RGB-D SLAM*

An Overview

Syed Muhammad Abbas, Abubakr Muhammad

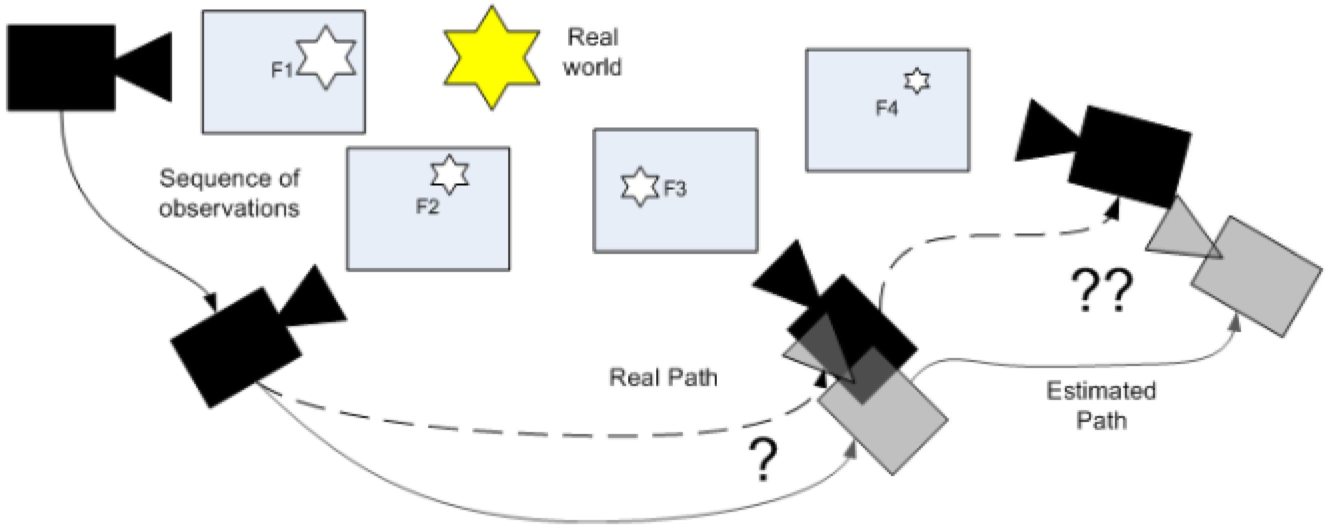
Laboratory for Cyber physical networks & systems
Department of Electrical Engineering,
School of Science and Engineering, LUMS.

Slides based on material from:
MS Thesis. Abbas. LUMS. 2012
MS Thesis. Hogman. KTH. 2011.

Introduction

SLAM: To navigate in an unknown environment, a mobile robot needs to build a map of the environment and localize itself in the map at the same time.

Necessity: Generally robot's position problem is solved by a GPS which provides a good accuracy for the robot. However, in places where the GPS data is not available, or not reliable enough, we need some other reliable method to estimate the robot's position precisely.



The estimations generally drift with respect to the real trajectory, and the uncertainty grows over time.

http://www.csc.kth.se/~aydemir/thesis_virgile_hogman.pdf

RGB-D Limitations & adaptations

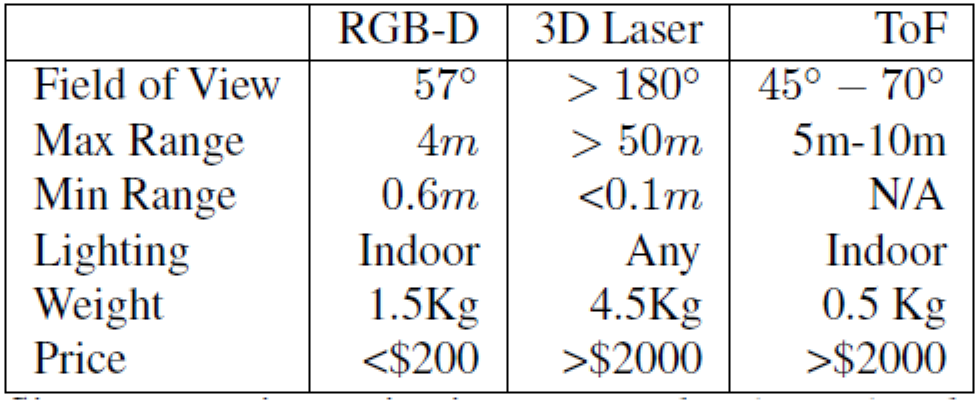
RGB-D cameras (such as Kinect) operational limitations:

•Limited field of view preventing an agile operation.

•Short range, not providing the scale for typical outdoor applications.

•Infrared saturation in direct sunlight.

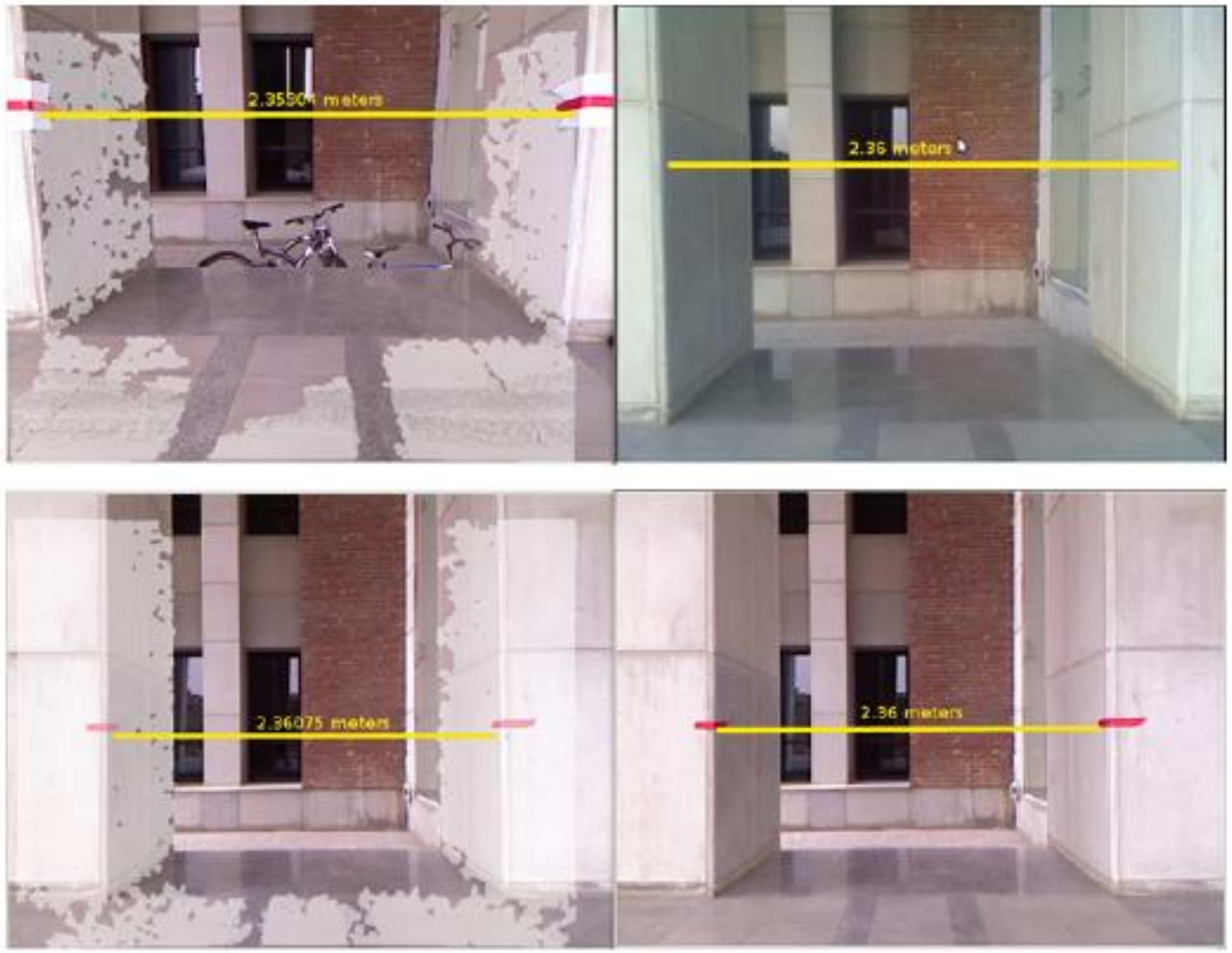
Comparisons chart:



	RGB-D	3D Laser	ToF
Field of View	$57°$	$> 180°$	$45° - 70°$
Max Range	$4m$	$> 50m$	5m-10m
Min Range	$0.6m$	$<0.1m$	N/A
Lighting	Indoor	Any	Indoor
Weight	1.5Kg	4.5Kg	0.5 Kg
Price	<$200	>$2000	>$2000

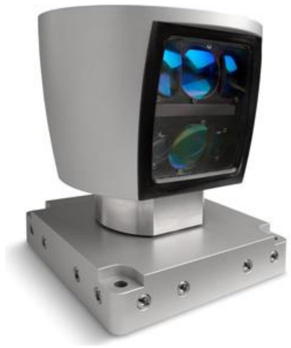
Microsoft Kinect

Kinect Depth and RGB data



Sensors and techniques

Laser-based SLAM (2D/ 3D Laser scanners)



Visual Monocular SLAM (Monocular Camera)

RGB-D SLAM (Kinect/Asus)

Stereo SLAM (Stereo Camera pair)

Why RGBD SLAM?

Currently, most of robotic mapping is performed using sensors that offers only a 2D cross section of the environment around them.

Acquiring high quality 3D data was either very expensive or had hard constraints on the robot movements.

Research has mainly focused on laser scanners to solve the SLAM problem and developed some methods making use of stereo and mono cameras.

Kinect Camera developed by Prime Sense and Microsoft has considerably changed the situation, providing a 3D camera at a very affordable price.

VSLAM (Visual SLAM)

Goal: Build a 3D map from RGB and depth information provided by a camera, considering a 6 Degrees-of-Freedom (DOF) motion system.

Idea Summary

Estimating the poses of the camera from its data stream (video and depth), in order to reconstruct the entire environment while the camera is moving. As the sensory noise leads to deviations in the estimations of each camera poses with respect to the real motion, the goal is to build a 3D map which is close, as much as possible, to the real environment.

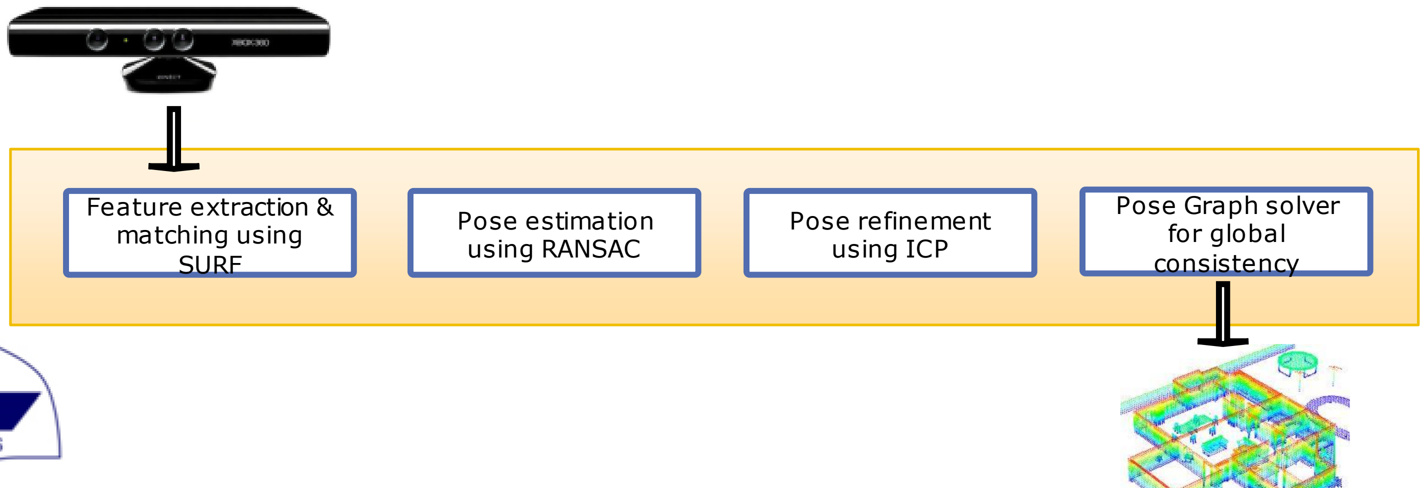
RGB-D SLAM

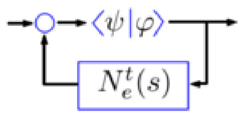
In the past, to solve the inherent problem of drift and provide a reliable estimation of the camera poses, most of the projects have used techniques such as Extended Kalman Filtering (EKF) or particle filters.

Approach described here is the RGB-D SLAM algorithm close to the technique developed by Univ of Freiburg.

Engelhard, Endres, Hess, Sturm, Burgard, "Real-time 3D visual SLAM with a hand-held RGB-D camera", RGB-D Workshop, European Robotics Forum, Vasteras, Sweden, 2011.

It is basically a consortium of different standard algorithms resulting in a very effective solution for 3D mapping using RGB-D camera.


Feature extraction & matching using SURF
Pose estimation using RANSAC
Pose refinement using ICP
Pose Graph solver for global consistency

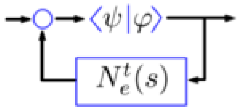


Features extraction techniques

Harris Corner : A corner detector, by Harris and Stephens
SIFT: Scalar Invariant Feature Transform, by David Lowe
SURF: Speeded Up Robust Feature
NARF: Normal Aligned Radial Feature
BRIEF: Binary Robust Independent Elementary Feature

Two aspects concerning a feature: the detection of a ***keypoint***, which identifies an area of interest, and its ***descriptor***, which characterizes its region.

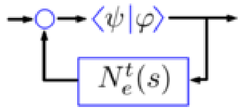
Feature	Detector	Descriptor
Harris Corner	Yes	No
SIFT	Yes	Yes
SURF	Yes	Yes
NARF	Yes	Yes
BRIEF	No	Yes



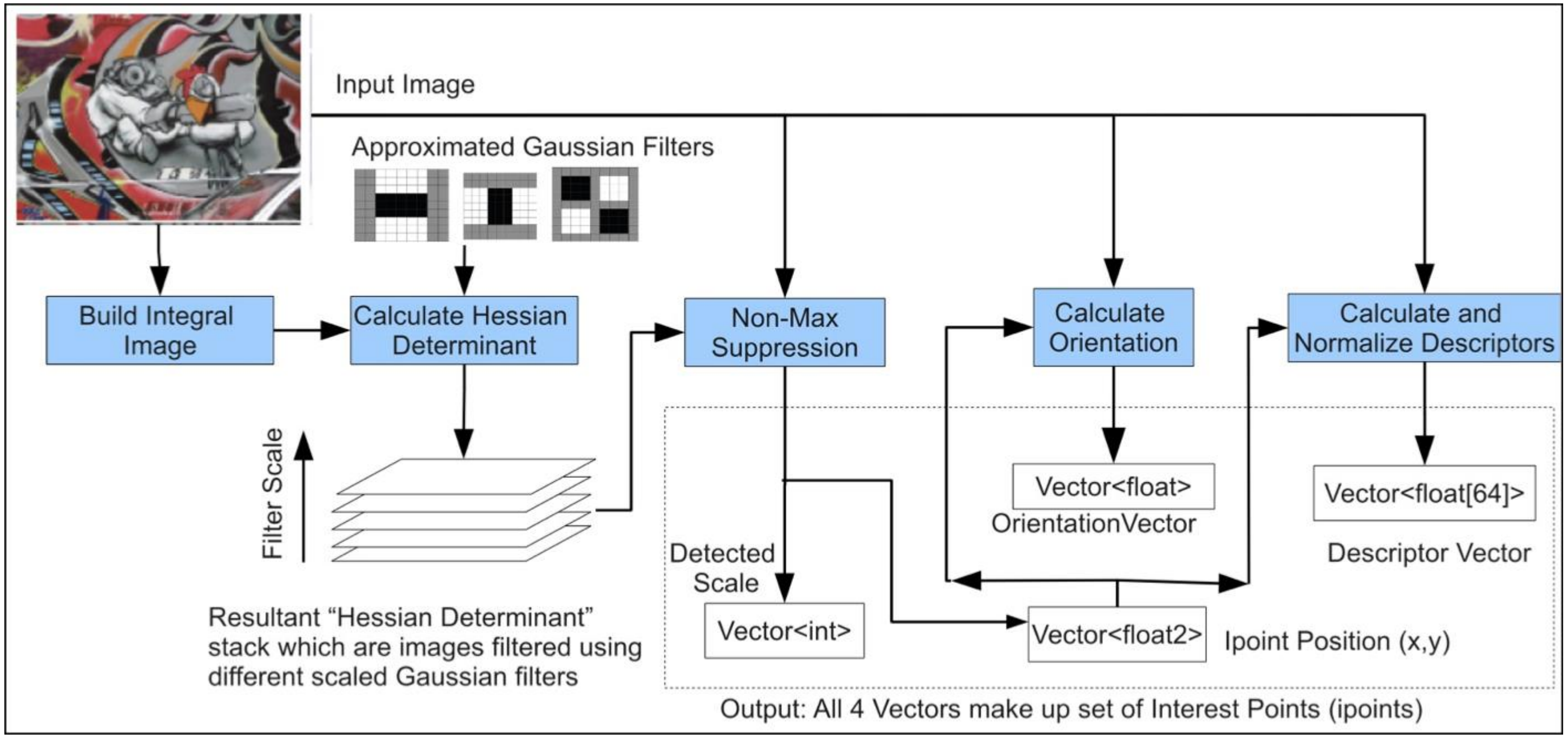
SURF

- **Speeded Up Robust Feature**
- **Used for feature extraction and matching**

STEP 1:
Interest points are selected at distinctive locations in the image like blob, corners and T-junctions.
STEP 2:
The neighborhood of every interest point is represented by a feature vector.
STEP 3:
Descriptor Vectors are matched between different images.

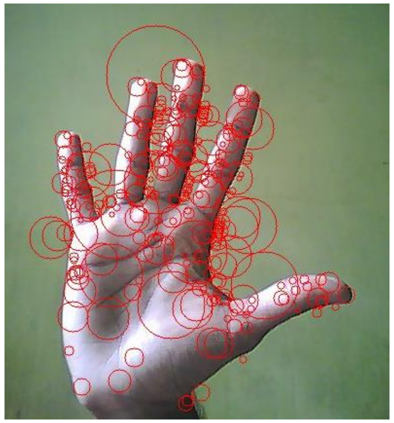

$$\begin{tikzcd} \arrow[r] & \circ \arrow[r] & \langle\psi|\varphi\rangle \arrow[r] \arrow[d] & {} \\ & & N_e^t(s) \arrow[ul] & \end{tikzcd}$$

SURF

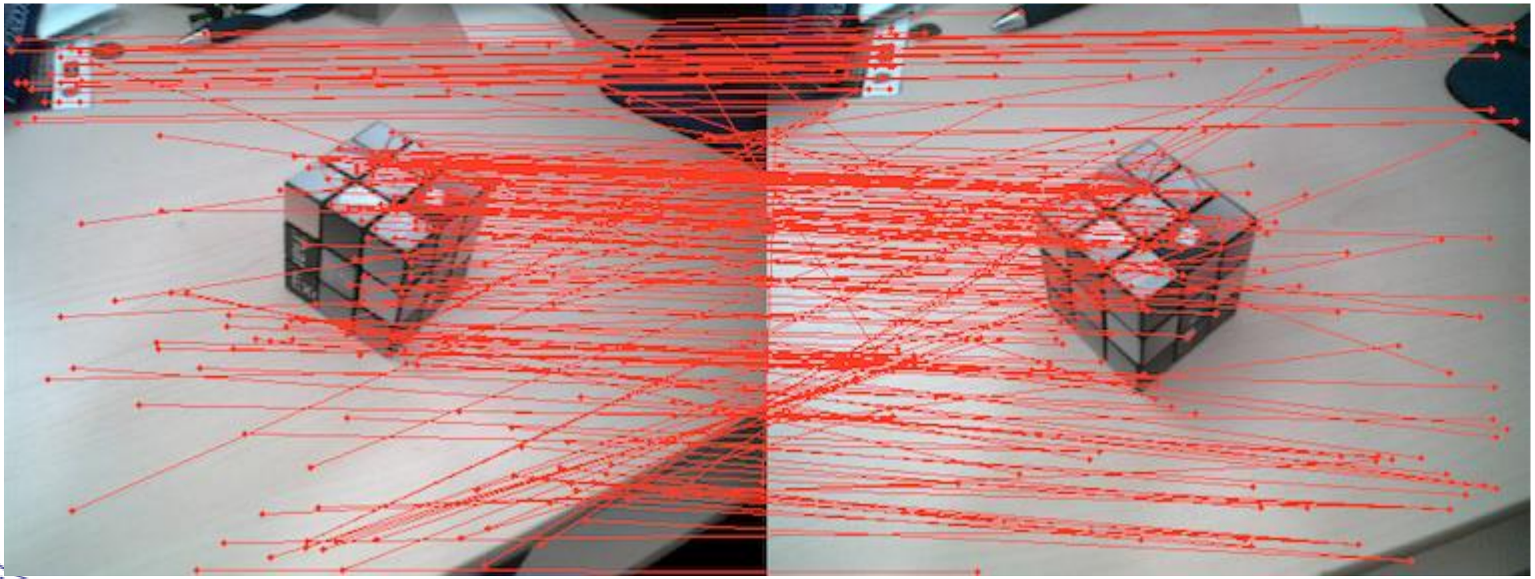

Input Image
Approximated Gaussian Filters
Build Integral Image
Calculate Hessian Determinant
Non-Max Suppression
Calculate Orientation
Calculate and Normalize Descriptors
Filter Scale
Vector<float>
OrientationVector
Vector<float[64]>
Descriptor Vector
Detected Scale
Vector<int>
Vector<float2>
Ipoint Position (x,y)
Resultant "Hessian Determinant" stack which are images filtered using different scaled Gaussian filters
Output: All 4 Vectors make up set of Interest Points (ipoints)

Surf

Features Extracted using SURF

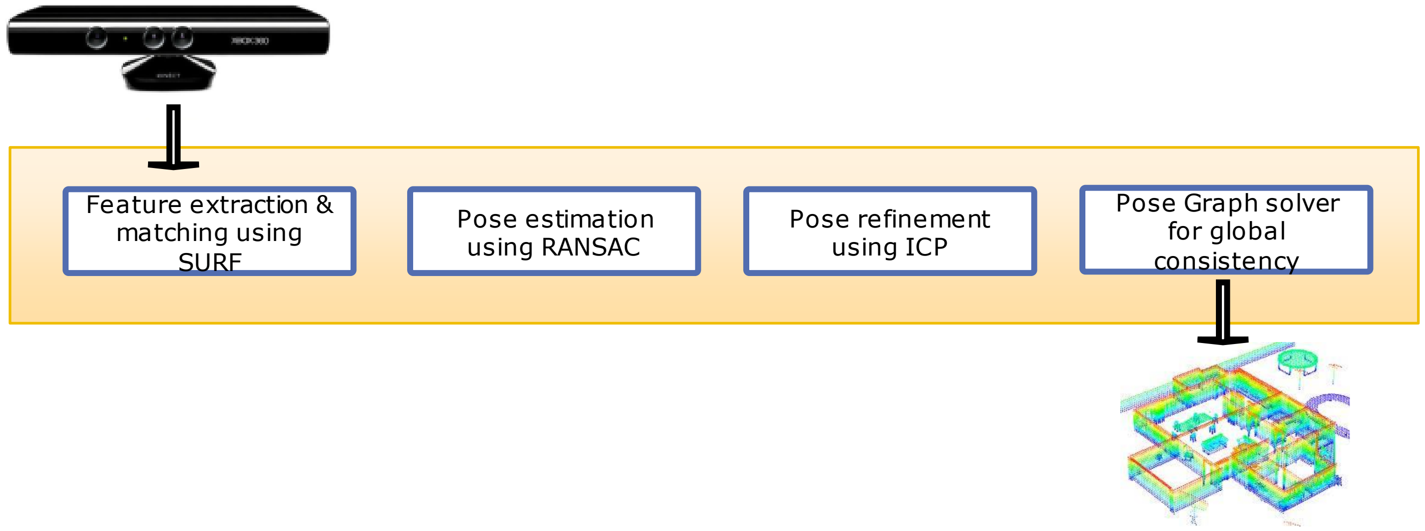


Features Matching:

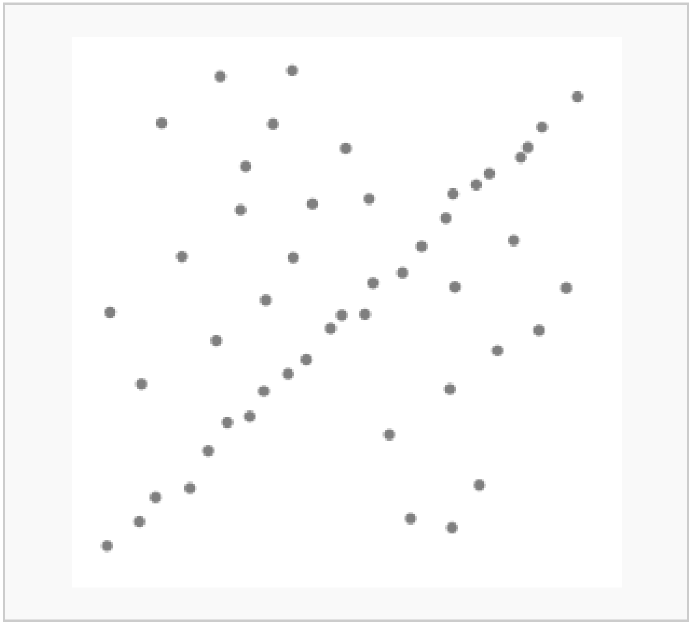


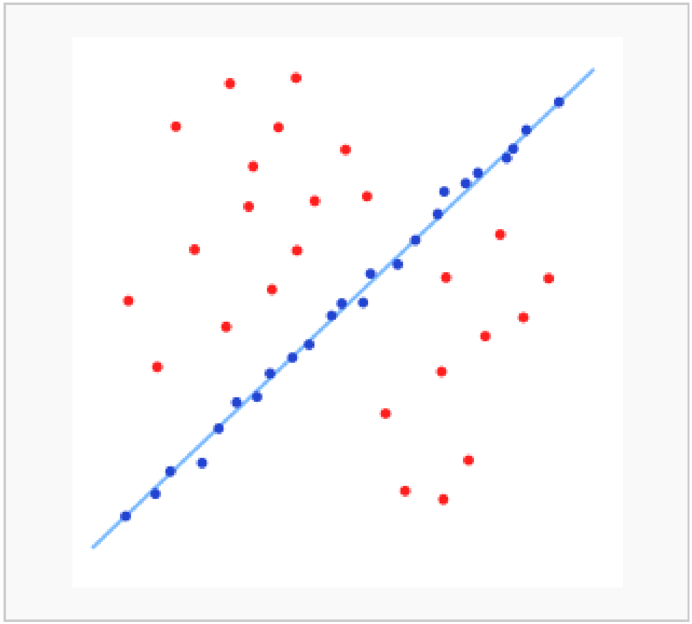
Frame t

Frame t+1


Feature extraction & matching using SURF
Pose estimation using RANSAC
Pose refinement using ICP
Pose Graph solver for global consistency

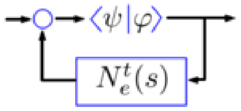
RANSAC: RANdom SAmple Consensus


A data set with many outliers for which a line has to be fitted.


Fitted line with RANSAC, outliers have no influence on the result.

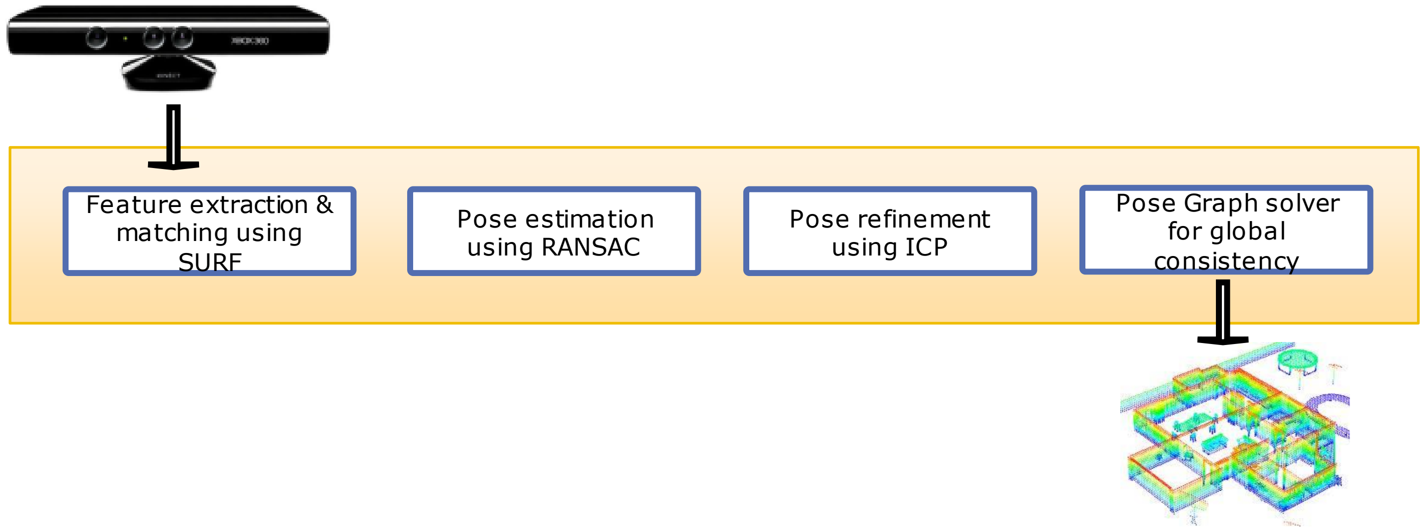
Ransac …. Algorithms

```
input:
    data - a set of observations
    model - a model that can be fitted to data
    n - the minimum number of data required to fit the model
    k - the number of iterations performed by the algorithm
    t - a threshold value for determining when a datum fits a model
    d - the number of close data values required to assert that a model fits well to data
output:
    best_model - model parameters which best fit the data (or nil if no good model is found)
    best_consensus_set - data points from which this model has been estimated
    best_error - the error of this model relative to the data
```



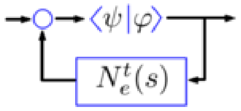
RANSAC ….. pseudo code

```
while iterations < k
    maybe_inliers := n randomly selected values from data
    maybe_model := model parameters fitted to maybe_inliers
    consensus_set := maybe_inliers

    for every point in data not in maybe_inliers
        if point fits maybe_model with an error smaller than t
            add point to consensus_set

    if the number of elements in consensus_set is > d
        (this implies that we may have found a good model,
        now test how good it is)
        this_model := model parameters fitted to all points in consensus_set
        this_error := a measure of how well this_model fits these points
        if this_error < best_error
            (we have found a model which is better than any of the previous ones,
            keep it until a better one is found)
            best_model := this_model
            best_consensus_set := consensus_set
            best_error := this_error

    increment iterations

return best_model, best_consensus_set, best_error
```


Feature extraction & matching using SURF
Pose estimation using RANSAC
Pose refinement using ICP
Pose Graph solver for global consistency

Pose refinement using ICP

•Iterative Closest Point algorithm
•Points in the source cloud are matched with their nearest neighboring points in a target cloud.
•Then a rigid transformation is found by minimizing the 3-D error between associated points.
•This may change the nearest neighbor for points.
•This algorithm is effective when two cloud are already nearly aligned.

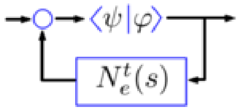


Pose graph optimization

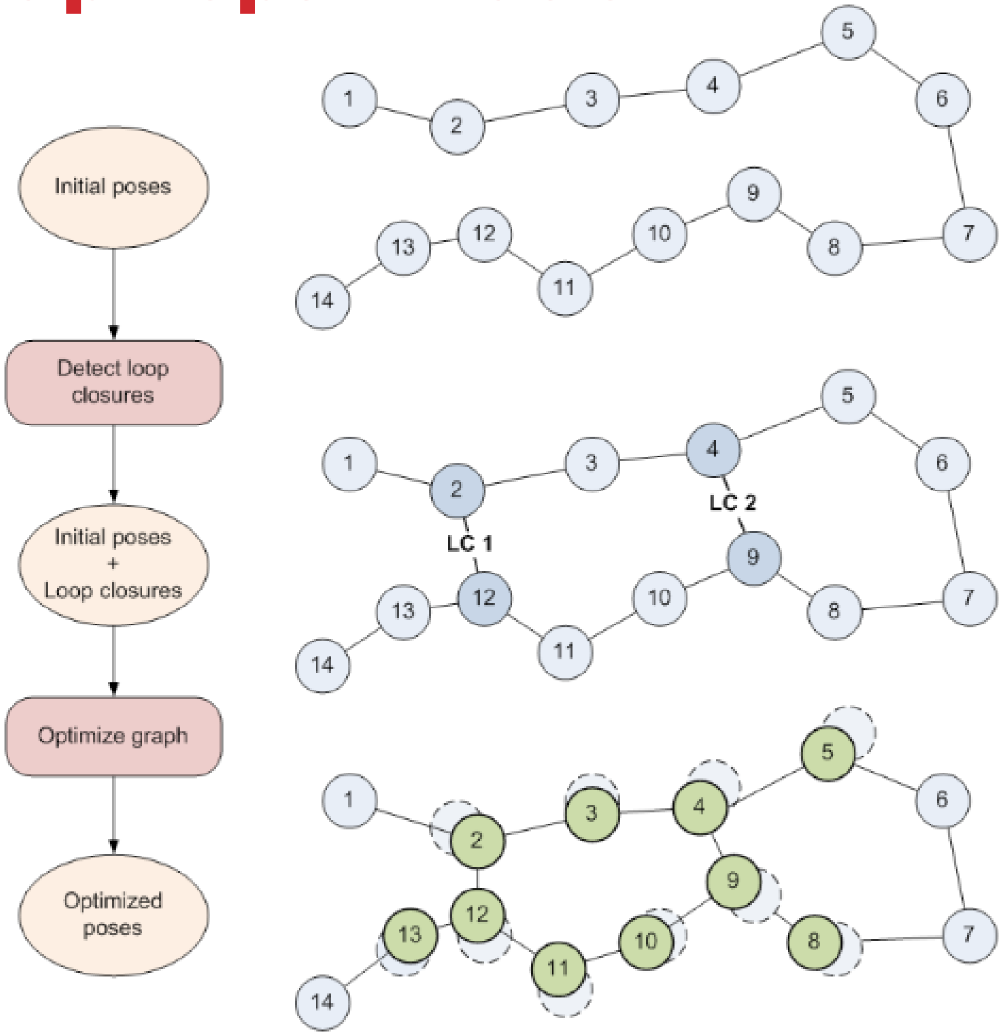
•As the pair-wise pose estimates between frames are not globally consistent.

•So we optimize the resulting pose graph using pose graph solver.

•Output is globally consistent 3D model of the environment.



Pose graph optimization



Graph optimization procedure

SLAM experiments

•We experimented with three different types of robots in varying ground profiles and lighting conditions.
•All three robots were custom built in our lab and controlled by ROS (Robot Operating System).

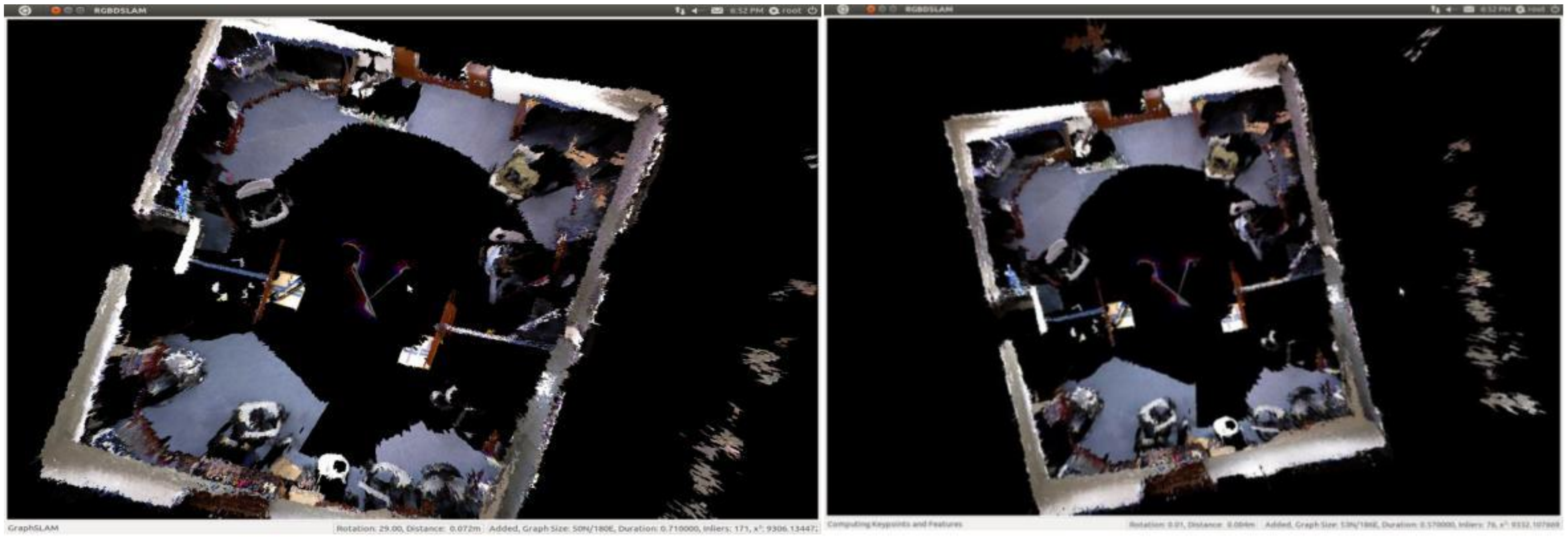
Hanoon

Marwa

Dul-dul

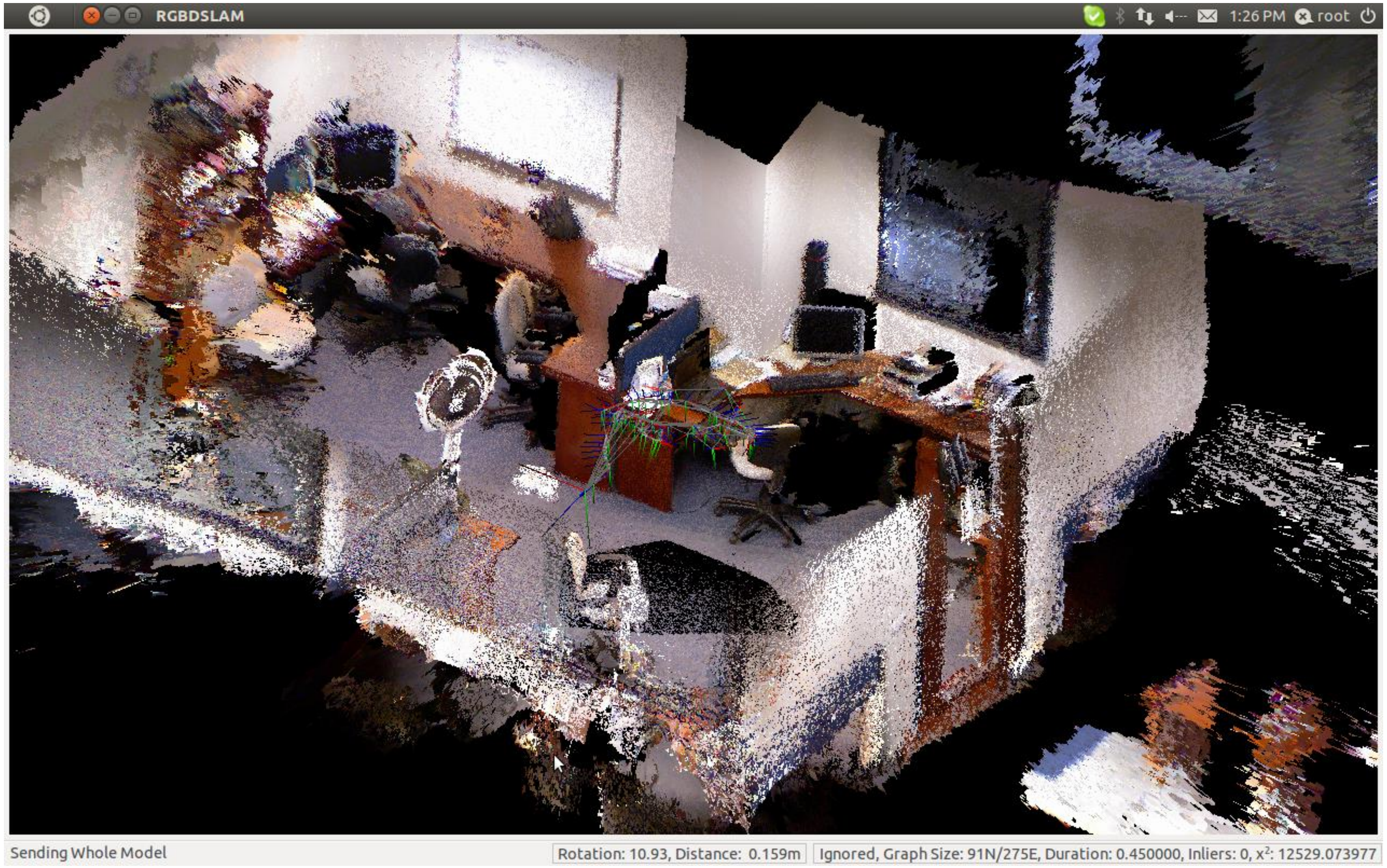
Indoor Slam (EE Dept, LUMS Campus)

•To establish baseline performance, we ran the robots in different indoor situations under variations of light intensity, dynamic obstacles, environment scales etc.
•Despite a compact environment, the robot moved collision free and built the map.



Indoor SLAM produced by *Hanoon* in our lab.

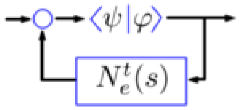
Indoor Slam



Sending Whole Model

Rotation: 10.93, Distance: 0.159m

Ignored, Graph Size: 91N/275E, Duration: 0.450000, Inliers: 0, x^2: 12529.073977


$$\begin{tikzcd} {} \arrow[r] & \circ \arrow[r] & \langle\psi|\varphi\rangle \arrow[r] & {} \arrow[d] \\ & & \boxed{N_e^t(s)} \arrow[ul] & {} \arrow[l] \end{tikzcd}$$

Outdoor slam

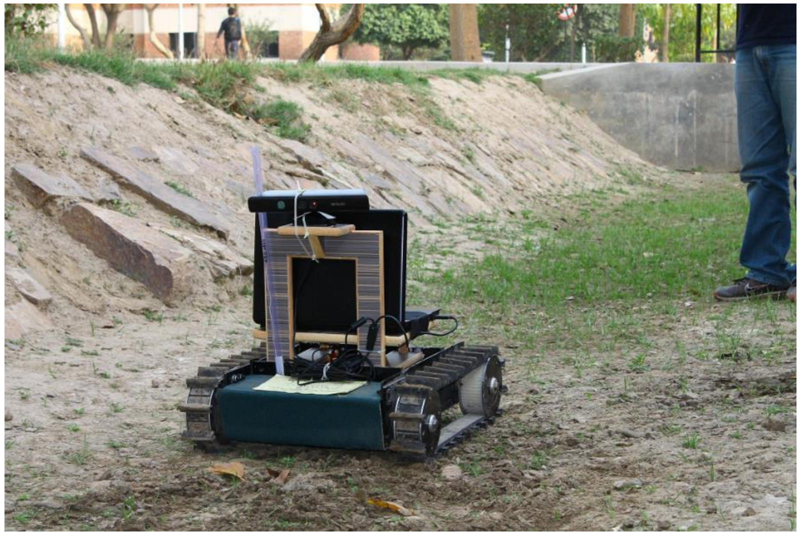
•We used different kinds of robots and three different kinds of environments.

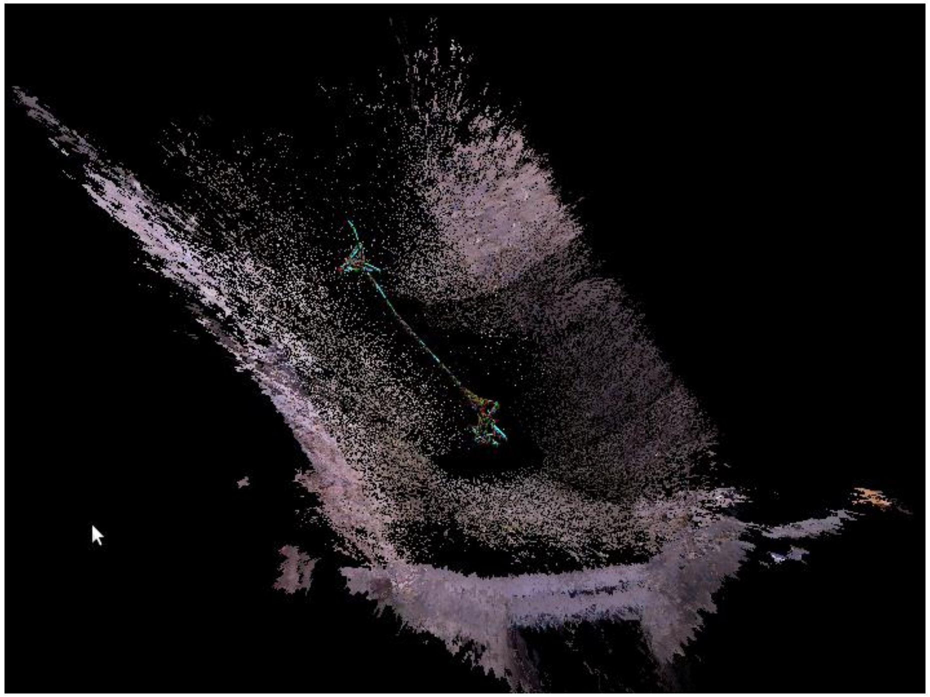
- •Outdoor Urban SLAM
- •Heavy vegetation SLAM
- •Dirt Road SLAM

•The operation was significantly slowed down as compared to indoors due to intermittent availability of data and occasional failure to find correspondence points.

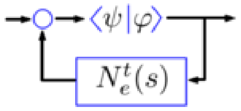
•Still, the overall results were satisfactory.

Dirt road slam (LUMS campus)

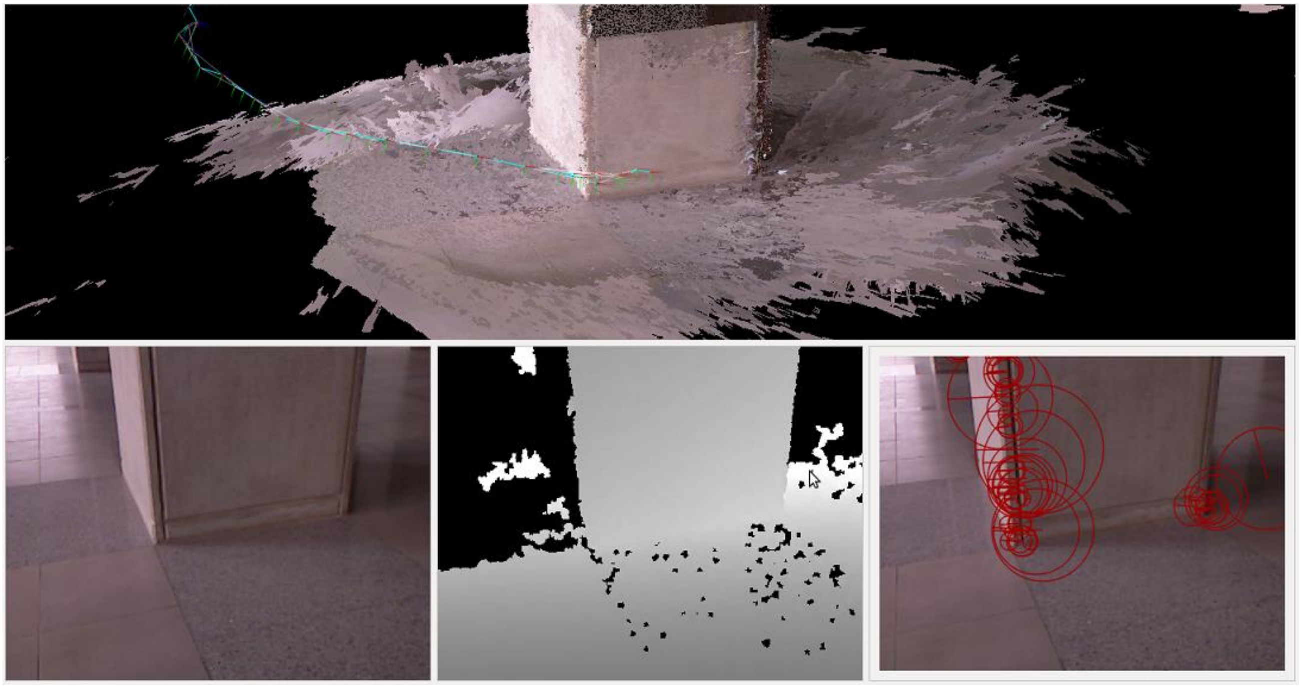




Dry canal profiling with robot *Dul-dul*.

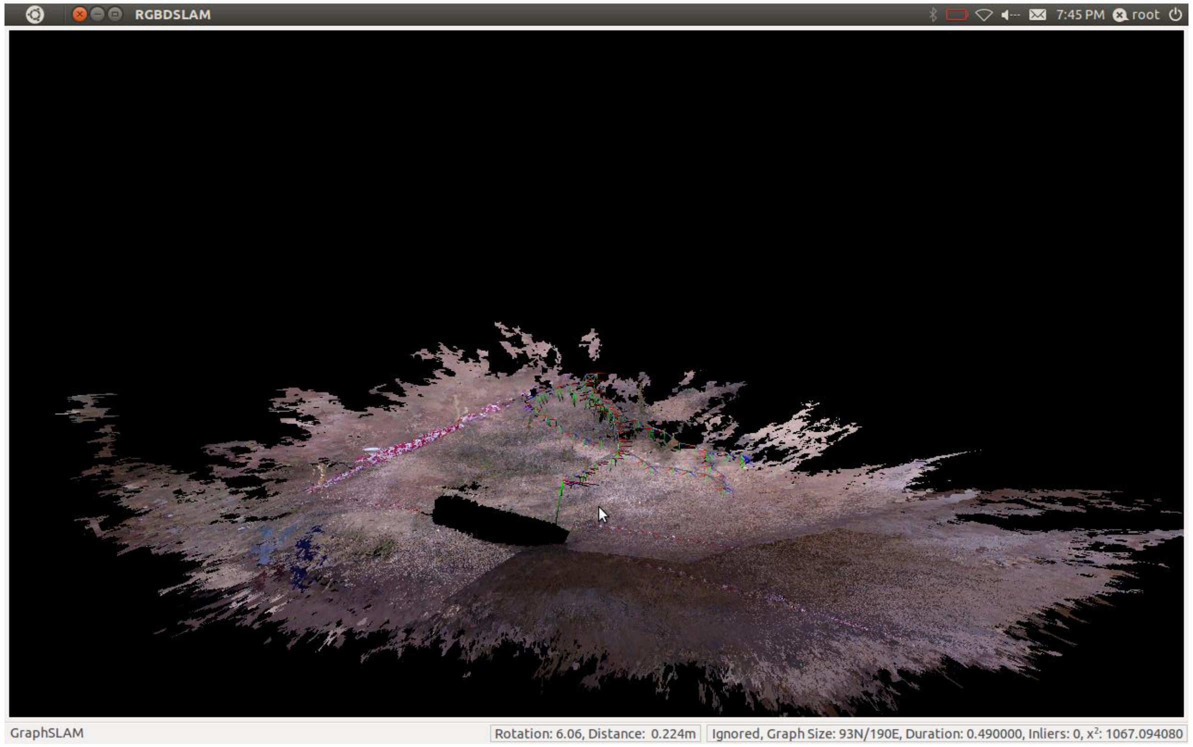


Outdoor urban slam (SSE Bldg, LUMS campus)



RGB-D SLAM with robot *Duldul* in an outdoor structured environment.

Heavy Vegetation slam



```
GraphSLAM    Rotation: 6.06, Distance: 0.224m    Ignored, Graph Size: 93N/190E, Duration: 0.490000, Inliers: 0, x²: 1067.094080
```

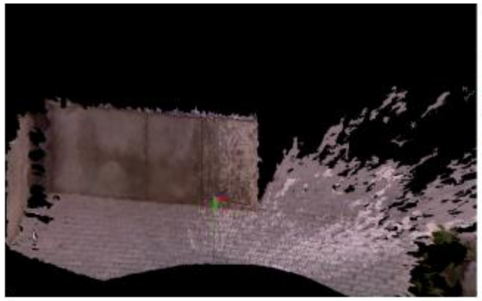
Arena mapped during Beirut trials.

Loop closing performance

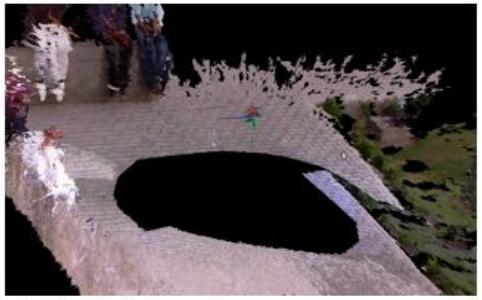
•We designed an experiment for outdoor environment with significant process noise in the robot motion.
•SLAM was performed by typical loop closing.
•We found out that in the first loop, the robot lost the global consistency and did not stitch the map so well.
•But in subsequent loops it improved the mapping performance and errors were subsequently reduced.
•In a typical experiment, the results after third loop was nearly globally consistent.
•This matches with the objectives of mine-sweeping.

Loop closing experiment (LUMS)

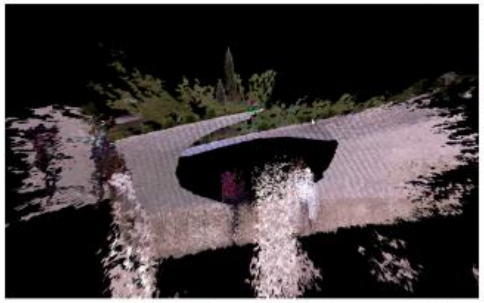
1



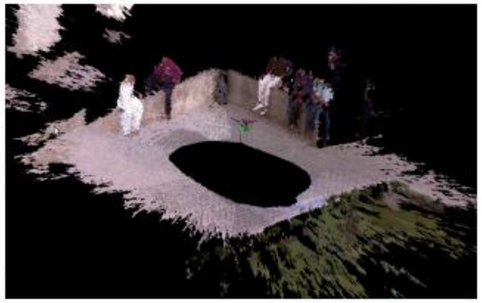
2



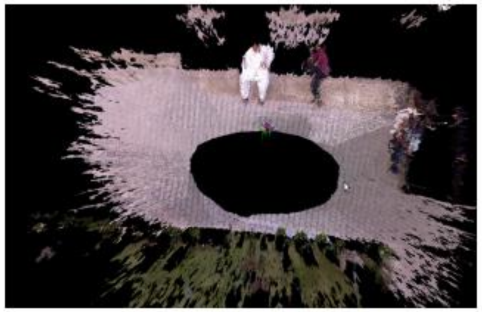
3



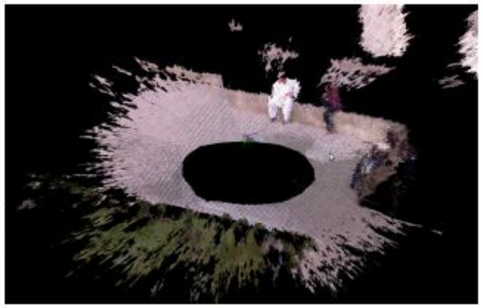
4



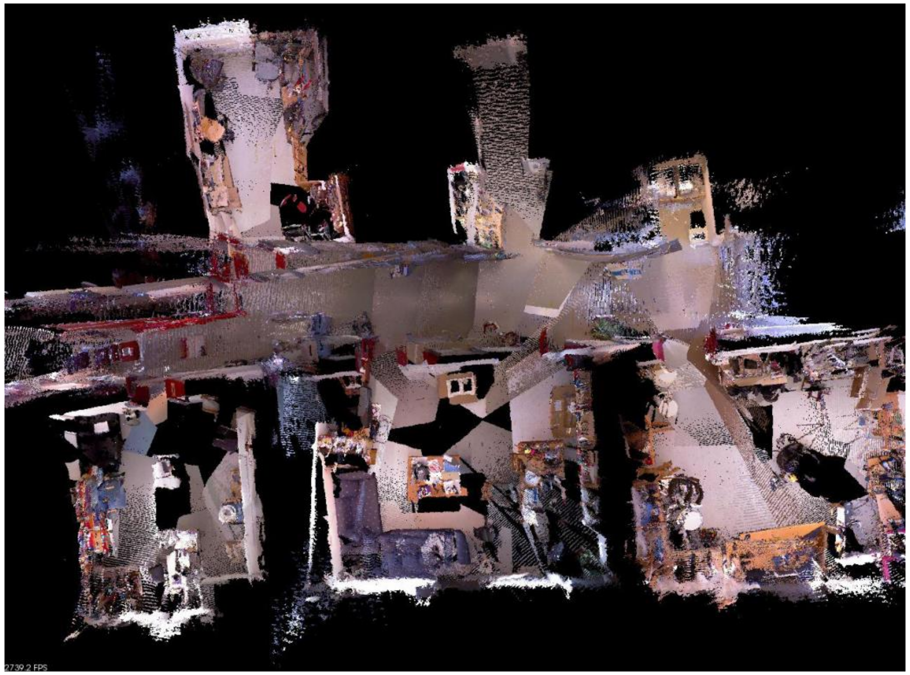
5

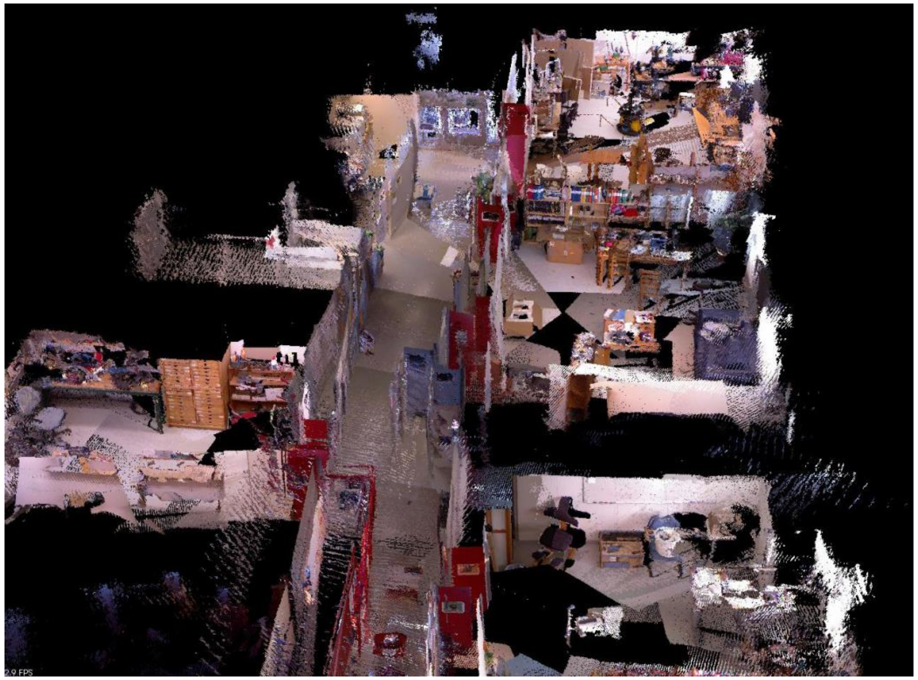


6



At Other Labs





Thank you !