Visual-Inertial Simultaneous Localization and Mapping with Multiple Cameras

Speaker: Pak Hong, Chui
Supervisors: Dr. Vladyslav Usenko,
Dr. Torsten Enßlin

Outline

1. Motivation
2. Introduction to VIO and VI-Mapping
3. Flow of proposed VI-SLAM
4. Extend to multi cameras

Advantages of using IMU

	Accurate when
IMU measurements	Small time interval
Triangulation (visual)	Large time interval (Large baseline)

The advantages of combining visual and IMU measurement are,

1. combine the strengths of both sensors
2. keep the map gravity aligned

VI-Odometry and VI-Mapping

	Feature type	Loop detection
VI-Odometry (online)	Optical Flow	No, tend to drift over time
VI-Mapping (offine)	Orb-feature	yes

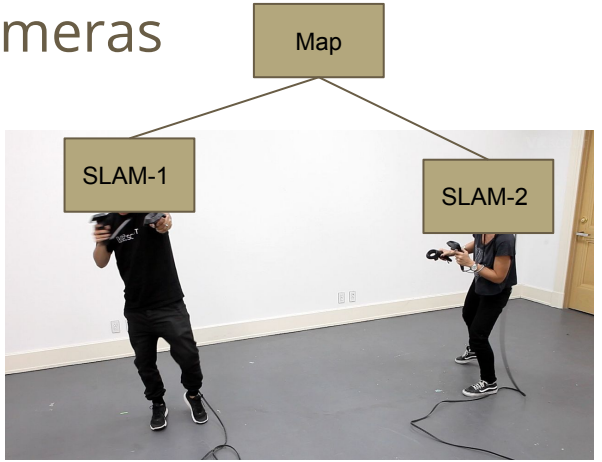
VI-SLAM ~ VI-Odometry + VI-Mapping

Objectives

- try to build a VI-SLAM
- capable of working with multiple cameras

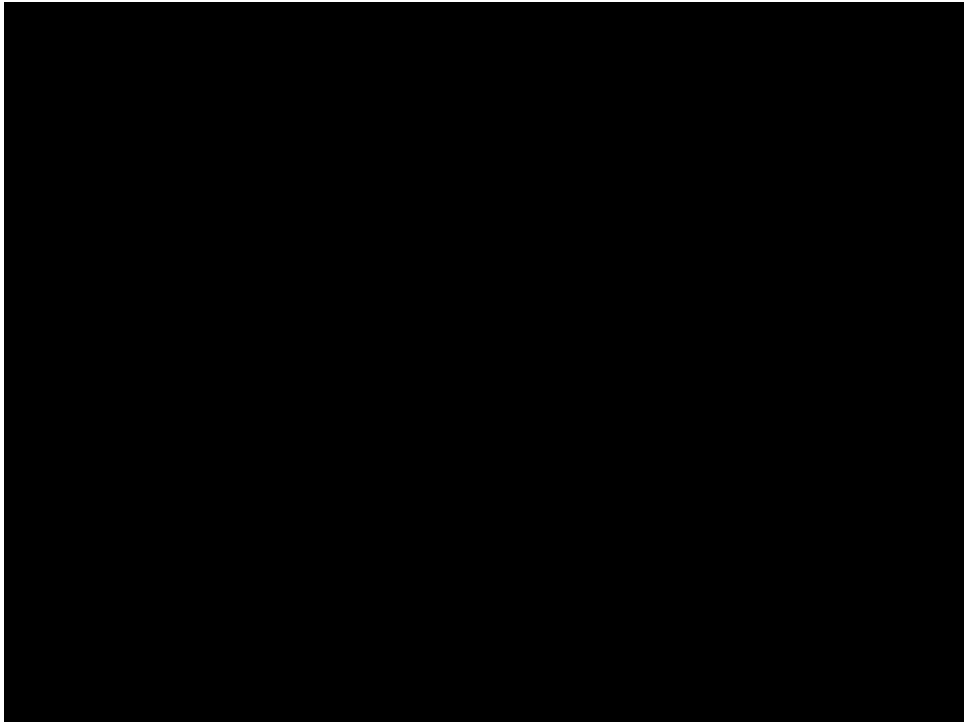
Use case:

for example, multi-player VR gaming etc...



source:
https://vrscout.com/news/cant-stop-playing-pool-nation-vr/

Demo of proposed method

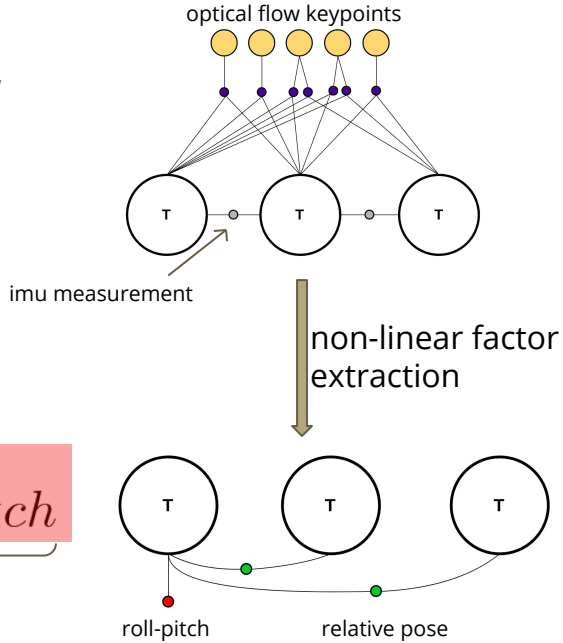


How the mapper reuse information from VIO?

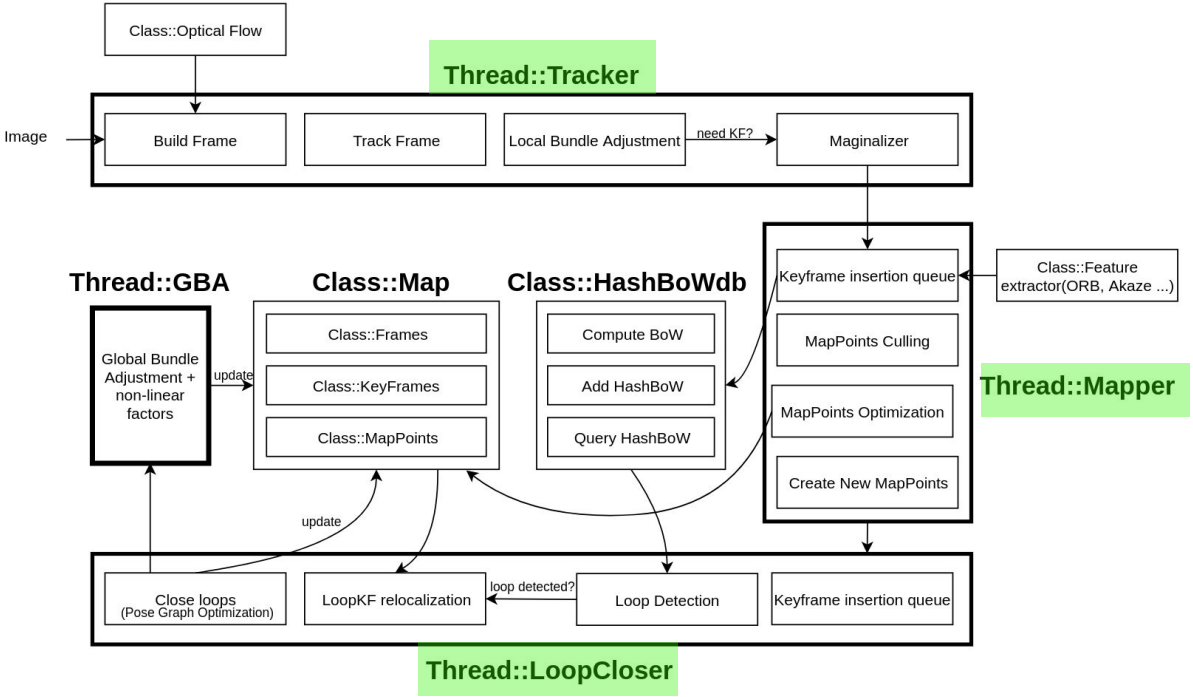
Turn optical flow kp and imu measurement to non-linear factors

- just need to store a few data for each frame in VIO
- smaller optimization problem in mapping

$$E = E^{orb}_{reproj} + \underbrace{E_{rel\ pose} + E_{roll\ pitch}}_{\text{from non-linear factors}}$$



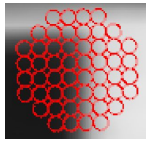
Structure of proposed VI-SLAM



Structure of proposed VI-SLAM (simplified)

Tracker

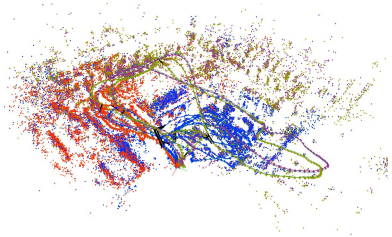
- track frame by Basalt-VIO



Credit: V. Usenko, N. Demmel, D. Schubert, J. Stckler, and D. Cremers, 2020

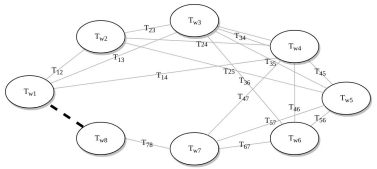
Mapper

- build and update map bases on non-linear factor (faster)



Loop closer

- detect and close loop by using hash-bow

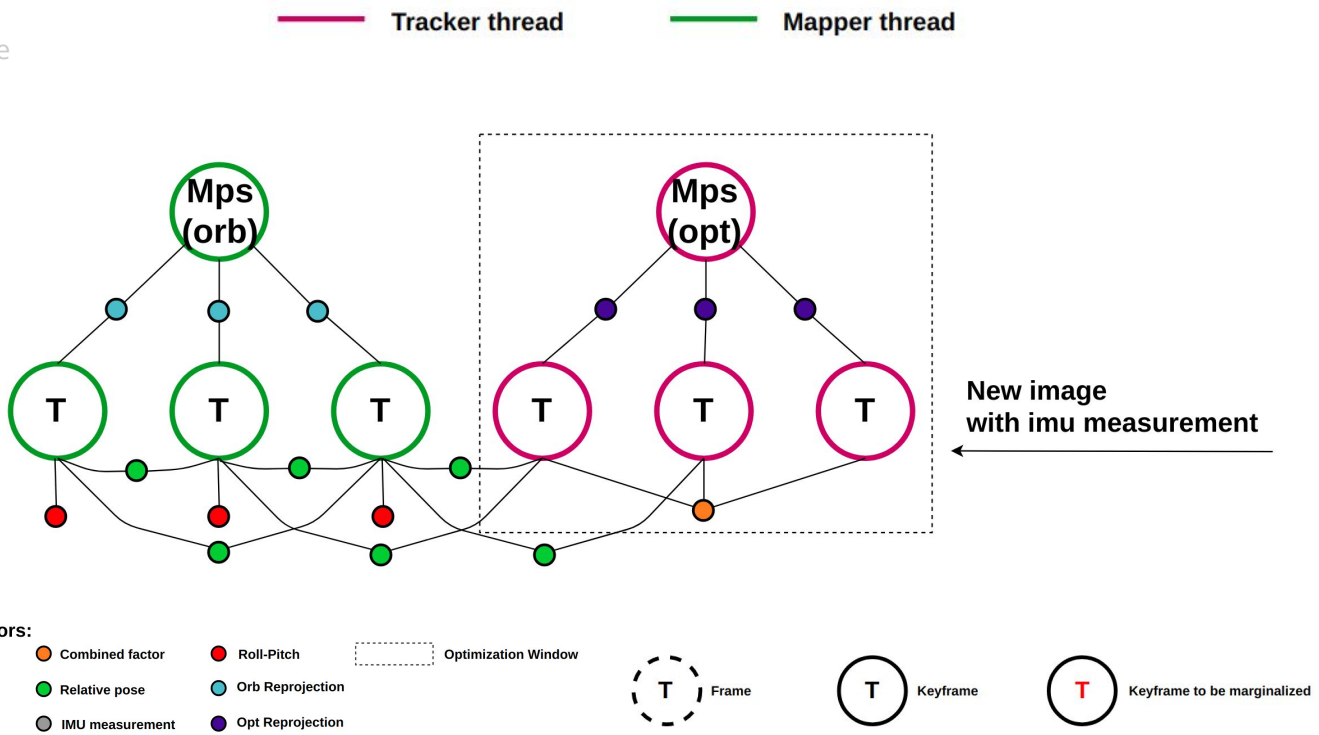


Flow of proposed VI-SLAM

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapp
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps corresponder
M3. detect loop
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2

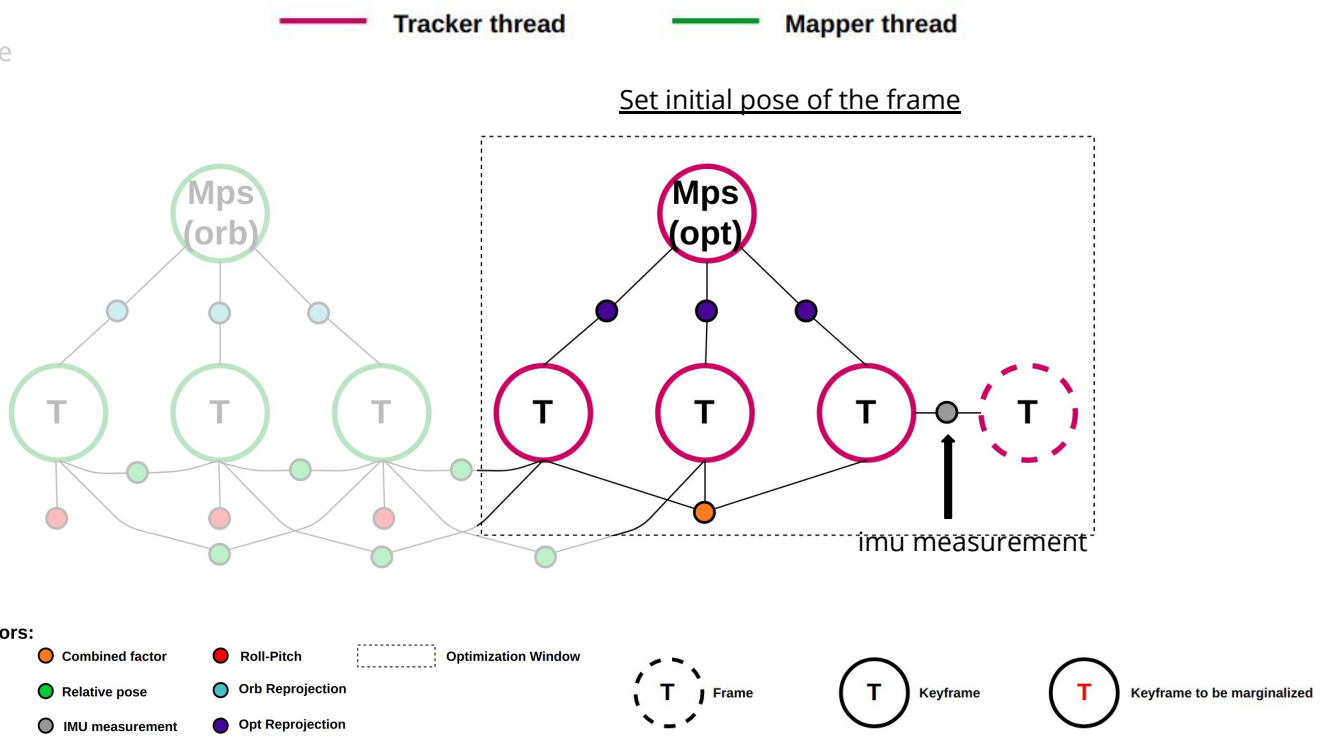


Flow of proposed VI-SLAM

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to map
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps corresponde
M3. detect loop
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2

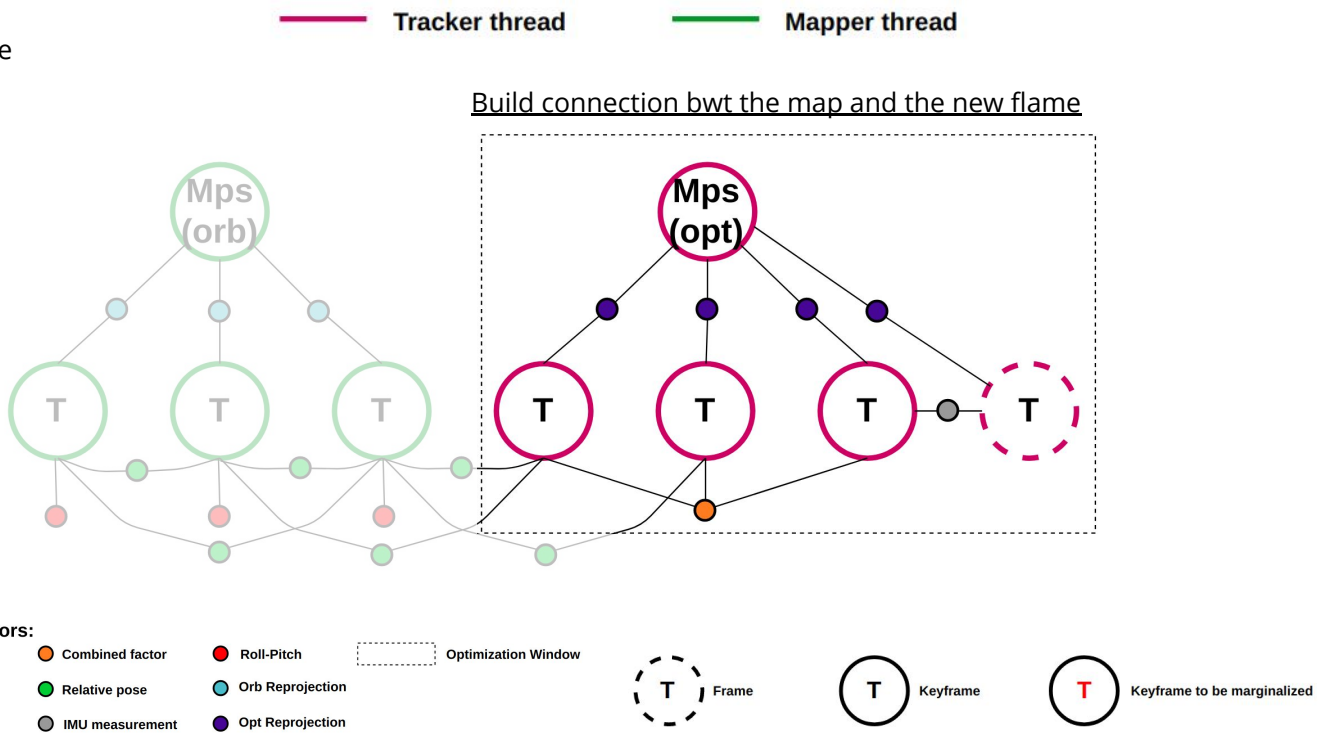


Flow of proposed VI-SLAM

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapp
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondenc
M3. detect loop
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2

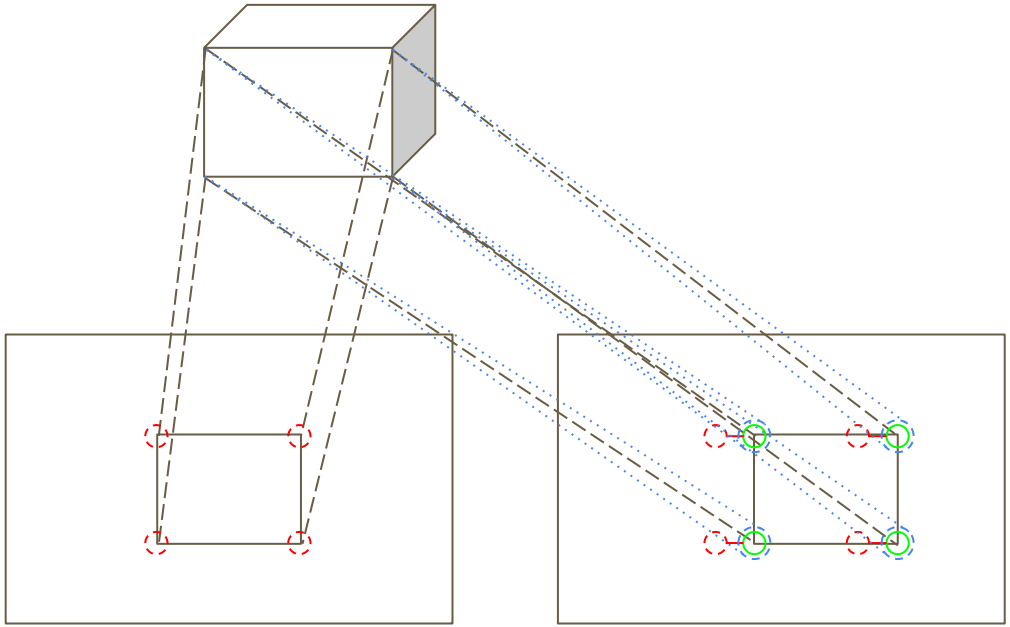


Track optical flow keypoints by re-projection

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapper
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondence
M3. detect loop
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2



Time t-1

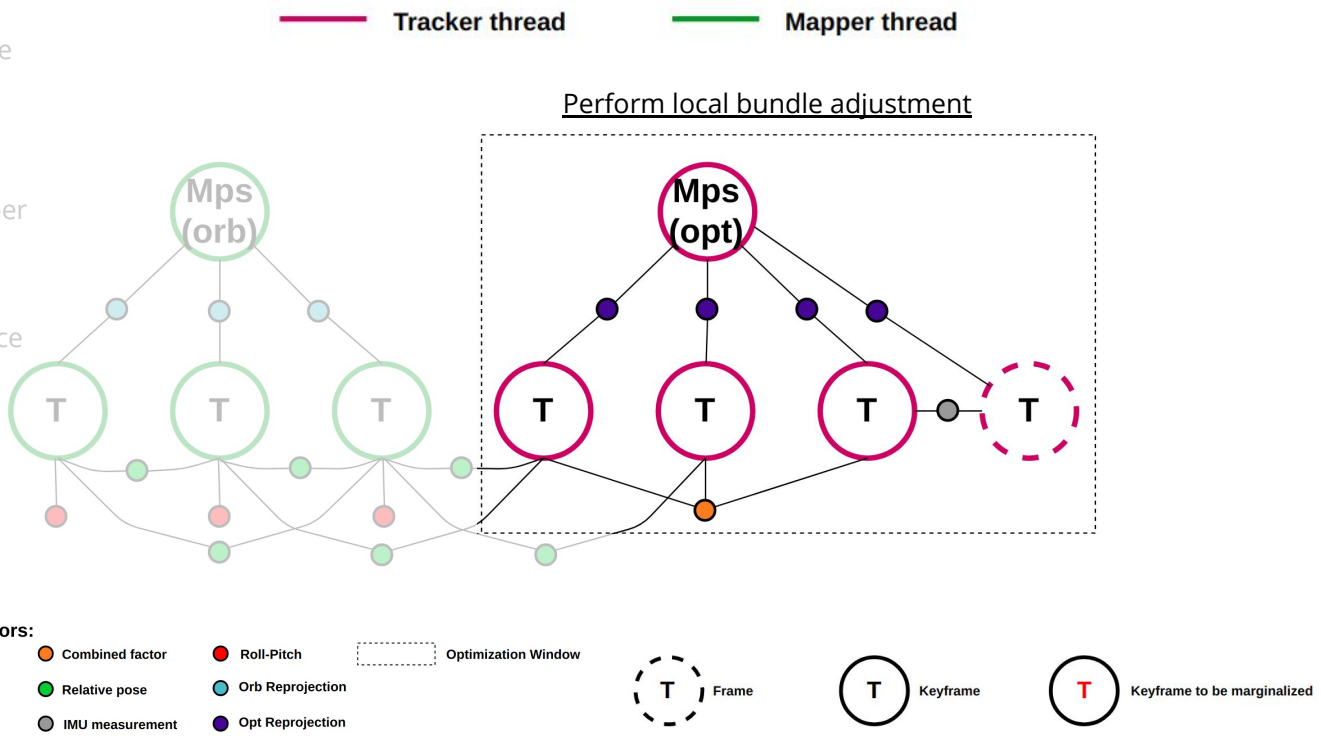
Time t

Flow of proposed VI-SLAM

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapper
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondence
M3. detect loop
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2

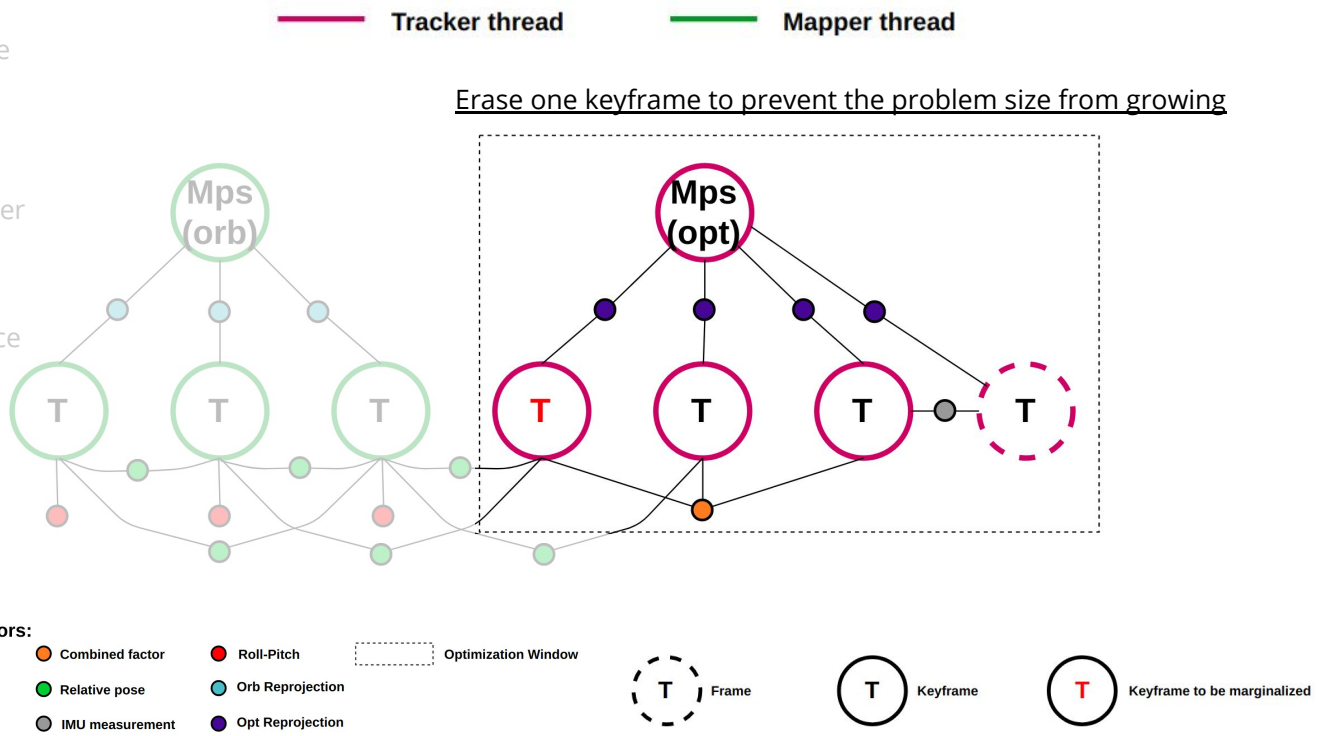


Flow of proposed VI-SLAM

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapper
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondence
M3. detect loop
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2

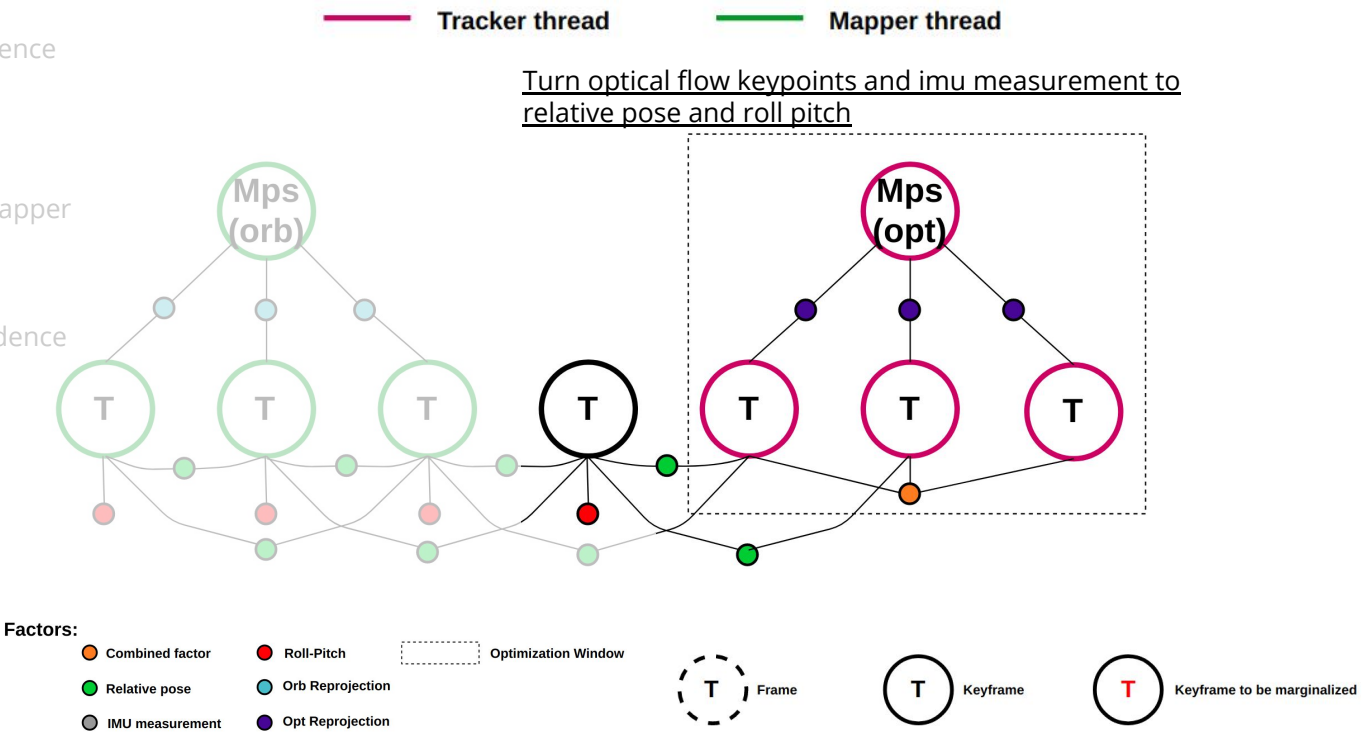


Flow of proposed VI-SLAM

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapper
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondence
M3. detect loop
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2

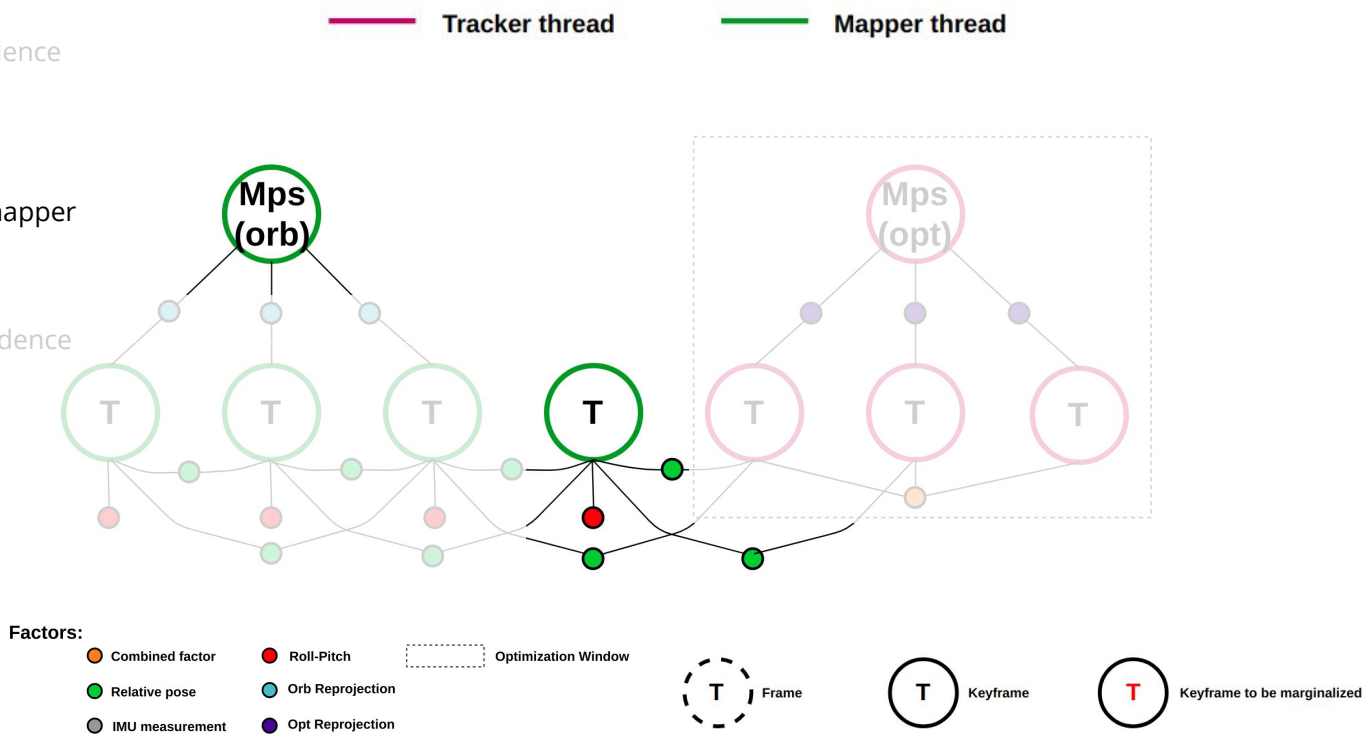


Flow of proposed VI-SLAM

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapper
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondence
M3. detect loop
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2

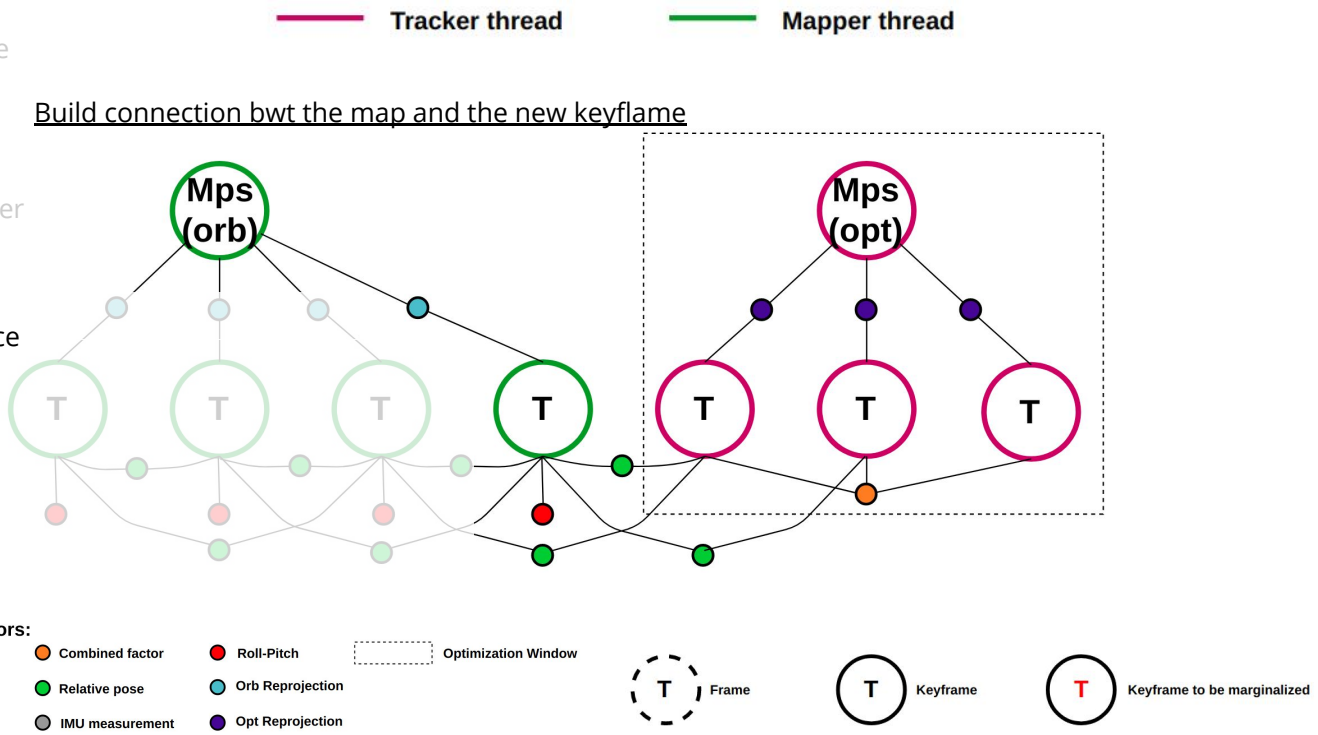


Flow of proposed VI-SLAM

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapper
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondence
M3. detect loop
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2

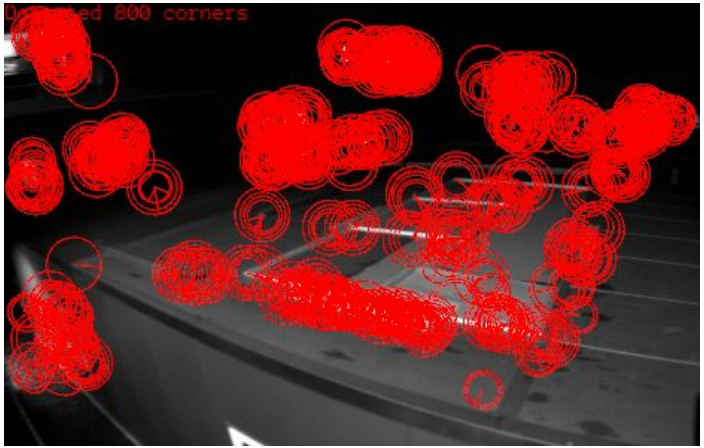


Scaled features and Non-maximal suppression

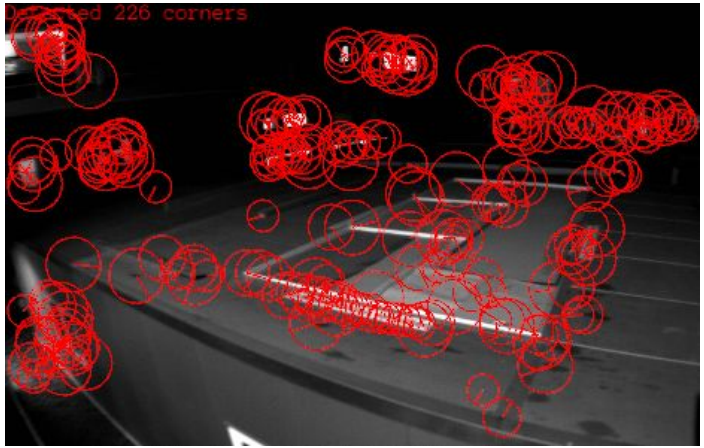
T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapper
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondence
M3. insert kf to loop closer
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2



without non-maximal suppression



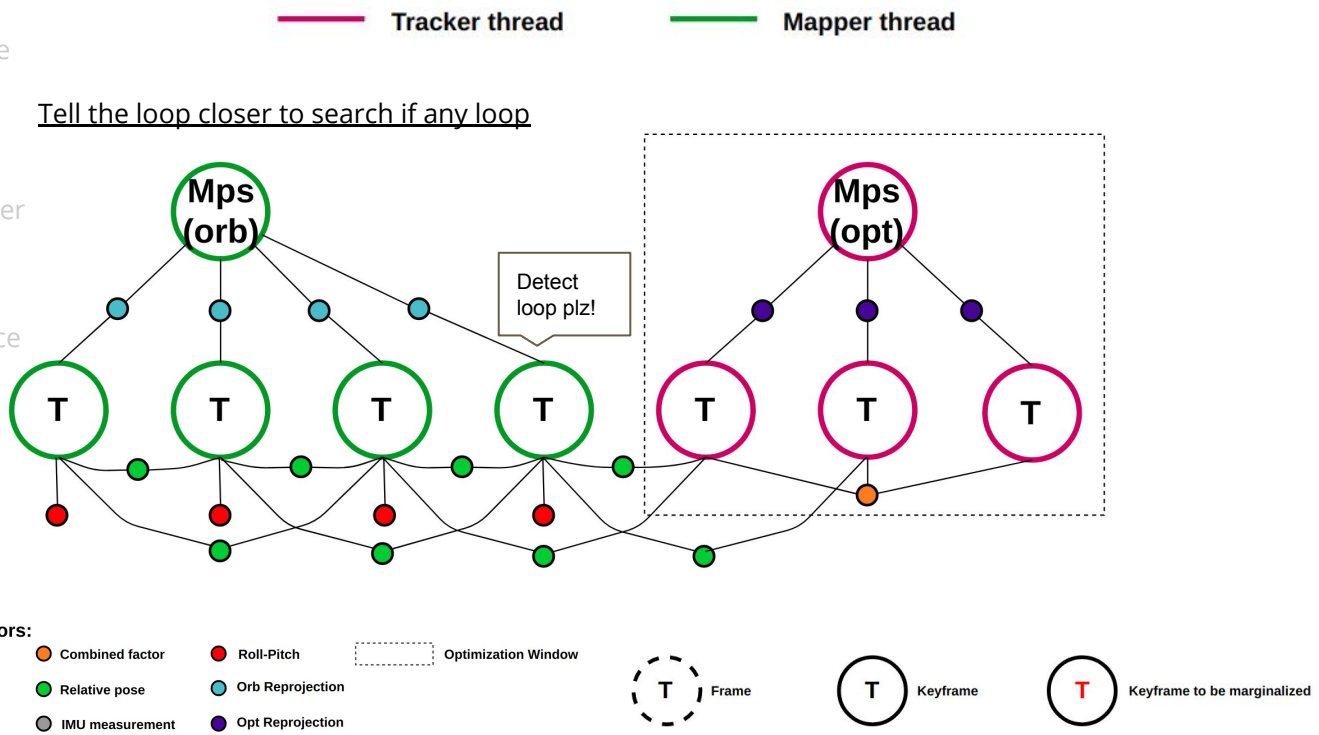
with non-maximal suppression

Flow of proposed VI-SLAM

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapper
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondence
M3. detect loop
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2

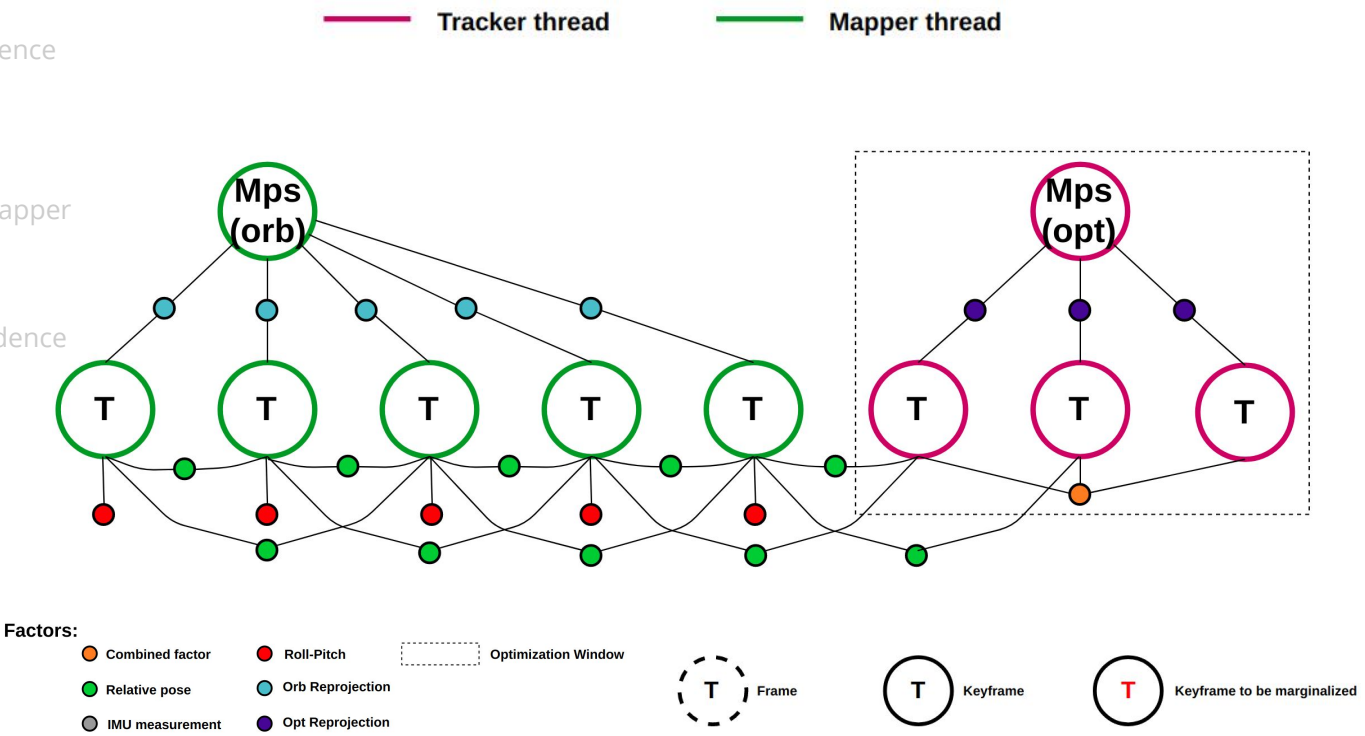


Flow of proposed VI-SLAM

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapper
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondence
M3. detect loop
M4. repeat step M1-M3

L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2

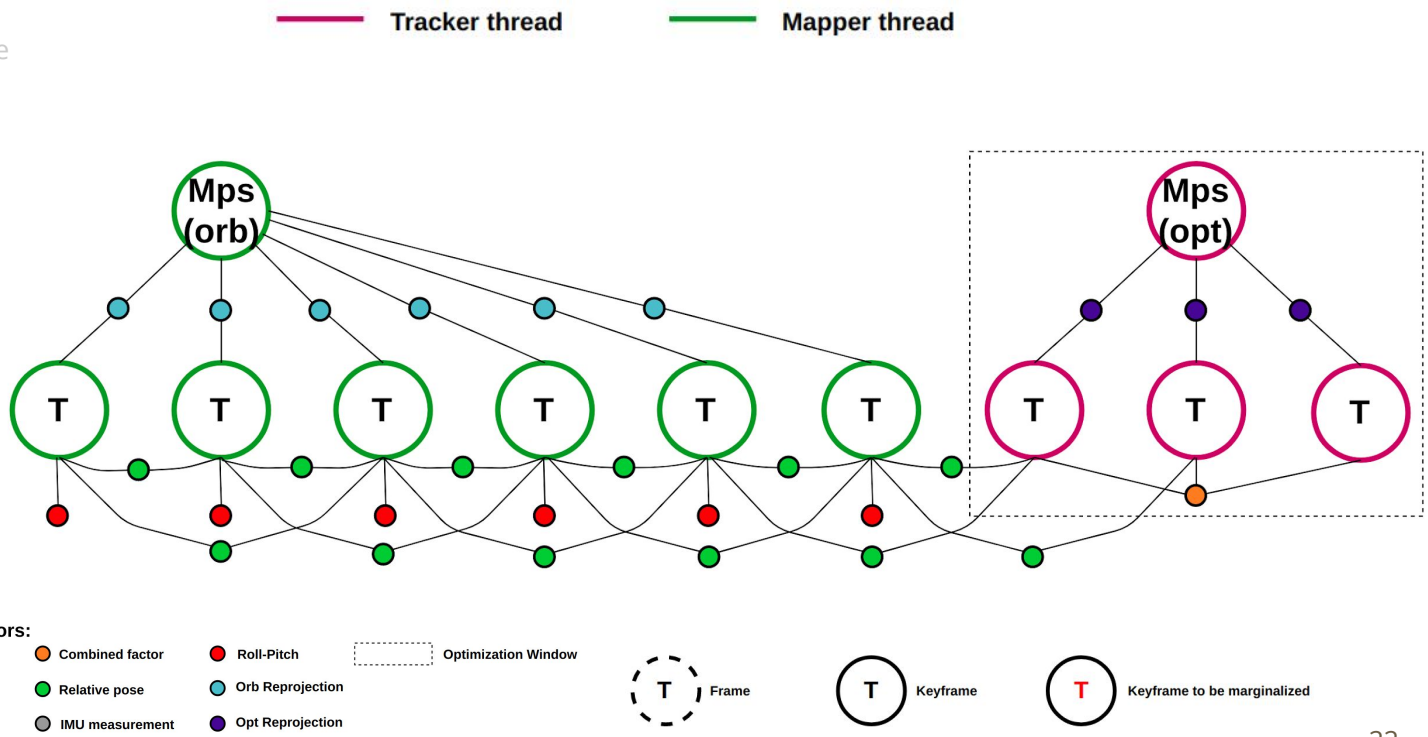


Flow of proposed VI-SLAM

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapp
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondenc
M3. detect loop
M4. repeat step M1-M3

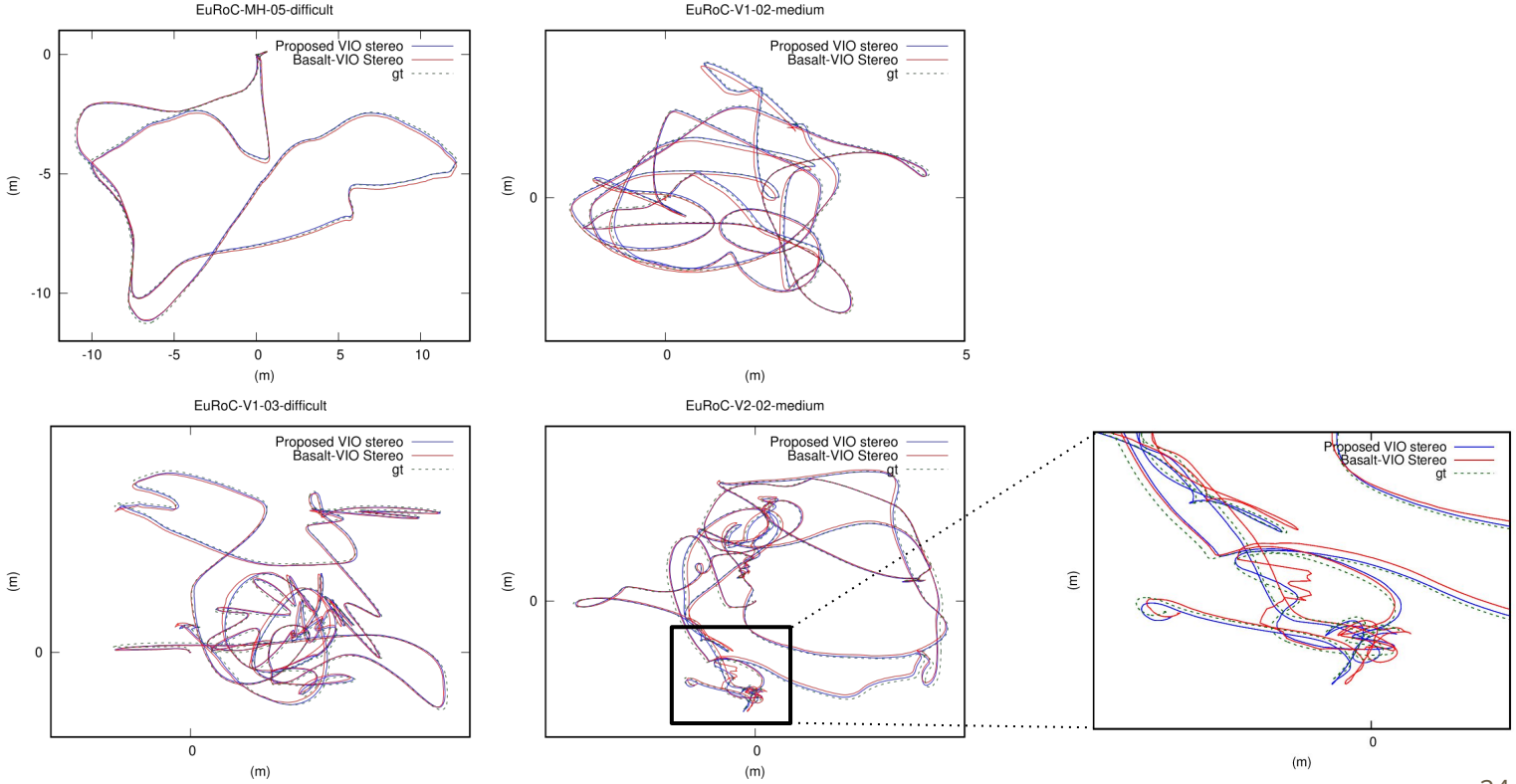
L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2



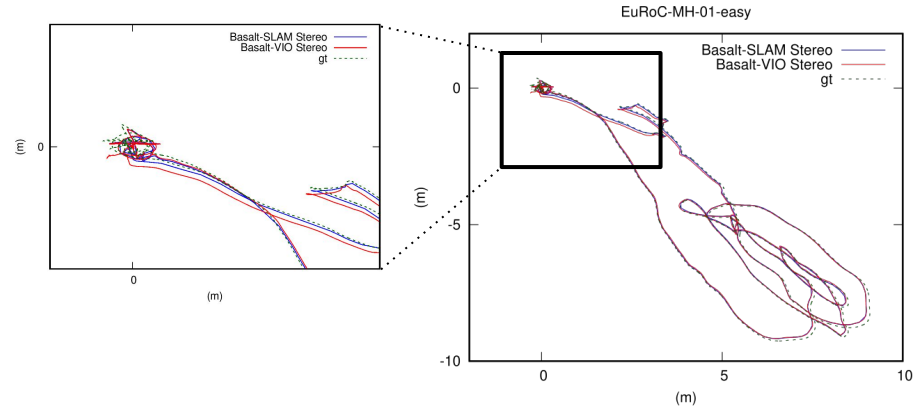
Quantitative results

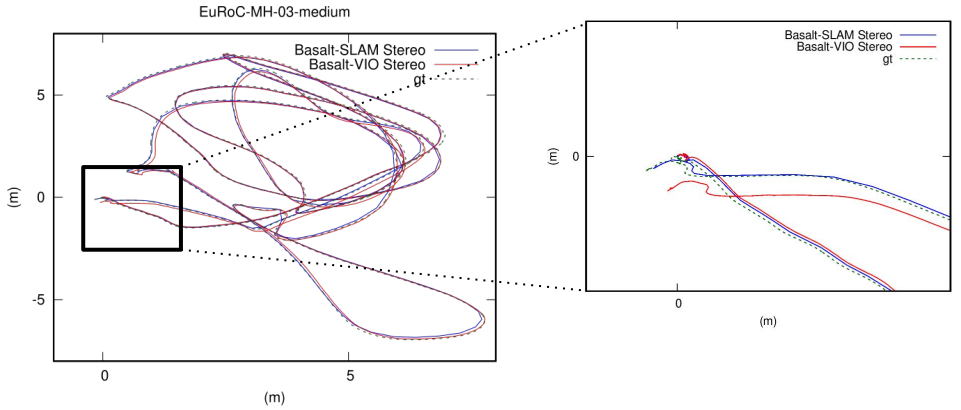
Sequence	MH_01	MH_02	MH_03	MH_04	MH_05	V1_01	V1_02	V1_03	V2_01	V2_02
Basalt VIO,stereo [33]	**0.09**	**0.05**	0.09	**0.11**	0.11	**0.04**	0.06	0.07	**0.04**	0.06
Proposed VIO,stereo	**0.09**	0.06	**0.08**	**0.11**	**0.08**	**0.04**	**0.04**	**0.05**	0.05	**0.04**
Basalt VI Mapping,stereo, KF [33]	0.08	0.06	**0.05**	0.11	**0.09**	**0.04**	**0.03**	**0.03**	**0.03**	**0.02**
Proposed SLAM (PGO),stereo, KF	0.08	**0.05**	**0.05**	**0.09**	**0.09**	0.05	0.04	0.05	0.05	0.05
Proposed SLAM (GBA),stereo, KF	**0.07**	**0.05**	**0.05**	**0.09**	**0.09**	**0.04**	**0.03**	0.05	0.04	0.04

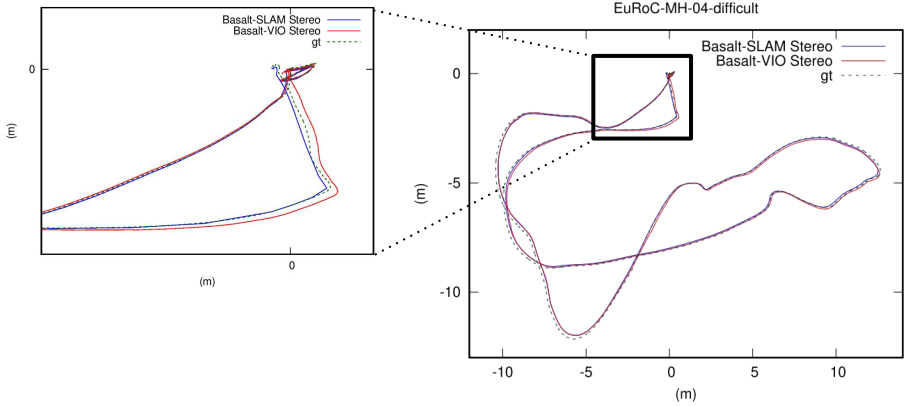
Improve lost tracking

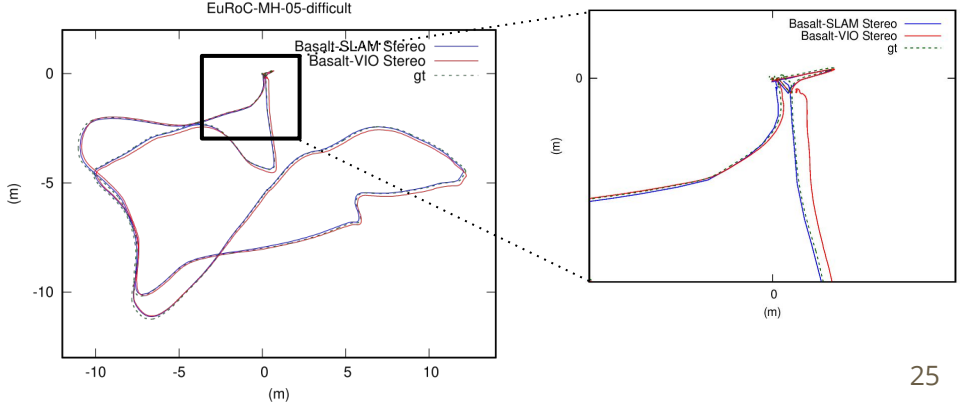


Drift correction

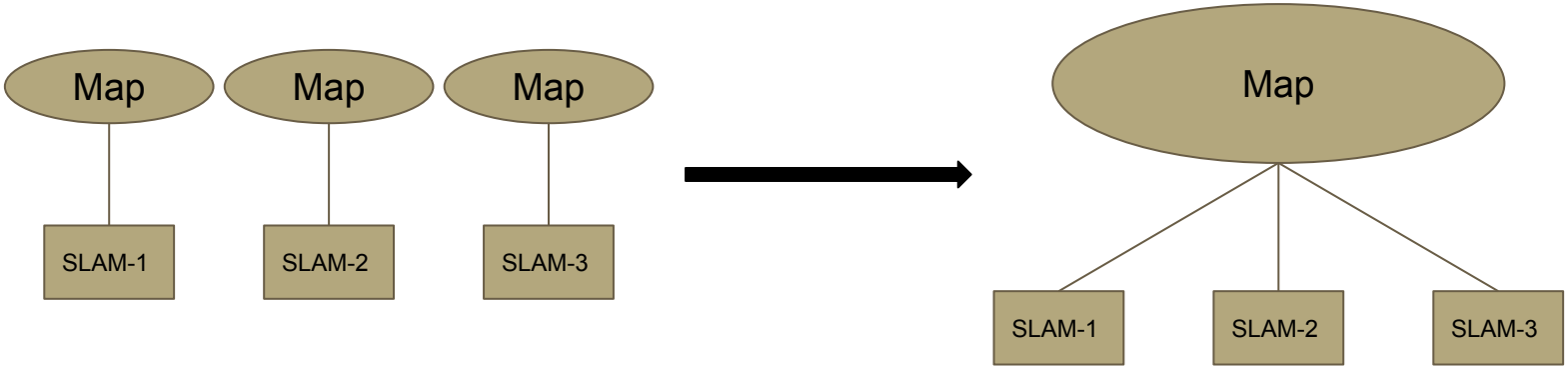




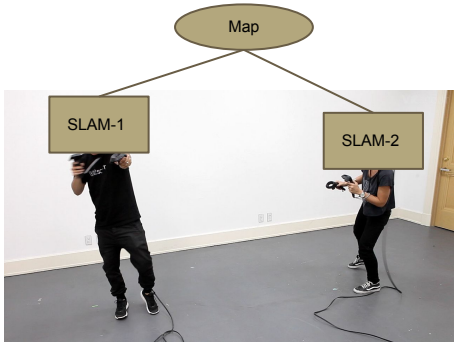




Structure of multi camera slam

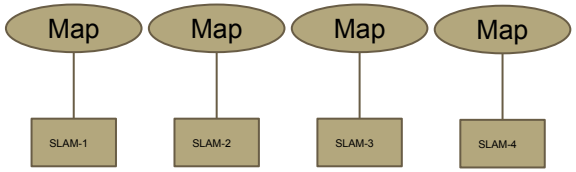


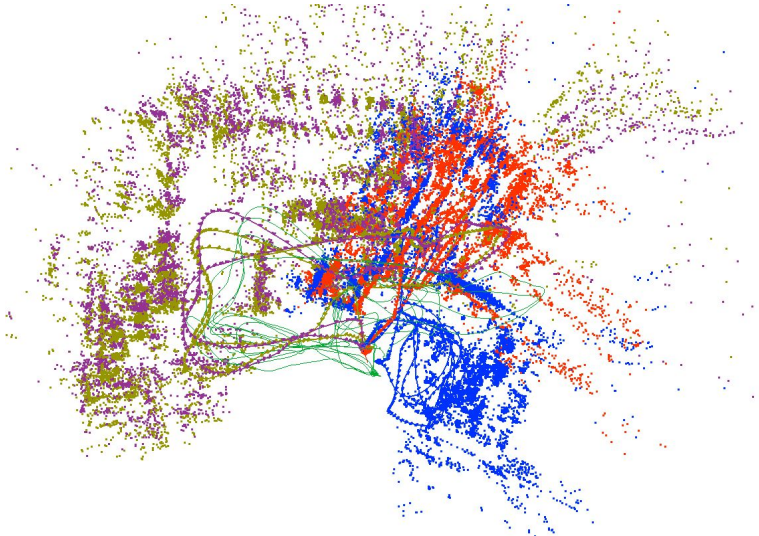
example:



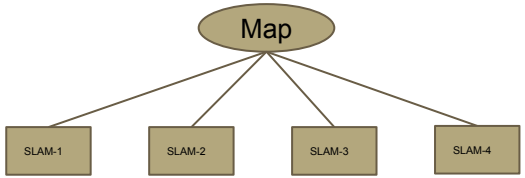
source: https://vrscout.com/news/cant-stop-playing-pool-nation-vr/

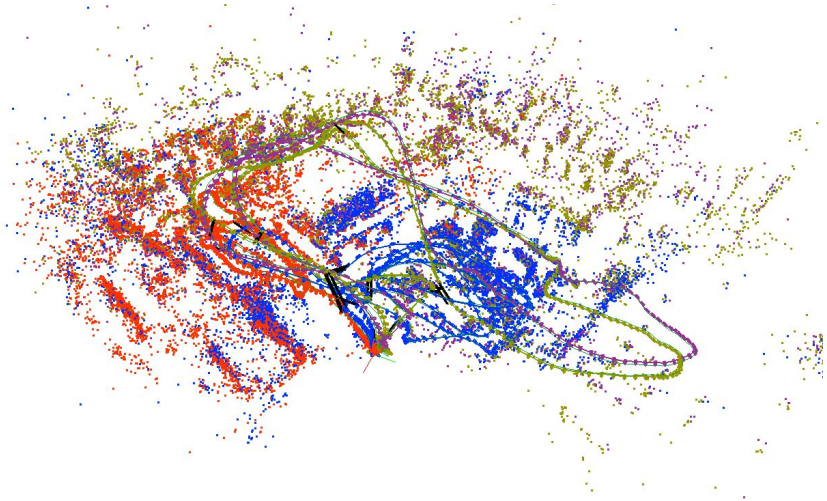
Qualitative results





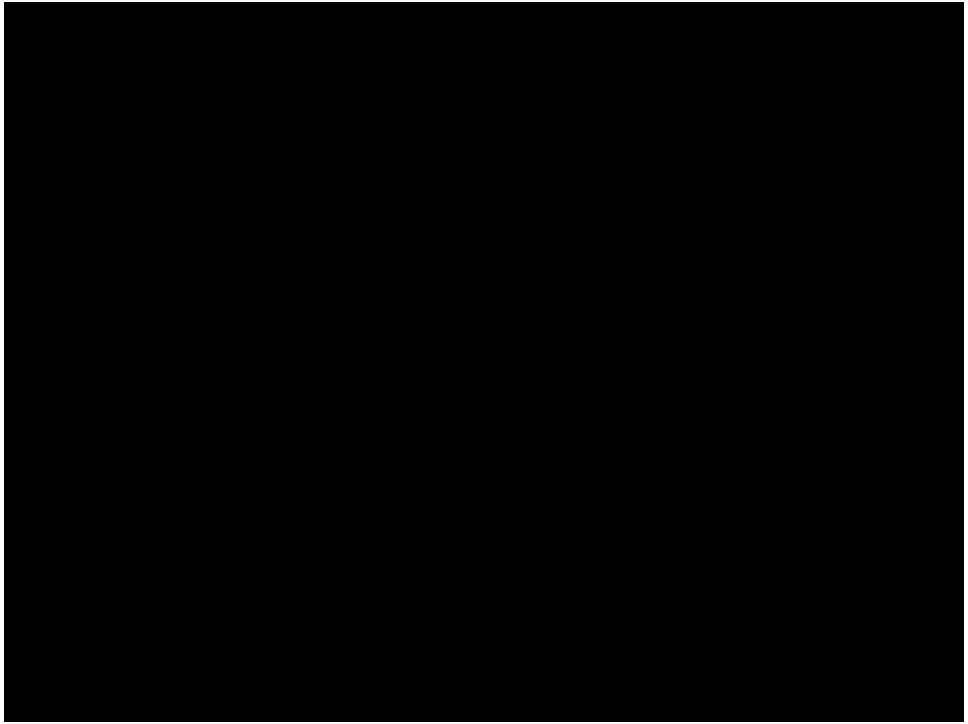
4 SLAM systems with 4 maps





4 SLAM systems with 1 map

Demo of proposed method



End

Thank you !!!

Demo

1 camera Basalt-SLAM (1 camera)

2 cameras Basalt-SLAM (2 cameras)

3 cameras Basalt-SLAM (3 cameras) 4x

4 cameras Basalt-SLAM (4 cameras) 4x

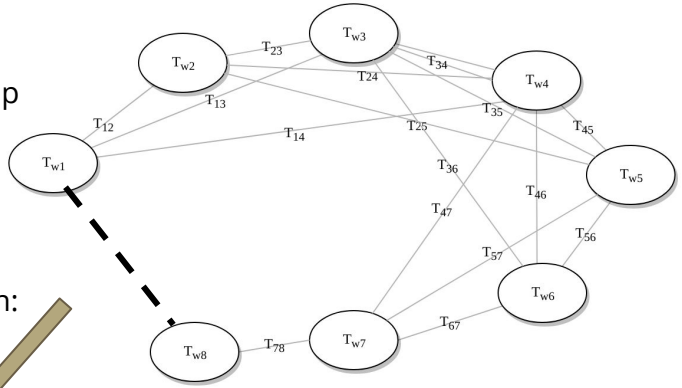
Appendix: Merge loop

T1. input data
T2. set ini pose
T3. search opt kps correspondence
T4. local BA within window
T5a. frame to kf
T5b. select a kf to marginalize
T6. extract non-linear factor
T7. insert marginalized kf to mapper
T8. repeat step T1-T7

M1. receive a kf from tracker
M2. search orb kps correspondence
M3. insert kf to loop closer
M4. repeat step M1-M3

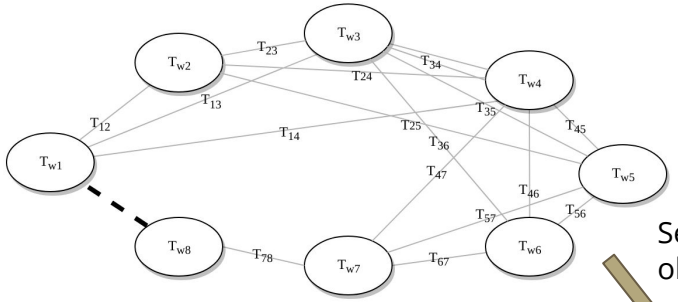
L1. receive a kf from mapper
L2. detect and merge loops
L3. repeat step L1-L2

Use Hash-bow to find potential loop

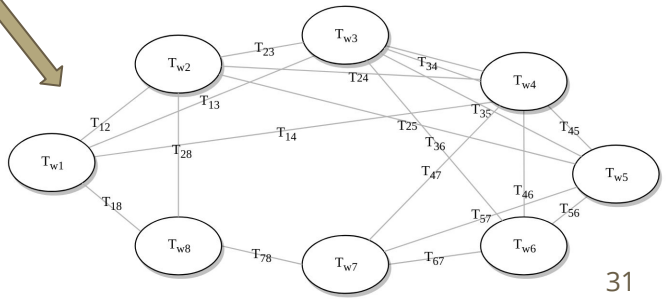


Pose graph optimization with error function:

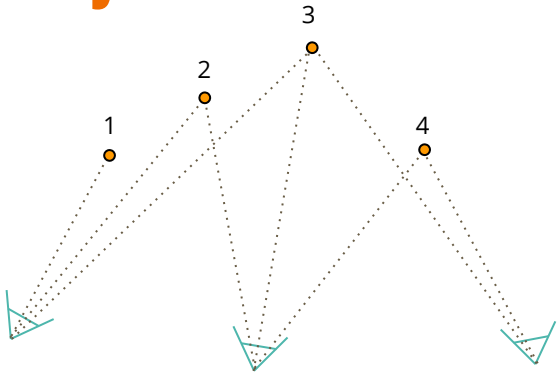
$$E = E^{nfr}_{rel\ pose} + E^{nfr}_{roll\ pitch} + E^{loop\ edge}_{rel\ pose}$$



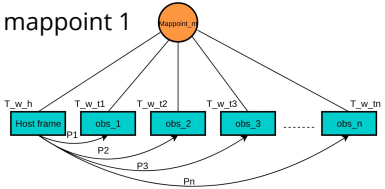
Search observations



Appendix: Bundle adjustment

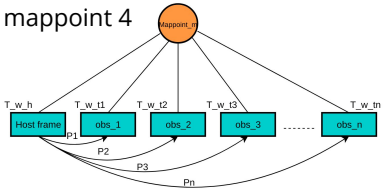


mappoint 1



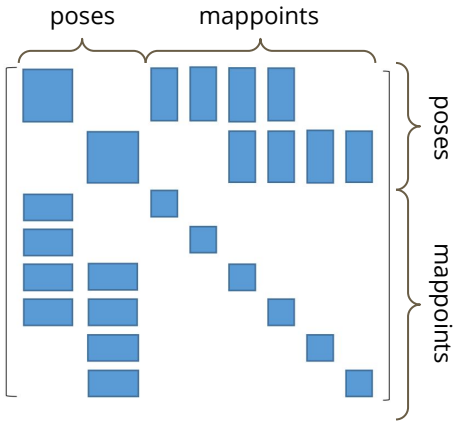
$= H^m_{\text{rel}} = \begin{pmatrix} H^m_{P_1P_1} & & & & H^m_{P_1m} \\ & H^m_{P_2P_2} & & & H^m_{P_2m} \\ & & \ddots & & \vdots \\ & & & H^m_{P_nP_n} & H^m_{P_nm} \\ H^m_{mP_1} & H^m_{mP_2} & \cdots & H^m_{mP_nm} & H^m_{mm} \end{pmatrix}$

mappoint 4

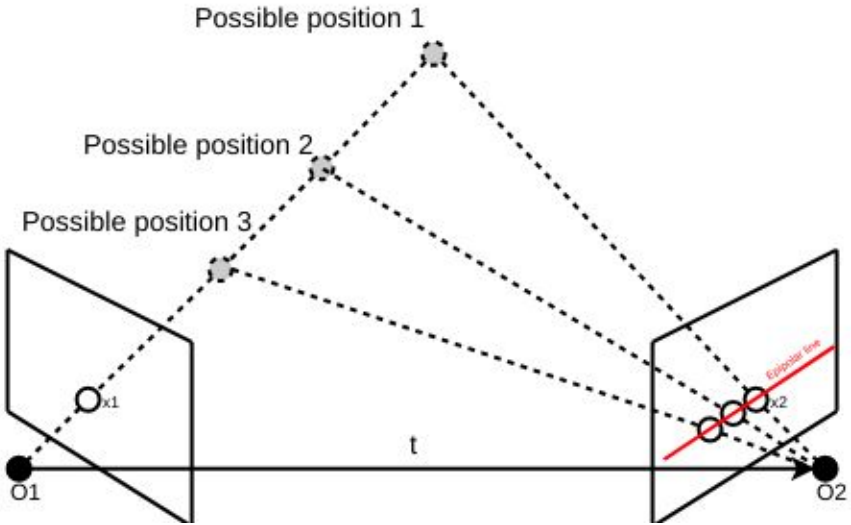


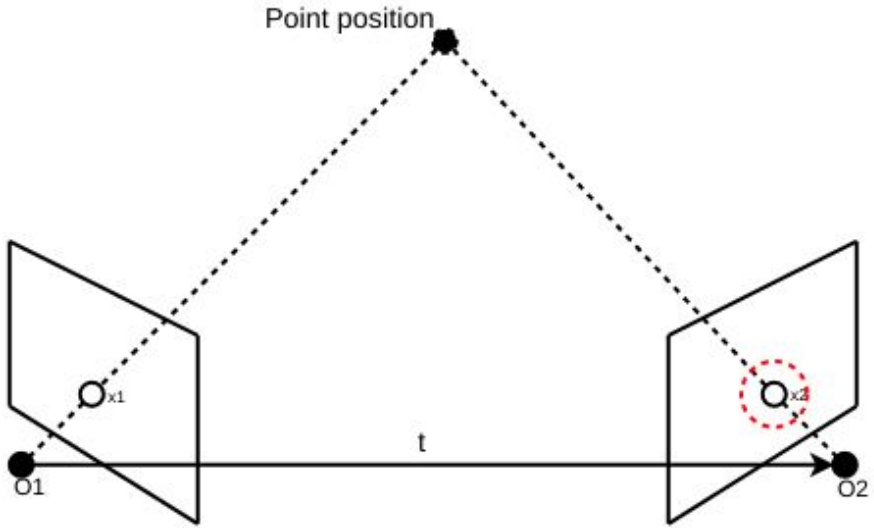
$= H^m_{\text{rel}} = \begin{pmatrix} H^m_{P_1P_1} & & & & H^m_{P_1m} \\ & H^m_{P_2P_2} & & & H^m_{P_2m} \\ & & \ddots & & \vdots \\ & & & H^m_{P_nP_n} & H^m_{P_nm} \\ H^m_{mP_1} & H^m_{mP_2} & \cdots & H^m_{mP_nm} & H^m_{mm} \end{pmatrix}$

H_abs =



Appendix: Track optical flow keypoints by re-projection





Appendix: Optical flow

$$E(\xi) = \sum_{p_i \in \Omega} \| \text{res}_{i,\Omega}(\xi) \|^2 = \sum_{p_i \in \Omega} \| \frac{I_{t+1}(e^{\xi^\wedge} p_i)}{\overline{I_{t+1,\Omega}}(\xi)} - \frac{I_t(p_i)}{\overline{I_{t,\Omega}}(\mathbf{0})} \|^2 \tag{3.1}$$

where $\overline{I_{t,\Omega}}(\mathbf{0}) \equiv \frac{1}{|\Omega|} \sum_{p_i \in \Omega} I_t(p_i)$ and $\overline{I_{t+1,\Omega}}(\xi) \equiv \frac{1}{|\Omega|} \sum_{p_i \in \Omega} I_{t+1}(e^{\xi^\wedge} p_i)$

To calculate $\delta\xi$ for each iteration, we will use Gauss-Newton method to do so. First, we calculate the jacobian

$$J_i(\xi) = \frac{\partial \text{res}_{i,\Omega}(\xi)}{\partial \xi} = \frac{\partial}{\partial \xi} \left(\frac{I_{t+1}(exp^{\xi^\wedge} p_i)}{\overline{I_{t+1,\Omega}}(\xi)} \right) = \frac{\partial}{\partial p'} \left(\frac{I_{t+1}(p')}{\overline{I_{t+1,i}}(\xi)} \right) \frac{\partial p'}{\partial \xi} |_{p'=exp^{\xi^\wedge} p_i} \tag{3.2}$$

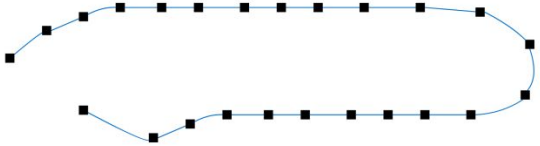
By using chain rule, we have

$$J_i(\xi) = \frac{\partial}{\partial p'} \left(\frac{I_{t+1}(p')}{\overline{I_{t+1,i}}(\xi)} \right) \frac{\partial p'}{\partial \xi} |_{p'=e^{\xi^\wedge}(p_i+x)} \tag{3.3}$$

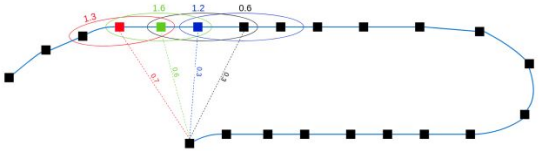
$\frac{\partial p'}{\partial \xi} |_{p'=e^{\xi^\wedge} p_i}$ can be calculated by eq.2.26 and the updated ξ_{new} is as follows,

$$\xi_{\text{new}} = \xi_{\text{old}} - \left(\sum_i J_i(\xi_{\text{old}}) (J_i(\xi_{\text{old}}))^T \right)^{-1} \left(\sum_i J_i(\xi_{\text{old}})^T \text{res}_{i,\Omega}(\xi_{\text{old}}) \right) \tag{3.4}$$

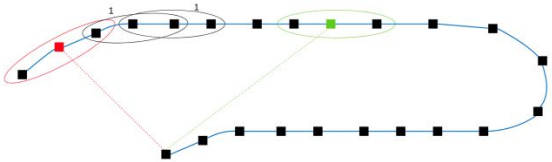
Appendix: Steps of loop detection



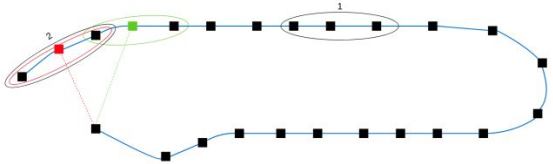
(a) Without loop detection.



(b) At time t, four query results with loop score 0.7 (red), 0.6 (green), 0.3 (blue) and 0.3 (black) respectively. The best accumulated score is 1.3, so the keyframes in red, green and blue are good loop kf candidates



(c) At time t+1, two consistent groups with consistant score 1, two new loop keyframe candidates (red, green).



(d) At time t+2, two consistent groups with consistent score 1 and consistant score 2 respectively, two new loop keyframe candidates (red and green).