Simultaneous Localization And Mapping (SLAM)

Advanced Seminar Computer Engineering

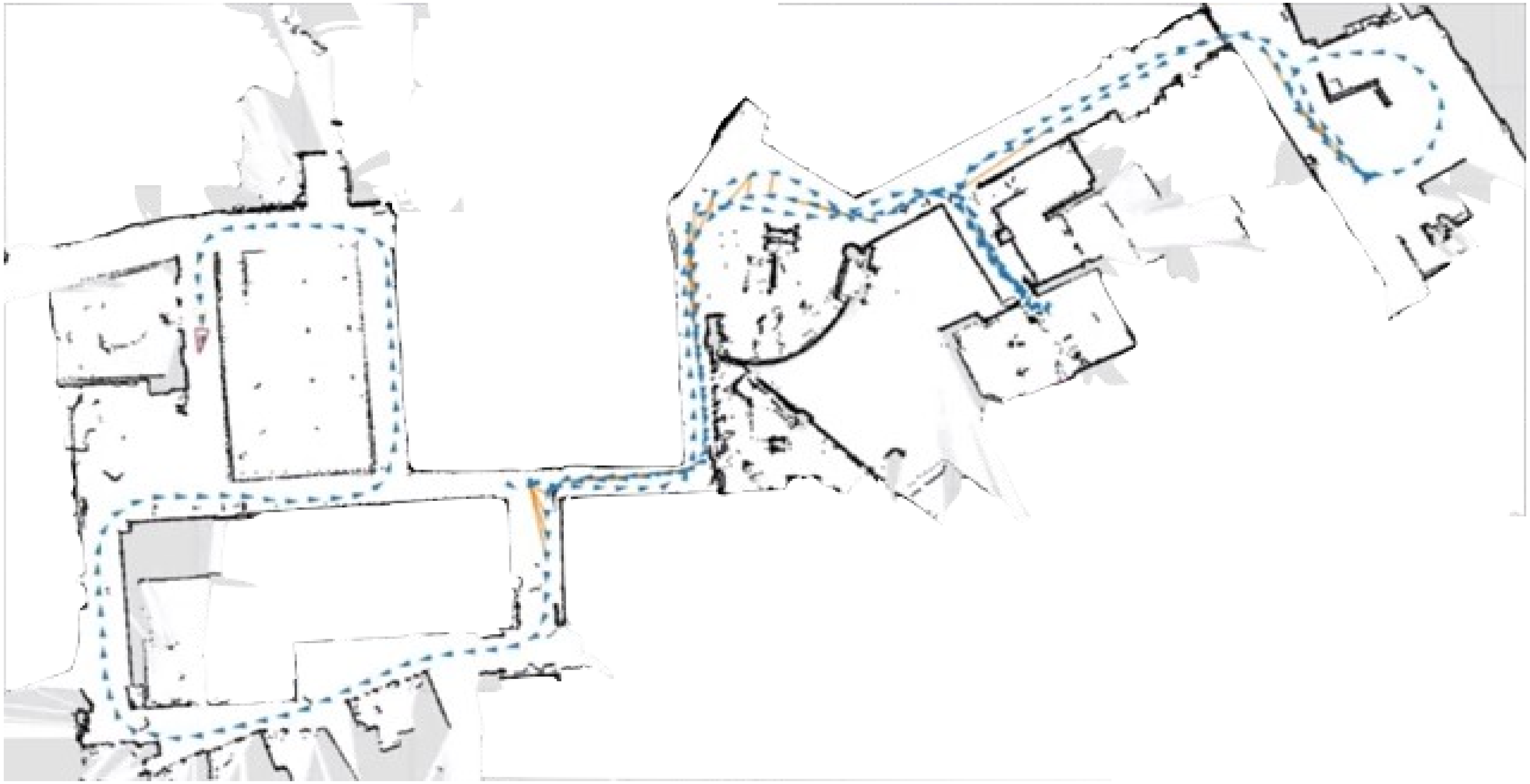
Habib Gahbiche
Supervisor: Dr. Andreas Kugel

Agenda

- Motivation
- Map Representation
- Depth Acquisition
- Monocular Feature Based SLAM
- Monocular LSD-SLAM
  - Tracking
  - Depth Map Estimation
  - Map Optimization
- Conclusion

Motivation

Goal: Build a **map** of the environment while simultaneously determining one's own **location** within it.

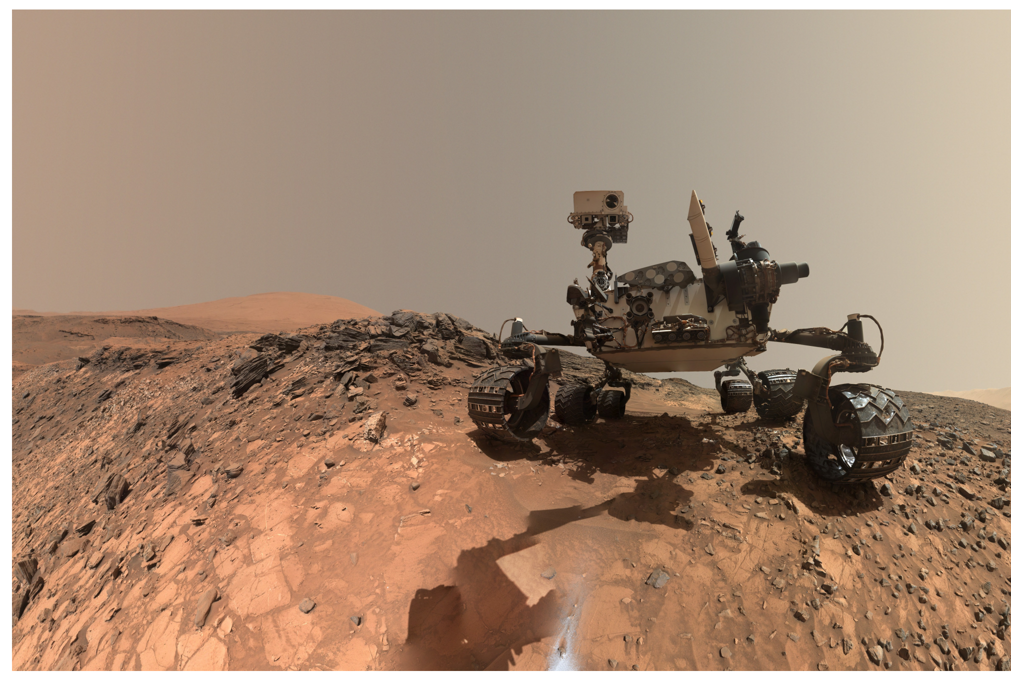


Demo

Motivation – Applications

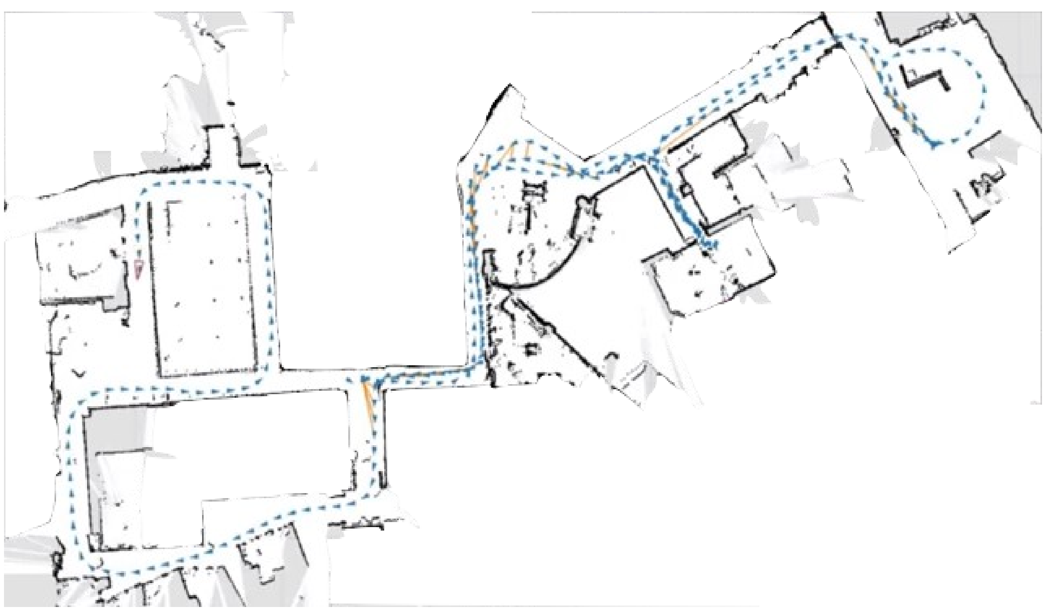
- Augmented reality

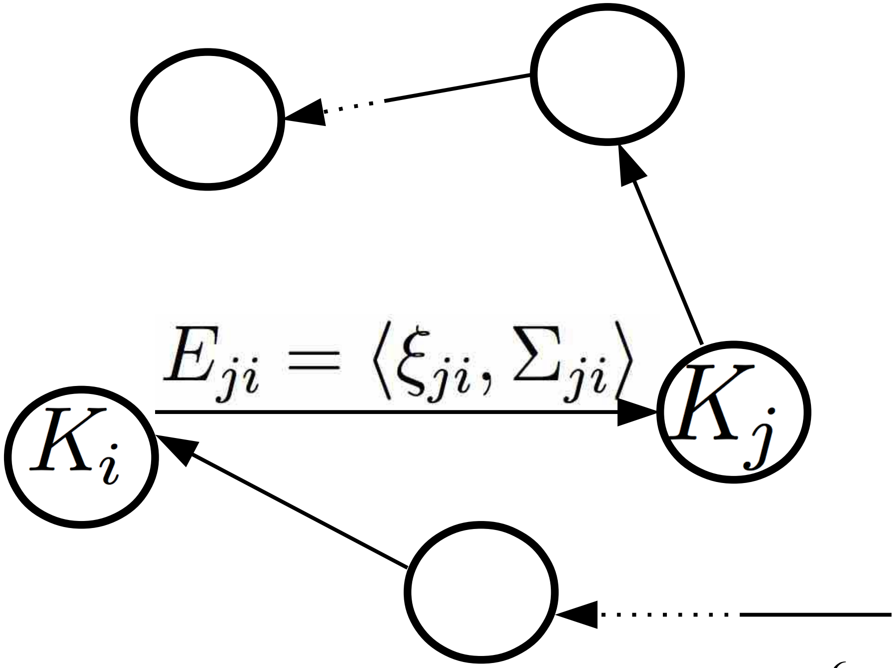
- Autonomous robots



Map Representation

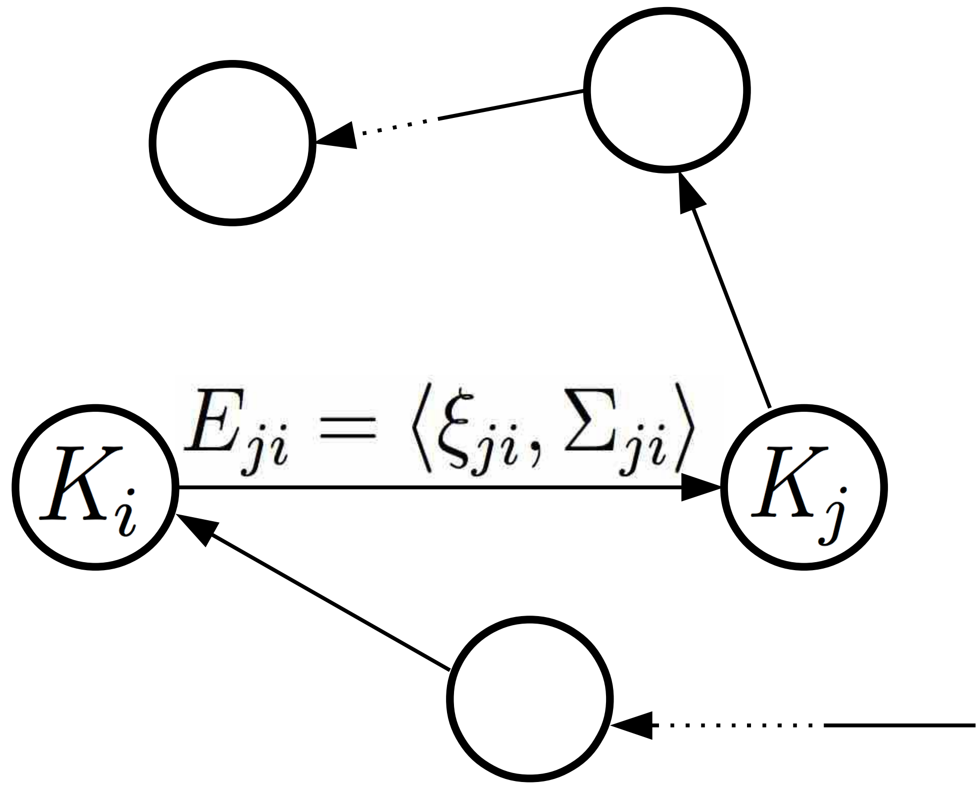
- Map is represented by a **pose graph**
- **Node = keyframe**
- **Edge = similarity transform & covariance matrix**





Map Representation

- **Nodes = keyframes**
- $K_i = \langle I_i, D_i, V_i \rangle$
- I : images
- D: depth maps
- V: variance of depth map



Map Representation

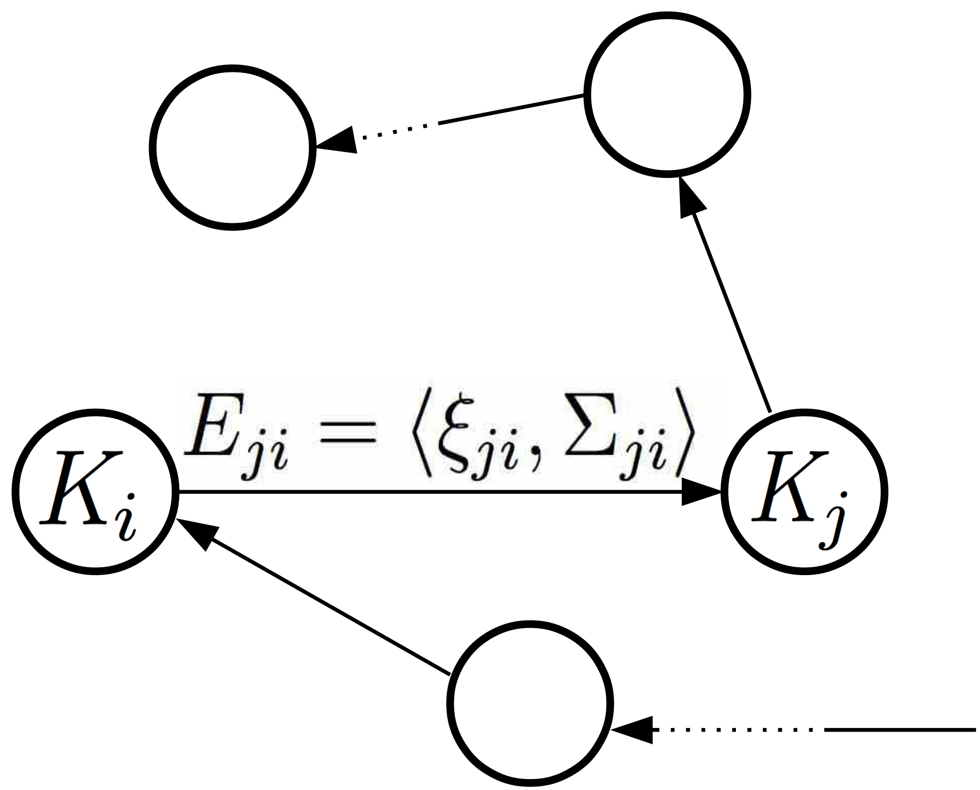
- Edges = similarity transforms & covariance matrices
- **Similarity transform**:

$$\xi = \log_{SE(3)} \begin{pmatrix} \mathbf{sR} & \mathbf{t} \\ 0 & 1 \end{pmatrix}$$

R: Rotation matrix

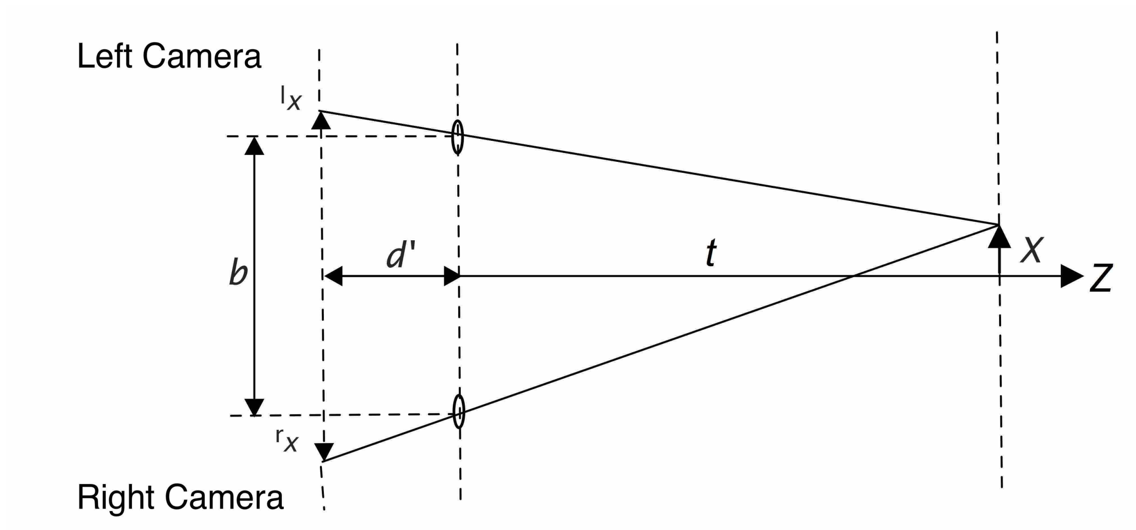
t: Translation vector

s: Scaling factor



- **Covariance matrix**: “How correlated are the two keyframes?”

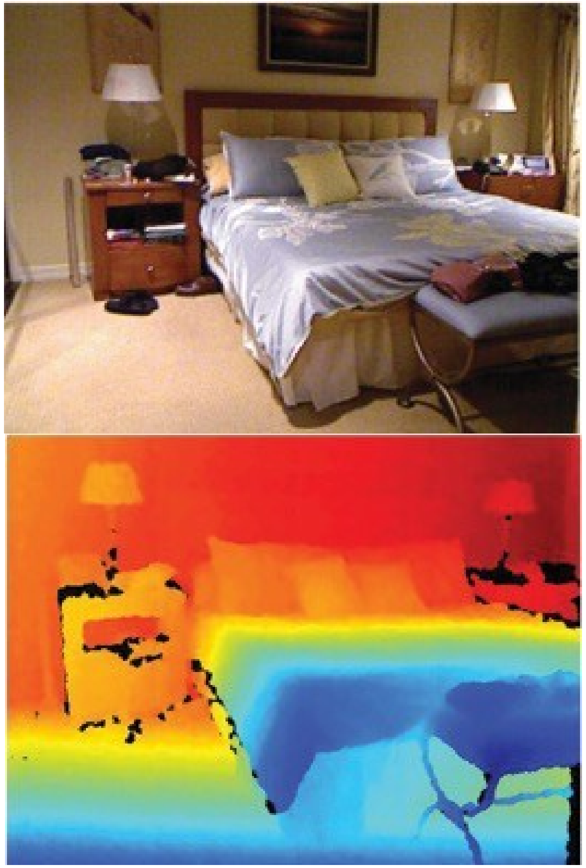
Depth Acquisition – Stereo setup



$$p = l_x - r_x = b\frac{d'}{t} \implies t = b\frac{d'}{p}$$

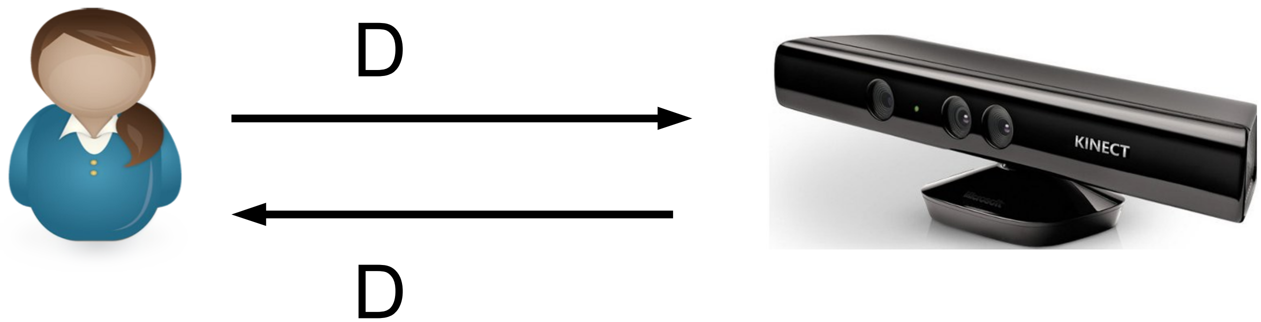
- Depth is estimated from **one** measurement (frame)

Depth Acquisition – RGB-D camera



- RGB-D: Red, Green, Blue, Depth
- Depth is estimated from **one** measurement (frame)

Depth Acquisition – RGB-D camera



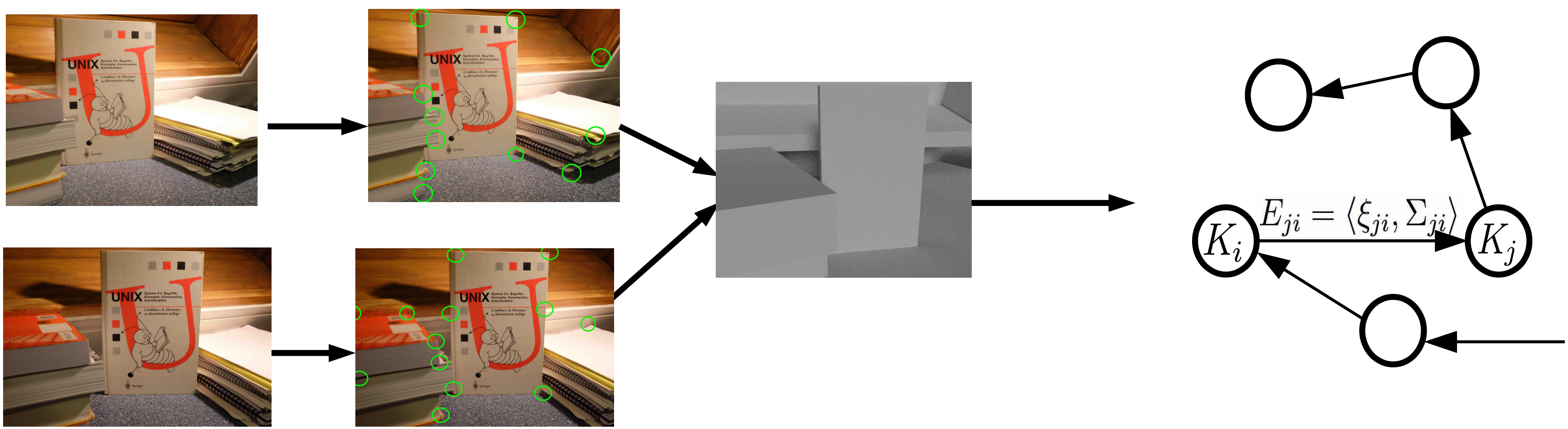
- Get D with Time of Flight (ToF)
- D = t c / 2
- Where c is the speed of light.

Depth Acquisition – Monocular setup

- Monocular: use one single RGB camera.
- Need **two** frames to estimate depth!

Monocular Feature Based SLAM

Images → extract features → estimate depth → create/update pose graph



Monocular Feature Based SLAM

- Extract features (e.g. using SIFT, HOG...)
- Compute camera position and scene geometry based on features.
- (+) Simplifies problem
- (-) Only information relevant to **feature type** is used!
- → Use direct method!

Monocular LSD-SLAM Algorithm

- Monocular Large-Scale Direct SLAM algorithm:

  Monocular: single RGB camera

  Large-Scale: Large variations in scene scale are allowed

  Direct: no feature extraction used!

- Three main steps:

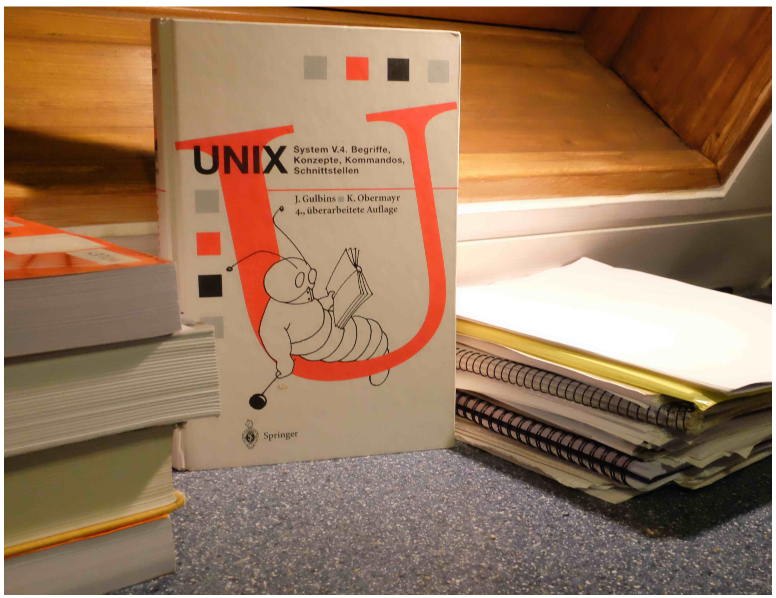
  1) Tracking

  2) Depth Map Estimation

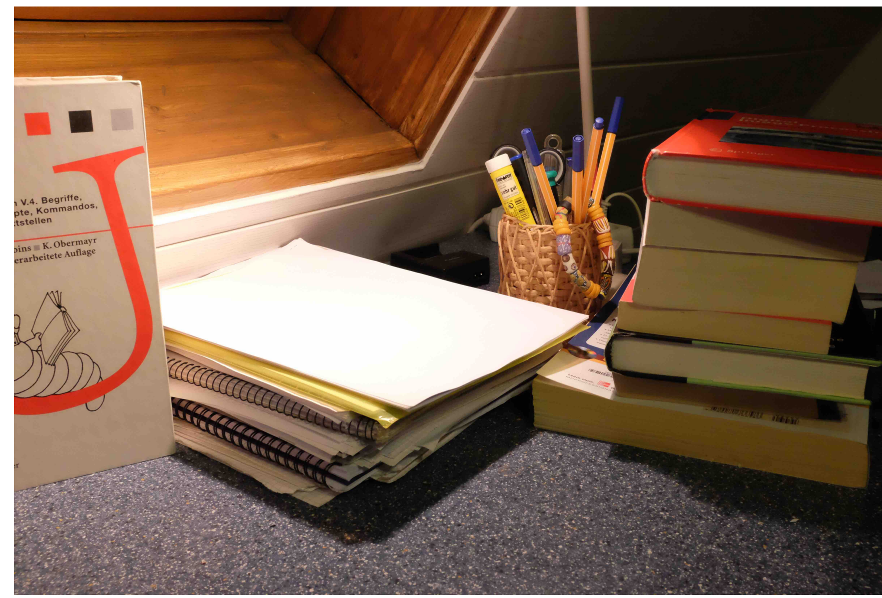
  3) Map Optimization

LSD-SLAM Algorithm
1 – Tracking

Tracking = estimating ξ_{in} between **initial** and **new** keyframes



$$\xrightarrow{\xi_{in}}$$



K_i

K_n

Initial position

(initial keyframe)

new position

new camera frame

LSD-SLAM Algorithm
1 – Tracking

- Estimating ξ_{in} :

  - Define **error** function $Err(\xi_{in}) = \sum_{\mathbf{p} \in I} \left(\xi_{in}[I_i(\mathbf{p})] - I_n(\mathbf{p})\right)^2$

  - Minimize $Err(\xi_{in})$ with **iterative** Gauss-Newton algorithm
  - Initialize $\xi^{(0)}$ with **random** values
  - $\xi_{in}^{(n+1)} = \delta\xi_{in}^{(n)} \circ \xi_{in}^{(n)}$

$$\delta\xi \propto \mathbf{J}(\xi)$$

- Propose ξ_{in} to Depth Map Estimator

LSD-SLAM Algorithm
2 – Depth Map Estimation

- If camera movement is significant

  - **Create** new keyframe.

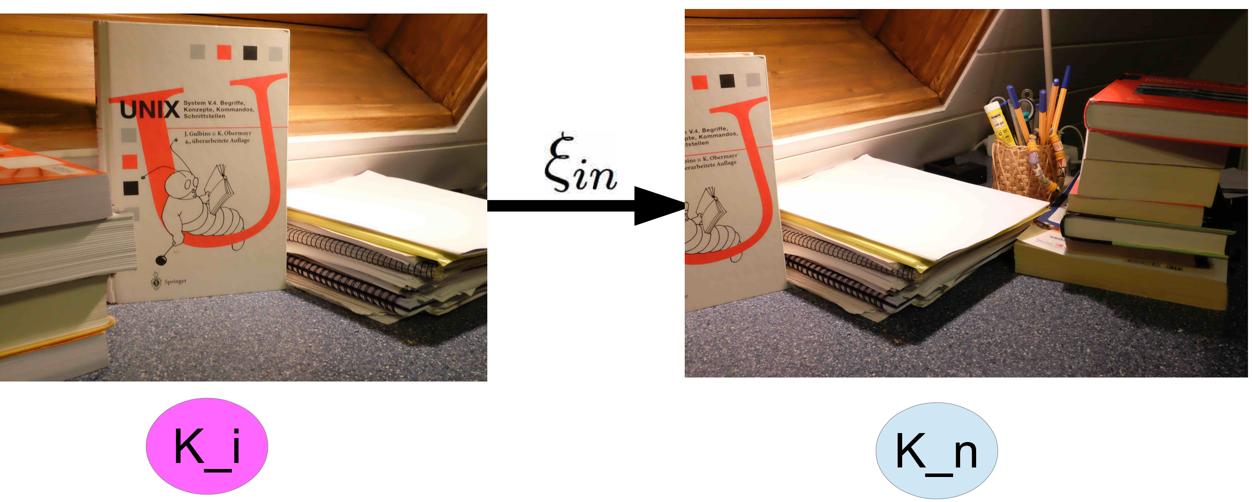
- Else

  - **Refine** current keyframe.

- (Remember: $K_i = \langle I_i, D_i, V_i \rangle$)

LSD-SLAM Algorithm
2 – Depth Map Estimation

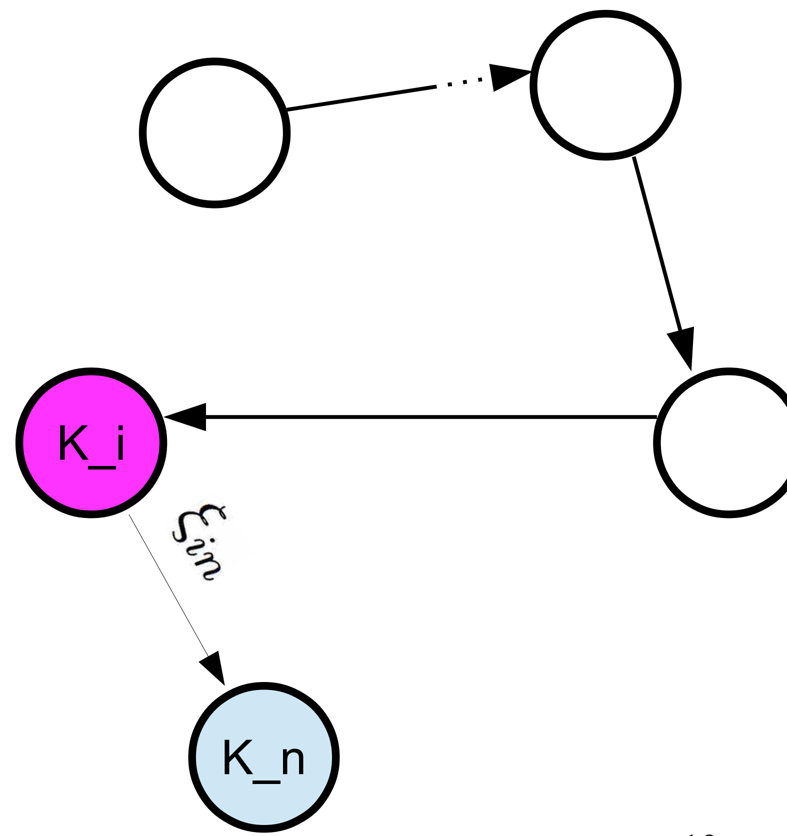
Current tracked frame



$\rightarrow$ Decision: **Create** $\mathbf{K_n}$!

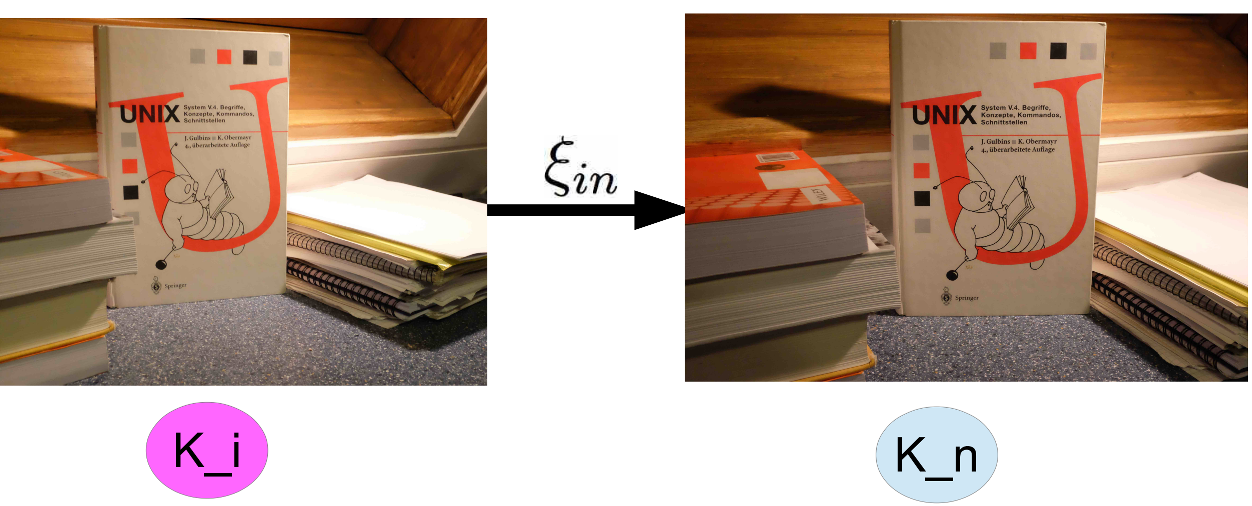
- Project D_i into D_n

Current pose graph



LSD-SLAM Algorithm
2 – Depth Map Estimation

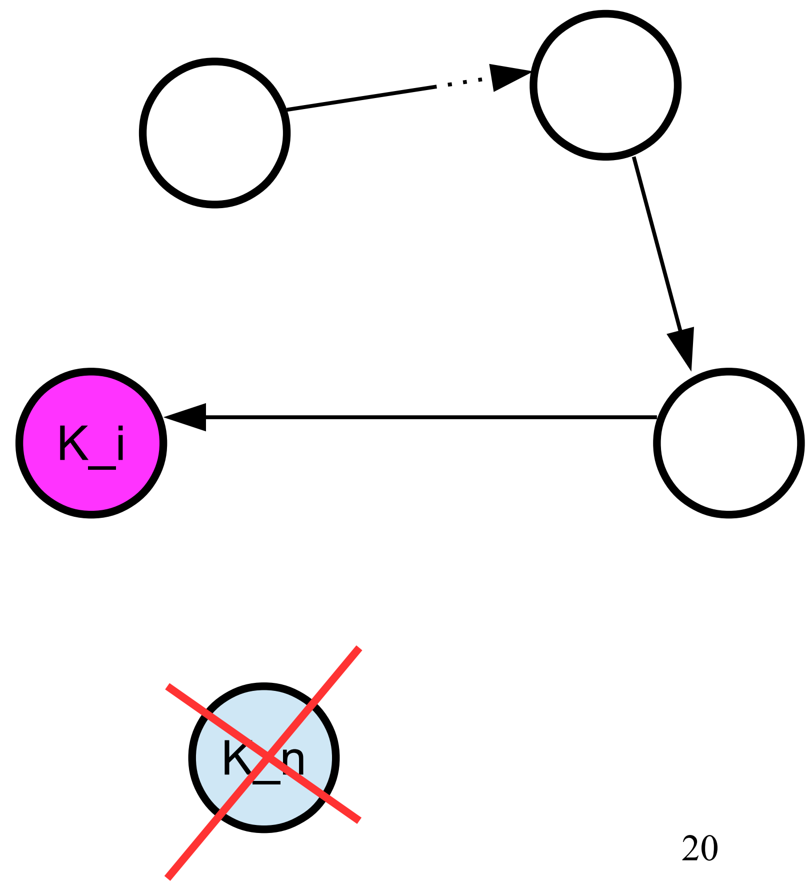
<u>Current tracked frame</u>



→ Decision: **Refine** $\mathbf{K_i}$!

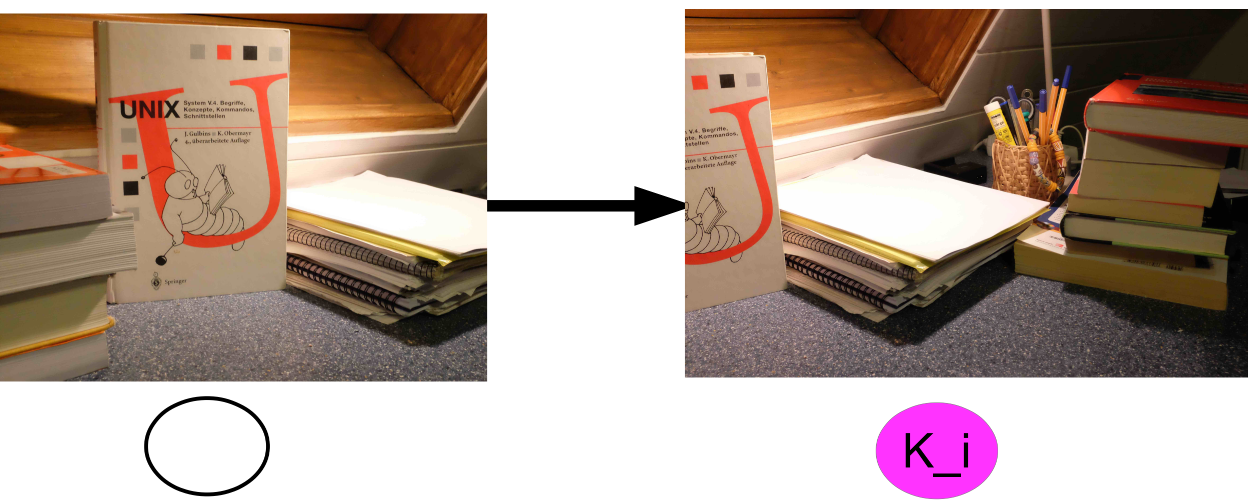
→ Improve $D_i \subset \mathbf{K_i}$ with stereo comparisons

<u>Current pose graph</u>



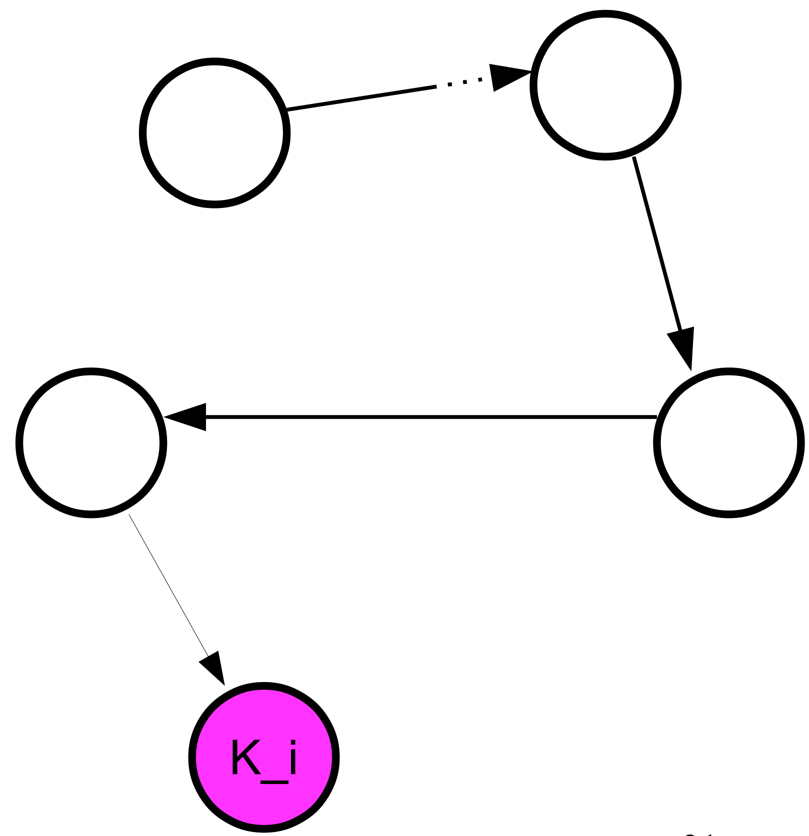
LSD-SLAM Algorithm
2 – Depth Map Estimation

<u>Current tracked frame</u>



- Set $\mathbf{K_n}$ as initial frame
- Back to Tracking...

<u>Current pose graph</u>



LSD-SLAM Algorithm
3 – Map Optimization

- Optimization includes:

  - Loop closures

  - Rendering: Pose Graph → Global Map

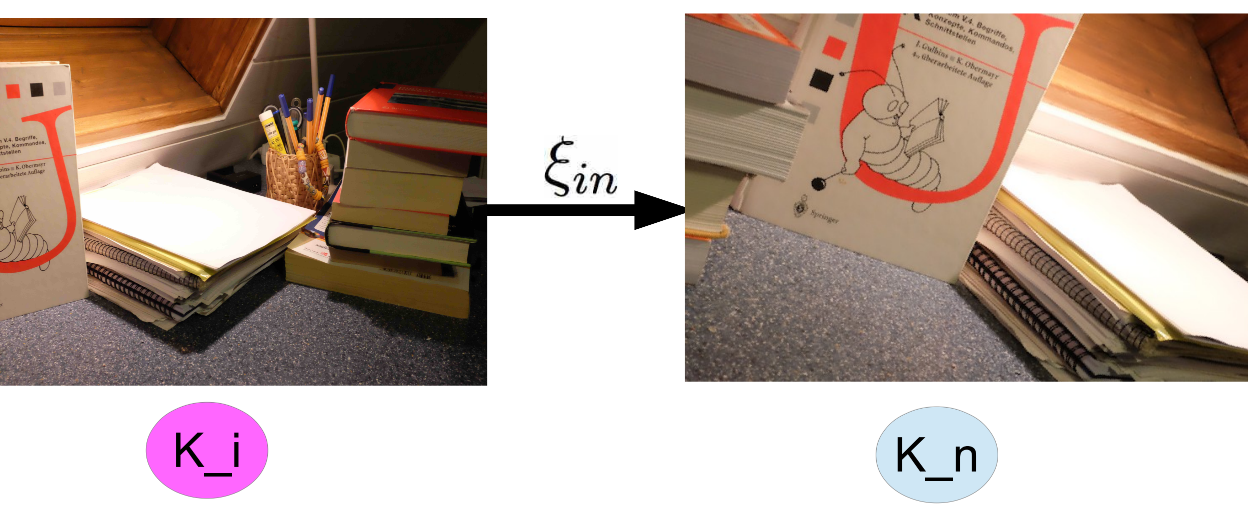
- Optimization after each iteration **not** necessary!

  - Loop closure after 5-10 new keyframes.

  - Render after 2 new keyframes.

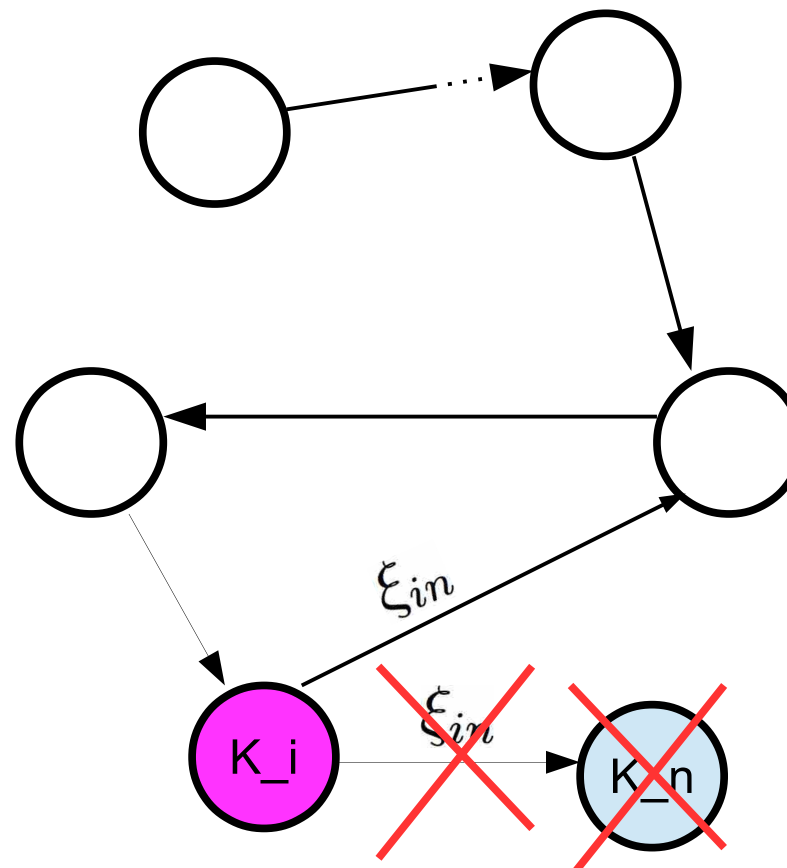
LSD-SLAM Algorithm
3 – Map Optimization (loop closure)

Current tracked frame



→ **Loop detected!**

Current pose graph

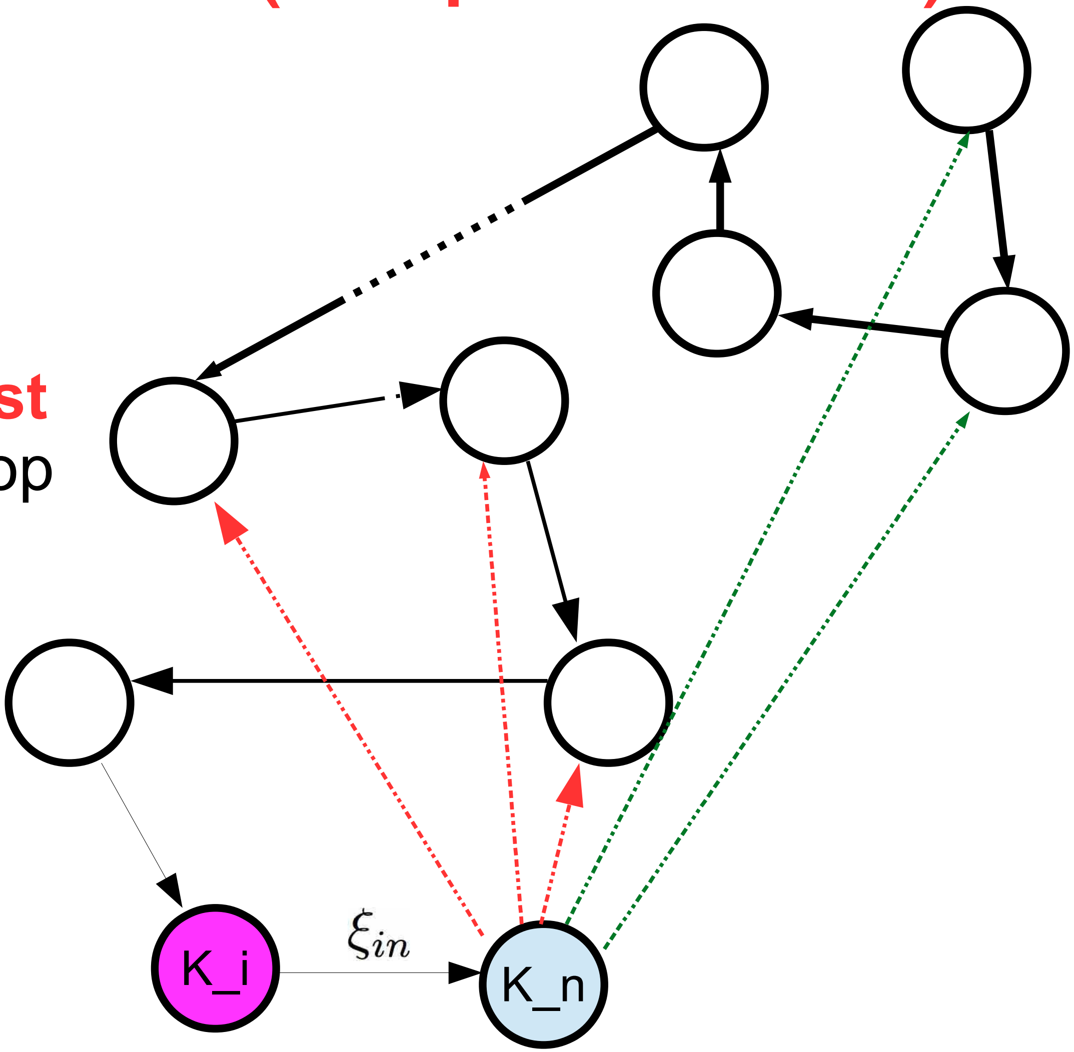


LSD-SLAM Algorithm
3 – Map Optimization (loop closure)

- Loop closure:

  - Compare K_i with **10 nearest** keyframes for **small scale** loop

  - Use proposals from **appearance-based** mapping algorithm for **large scale** loops.



Evaluation LSD-SLAM

- Real time SLAM possible

  - 2,5 GHz Intel Core i5 | 10-15 FPS | 640x480 image size

  - 100 MHz Zynq-7020, ARM Cortex A9 | 4 FPS | 320x240

Demo

Conclusion

- SLAM is the problem of building a **map** and determining the camera **position** within this map **simultaneously**.

- LSD-SLAM solves the SLAM problem with a monocular direct SLAM algorithm

References

- J. Engel, T. Schops, and D. Cremers, ”LSD-SLAM: Large-scale direct monocular SLAM,” in European Conference on Computer Vision (ECCV), September 2014.
- K. Boikos and C. S. Bouganis, ”Semi-dense SLAM on an FPGA SoC,” 2016 26th Inter- national Conference on Field Programmable Logic and Applications (FPL), Lausanne, 2016, pp. 1-4. doi: 10.1109/FPL.2016.7577365
- Kerl, C., Sturm, J., Cremers, D.: Dense visual SLAM for RGB-D cameras. In: Intl. Conf. on Intelligent Robot Systems (IROS) (2013)
- Digital Image Processing, Bernd Jähne, 6th edition
- Simultaneous Localisation and Mapping (SLAM):Part I The Essential AlgorithmsHughDurrant-Whyte, Fellow, IEEE, and Tim Bailey
- Colored depth map: ImageNet.com
- Mars Rover and Augmented reality: https://wikimedia.org/wikipedia/commons/d/dc/PIA16239_High-Resolution_Self-Portrait_by_Curiosity_Rover_Arm_Camera.jpg

  http://www.augment.com/blog/wp-content/uploads/2016/08/augmented-reality-Augment-blog.jpg