SLAM in Egocentric Videos

- *Sarthak Ahuja*

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY
DELHI

Overview

Fold I

1. Motivation
2. Recap

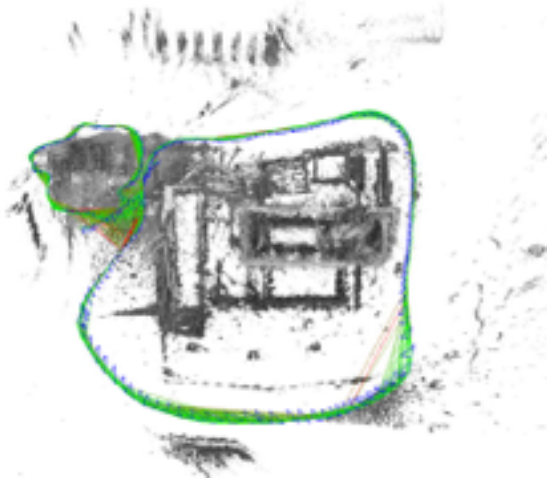


Figure: Example SLAM

Fold II

1. Theory
   - I. Pre-requisites
   - II. Derivation
2. Implementation
   - I. MATLAB
   - II. ROS/C++
3. Dataset & Results
4. Demo
5. Future Work

FOLD I

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY
DELHI

Motivation

Figure: Google Glass

- GoPro cameras and Google Glass gaining immense popularity in recent years.
- The videos created by these wearable cameras give a plethora of information about his/her activities.
- Conditions have become ideal for the use of computer vision technologies in the field of egocentric vision.

Motivation

- Gaining information about the path followed by an observer, detecting the current pose and field of view of the person can be of immense importance.
  - It can be used as a **local positioning system** in an unknown environment, eventually turning the unknown environment into a much more known one.
  - Its applications extend to **detecting observer behavior** by tracking the pose of the user at various key frames to learn more about the field of interest of the user.
  - Large egoCentric videos can be broken into chapters based on pose estimates of the camera allowing **quick browsing**.
  - For visually impaired people these wearable cameras can have even more significance in **tracking their path** and generating stimuli according to their pose to give them needful information about their surrounding.

Motivation

- Google's Project Tango:

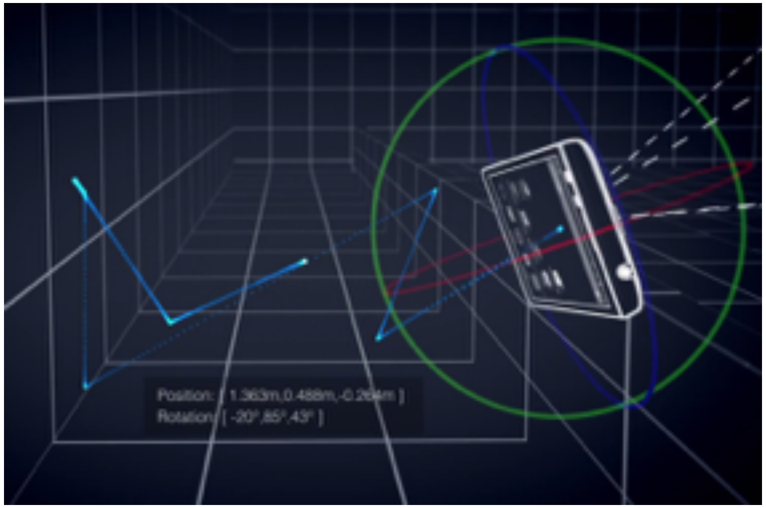

Position: [1.363m,0.488m,-0.264m]
Rotation: [-20°,85°,43°]

Figure: SLAM in Project Tango

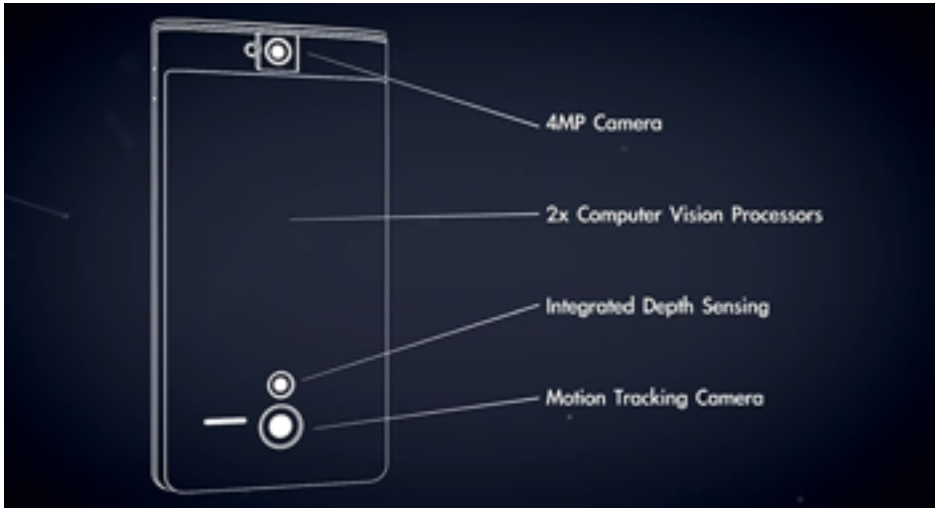
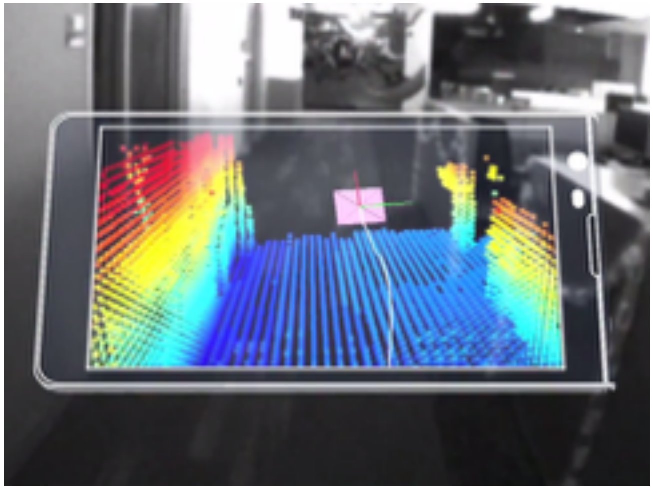


Figure: Device Anatomy

Recap

- SLAM is the abbreviation for Simultaneous Localization And Mapping.
- It is a problem involving the construction of a consistent map of an unknown environment when placed in an unknown location.
- It can be seen as a real time implementation of Structure from Motion with a focus on consistent mapping.
  - Camera Localization: Estimating the pose of the camera.
  - Triangulation/Mapping: Estimating the depth of the pixels in an image.



Recap

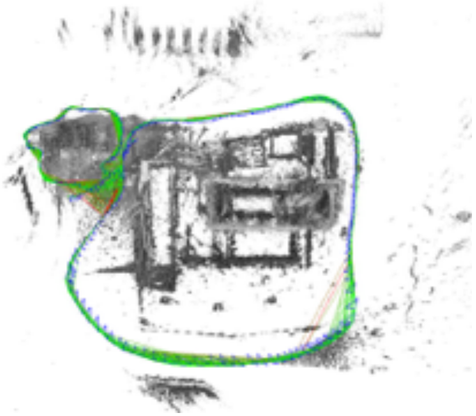


Figure: Example SLAM

- SLAM involves 3D reconstruction of the surrounding while at the same time tracking the camera pose at certain key frames.
- It is a very popular technique regularly being applied in robotic vision, now being extended to handheld and wearable devices possessing stereo or monocular cameras.
- SLAM can be positively applied to the case of egoCentric videos to solve the challenge of tracking.

Recap

- SLAM Algorithms:
  - PTAM
    - Highlights: Stereo Initialization, Map is optimized with bundle adjustment, **Image Pyramid**, FAST Corner Features, **Separate Threads for Tracking and Mapping.**
    - Drawbacks: the need of human intervention to initialize the map, inability to generate dense maps, and limited accuracy to only indoor environments.
  - DTAM
    - Highlights: **Photometric Error Minimization**, Direct Image Alignment, **More features.**
    - Drawbacks: Not possible in real time.

Recap

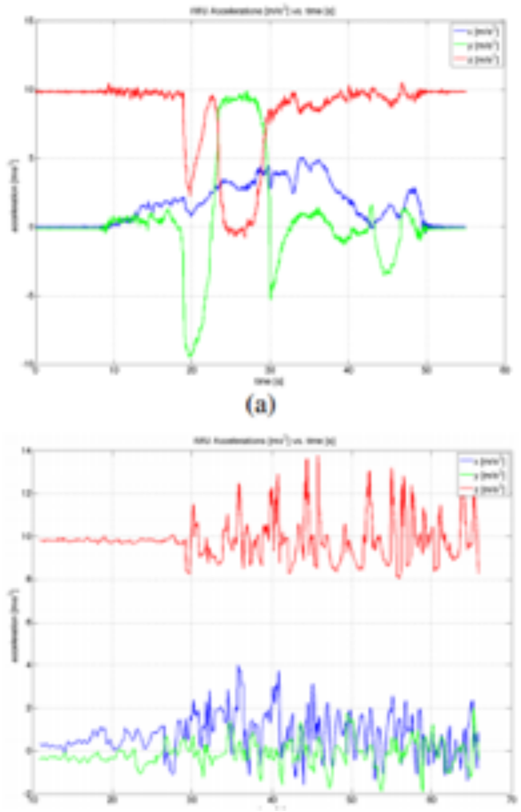
- SLAM Algorithms:
  - RGB-D SLAM
    - Highlights: Depth Data Available, Handled Moving Objects,
  - LSD-SLAM
    - Highlights: Direct Monocular SLAM

FOLD II

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY
DELHI

Initial Results & Challenges

- Applying SLAM to the egocentric context is a tricky task.



(a)

Figure: Comparison of varying acceleration in handheld cameras and wearable cameras

Initial Results & Challenges

LSD SLAM Results on Test Dataset:

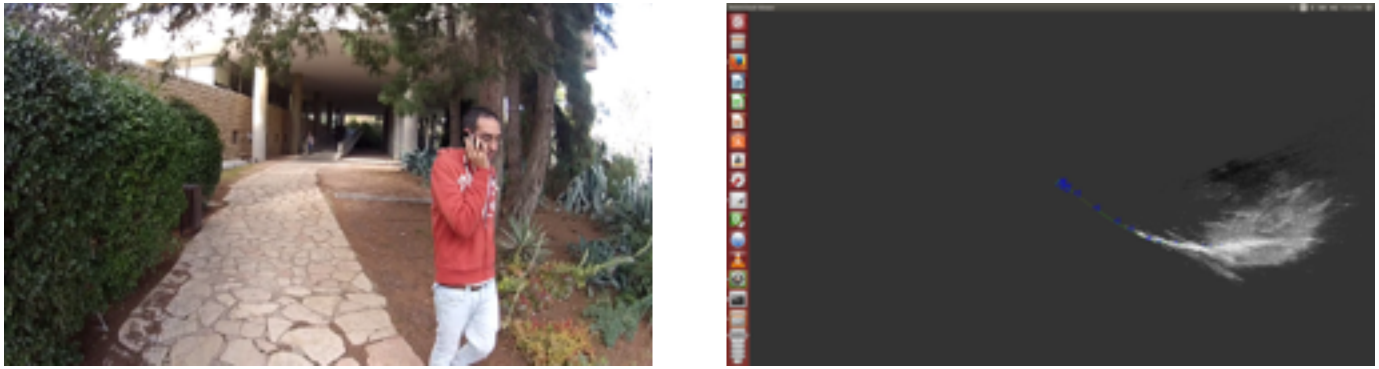


Figure: Huji_Chetan_1

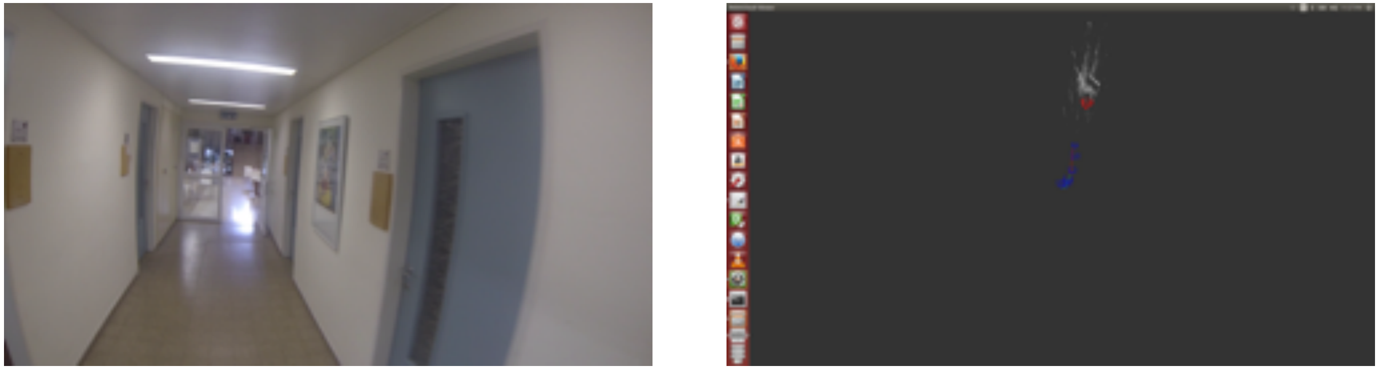


Figure: Huji_Chetan_2

Initial Results & Challenges

LSD SLAM Results on Test Dataset:

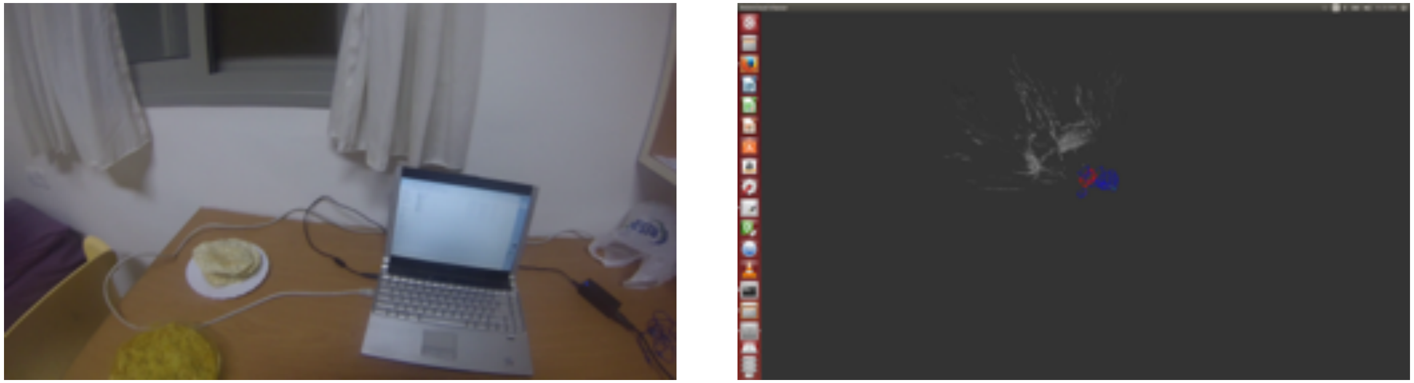


Figure: Huji_Chetan_1_Dinner_Part1

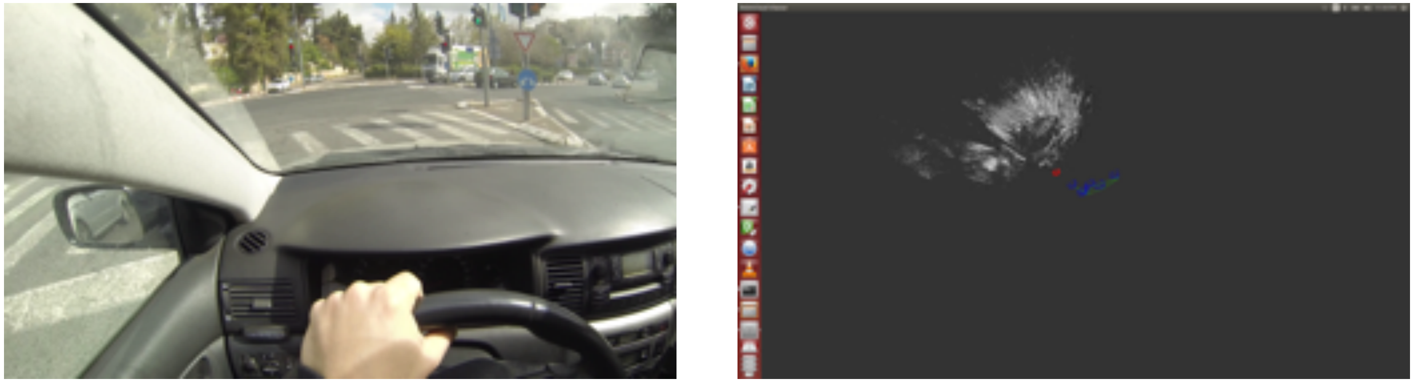


Figure: Huji_Yair_1_Part_1

Initial Results & Challenges

LSD SLAM Results on Test Dataset:

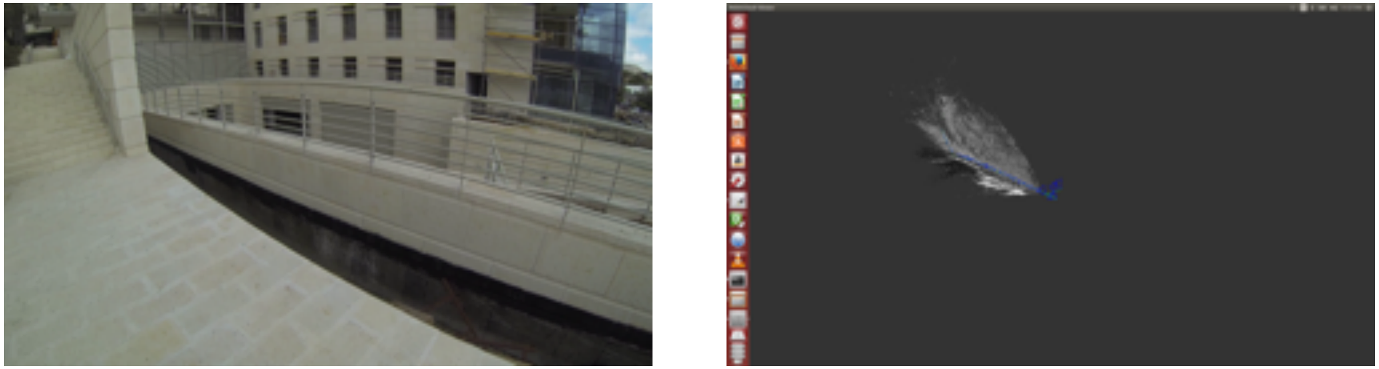


Figure: Huji_Yair_1_Part_2

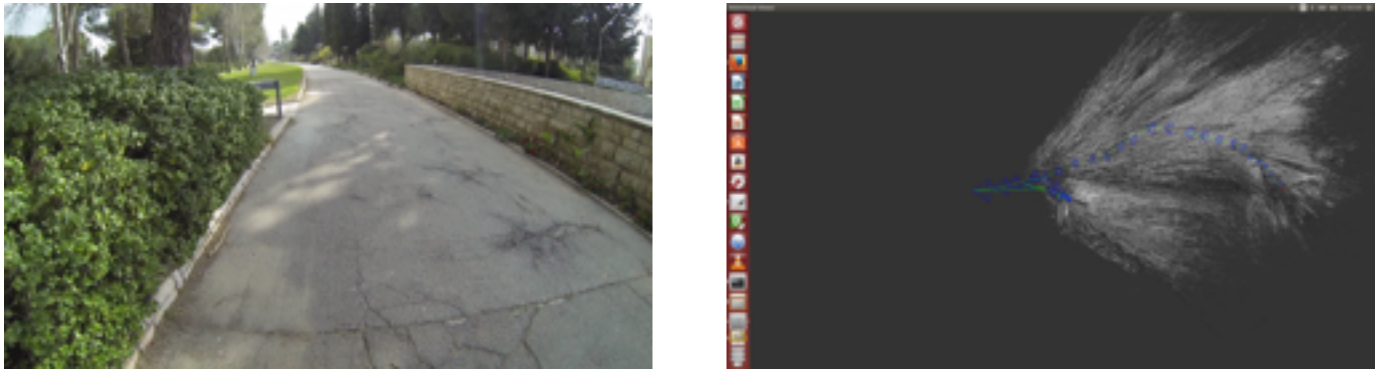


Figure: Huji_Yair_3

Initial Results & Challenges

LSD SLAM Results on Test Dataset:

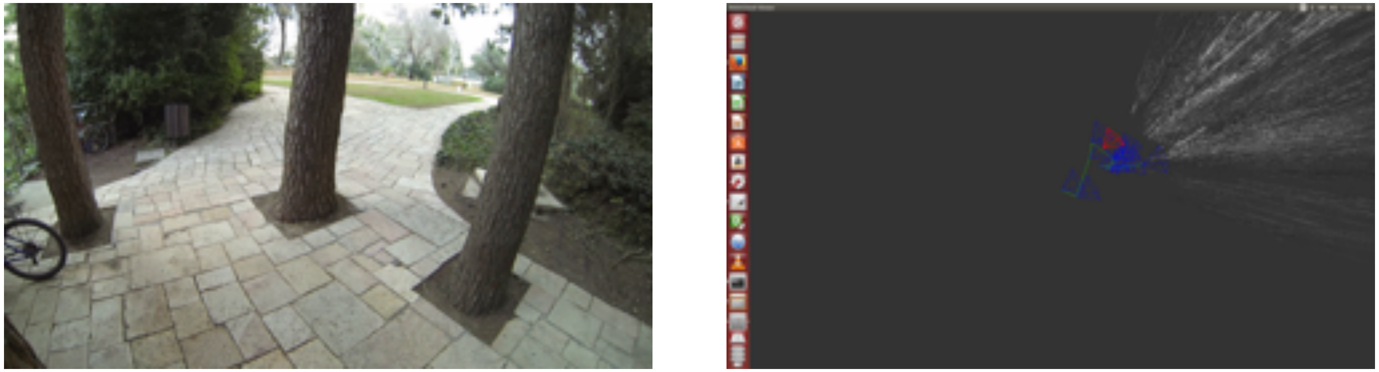


Figure: Huji_Yair_4

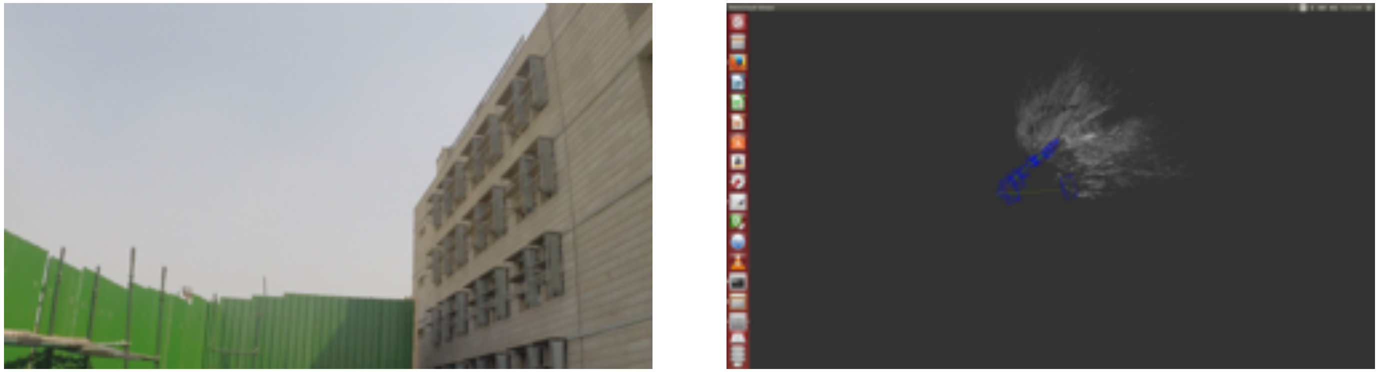


Figure: GOPRO336

Initial Results & Challenges

LSD SLAM Results on Test Dataset:

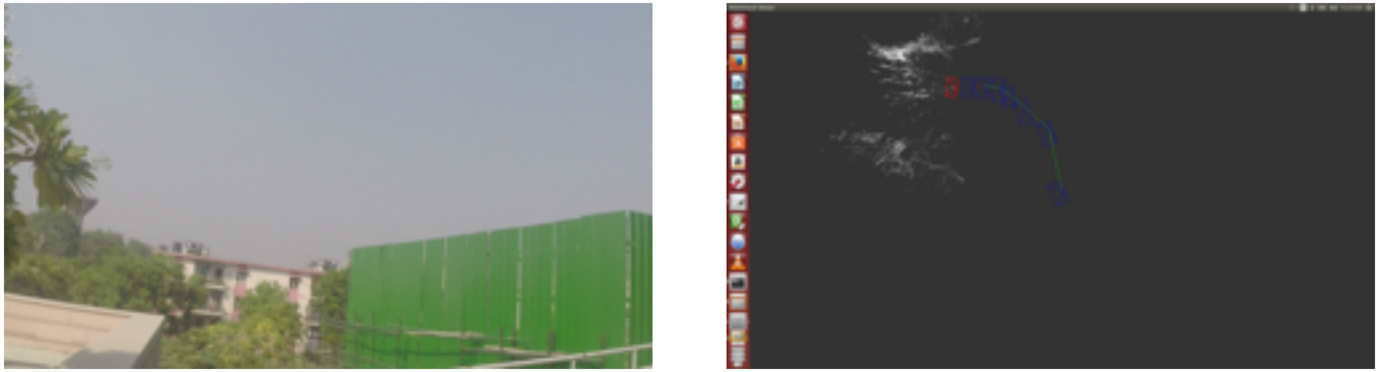


Figure: GOPRO346

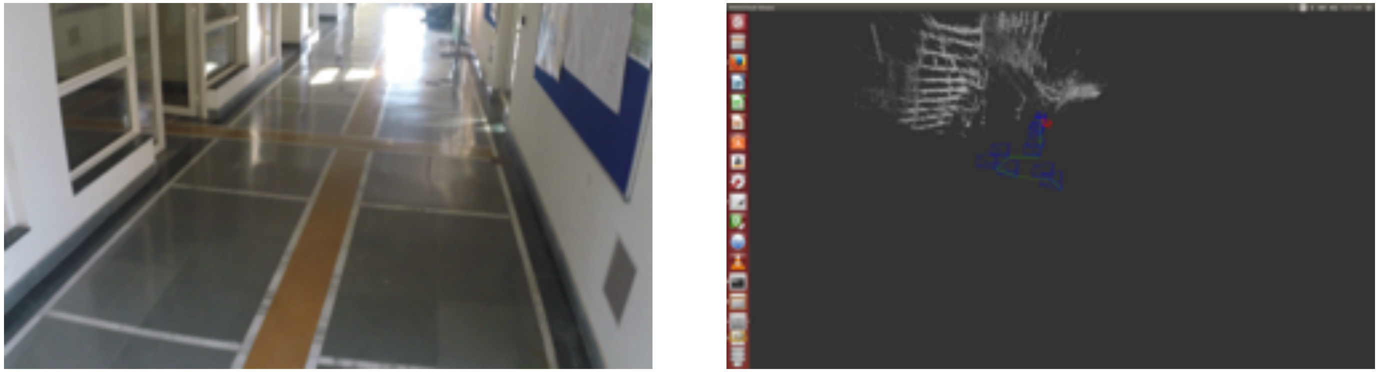


Figure: GOPRO418

Initial Results & Challenges

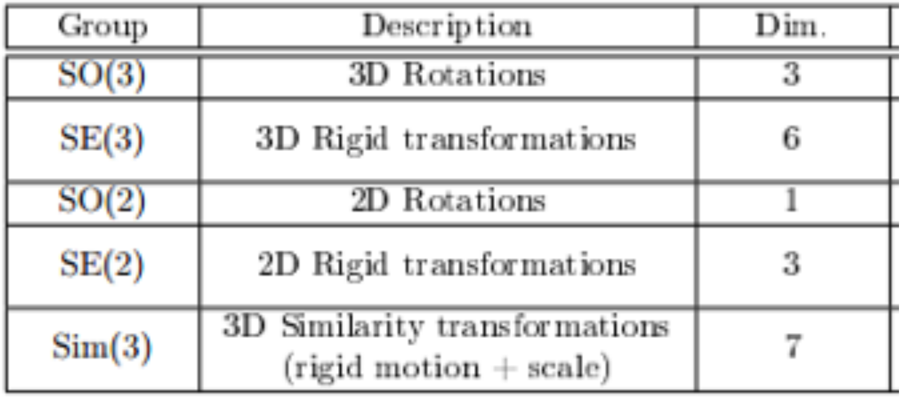
Challenges:

- *Biased Forward Movement of the Camera*
  - *Optical Axis Parallel to Direction of Motion.*
- *Variation in Acceleration*
  - *Natural Head Motion leads to blurring, which in turn leads to loss of features.*
- *Presence of Moving Objects*
  - *Moving Objects*
- *Map Initialization*
  - *Since we are not using a depth camera, and computing depth on the go. Initialization requires a period of translational motion, which may often not be the case.*
- *Length of the Video*
  - *LSD-SLAM is highly memory optimized but still faces a significant lag due to the sheer length of an egocentric video.*
- *Gradient Loss*
  - *Illumination in open scenes, often leads to loss of gradient.*

Theory: Prerequisites

- Lie Groups and Lie Algebra
    - A group on a smooth manifold with some nice properties.
    - With it is an associated Lie Algebra - a tangential vector space.
    - Why? Transformations must be composed, inverted, differentiated and interpolated.
    - Differentiating group transformations along chosen directions in the space, at the identity transformation.
    - The basis elements of the lie algebra are called generators.
    - Optimal space to represent differential quantities related to the group - Jacobian
        - Exponential Map converts any element exactly into a transformation.

Theory: Prerequisites



Group	Description	Dim.
SO(3)	3D Rotations	3
SE(3)	3D Rigid transformations	6
SO(2)	2D Rotations	1
SE(2)	2D Rigid transformations	3
Sim(3)	3D Similarity transformations (rigid motion + scale)	7

Lie Groups

$$G_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}, \; G_2 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}, \; G_3 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Differentiating at Identity

$$\begin{aligned} \boldsymbol{\omega} &\in \mathbb{R}^3 \\ \omega_1 G_1 + \omega_2 G_2 + \omega_3 G_3 &\in \mathrm{so}(3) \end{aligned}$$

Composition of a lie algebra element in SO(3)

Theory

Preliminaries:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Transformation Function: $T(g, \mathbf{p}) = R\mathbf{p} + t$

Projection Function: $\pi(T(g,p)) = (\frac{fx}{z} + c_x, \frac{fy}{z} + c_y)^T$

Warping Function: $w(\xi, \mathbf{x}) = \pi(T(g(\xi), \mathbf{p}))$

Residual/pixel: $r_i = (I_2(w(\xi, \mathbf{x}_i)) - I_1(\mathbf{x}_i))$

Theory

Lie Algebra SE(3):

Generator Matrices:

$$G_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} G_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$G_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} G_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$G_5 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} G_6 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Lie Composition: $se(3) = v_1 G_1 + v_2 G_2 + v_3 G_3 + w_1 G_4 + w_2 G_5 + w_3 G_6$

Theory

Jacobian:

$$\frac{\partial \mathbf{p}'}{\partial \xi} = \frac{\partial}{\partial \xi}(exp(\xi)\mathbf{p}$$

$$\frac{\partial \mathbf{p}'}{\partial \xi} = \frac{\partial \mathbf{p}'}{\partial \mathbf{C}} \frac{\partial \mathbf{C}}{\partial \xi}$$

$$\frac{\partial \mathbf{p}'}{\partial C} = \begin{bmatrix} \mathbf{p}' & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{p}' & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{p}' \end{bmatrix} 3 \times 12$$

$$\frac{\partial \mathbf{p}'}{\partial \xi} = (\mathbf{I}| - \mathbf{p}'_{\times})$$

$$\frac{\partial \mathbf{C}}{\partial \xi} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} 12 \times 6$$

$$\frac{\partial \mathbf{p}'}{\partial \xi} = (G_1.(\mathbf{p}')|G_2.(\mathbf{p}')|G_3.(\mathbf{p}')|G_4.(\mathbf{p}')|G_5(\mathbf{p}')|G_6.(\mathbf{p}'))$$

Theory

Probabilistic Definition:

$$p(r|\xi) = \prod p(r_i|\xi)$$

$$p(\xi|r) = \frac{p(r|\xi)p(\xi)}{p(r)}$$

$$\xi = argmax(p(\xi|r))$$

$$\xi = argmax(\prod p(r|\xi)p(\xi))$$

$$\xi = argmin(-\sum log(p(r|\xi) - log(p(\xi)))$$

$$\frac{\partial(log(p(r_i|\xi)))}{\partial\xi} = 0$$

$$\frac{\partial(log(p(r_i|\xi)))}{\partial r_i}\frac{\partial(r_i)}{\partial\xi} = 0$$

On substituting $w(r_i) = \frac{\partial log(p(r_i|\xi))}{\partial r_i}\frac{1}{r_i}$:

$$\frac{\partial(r_i)}{\partial\ \xi}w(r_i)r_i = 0$$

$$\xi = argmin(\sum w(r_i)r_i^2(\xi))$$

This model allows for flexibility in adding a Motion Prior and a Outlier removal model.

Theory

Linearization:

$$r(\Delta\xi) = (I_2 - I_2^{predicted})^2$$

$$0 = I_2 - \frac{\partial(I_1(\xi))}{\partial\ \xi}\Delta\xi - I_1$$

$$0 = r(\xi) - \frac{\partial(I_1(\xi))}{\partial\ \xi}\Delta\xi$$

$$0 = r(\xi) - \frac{\partial(I_1(\xi))}{\partial\ \mathbf{x}}\frac{\partial(\mathbf{x})}{\partial\ \xi}\Delta\xi$$

$$r(\xi) = J(\xi)\Delta\xi$$

$$\Delta\xi = (J^T J)^{-1} J^T r(\xi) \qquad J^T W J \Delta\boldsymbol{\xi} = -J^T W \mathbf{r}(\mathbf{0}),$$

$$\boldsymbol{\xi}^{(k+1)} = \log(\exp(\boldsymbol{\xi}^{(k)})\exp(\Delta\boldsymbol{\xi}))$$

Theory

Final Jacobian:

$$J_i = \frac{\partial(I_{1i}(\xi))}{\partial\ \mathbf{x_i}} \frac{\partial(\mathbf{x_i})}{\partial\ \xi}$$

$$J_i = \frac{\partial(I_{1i})}{\partial\ \mathbf{x_i}} \frac{\partial(\mathbf{x_i})}{\partial\ \mathbf{p_i}}\ \frac{\partial(\mathbf{p_i})}{\partial\ \xi}$$

$$J_i = grad^T \frac{\partial(\mathbf{p})}{\partial\xi}$$

$$grad = \frac{\partial(I_{1i})}{\partial\ \mathbf{x_i}} \frac{\partial(\mathbf{x_i})}{\partial\ \mathbf{p_i}}$$

$$grad = \begin{bmatrix} \frac{\partial I(\mathbf{x})}{\partial u} & \frac{\partial I(\mathbf{x})}{\partial v} \end{bmatrix} \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial'} & \frac{\partial v}{\partial z} \end{bmatrix}$$

$$\frac{\partial\mathbf{p}}{\partial\xi} = (\mathbf{I}| - \mathbf{p}_\times)$$

$$grad = \begin{bmatrix} \frac{\partial I(\mathbf{x})}{\partial u} & \frac{\partial I(\mathbf{x})}{\partial v} \end{bmatrix} \begin{bmatrix} \frac{1}{z} & 0 & -\frac{x}{z^2} \\ 0 & \frac{1}{z} & -\frac{y}{z^2} \end{bmatrix}$$

$$J_i = grad^T . \frac{\partial p}{\partial\xi}$$

Theory

Final Jacobian:

$$J_i = grad^T . \frac{\partial p}{\partial \xi}$$

$$J_i = \begin{bmatrix} \frac{\partial I(\mathbf{x})}{\partial u} & \frac{\partial I(\mathbf{x})}{\partial v} \end{bmatrix} \begin{bmatrix} \frac{1}{z} & 0 & -\frac{x}{z^2} \\ 0 & \frac{1}{z} & -\frac{y}{z^2} \end{bmatrix} .(\mathbf{I}| - p_\times)$$

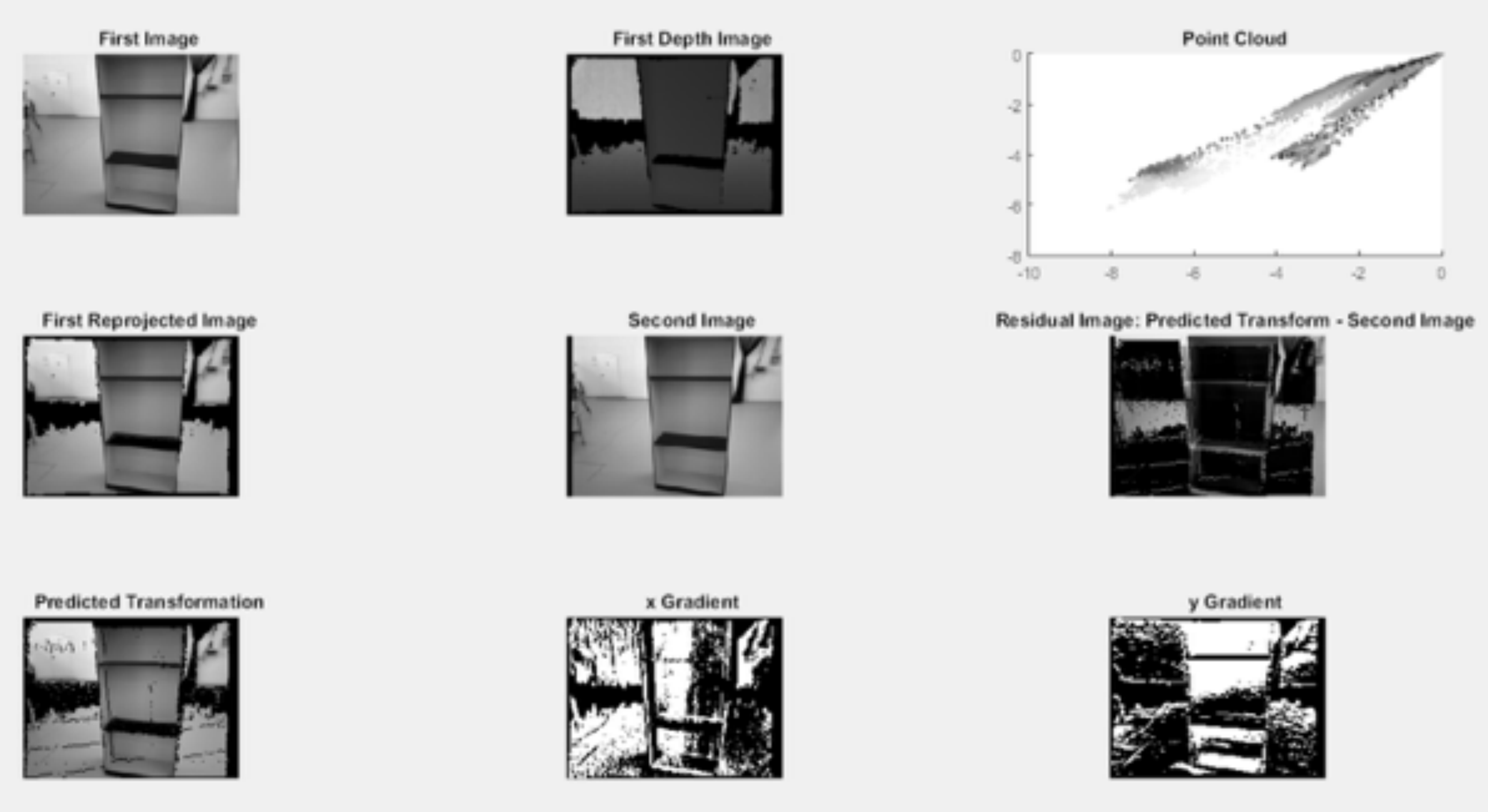
$$J_i = \begin{bmatrix} \frac{\partial I(\mathbf{x})}{\partial u} & \frac{\partial I(\mathbf{x})}{\partial v} \end{bmatrix} \begin{bmatrix} \frac{1}{z} & 0 & -\frac{x}{z^2} \\ 0 & \frac{1}{z} & -\frac{y}{z^2} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & z & -y \\ 0 & 1 & 0 & -z & 0 & x \\ 0 & 0 & 1 & y & -x & 0 \end{bmatrix}$$

$$J_i = \begin{bmatrix} \frac{\partial I(\mathbf{x})}{\partial u} & \frac{\partial I(\mathbf{x})}{\partial v} \end{bmatrix} \begin{bmatrix} \frac{1}{z} & 0 & -\frac{x}{z^2} & -\frac{xy}{z^2} & 1 + \frac{x^2}{z^2} & \frac{x}{z} \\ 0 & \frac{1}{z} & -\frac{y}{z^2} & -(1 + \frac{y^2}{z^2}) & \frac{xy}{z^2} & -\frac{y}{z} \end{bmatrix}$$

$$J_i = \begin{bmatrix} \frac{\partial I(\mathbf{x})}{\partial u} & \frac{\partial I(\mathbf{x})}{\partial v} \end{bmatrix} \begin{bmatrix} \frac{1}{Z(\mathbf{x})} & 0 & -\frac{u}{Z(\mathbf{x})} & -uv & 1 + u^2 & -u \\ 0 & \frac{1}{Z(\mathbf{x})} & -\frac{v}{Z(\mathbf{x})} & -(1 + v^2) & uv & -v \end{bmatrix}$$

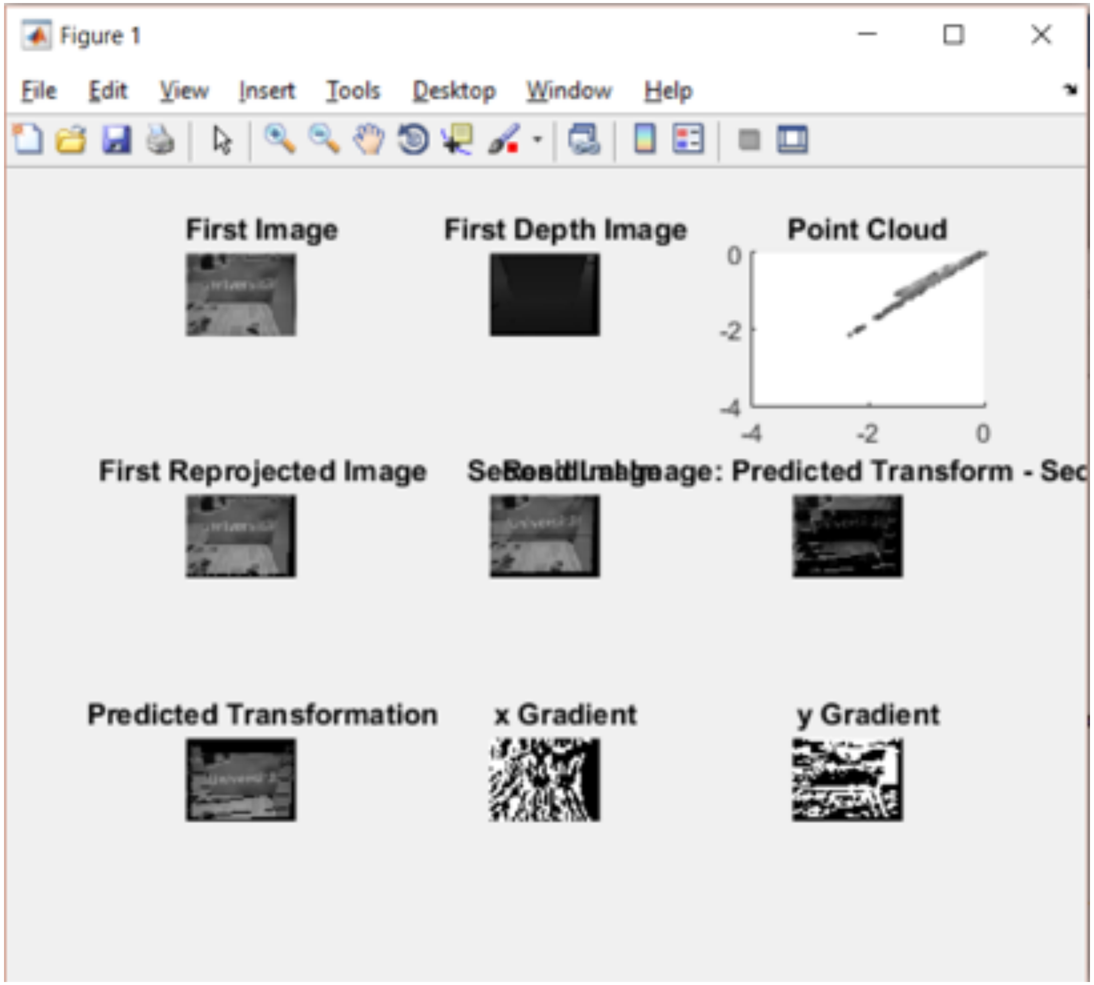
Implementation

- MATLAB



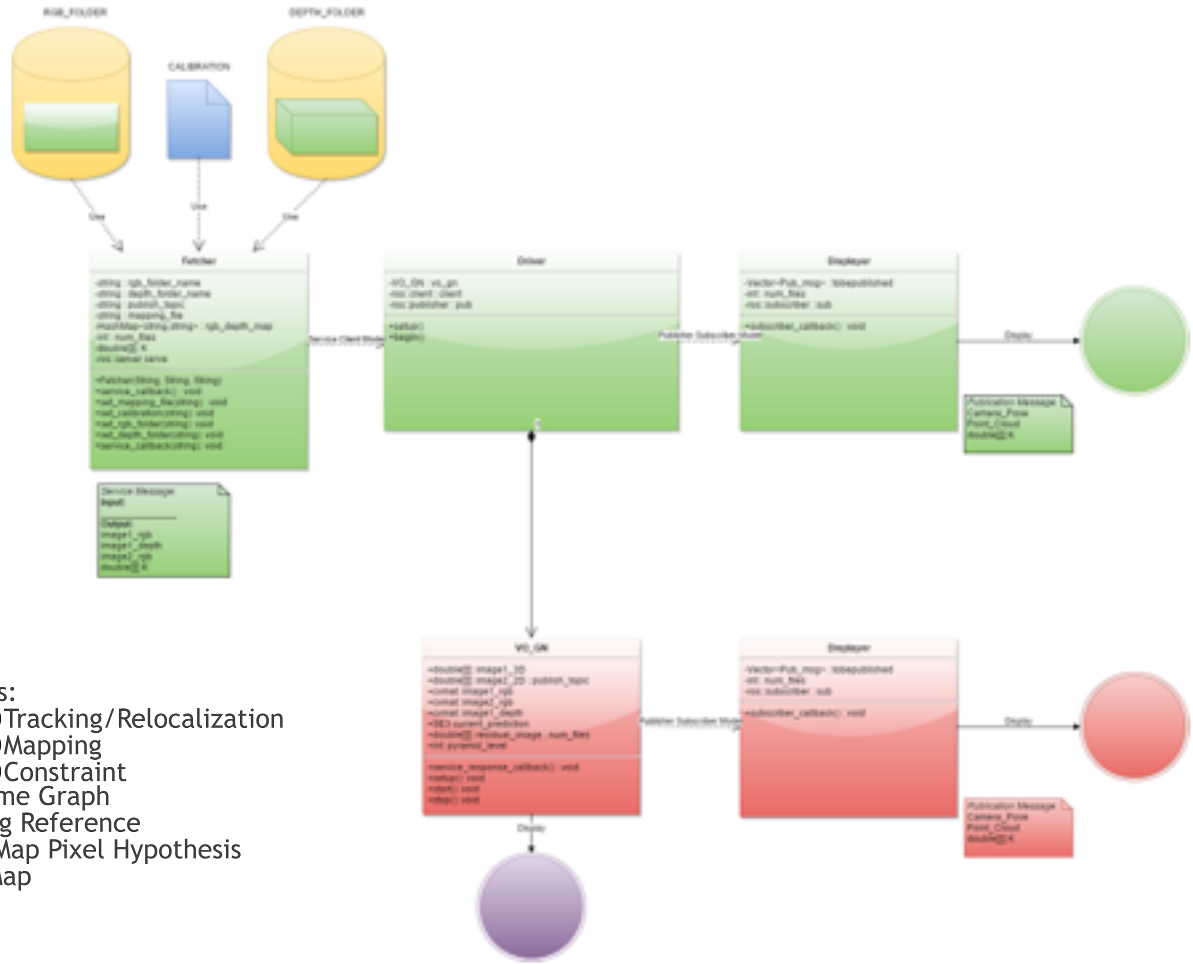
Implementation

- MATLAB



Implementation: LSD SLAM

- ROS/C++



- Extends to:
  - Threads:
    - Tracking/Relocalization
    - Mapping
    - Constraint
  - KeyFrame Graph
  - Tracking Reference
  - Depth Map Pixel Hypothesis
  - DepthMap

Results

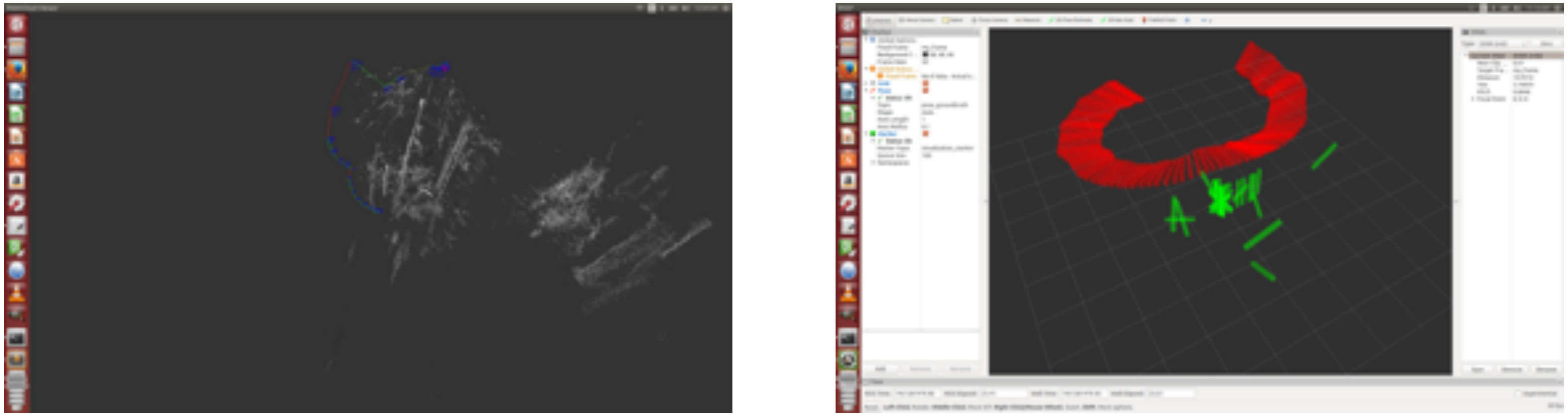


Fig: Around a Cupboard (TUM Dataset)

Results

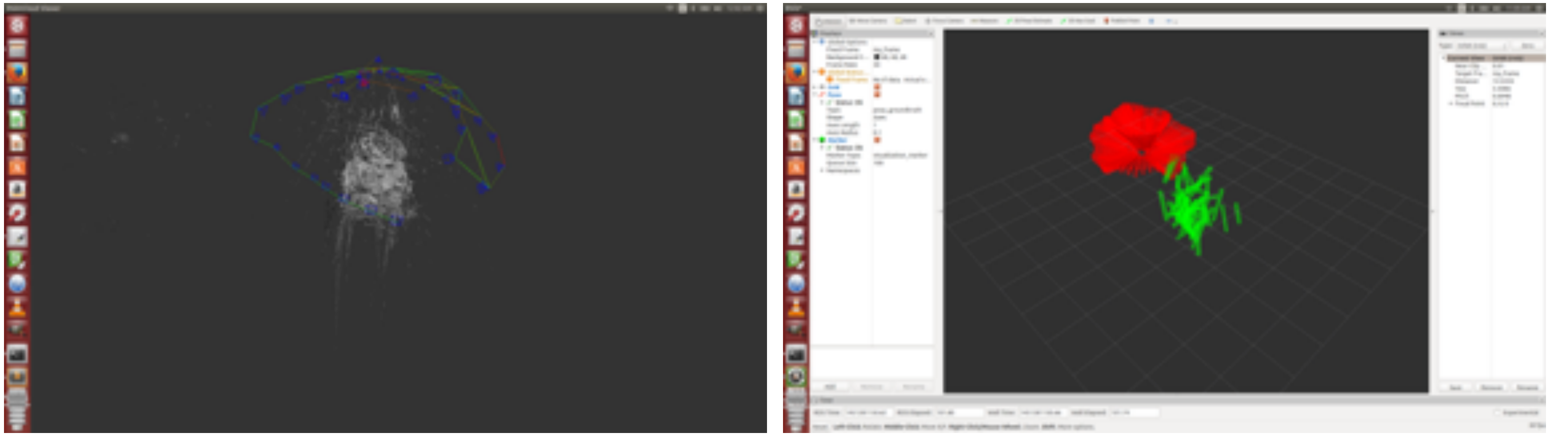
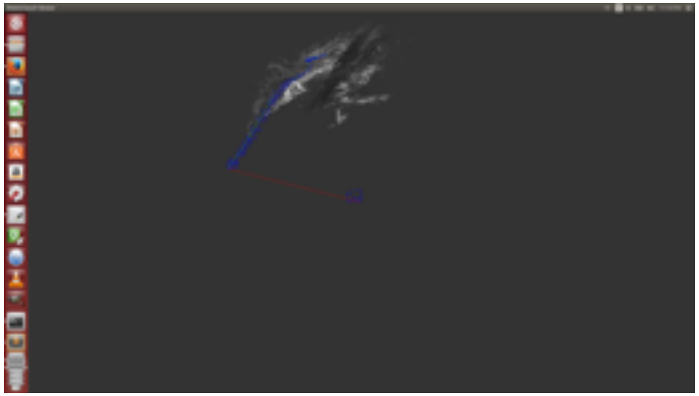


Fig: Teddy Bear (TUM Dataset)

Results



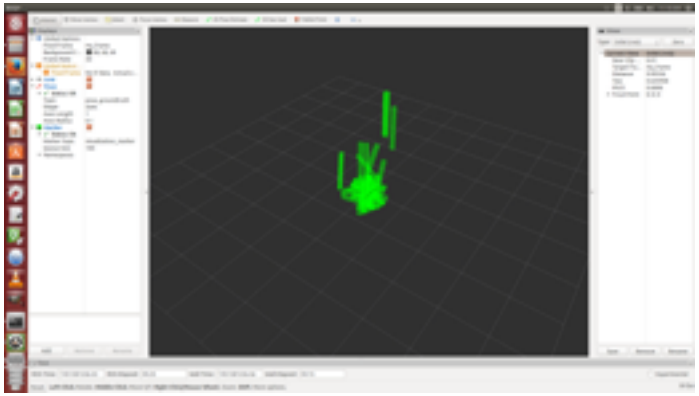


Fig: E-Rickshaw IIIT-Delhi

References

- C. Kerl, J. Sturm, and D. Cremers. Robust odometry estimation for rgb-d cameras. May 2013
- J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. September 2014
- Please refer to the report for other references.

Thank You!

APPENDIX

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY
DELHI

Theory: Prerequisites

- Gauss-Newton Optimization

$$S(\mathbf{x}) = (\mathbf{f}(\mathbf{x}) - \mathbf{z})^{\mathsf{T}}(\mathbf{f}(\mathbf{x}) - \mathbf{z}) = |\mathbf{f}(\mathbf{x}) - \mathbf{z}|^2$$

$$\mathbf{x} \leftarrow \mathbf{x} + \boldsymbol{\delta}$$

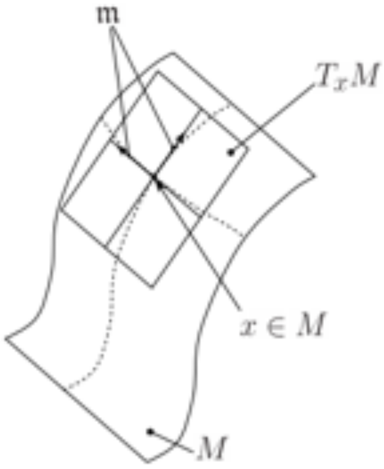
$$\left.\frac{\partial S(\mathbf{x} + \boldsymbol{\delta})}{\partial \boldsymbol{\delta}}\right|_{\boldsymbol{\delta}=0} = 0$$

Theory: Prerequisites

- Semi-Depth
  - Instead of reducing the images to a sparse set of feature observations however, continuously estimates a semi-dense inverse depth map for the current frame, i.e., a dense depth map covering all image regions with non-negligible gradient.
- Frames, while at the same time greatly reducing computational complexity compared to volumetric methods. The estimated depth map is propagated from frame to frame, and updated with variable-baseline stereo comparisons. We explicitly use prior knowledge about a pixel's depth to select a suitable reference frame on a per-pixel basis, and to limit the disparity search range.
- First, a subset of pixels is selected for which the accuracy of a disparity search is sufficiently large. For this we use three intuitive and very efficiently computable criteria:
  - photometric disparity error: depending on the magnitude of the image gradient along the epipolar line,
  - geometric disparity error: depending on the angle between the image gradient and the epipolar line
- For each selected pixel, we then individually select a suitable reference frame, and perform a one-dimensional disparity search.

Theory: Prerequisites

- **More on Lie Groups and Lie Algebra**
  - A group G is a structure consisting of a finite or infinite set of elements plus some binary operation (the group operation), which for any two group elements A, B ∈ G is denoted as the multiplication AB. A group is said to be a group under some given operation if it fulfills the following conditions:
    - Closure
    - Associativity
    - Identity
    - Inverse
- An N-dimensional manifold M is a topological space where every point p ∈ M is endowed with local Euclidean structure.
- A D-dimensional manifold M embedded in R N (with N ≥ D) has associated an N-dimensional tangent space for every point p ∈ M. This space is denoted as TxM ts has a dimensionality of D (identical to that of the manifold).
- Comes handy in calculating the Jacobian:



$$\begin{aligned}
\frac{\partial \mathbf{y}}{\partial \mathbf{R}} &= \frac{\partial}{\partial \omega}\Big|_{\omega=0} (\exp(\omega) \cdot \mathbf{R}) \cdot \mathbf{x} \\
&= \frac{\partial}{\partial \omega}\Big|_{\omega=0} \exp(\omega) \cdot (\mathbf{R} \cdot \mathbf{x}) \\
&= \frac{\partial}{\partial \omega}\Big|_{\omega=0} \exp(\omega) \cdot \mathbf{y} \\
&= (\ G_1\mathbf{y} \mid G_2\mathbf{y} \mid G_3\mathbf{y}\ ) \\
&= -\mathbf{y}_\times
\end{aligned}$$

Theory: Pre-Requisites

- **More on Lie Groups and Lie Algebra**
- Exponentiation and Logarithm:

$$\begin{aligned}\exp(\boldsymbol{\omega}_\times) &\equiv \exp\begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \\ &= \mathbf{I} + \boldsymbol{\omega}_\times + \frac{1}{2!}\boldsymbol{\omega}_\times^2 + \frac{1}{3!}\boldsymbol{\omega}_\times^3 + \cdots\end{aligned}$$

$$\exp(\boldsymbol{\omega}_\times) = \mathbf{I} + \sum_{i=0}^{\infty}\left[\frac{\boldsymbol{\omega}_\times^{2i+1}}{(2i+1)!} + \frac{\boldsymbol{\omega}_\times^{2i+2}}{(2i+2)!}\right]$$

$$\boldsymbol{\omega}_\times^3 = -\left(\boldsymbol{\omega}^T\boldsymbol{\omega}\right)\cdot\boldsymbol{\omega}_\times$$

$$\begin{aligned}\theta^2 &\equiv \boldsymbol{\omega}^T\boldsymbol{\omega} \\ \boldsymbol{\omega}_\times^{2i+1} &= (-1)^i\theta^{2i}\boldsymbol{\omega}_\times \\ \boldsymbol{\omega}_\times^{2i+2} &= (-1)^i\theta^{2i}\boldsymbol{\omega}_\times^2\end{aligned}$$

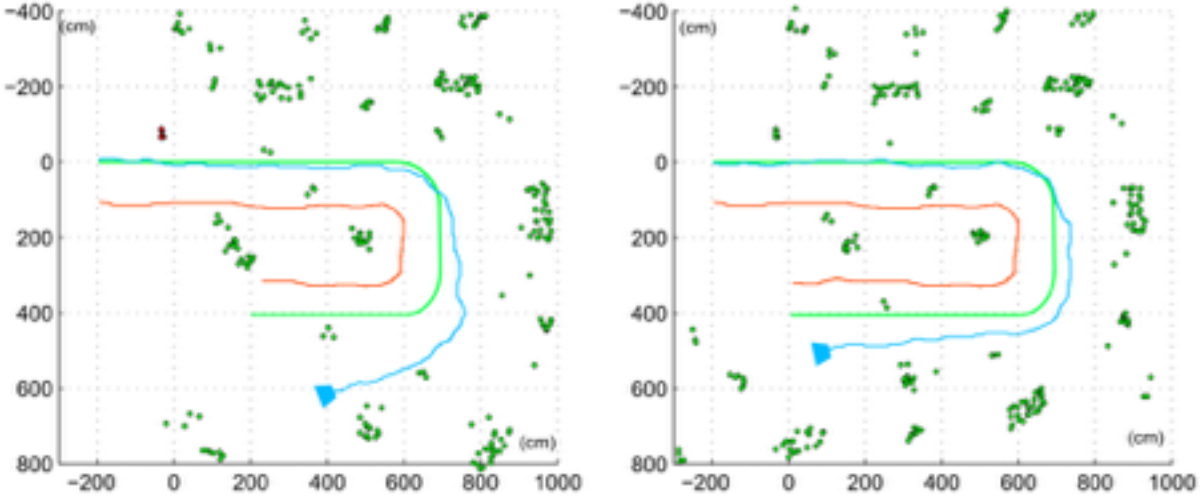
$$\begin{aligned}\exp(\boldsymbol{\omega}_\times) &= \mathbf{I} + \left(\sum_{i=0}^{\infty}\frac{(-1)^i\theta^{2i}}{(2i+1)!}\right)\boldsymbol{\omega}_\times + \left(\sum_{i=0}^{\infty}\frac{(-1)^i\theta^{2i}}{(2i+2)!}\right)\boldsymbol{\omega}_\times^2 \\ &= \mathbf{I} + \left(1 - \frac{\theta^2}{3!} + \frac{\theta^4}{5!} + \cdots\right)\boldsymbol{\omega}_\times + \left(\frac{1}{2!} - \frac{\theta^2}{4!} + \frac{\theta^4}{6!} + \cdots\right)\boldsymbol{\omega}_\times^2 \\ &= \mathbf{I} + \left(\frac{\sin\theta}{\theta}\right)\boldsymbol{\omega}_\times + \left(\frac{1-\cos\theta}{\theta^2}\right)\boldsymbol{\omega}_\times^2\end{aligned}$$

$$\boxed{\mathbf{R} = \mathbf{I} + (\sin\theta)\mathbf{K} + (1-\cos\theta)\mathbf{K}^2\,.}$$

$$\begin{aligned}\mathbf{R} &\in \mathrm{SO}(3) \\ \theta &= \arccos\left(\frac{\mathrm{tr}(\mathbf{R})-1}{2}\right) \\ \ln(\mathbf{R}) &= \frac{\theta}{2\sin\theta}\cdot\left(\mathbf{R}-\mathbf{R}^T\right)\end{aligned}$$

Theory: Pre-Requisites

- Scale Drift



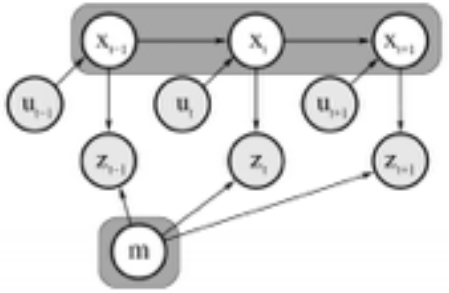
Last Presentation

Introduction to SLAM

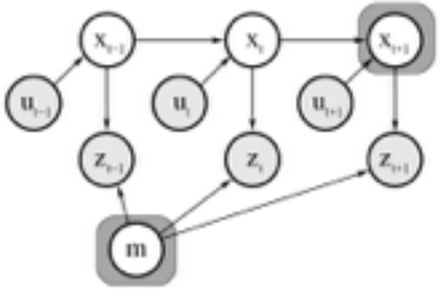
- While online SLAM is only concerned with the most recent position and map, Full SLAM estimates the entire path and map.

Some preliminaries before we move forward:

- x_i: The state vector describing the location and orientation of the observer camera.
- u_i: The control vector, applied at a previous instance of time to bring the camera to its current position. In case of a camera this would be the camera transform.
- m_i: A vector describing the current location of the ith landmark whose true location is assumed to be time invariant.
- z_{ik}: Observations taken from the camera of the landmark m_i at time k.

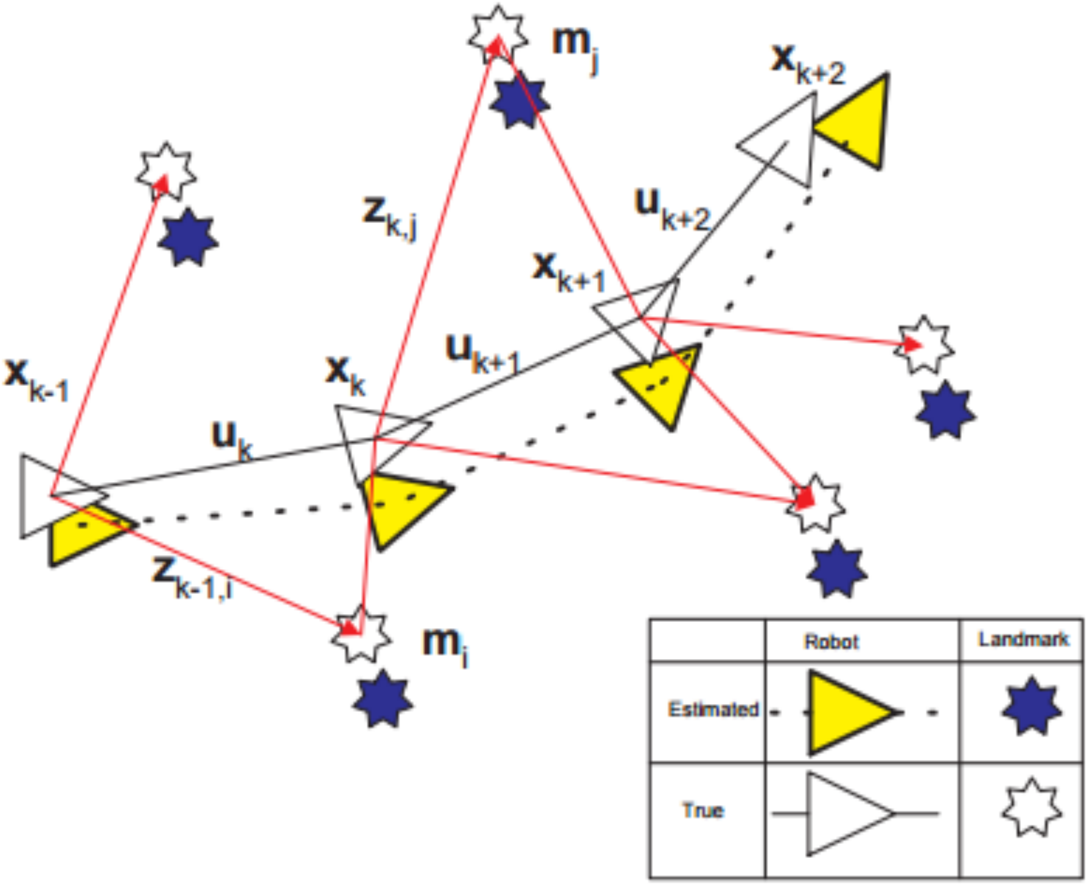


Full SLAM



Online SLAM

Introduction to SLAM



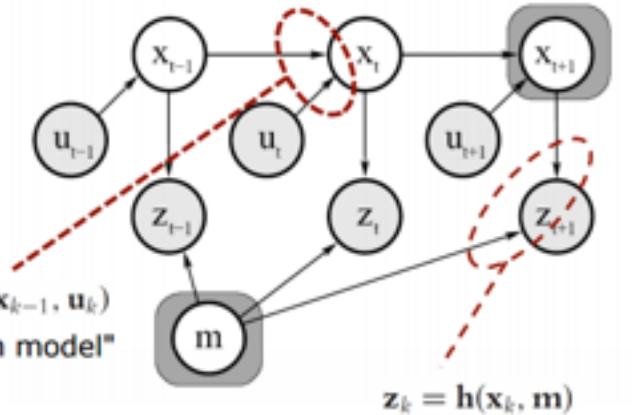
Model Depiction of the SLAM Algorithm

Introduction to SLAM

SLAM Equations:

- Our goal is to find the value of the following equation for all values of k.

$$P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0)$$



$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k)$

"Motion model"

$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{m})$

"Observation model"

Prediction Step:

$$P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0) = \int P(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{u}_k) \times P(\mathbf{x}_{k-1}, \mathbf{m}|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k-1}, \mathbf{x}_0) \mathrm{d}\mathbf{x}_{k-1}$$

Correction Step:

$$P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{x}_0) = \frac{P(\mathbf{z}_k|\mathbf{x}_k, \mathbf{m}) P(\mathbf{x}_k, \mathbf{m}|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{x}_0)}{P(\mathbf{z}_k|\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k})}$$

Camera as a Sensor

- While the conventional approaches to SLAM use lasers and IMU sensors, visual sensors have proved to be equally good if not better.
- The camera proves to be a **very light, very cheap and less power consuming sensor** which again makes it a good choice for detecting egomotion.
- A revisit to fundamental concepts of Computer Vision will allow us to see that there are several methods to capture the camera pose given useful images such as :
  - The calculation of the essential matrix under epipolar geometry
  - Stereo Vision depth estimation and
  - computation of 3D representation of points in SFM.
- When SLAM is performed using a Camera as the primary sensor it is often referred to as Visual SLAM.

Camera as a Sensor

- **Visual Odometry**
- It is a special case of structure from motion where the focus is on recovering the 3-D motion of the camera while in structure from motion it is on just finding the optimal camera poses.
- In visual odometry the consecutive images taken from the camera are used to detect the camera pose.
- This can be done either by **direct image alignment or by identifying feature points** on both images and performing a matching among them followed by localization (structure from motion).
- Generally this is achieved by a least square optimization over the **last 'n' poses of the camera**. (Incremental and Windowed Bundle Adjustment)

Camera as a Sensor

- **Visual SLAM**
- Visual SLAM is different from Visual Odometry as it is concerned about **the global map** while VO is majorly concerned only about the local scene. This often leads to drifting away from actual measurements.
- Apart from that Visual SLAM uses the generated 3D map of features extracted (or in some cases the entire image) to obtain localization.
- In the case of Monocular Visual SLAM there is an added step of having a **bootstrapping or initialization** of the 3D map. This can be done in various ways. One would be to use Stereo Vision for the first couple of frames and another would be to just randomly initialize the map with depths of infinite variance.
- After the initialization step when the next image in the pipeline arrives, the algorithm needs to match the features extracted from this new image and **align it along the current 3D map** to obtain it's pose.
- It is possible that after adding new features to the map, the overall structure of the pose and map would require some update. This is done by methods of **pose graph optimization**.

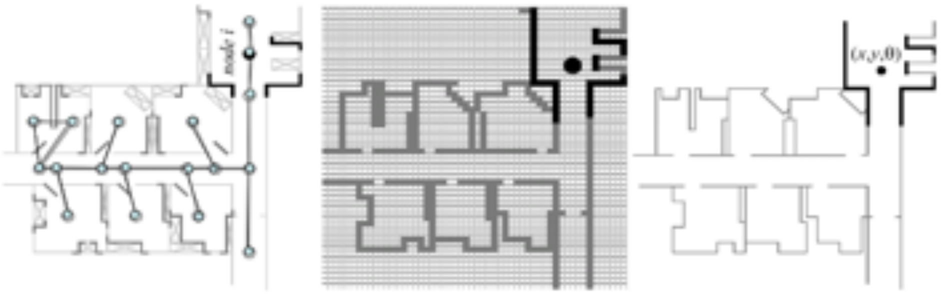
Classification of SLAM Techniques

SLAM Techniques can be broadly classified on the basis of:

- Kind of Map Generated
- Choice of Landmark Extraction
- Technique Used
- Camera Used

Classification of SLAM Techniques

Kind of Map Generated



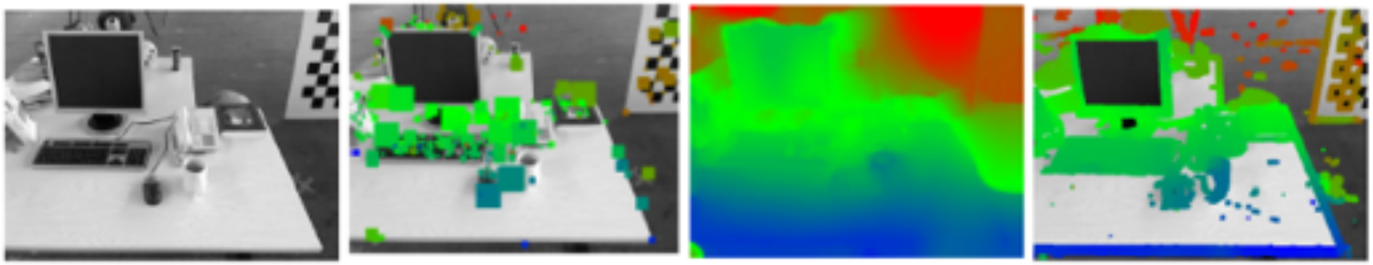
(a) Discrete Topological Map (b) Discrete Metric Map (c) Continuous Metric Map

Broadly there are 3 kinds of map that can be generated:

1. **Discrete Topological Map** – Some information present at each node. Example in the form of visual words.
2. **Continuous Metric Map** – Most generic and most common. Aims for an accurate real map.
3. **Discrete Metric Map** – Used mostly in simulations and in the case of 2D SLAM

Classification of SLAM Techniques

Choice of Landmark Extraction



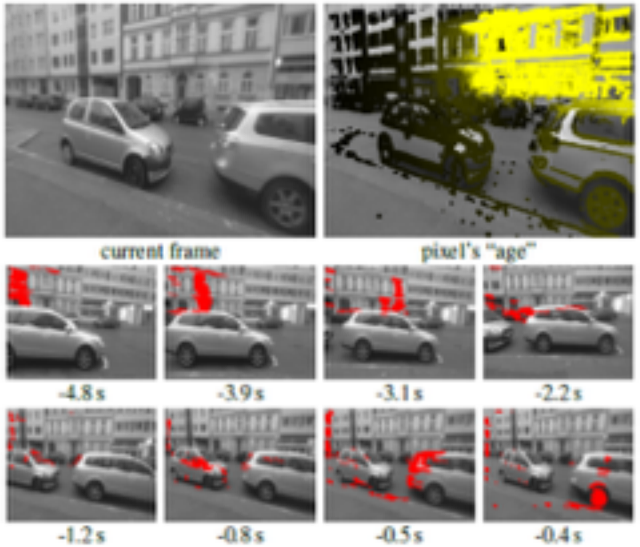
(a) Original Image (b) Feature Map (c) Dense Map (d) Semi Dense Map

Broadly there are 3 categories:

1. Sparse Feature Based – SIFT, SURF, FAST
2. Dense Feature Based – Use the entire image
3. Semi-Dense Feature Based – Only Important parts of the image are used

Classification of SLAM Techniques

- Semi-Dense Features:
- Estimate the Depth of each pixel which has a non negligble gradient
- For each selected pixel, a suitable reference frame is chosen and a one dimensional disparity search is initiated.
- Finally tracking is done by direct image alignment of the inverse depth map of the previous frame. It is based on the direct minimization of the photometric error given by:



(a) For each pixel in the new top left frame, a different stereo-reference frame is selected,based on how long the pixel was visible (older the pixel, higher the age, yellower the color). Red regions were used for stereo comparisons from the reference frame

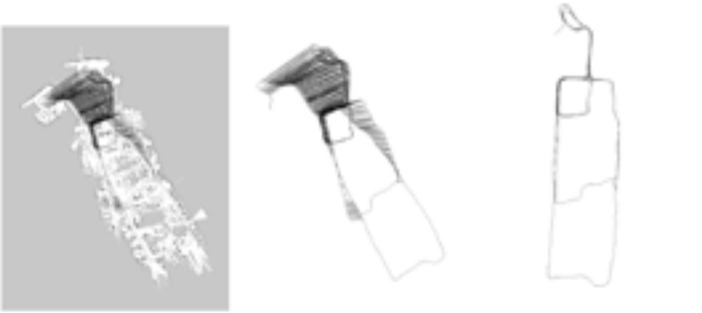
$$r_i(\xi) := (I_2(\omega(x_i, d_i; \xi)) - I_1(x_i))^2$$

Classification of SLAM Techniques

- Comparison:
  - Advantages of feature based SLAM include ease of computation, and ability to be computed in Real time on CPU. Disadvantages of feature based methods are that only information that conforms to the feature type can be used. In the real world, many features like consistent textures, curved edges, etc which these feature are not able to represent are missed out on.
  - Advantages of Dense feature based SLAM is that it does not disregard any important feature as in the previous case and following the concepts of multi-view stereo, it is able to give robust results. Disadvantages include the inability to perform in Real Time in the absence of a state-of-the-art GPU.
  - Semi-Dense feature SLAM beats both!

Classification of SLAM Techniques

Technique Used



(a) Original Image (b) Nodes(Poses) and Edges(Constraints) (c) After Optimization

Broadly there are 3 categories:

1. Kalman Filter Based – Assume Gaussian Noise, Combination of Gaussian Noise
2. Particle Filter Based – Consist of many hypothesis which over time build up to give accurate results
3. Graph or Keyframe Based: Every node in the graph is meant to represent pose of the camera while every edge is meant to correspond to the spatial constraints between them. Our goal at the end comes out find a configuration of the nodes such that the error introduced by the constraints is minimized.

Classification of SLAM Techniques

- To conclude a major drawback of the EKF based SLAM is that it is **relatively slow** when estimating high dimensional maps, because every single feature measurement leads to the update of the feature covariance matrix and the state matrix which starts to cause latency in the system after a certain point.
- An advantage of using particle filter based approach is that it does not make any assumptions about the distribution of the system like the **gaussian assumptions** made in the Kalman Filter. A disadvantage would be the time and computation complexity which makes it virtually impossible for pure SLAM to be done in real time.
- Advantages of Graph based SLAM is that it is very fast, it can be applied to large datasets, works under non-linear conditions and has consistently proved to **give better results** as compared to EKF SLAM and particle filter approaches.

Classification of SLAM Techniques

Camera Used

(a) Monocular Camera (b) Stereo Camera (c) Infra-red Camera (d) OmniDirectional Camera

Broadly there are 4 categories:

1. Monocular Camera -> Scale Ambiguity
2. Stereo Camera
3. RGB-D IR Camera
4. Omni-directional Camera

Classification of SLAM Techniques

- Comparison:
  - It’s Advantages are that it’s independence to scale allows it to be seamlessly be used in both indoor and outdoor environments. This is not the case with Scaled sensors which we shall see further. It’s disadvantage is the difficulty in achieving accuracy and the absence of depth information.
  - It allows for a much easier pose estimation of the camera as the points become known in 3D. It does not face the "scale ambiguity" like the monocular camera. Disadvantage is that even with perfect correspondences, the depth error in traditional stereo grows quadratically with depth, which means that the accuracy in the near range far exceeds that of the far range thus, making it inadequate to scale outdoors.
  - Advantage of IR Depth based system is, again an accurate and computationally inexpensive way to get depth of the image pixels. Disadvantage is the scale dependency of the sensor as it is only able to succeed in the indoor scene.
  - Under the testing conditions Omnidirectional Camera had 3 times less error as compared to a standard monocular camera. The extended field of view reduces the complexity of the problem by reducing the number of robust correspondences needed in different directions.

Prevalent Work

- Discussed are 4 Major works:
  - PTAM – Parallel Tracking and Mapping
  - DTAM – Dense Tracking and Mapping
  - RGBD-VO – RGBD Visual Odometry
  - LSD-SLAM – LSD SLAM

DTAM

- The aim is to estimate by using an energy minimization technique the inverse depth map of a keyframe r over a large number of short baseline frames where the energy is the sum of the photometric data error term.

$$\mathbf{C}_r(\mathbf{u},d) = \frac{1}{|\mathcal{I}(r)|}\sum_{m\in\mathcal{I}(r)} \|\rho_r\left(\mathbf{I}_m,\mathbf{u},d\right)\|_1 ,$$

$$\rho_r\left(\mathbf{I}_m,\mathbf{u},d\right) = \mathbf{I}_r\left(\mathbf{u}\right) - \mathbf{I}_m\left(\pi\left(\mathrm{K}\mathrm{T}_{mr}\pi^{-1}\left(\mathbf{u},d\right)\right)\right)$$

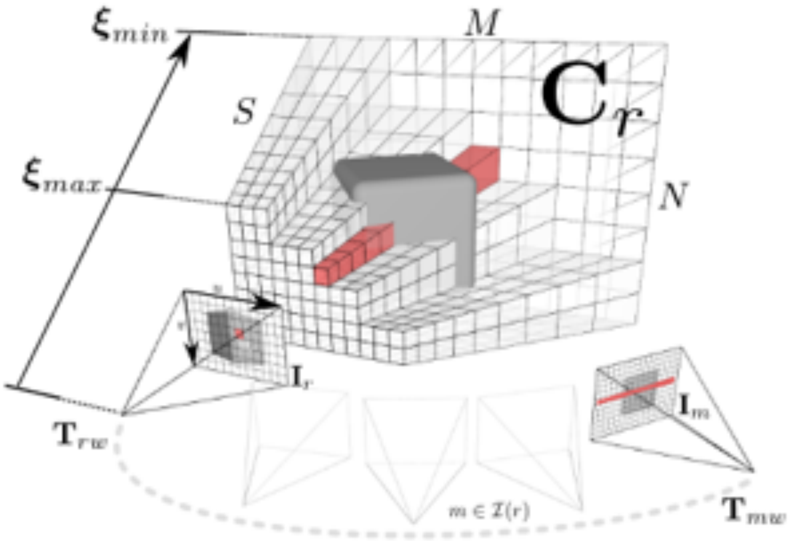
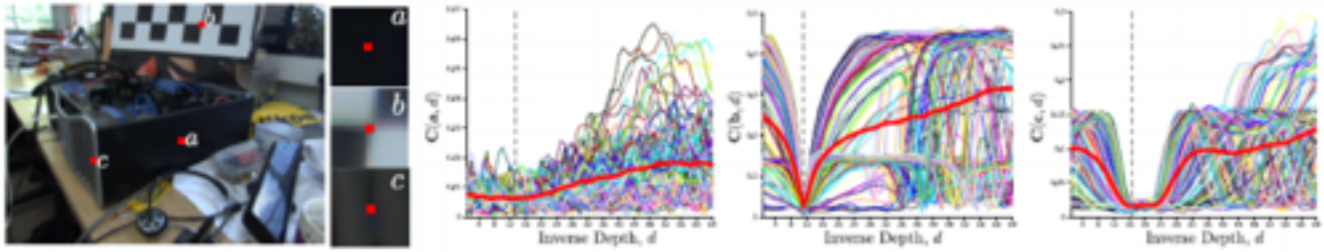


Figure 1. A keyframe r consists of a reference image $\mathbf{I}_r$ with pose T_{rw} and data cost volume $\mathbf{C}_r$. Each pixel of the reference frame $\mathbf{u}_r$ has an associated row of entries $\mathbf{C}_r(\mathbf{u})$ (shown in red) that store the average photometric error or cost $\mathbf{C}_r(\mathbf{u}, d)$ computed for each inverse depth $d \in \mathcal{D}$ in the inverse depth range $\mathcal{D} = [\xi_{min}, \xi_{max}]$. We use tens to hundreds of video frames indexed as $m \in \mathcal{I}(r)$, where $\mathcal{I}(r)$ is the set of frames nearby and overlapping r, to compute the values stored in the cost volume.

DTAM



(a) Plots for the single pixel photometric functions p(u) and the resulting total data cost row C(u) are shown for three example pixels in the reference frame, chosen in 3 different regions. (a) being texture less does not give significant location information; (b) being strongly textured is a region where a point feature might be detected; and (c) is in a region of linear repeating texture. As can be seen, while the individual costs i.e. C(U), might exhibit many local minima, the total cost shows clearly a clear minimum in the images

- Feature less points do not give good depth estimates which is corrected by adding a regularizer term.
- The drawback of this approach is that it requires state-of-the-art **GPUs** and does not work in real time on a regular CPU nor can it be extended to a longer duration of video. The **constraints on the depth range** are not well defined which does not allow it to be scalable to outdoor larger scenes.

Depth map without regularizer term

PTAM

- Parallel Tracking and Mapping was a novel approach towards improving SLAM by parallelising the Tracking and Mapping functions.
- Uses Large number of points in the map- FAST corners points.
- Doesn't update the map every frame. Introduced the idea of key frames.
- Splits the tracking and mapping into two different threads.
- Uses pyramid levels for tracking and mapping.
- Complete Map is optimized with bundle adjustment on addition of a keyframe.
- It is also completely functional on a **parallelizable CPU**. The drawbacks of PTAM were the need of human intervention to initialize the map, inability to **generate dense maps, and limited accuracy to only indoor environments**.

RGBD-VO

- This novel approach uses the depth data from an IR sensor along with RGB data to perform direct motion estimation.
- This approach uses the residual of two consecutive images to estimate the pose of the camera. The residual image is defined as the difference or brightness between the first and second warped image.

$$r_i(\boldsymbol{\xi}) := I_2(\tau(\boldsymbol{\xi}, \mathbf{x}_i)) - I_1(\mathbf{x}_i).$$

- Following this it is assumed that the residual of each pixel is independent of the other pixels.

$$p(\mathbf{r} \mid \boldsymbol{\xi}) = \prod_i p(r_i \mid \boldsymbol{\xi}).$$

- The authors then derive a probabilistic definition of the camera parameters with the aim of maximizing the posterior as follows:

$$p(\boldsymbol{\xi} \mid \mathbf{r}) = \frac{p(\mathbf{r} \mid \boldsymbol{\xi}) p(\boldsymbol{\xi})}{p(\mathbf{r})}. \qquad \arg\max_{\boldsymbol{\xi}} p(\boldsymbol{\xi} \mid \mathbf{r}).$$

- On taking a log of the equation and equating its derivative to zero we get

$$\sum_i \frac{\partial \log p(r_i \mid \boldsymbol{\xi})}{\partial \boldsymbol{\xi}} = \sum_i \frac{\partial \log p(r_i)}{\partial r_i} \frac{\partial r_i}{\partial \boldsymbol{\xi}} = 0.$$

RGBD-VO

- On following the substitution $w(r_i) = \partial \log p(r_i) / \partial r_i \cdot 1/r_i$ we obtain $\arg\min_{\xi} \sum_i w(r_i)(r_i(\xi))^2$
- In order to find the best estimate of the camera parameters the above equation is linearized using Gauss-Newton Optimization.
- The motion prior term which was assumed to be a constant in the above equations can be replaced by a motion prior model to increase the accuracy provided the motion of the camera is known or can be predicted via a sensor like IMU.
- Besides being reasonably accurate the algorithm runs on a CPU. The only drawback would be the use of an RGB-D sensor which restricts the scalability of the algorithm to an indoor scene.

LSD-SLAM

- Large Scale Direct Monocular SLAM[18] is a SLAM approach that makes significant contribution to improving the performance of SLAM Algorithms in large scale conditions.

EgoVision

- The rapid development in wearable camera systems and devices have created encouraging conditions for the use of computer vision and multimedia technologies to augment experience in all aspects of human life.
- These systems are exploited for collecting and analyzing a variety of features, in our case primarily videos. These devices have shown great potential, as can be seen in the research work they have encouraged in the domain of egocentric vision. Established and solved computer vision problems can be extended to this new field of view.
- By taking advantage of the first-person point-of-view paradigm, there have been a number of advances in the domain which is being seen as an entirely new and growing domain in Computer Vision. Some of the recent research work in the domain includes:
- Temporal Segmentation
- Social Interactions
- Video Summarization
- Activity Recognition