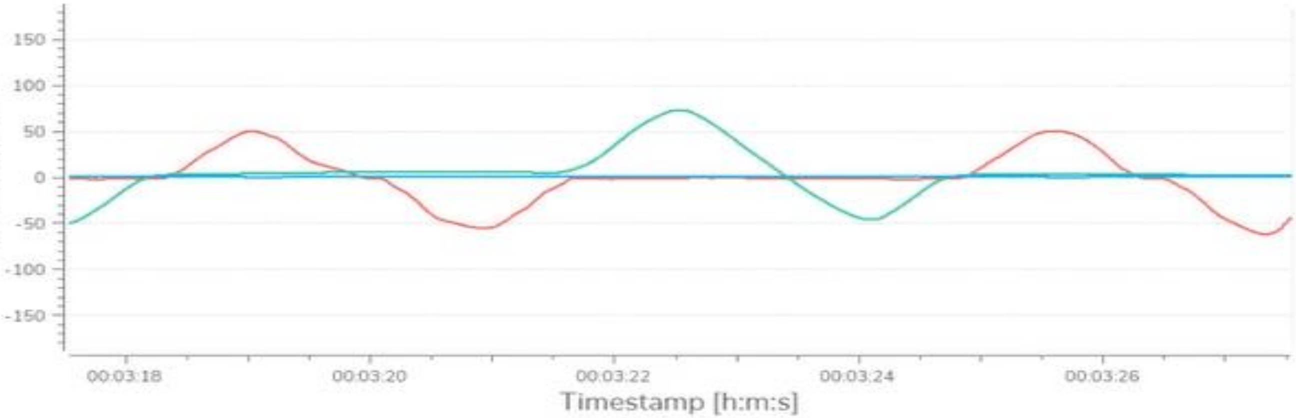
- The advantage of this method is that it does not require any complicated equipment.
- All you need is:
  - Printed April tags and wall mounting them
  - Small embedded computer(Raspberry Pi or similar) with camera (Global shutter preferred)

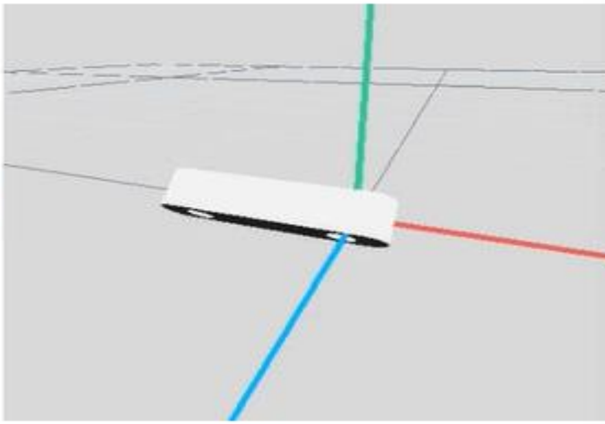
SOFTWARE TOOLS

- ROS noetic
- PYTHON3
- RVIZ
- RTAB Map
- C++
- OpenCV
- apriltag_ros wrapper
- rtabmap_ros
- zed_ros_wrapper

HARDWARE

- ZED-Mini camera



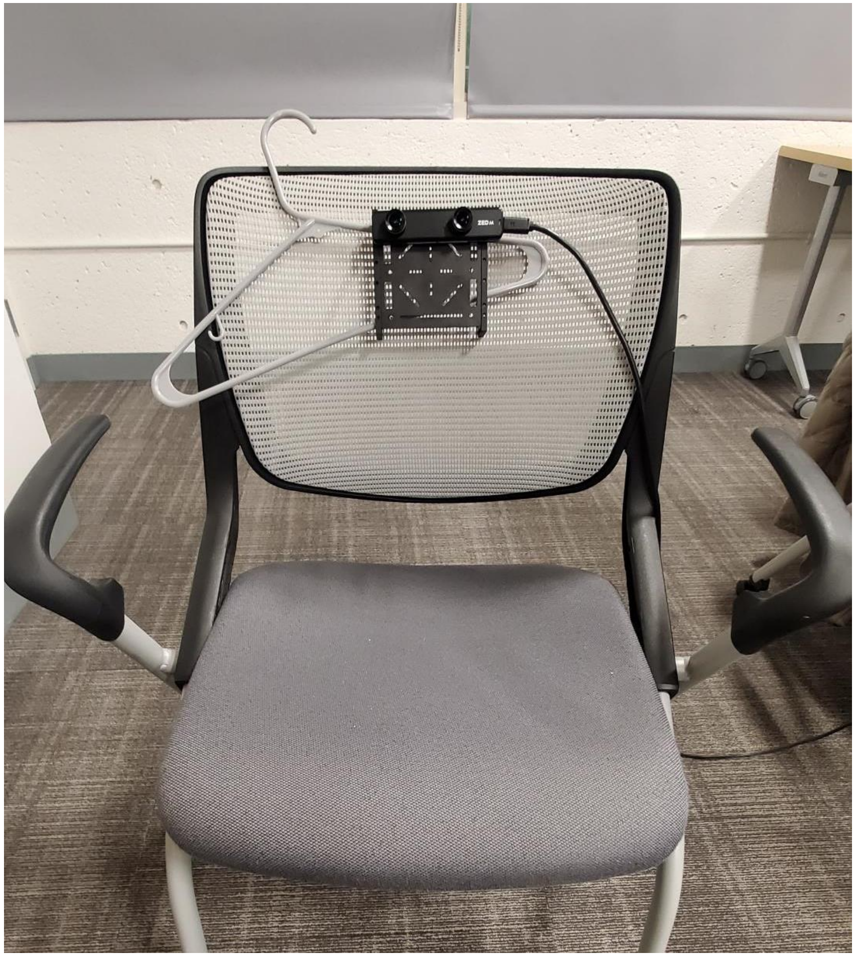


- The ZED family of depth cameras is a multi-sensor platform. The cameras have built-in sensors to add position and motion-assisted capabilities to your app, like accelerometer and gyroscope sensors.
- The sensors can be used to detect camera movements, compute the camera orientation according to the north magnetic pole, detect relative altitude variations, and much more.

ZED MINI SPECS

1. In-built IMU (Accelerometer, Gyroscope)
2. Depth range - 0.1 - 15 m (0.32 to 50 ft)
3. Image sensor - Dual 4M pixels sensors with large 2-micron pixels
4. Connectivity - USB 3.0 Type-C port
5. Power - Power via USB 5V / 380mA
6. Compatible OS - Windows, Ubuntu, Debian, Jetson
7. Third-party Integrations – ROS, Python, C++, C#,MATLAB, OpenCV, pcl , Unity, TensorFlow
8. SDK System Requirements - Dual-core 2.3GHz or faster processor|4 GB RAM or more| Nvidia GPU with compute capability > 3.0

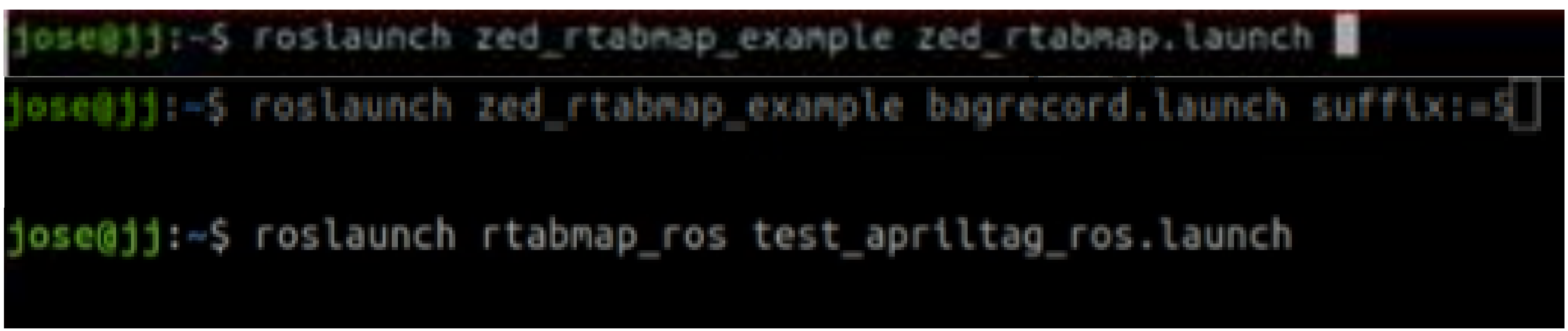
SET UP



- Camera fixed to a chair for steady Z axis stability.
- Ensured ease of movement.
- Observed better locali zation and loop closures when fixed to a chair.

IMPLEMENTATION

- Launch file for rtabmap (zed-ros-examples).
- Launch file for recording rosbag.
- Launch file for April tag detection.
- SLAM mode
- Switch to localization



ISSUES FACED IN IMPLEMENTATION

Time sync error

Transformation error (from base link to odom-frame)

Compressed image transport error

cv::bridge error

```
[ WARN] [1650561114.538549157]: Could not get transform from odom to base_link after
0.200000 seconds (for stamp=1650561113.822612)! Error="Lookup would require extrapolation
-0.684948310s into the future. Requested time 1650561113.822611809 but the latest data is
at time 1650561113.137663603, when looking up transform from frame [base_link] to frame [odom].
canTransform returned after 0.20186 timeout was 0.2.". [ERROR] [1650561114.558475290]: Compressed Depth
Image Transport - Compression requires single-channel 32bit-floating point or 16bit raw depth images (input format is: bgra8).
```

Observations:

- Insufficient features in the environments (for example, white walls).
- Dedicated depth sensors (in RGB-D) or depth from LiDAR scans give better results for 3D reconstruction than the depth obtained from stereo depth sensing.

PARAMETER TUNING

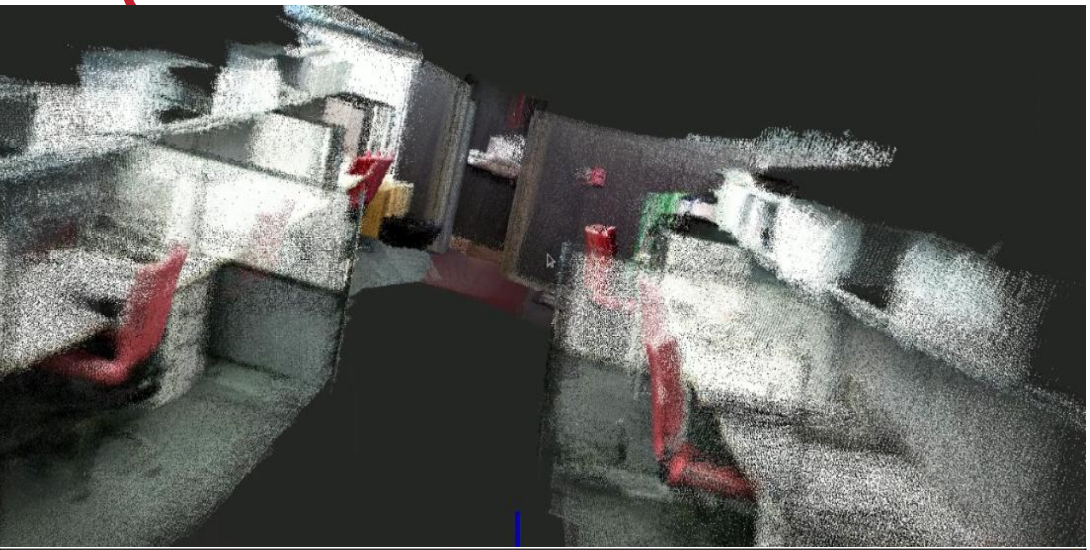
- RTAB-Map's performance can be improved by tuning various parameters.
- Variety of strategies can be used for Odometry, Optimizer, Feature Detection, Feature registration.
- Odometry Strategy: Frame to Frame
- Feature Detection Strategy: GFTT/ORB
- Optimizer: GTSAM

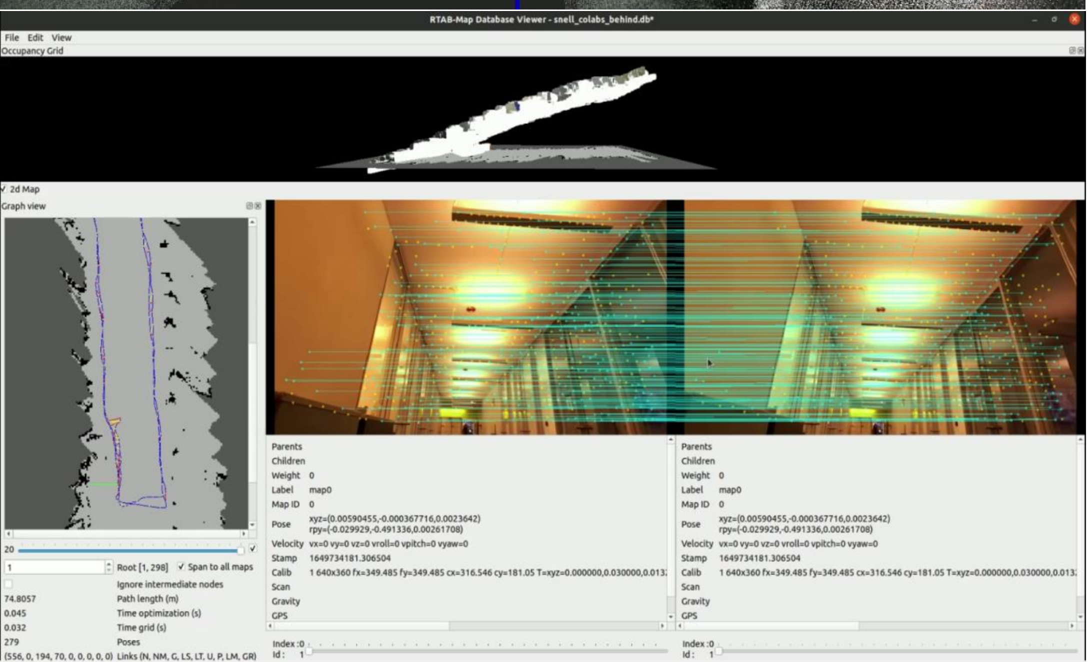
PARAMETER TUNING: COMPARISON

```yaml
Odom:
    Strategy:                 1
    ResetCountdown:           1
    FilteringStrategy:        1

# OdomF2M:
#     MaxSize:                 2000

Vis:
    CorType:                  0
    MaxFeatures:              900
    BundleAdjustment:         1
    MinInliers:               30

GFTT:
    K:                        0.04
    MinDistance:              7
    QualityLevel:             0.0001
    UseHarrisDetector:        false

RGBD:
    Enabled:                        true
    CreateOccupancyGrid:            true
    LoopClosureReextractFeatures:   true
    AngularSpeedUpdate:             0.00
    AngularUpdate:                  0.05
    GoalReachedRadius:              0.5
    GoalsSavedInUserData:           false
    LinearSpeedUpdate:              0
    LinearUpdate:                   0.05
    LocalBundleOnLoopClosure:       false
    LocalImmunizationRatio:         0.25
    LocalRadius:                    10
    LoopClosureIdentityGuess:       false
    LoopCovLimited:                 false
    MarkerDetection:                false
    MaxLocalRetrieved:              0
    MaxLoopClosureDistance:         0.0
    MaxOdomCacheSize:               10
    NeighborLinkRefining:           false
    NewMapOdomChangeDistance:       0
    OptimizeFromGraphEnd:           true
    OptimizeMaxError:               1
    PlanAngularVelocity:            0
    PlanLinearVelocity:             0
    PlanStuckIterations:            0
    ProximityAngle:                 45
    ProximityBySpace:               1
    ProximityByTime:                false
    ProximityGlobalScanMap:         false
    ProximityMaxGraphDepth:         50
```

```cpp
// Odometry
RTABMAP_PARAM(Odom, Strategy,                    int, 0,
RTABMAP_PARAM(Odom, ResetCountdown,              int, 0,
RTABMAP_PARAM(Odom, Holonomic,                   bool, true,
RTABMAP_PARAM(Odom, FillInfoData,                bool, true,
RTABMAP_PARAM(Odom, ImageBufferSize,         unsigned int, 1,
RTABMAP_PARAM(Odom, FilteringStrategy,           int, 0,
// RGB-D SLAM
RTABMAP_PARAM(RGBD, Enabled,                     bool, true,  "Ac
RTABMAP_PARAM(RGBD, LinearUpdate,                float, 0.1,  "Mi
RTABMAP_PARAM(RGBD, AngularUpdate,               float, 0.1,  "Mi
RTABMAP_PARAM(RGBD, LinearSpeedUpdate,           float, 0.0,  "Ma
RTABMAP_PARAM(RGBD, AngularSpeedUpdate,          float, 0.0,  "Ma
RTABMAP_PARAM(RGBD, NewMapOdomChangeDistance,    float, 0,    "A
RTABMAP_PARAM(RGBD, OptimizeFromGraphEnd,        bool, false, "Op
RTABMAP_PARAM(RGBD, OptimizeMaxError,            float, 3.0,   uF
RTABMAP_PARAM(RGBD, MaxLoopClosureDistance,      float, 0.0,   "F
RTABMAP_PARAM(RGBD, StartAtOrigin,               bool, false, uFo
RTABMAP_PARAM(RGBD, GoalReachedRadius,           float, 0.5,  "Go
RTABMAP_PARAM(RGBD, PlanStuckIterations,         int, 0,      "Ma
RTABMAP_PARAM(RGBD, PlanLinearVelocity,          float, 0,    "Li
RTABMAP_PARAM(RGBD, PlanAngularVelocity,         float, 0,    "An
RTABMAP_PARAM(RGBD, GoalsSavedInUserData,        bool, false, "Wh
RTABMAP_PARAM(RGBD, MaxLocalRetrieved,           unsigned int, 2,
RTABMAP_PARAM(RGBD, LocalRadius,                 float, 10,   "Lo
RTABMAP_PARAM(RGBD, LocalImmunizationRatio,      float, 0.25, "Ra
RTABMAP_PARAM(RGBD, ScanMatchingIdsSavedInLinks, bool, true,
RTABMAP_PARAM(RGBD, NeighborLinkRefining,          bool, false,
RTABMAP_PARAM(RGBD, LoopClosureIdentityGuess,      bool, false,
RTABMAP_PARAM(RGBD, LoopClosureReextractFeatures,  bool, false,
RTABMAP_PARAM(RGBD, LocalBundleOnLoopClosure,      bool, false,
RTABMAP_PARAM(RGBD, CreateOccupancyGrid,           bool, false,
RTABMAP_PARAM(RGBD, MarkerDetection,               bool, false,
RTABMAP_PARAM(RGBD, LoopCovLimited,                bool, false,
RTABMAP_PARAM(RGBD, MaxOdomCacheSize,              int,  10,
```

QUALITATIVE ANALYSIS OF RESULTS



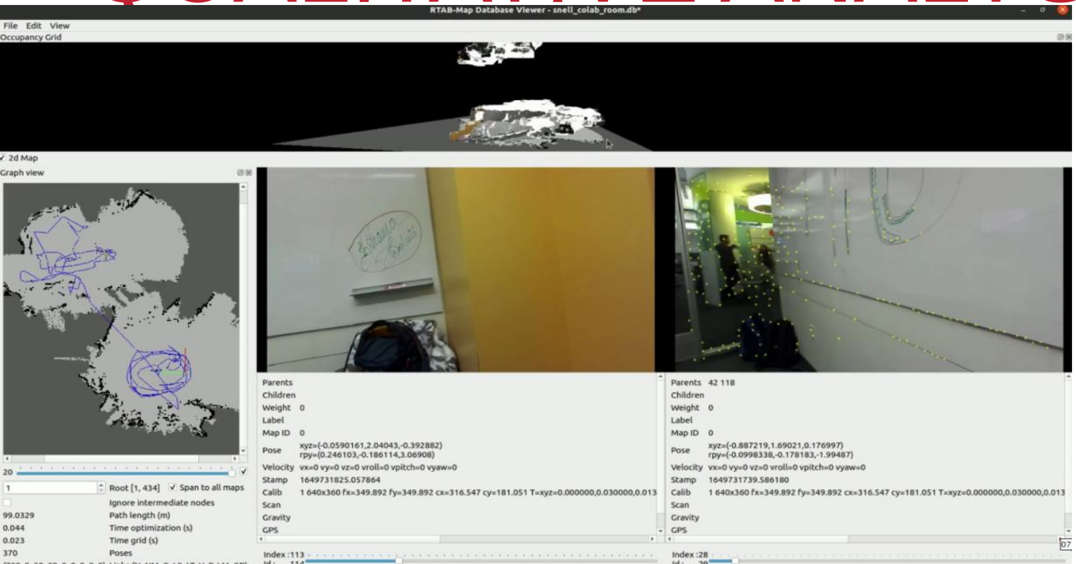


2loops_workspace_3IT.db

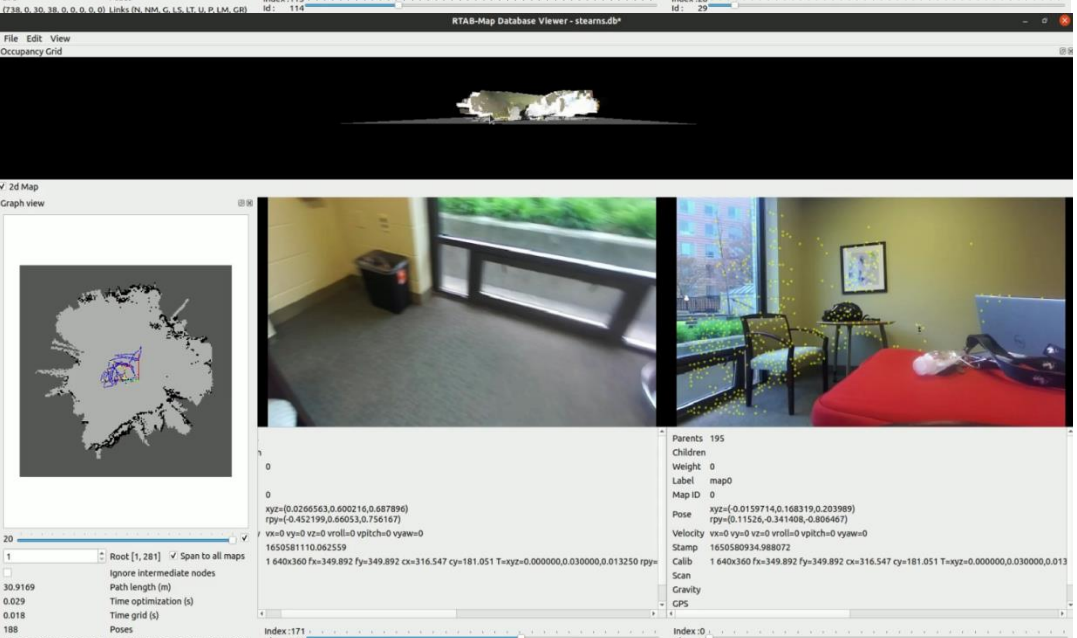
Hardware: Xtion Pro Live Camera with dedicated depth sensing

Major z-drift in initial collected databases
-But obtained loop closure after a few loops (refer graph view in rsn_vid1 video)

QUALITATIVE ANALYSIS OF RESULTS

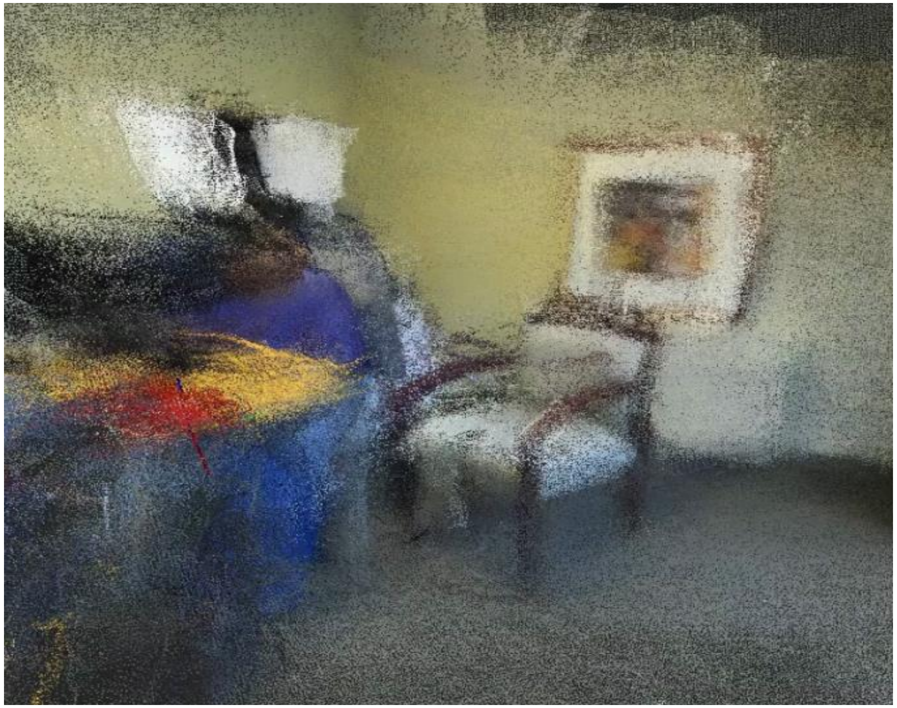


Transparent surface during mapping
-Produces “pseudo” features
-resulting in new map(refer graph view)



Fixed z-drift
-setting Odom:FilteringStrategy parameter to 1(Kalman Filter)

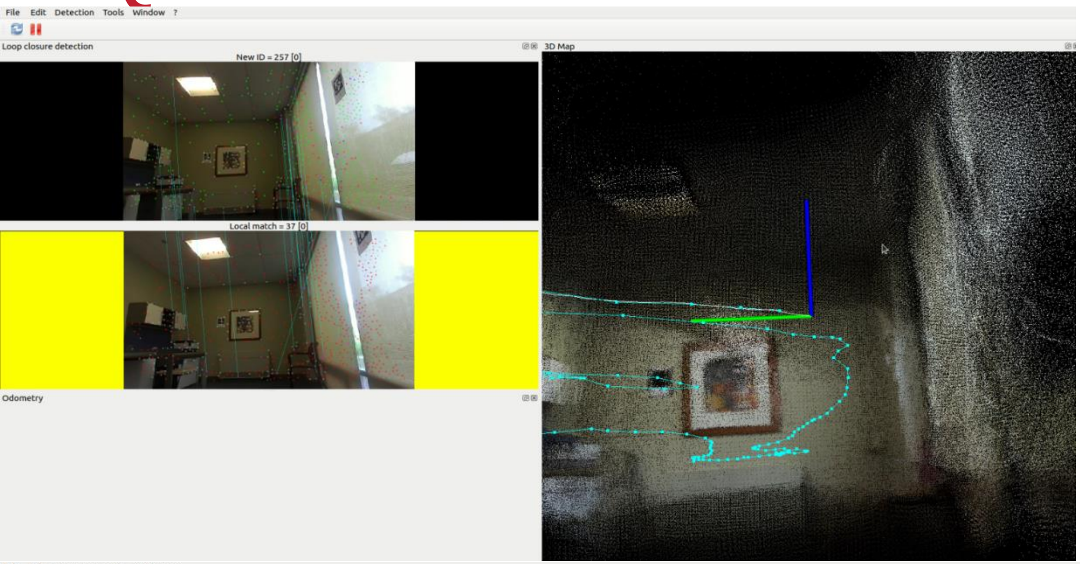
QUALITATIVE ANALYSIS OF RESULTS



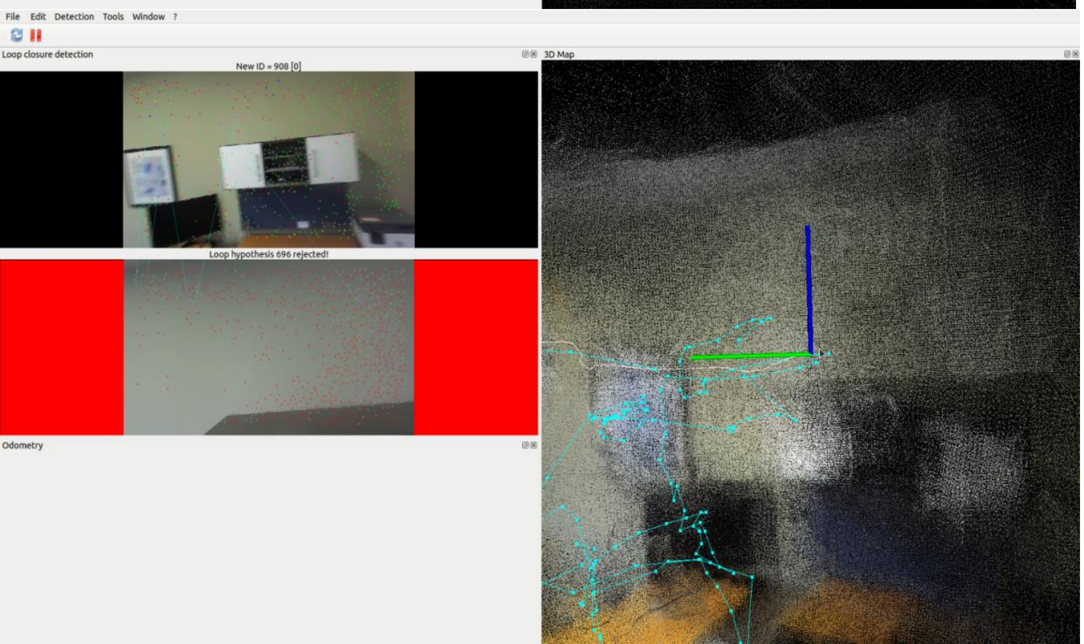
Parameter tuning resulting in better localization and 3D reconstruction
-But still environment had glass surfaces(refer rsn_vid1 video)

Smaller environment with contrasting features
-results in better frame matching and 3D reconstruction

QUALITATIVE ANALYSIS OF RESULTS

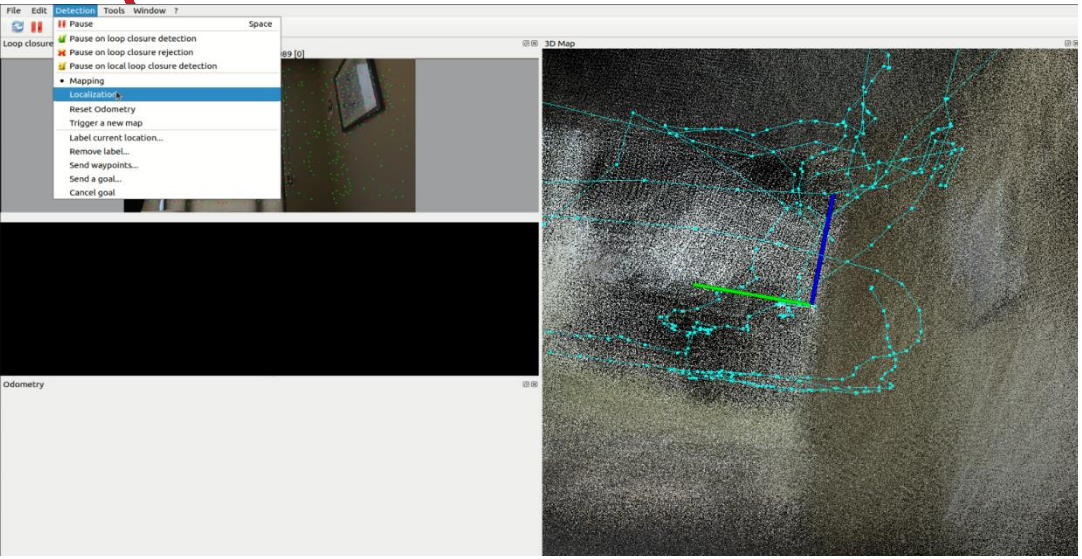


Rosbag issue fixed & final mapping
-First 2 loops with chair setup for constant Z level

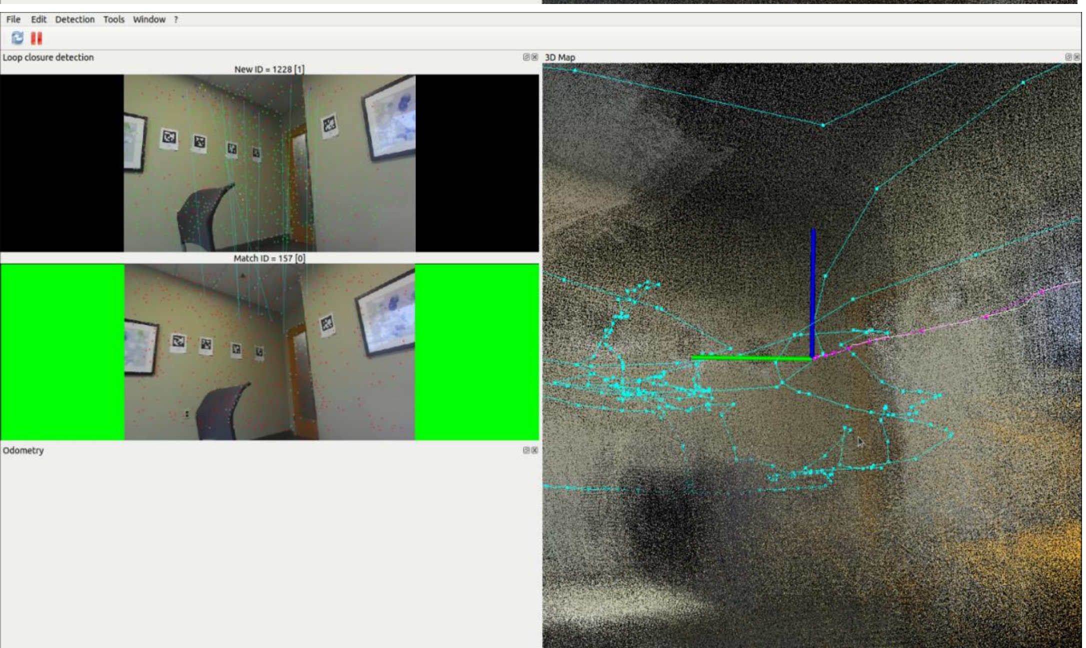


Handheld mapping to get different views of objects in the room

QUALITATIVE ANALYSIS OF RESULTS

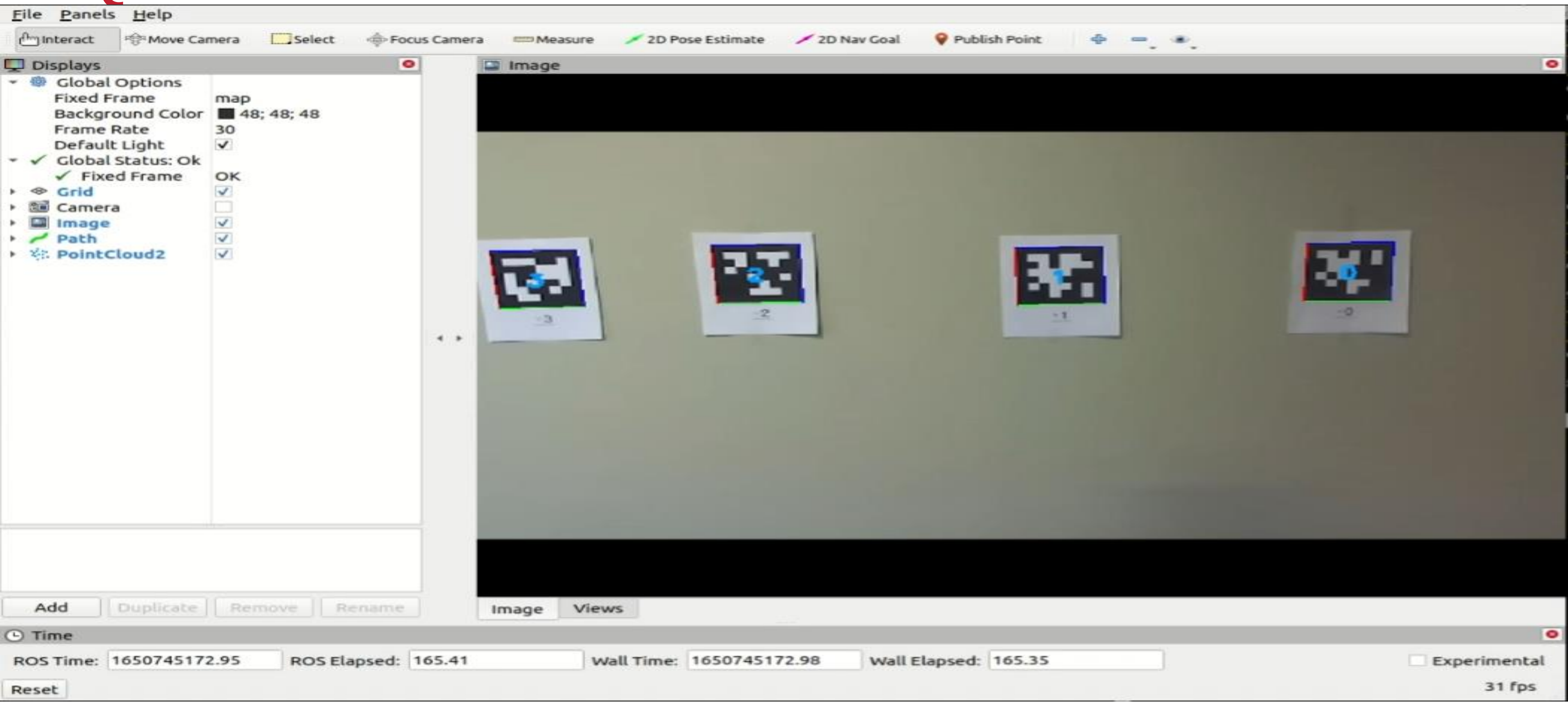


Switching to pure localization



Pink trajectory shows the localization after the environment is sufficiently mapped

QUALITATIVE ANALYSIS OF RESULTS



April tags detected with bounding boxes
and central TagID# display

iPhone RTAB MAP

CONCLUSION AND FUTURE SCOPE

- RTAB-Map was successfully implemented for performing SLAM followed by standalone localization within a prebuilt map.
- Loop closures were frequently observed which is a qualitative indicator of SLAM performance.
- Glass windows and door were a problem we came across, RTAB couldn’t do 3D reconstruction perfectly since the features on the other side of the glass were also detected and registered.
- Illumination effects affected the performance of RTAB-Map. Excessive and insufficient illumination contributed to poor 3D reconstruction.
- **Future scope:** Combine RTAB-SLAM with April tag detection and implement path planning on autonomous platforms.

REFERENCES

1. https://www.automate.org/blogs/what-is-visual-slam-technology-and-what-is-it-used-for#:~:text=How%20Does%20Visual%20SLAM%20Technology,information%20to%20approximate%20camera%20pose.
2. https://www.mathworks.com/help/vision/ug/visual-simultaneous-localization-and-mapping-slam-overview.html
3. http://introlab.github.io/rtabmap/#:~:text=Overview,location%20or%20a%20new%20location
4. https://www.researchgate.net/publication/264838423
5. https://www.stereolabs.com/zed-mini/
6. http://wiki.ros.org/apriltag_ros

TO

Sushant Govindraj Xavier Hubbard