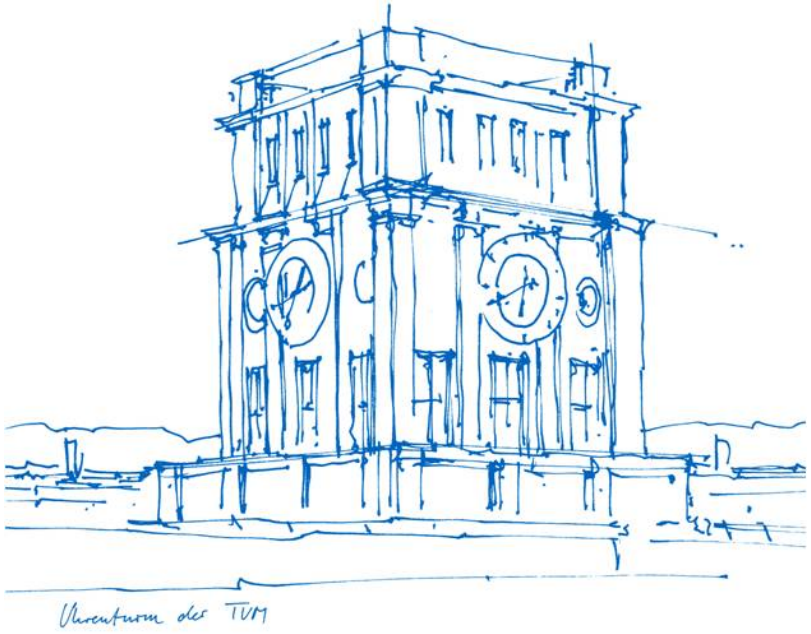
BAD SLAM: Bundle Adjusted Direct RGB-D SLAM

Thomas Schöps, Torsten Sattler, Marc Pollefeys

CVPR 2019

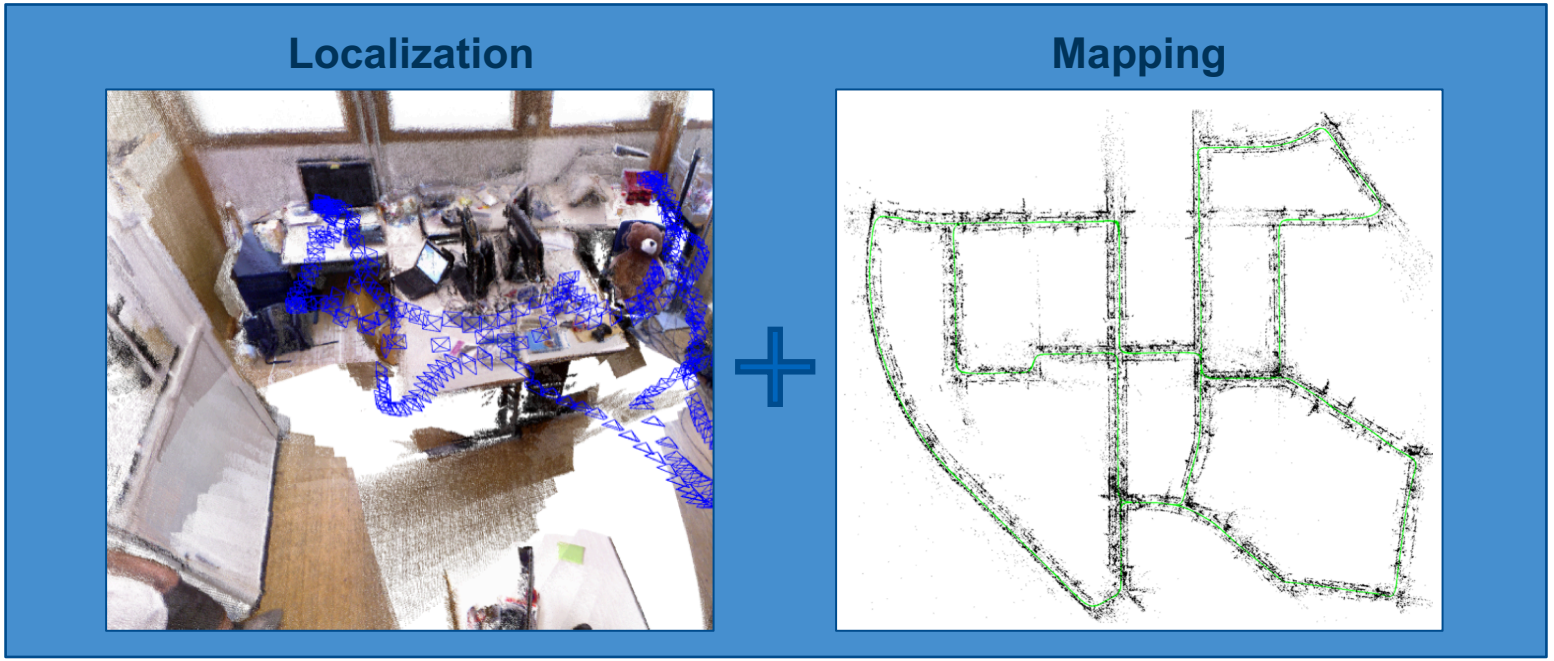


Michael Gentner

Technische Universität München

München, 15. April 2020

Introduction – SLAM



Introduction – State-of-the-art

Indirect[1]

→Only feature based, not using all available information.

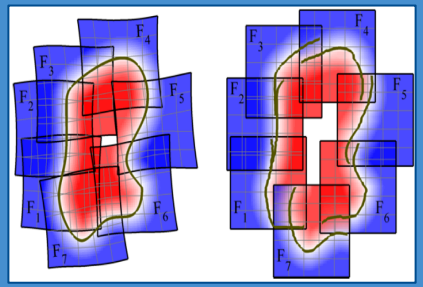
[1] ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras, Artal et. al., 2016
[2] Large-Scale and Drift-Free Surface Reconstruction Using Online Subvolume Registration, Fioraio et. al., 2015
[3] LDSO: Direct Sparse Odometry with Loop Closure, Gao et. al., 2018

Introduction – State-of-the-art

Indirect[1]

→Only feature based, not using all available information.

Not real-time[2]



→Requires some offline computing.

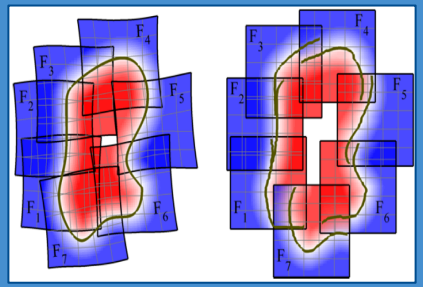
[1] ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras, Artal et. al., 2016
[2] Large-Scale and Drift-Free Surface Reconstruction Using Online Subvolume Registration, Fioraio et. al., 2015
[3] LDSO: Direct Sparse Odometry with Loop Closure, Gao et. al., 2018

Introduction – State-of-the-art

Indirect[1]

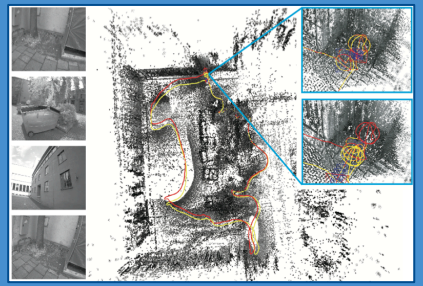
→Only feature based, not using all available information.

Not real-time[2]



→Requires some offline computing.

Approximate[3]



→Reduces problem size by simplifications.

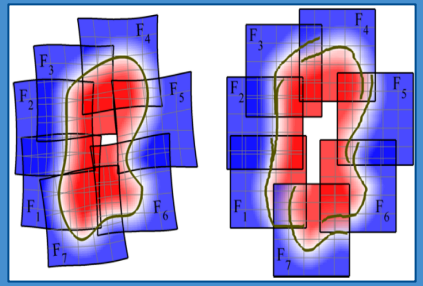
[1] ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras, Artal et. al., 2016
[2] Large-Scale and Drift-Free Surface Reconstruction Using Online Subvolume Registration, Fioraio et. al., 2015
[3] LDSO: Direct Sparse Odometry with Loop Closure, Gao et. al., 2018

Introduction – State-of-the-art

Indirect[1]

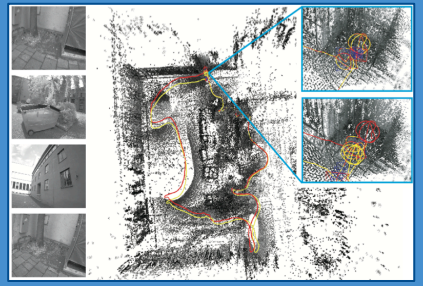
→Only feature based, not using all available information.

Not real-time[2]



→Requires some offline computing.

Approximate[3]



→Reduces problem size by simplifications.

Direct + Realtime + Global

[1] ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras, Artal et. al., 2016
[2] Large-Scale and Drift-Free Surface Reconstruction Using Online Subvolume Registration, Fioraio et. al., 2015
[3] LDSO: Direct Sparse Odometry with Loop Closure, Gao et. al., 2018

Outline

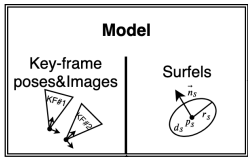
Method
- Overview
- Cost function
- Back-end
- Dataset

Results
- Accuracy
- Runtime

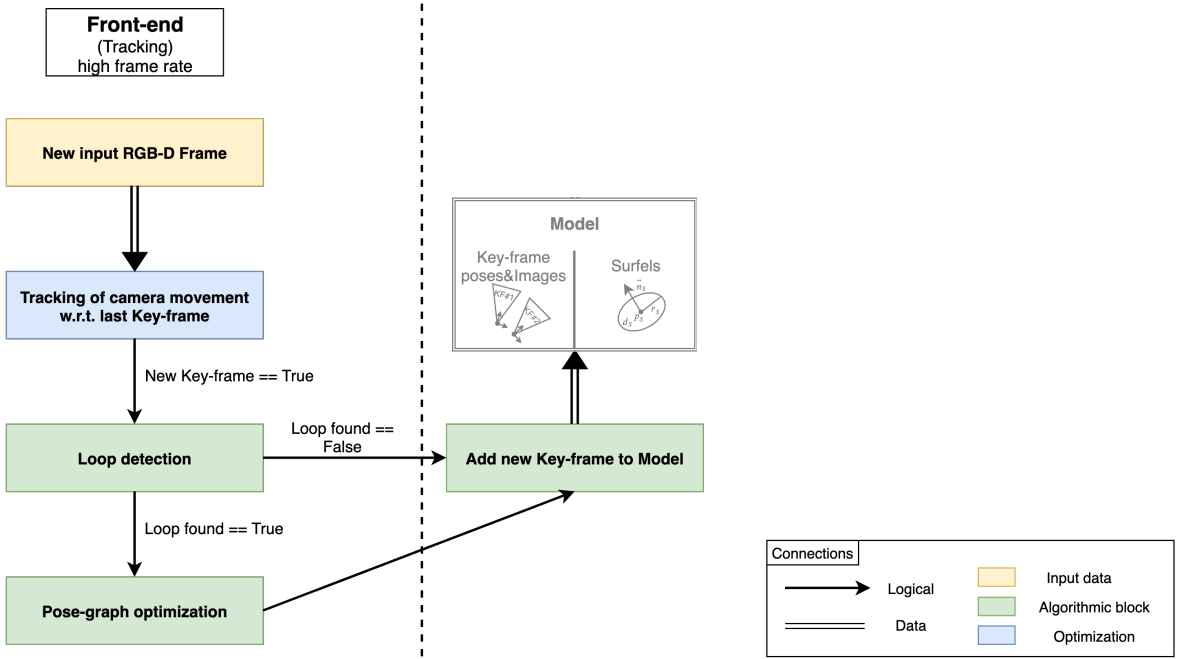
Personal comments

Summary

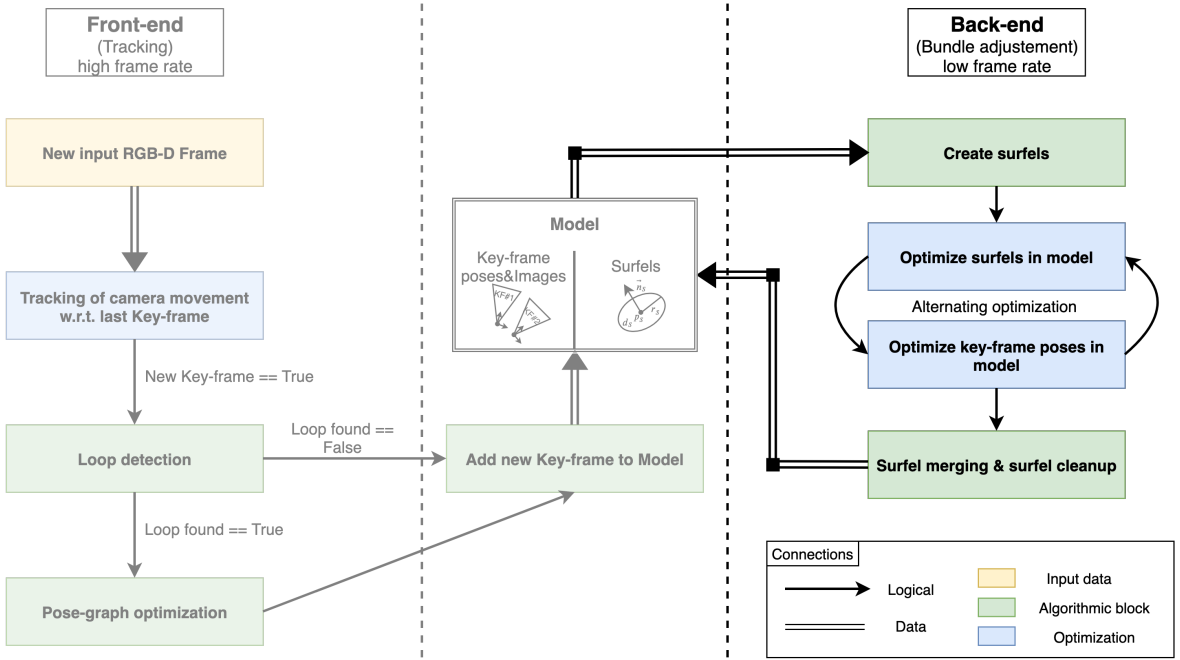
Method – Overview



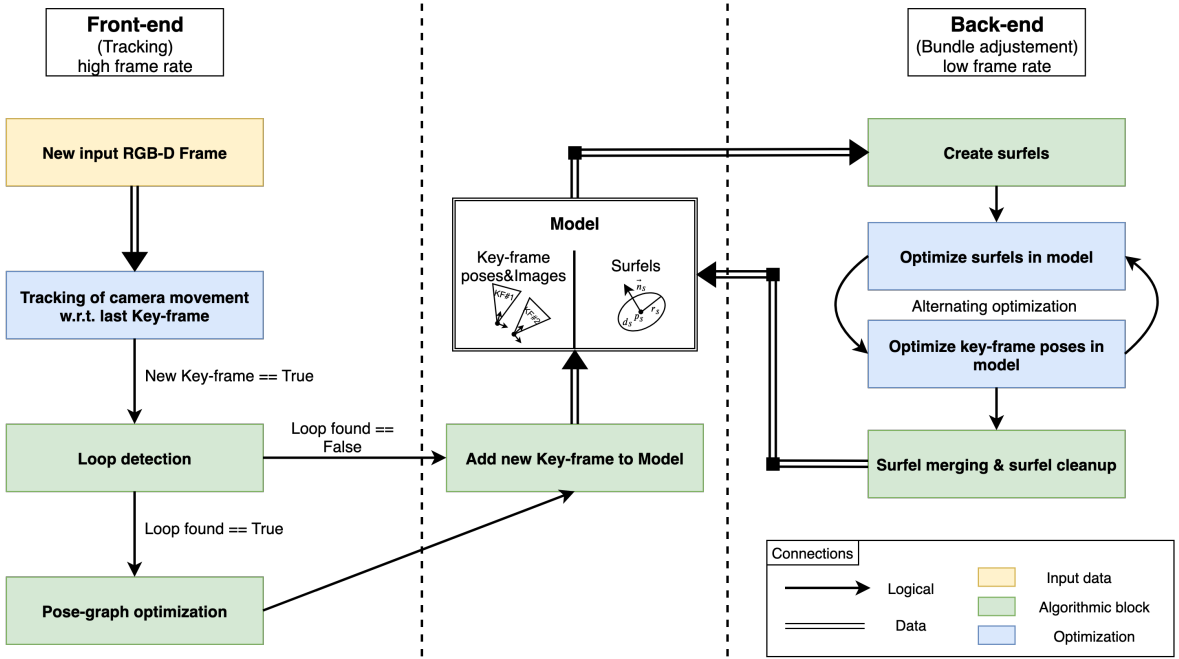
Method – Overview



Method – Overview

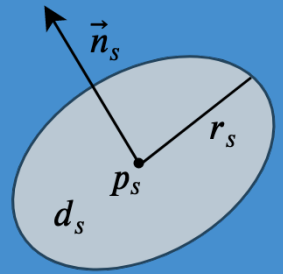


Method – Overview



Method – Model

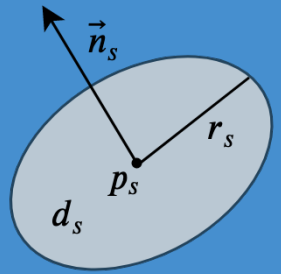
Surfel s



- $\vec{n}_s$: Normal vector
- p_s: Center point
- r_s: Radius
- d_s: Visual descriptor

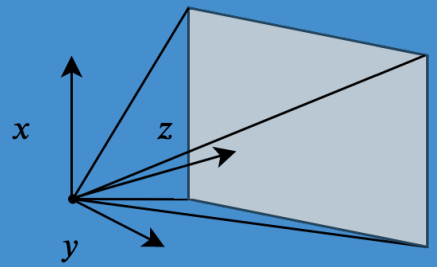
Method – Model

Surfel s



- $\vec{n}_s$: Normal vector
- p_s: Center point
- r_s: Radius
- d_s: Visual descriptor

Keyframe k



- 6-DOF position
- RGB image + Depth image
- Shared intrinics
- Intensity is interpolated bilinearly

Method – Cost function

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey}\left(\sigma_D^{-1} r_{geom}(s,k)\right) + w_{photo} \cdot \rho_{Huber}\left(\sigma_p^{-1} r_{photo}(s,k)\right) \right)$$

K: Set of all keyframes

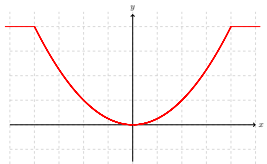
S_k: Set of all corresponding surfels

Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey}\left(\sigma_D^{-1} r_{geom}(s,k)\right) + w_{photo} \cdot \rho_{Huber}\left(\sigma_p^{-1} r_{photo}(s,k)\right)\right)$$

$$\rho_{Tukey}(x) = \begin{cases} \left(\frac{x}{c}\right)^2, & |x| < c \\ 0, & |x| \geq c \end{cases}$$



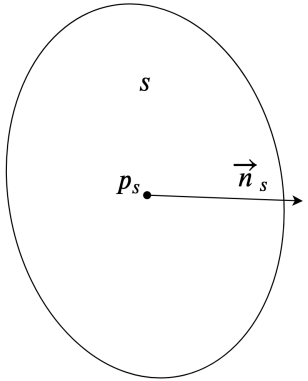
σ_D: Standard deviation of geometric residual

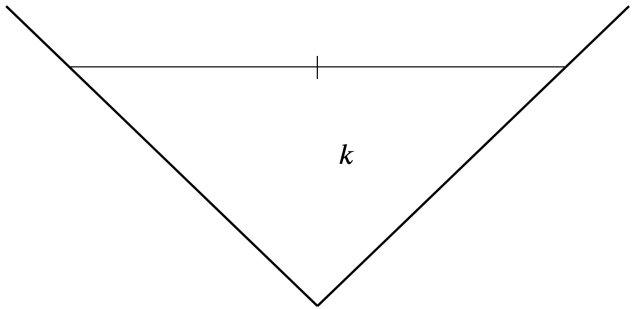
$r_{geom}(s,k)$: Geometric residual

Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$



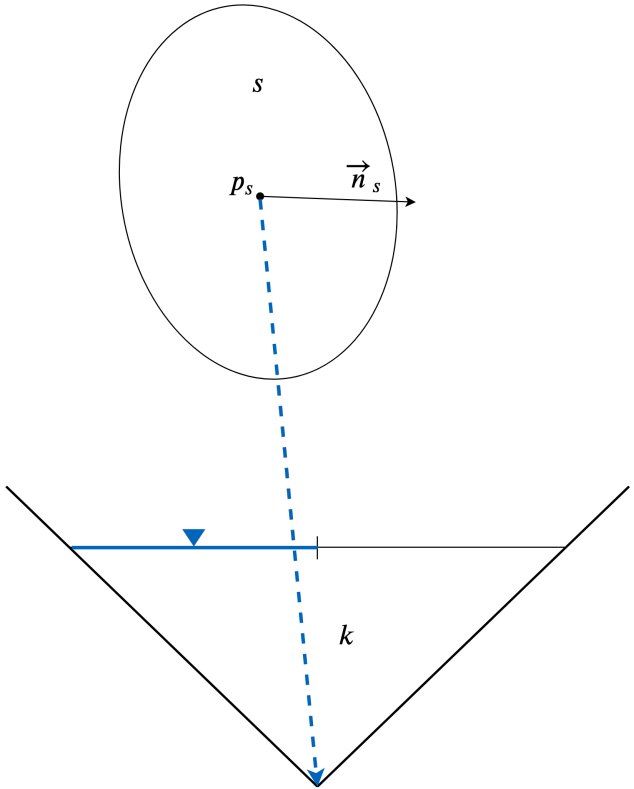


Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

$\hat{\pi}_{D,k}\left(\boldsymbol{T}_G^k \boldsymbol{p}_s\right)$

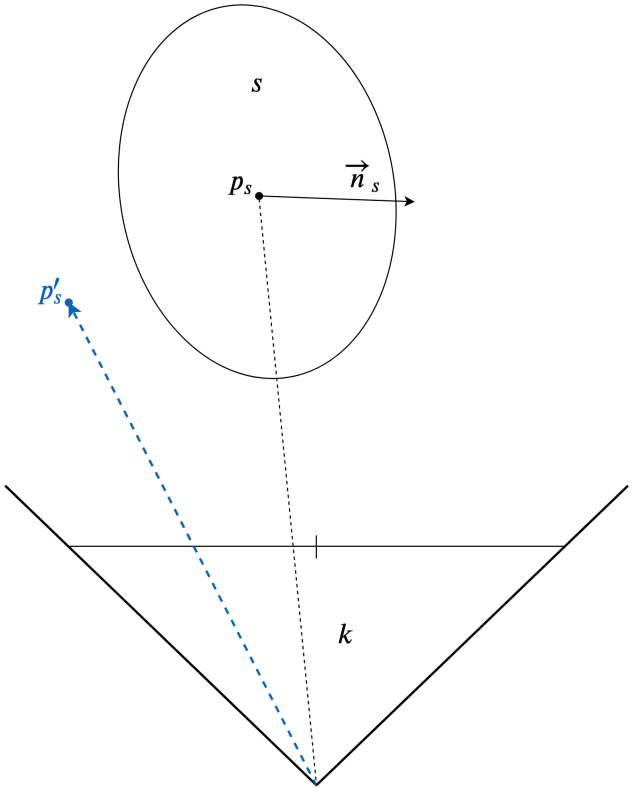


Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

$$\pi_{D,k}^{-1} \left(\hat{\pi}_{D,k} \left(\boldsymbol{T}_G^k \boldsymbol{p}_s \right) \right)$$

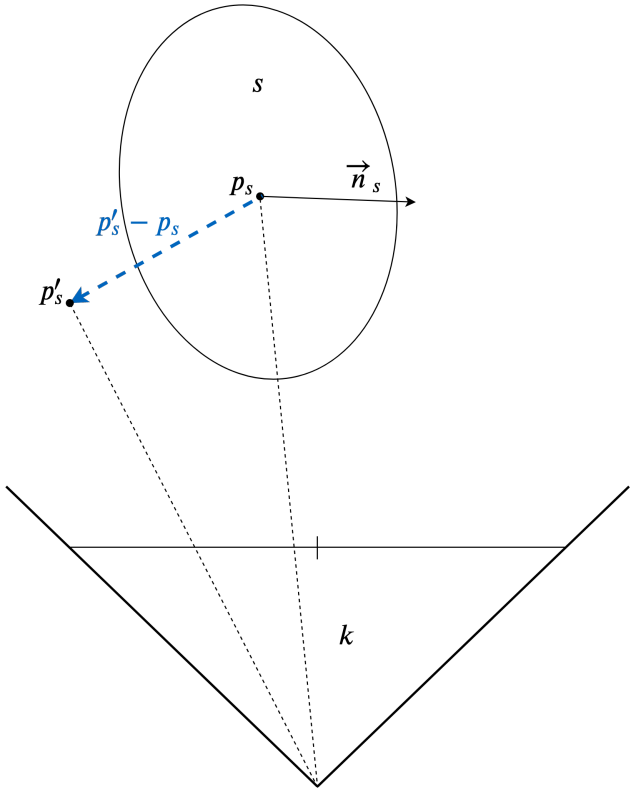


Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

$$\left(\pi_{D,k}^{-1} \left(\hat{\pi}_{D,k} \left(\boldsymbol{T}_G^k \boldsymbol{p}_s \right) \right) - \boldsymbol{T}_G^k \boldsymbol{p}_s \right)$$

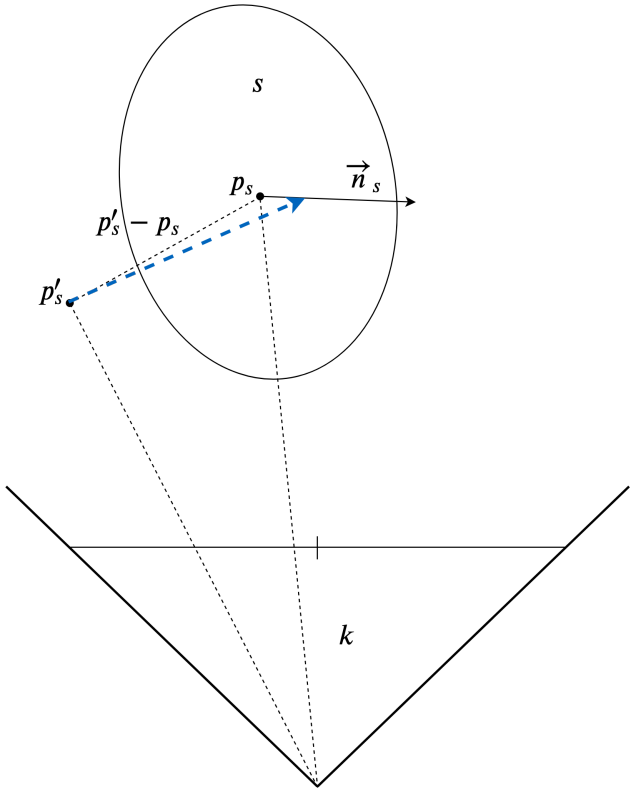


Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

$$r_{geom}(s,k) = \left(\boldsymbol{T}_G^k \boldsymbol{n}_s \right)^T \left(\pi_{D,k}^{-1} \left(\hat{\pi}_{D,k} \left(\boldsymbol{T}_G^k \boldsymbol{p}_s \right) \right) - \boldsymbol{T}_G^k \boldsymbol{p}_s \right)$$

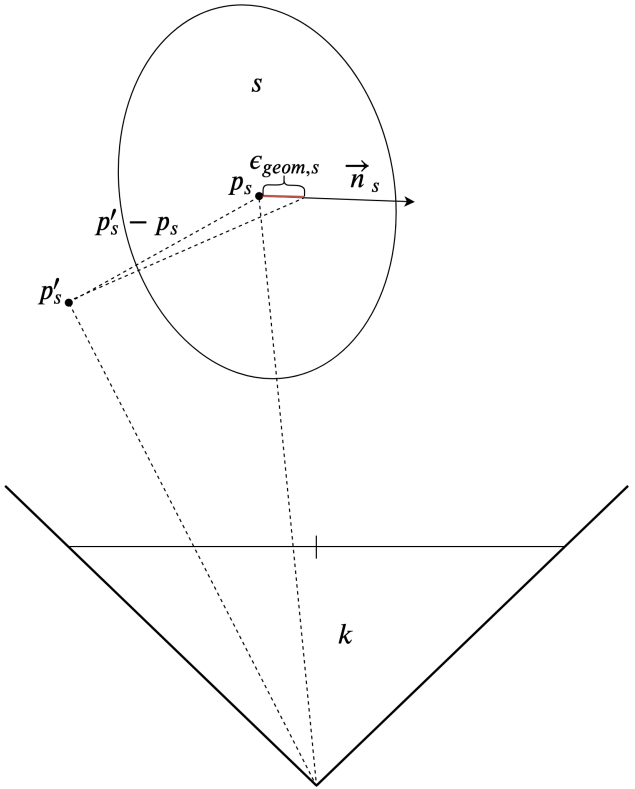


Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

$$r_{geom}(s,k) = \left(\boldsymbol{T}_G^k \boldsymbol{n}_s \right)^T \left(\pi_{D,k}^{-1} \left(\hat{\pi}_{D,k} \left(\boldsymbol{T}_G^k \boldsymbol{p}_s \right) \right) - \boldsymbol{T}_G^k \boldsymbol{p}_s \right) = \epsilon_{geom,s}$$

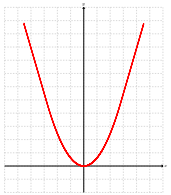


Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey}\left(\sigma_D^{-1} r_{geom}(s,k)\right) + w_{photo} \cdot \rho_{Huber}\left(\sigma_p^{-1} r_{photo}(s,k)\right) \right)$$

$$\rho_{Huber}(x) = \begin{cases} \frac{1}{2}x^2, & |x| < c \\ c(|x| - \frac{1}{2}c), & |x| \geq c \end{cases}$$



w_{photo}: Weighting factor

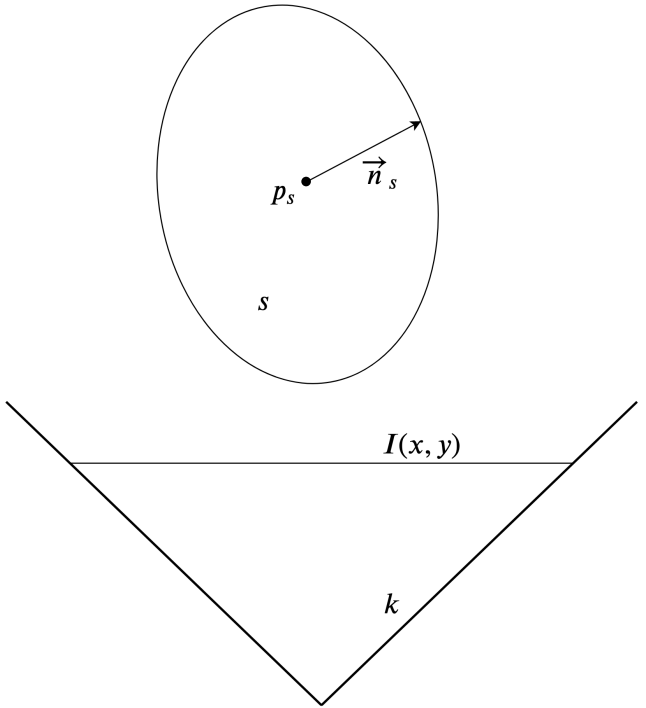
σ_p: Standard deviation of photometric residual

$r_{photo}(s,k)$: Photometric residual

Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

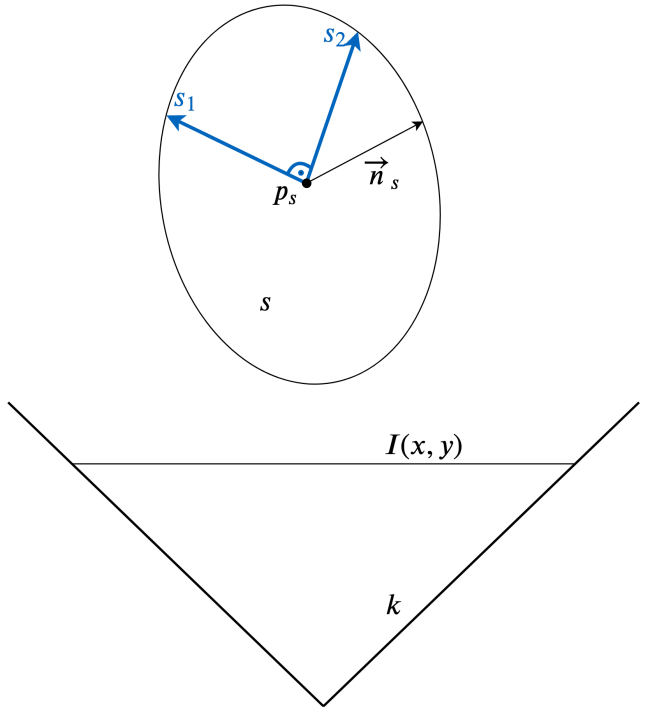


Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

$\boldsymbol{s}_1$
$\boldsymbol{s}_2$



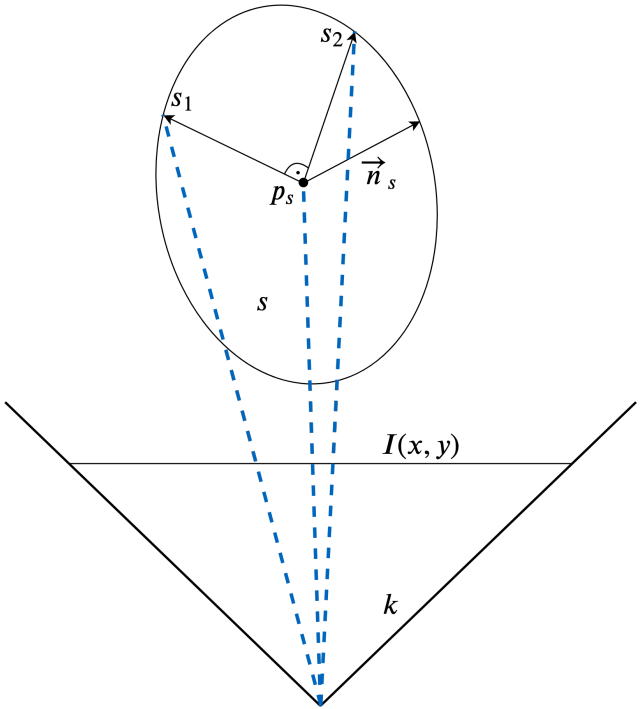
Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey}\left(\sigma_D^{-1} r_{geom}(s,k)\right) + w_{photo} \cdot \rho_{Huber}\left(\sigma_p^{-1} r_{photo}(s,k)\right) \right)$$

$\pi_{I,k}(\boldsymbol{s}_1)$ $\pi_{I,k}(\boldsymbol{p}_s)$

$\pi_{I,k}(\boldsymbol{s}_2)$ $\pi_{I,k}(\boldsymbol{p}_s)$

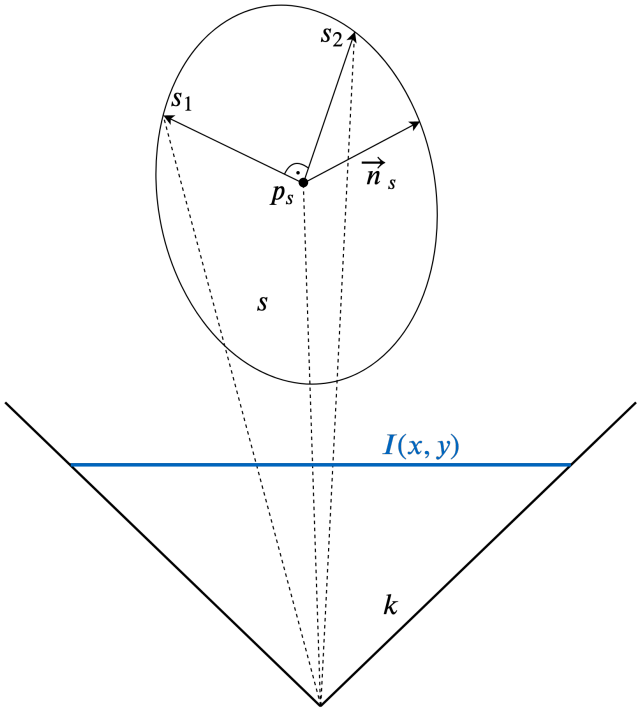


Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey}\left(\sigma_D^{-1} r_{geom}(s,k)\right) + w_{photo} \cdot \rho_{Huber}\left(\sigma_p^{-1} r_{photo}(s,k)\right) \right)$$

$$I\left(\pi_{I,k}(\boldsymbol{s}_1)\right) \quad I\left(\pi_{I,k}(\boldsymbol{p}_s)\right)$$
$$I\left(\pi_{I,k}(\boldsymbol{s}_2)\right) \quad I\left(\pi_{I,k}(\boldsymbol{p}_s)\right)$$

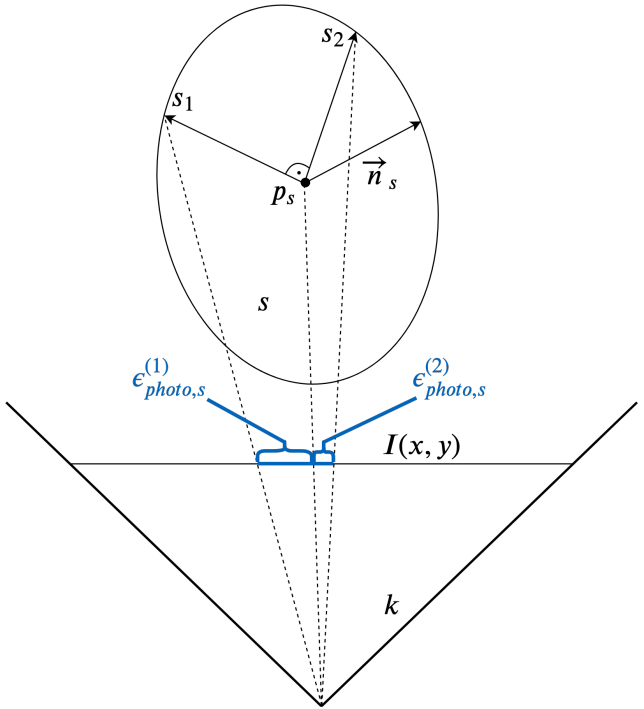


Method – Cost function

Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey}\left(\sigma_D^{-1} r_{geom}(s,k)\right) + w_{photo} \cdot \rho_{Huber}\left(\sigma_p^{-1} r_{photo}(s,k)\right) \right)$$

$$\left\| \begin{matrix} I\left(\pi_{I,k}(\boldsymbol{s}_1)\right) - I\left(\pi_{I,k}(\boldsymbol{p}_s)\right) \\ I\left(\pi_{I,k}(\boldsymbol{s}_2)\right) - I\left(\pi_{I,k}(\boldsymbol{p}_s)\right) \end{matrix} \right\|_2 = \left\| \begin{matrix} \epsilon^{(1)}_{photo,s} \\ \epsilon^{(2)}_{photo,s} \end{matrix} \right\|_2$$

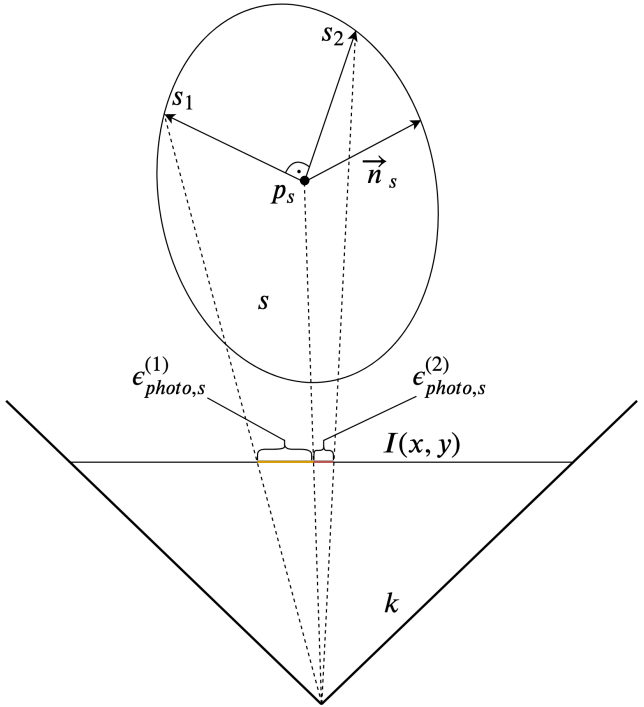


Method – Cost function

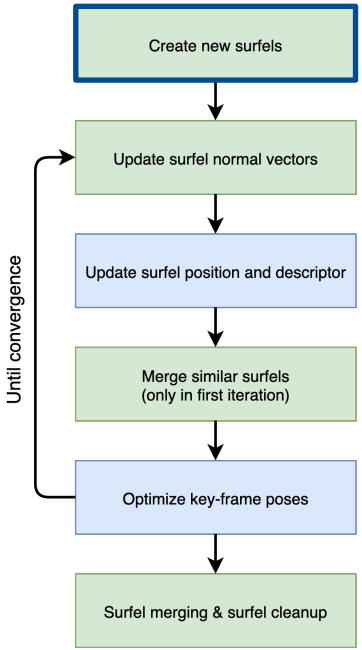
Geometric cost + Photometric cost

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey}\left(\sigma_D^{-1} r_{geom}(s,k)\right) + w_{photo} \cdot \rho_{Huber}\left(\sigma_p^{-1} r_{photo}(s,k)\right) \right)$$

$$r_{photo}(s,k) = \left\| \begin{matrix} I\left(\pi_{I,k}(\boldsymbol{s}_1)\right) - I\left(\pi_{I,k}(\boldsymbol{p}_s)\right) \\ I\left(\pi_{I,k}(\boldsymbol{s}_2)\right) - I\left(\pi_{I,k}(\boldsymbol{p}_s)\right) \end{matrix} \right\|_2 - d_s = \left\| \begin{matrix} \epsilon^{(1)}_{photo,s} \\ \epsilon^{(2)}_{photo,s} \end{matrix} \right\|_2 - d_s$$



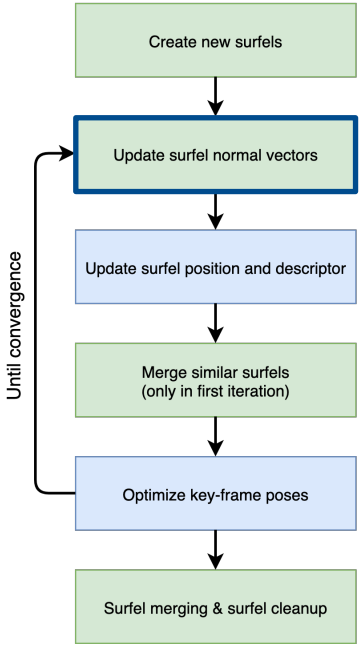
Method – Back-end



- **Create surfel if no measurement in 4x4 pixel cells corresponds to surfel:**

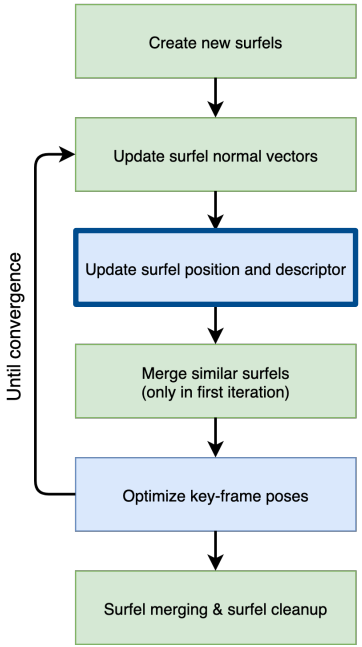
  $\boldsymbol{p}_s \leftarrow \boldsymbol{T}_k^G \pi_{D,k}^{-1}(p)$
  $\boldsymbol{n}_s \leftarrow$ centered finite differences on depth image
  $r_s \leftarrow$ minimum distance between $\boldsymbol{p}_s$ and the 3D 4-neighborhood of p
  $d_s \leftarrow$ according to photometric residual

Method – Back-end



- **Average all corresponding measurements $\pi_{D,k}^{-1}(p_k)$**
- **Renormalize to unit length**

Method – Back-end



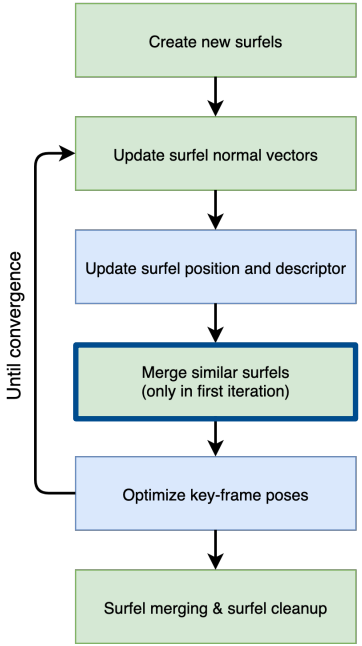
- **Optimizing $C(K,S)$ w.r.t. t and d_s with Gauss Newton:**

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

- **Update rule for position:**

$$\boldsymbol{p}_s^{(1)} = \boldsymbol{p}_s^{(0)} + t \cdot \vec{\boldsymbol{n}}_s$$

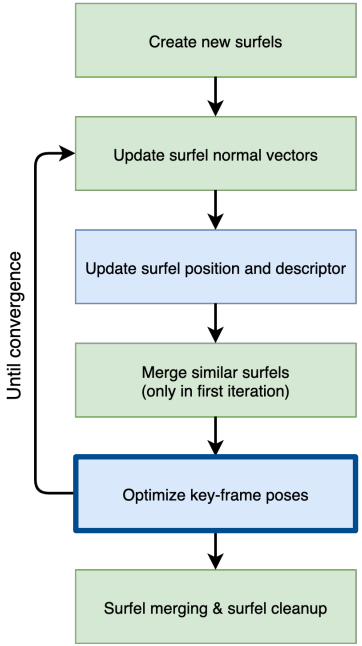
Method – Back-end



- **Merge two surfels s_1 and s_2 if:**

$$\angle(n_{s_1}, n_{s_2}) < 40° \quad + \quad d(p_{s_1}, p_{s_2}) < 4 \cdot 0.8 \cdot \min(r_{s_1}, r_{s_2})$$

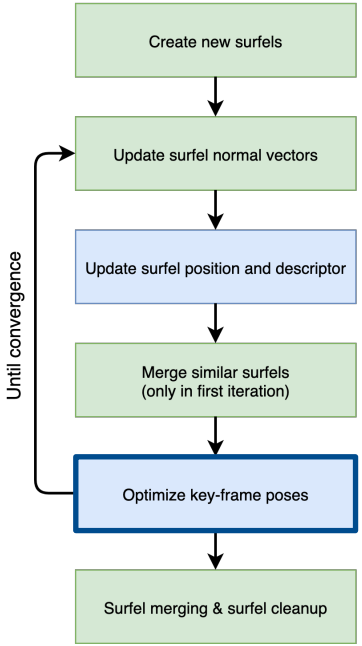
Method – Back-end



- **Optimizing $C(K,S)$ w.r.t. T_G^k with Gauss Newton:**

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

Method – Back-end



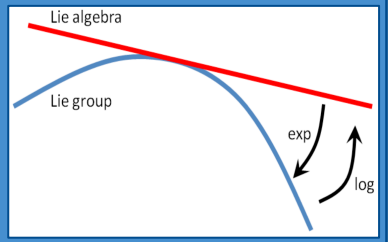
- **Optimizing $C(K,S)$ w.r.t. T_G^k with Gauss Newton:**

$$C(K,S) = \sum_{k \in K} \sum_{s \in S_k} \left(\rho_{Tukey} \left(\sigma_D^{-1} r_{geom}(s,k) \right) + w_{photo} \cdot \rho_{Huber} \left(\sigma_p^{-1} r_{photo}(s,k) \right) \right)$$

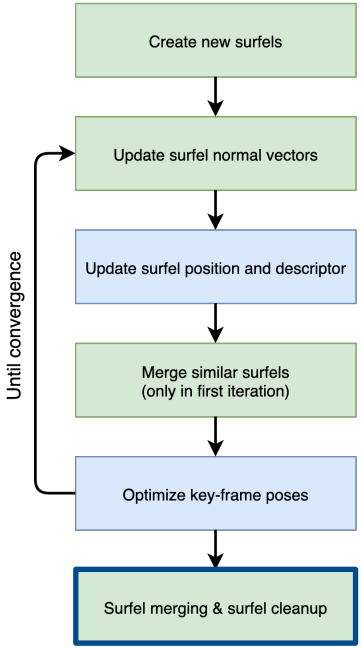
In lie algebra $\mathfrak{se}(3)$

- T_G^k belongs to lie group $SE(3)$
- Pose updates parametrized as local updates in $\mathfrak{se}(3)$

→Ensures valid transformations



Method – Back-end



- **Similar surfels are merged**
- **Radius is min. of corresponding measurements**
- **Outlier-filter is applied**

Method – Dataset

	fr1/desk	fr2/xyz	fr3/office	avg. rank
BundleFusion [7]	**1.6 (1)**	1.1 (3)	2.2 (4)	2.7 (2)
DVO SLAM [30]	2.1 (5)	1.8 (6)	3.5 (8)	6.3 (6)
ElasticFusion [69]	2.0 (4)	1.1 (3)	1.7 (2)	3.0 (4)
Kintinuous [68]	3.7 (8)	2.9 (9)	3.0 (6)	7.7 (8)
MRSMap [60]	4.3 (9)	2.0 (7)	4.2 (9)	8.3 (9)
ORB-SLAM2 [44]	**1.6 (1)**	**0.4 (1)**	**1.0 (1)**	**1.0 (1)**
PSM SLAM [70]	**1.6**	-	3.1	-
RGB-D SLAM [9]	2.3 (6)	0.8 (2)	3.2 (7)	5.0 (5)
VoxelHashing [47]	2.3 (6)	2.2 (8)	2.3 (5)	6.3 (6)
Ours (fixed intr.)	3.6	1.2	2.5	-
Ours	1.7 (3)	1.1 (3)	1.7 (2)	2.7 (2)

Results on the TUM RGB-D benchmark[4]

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Method – Dataset

	fr1/desk	fr2/xyz	fr3/office	avg. rank
BundleFusion [7]	**1.6 (1)**	1.1 (3)	2.2 (4)	2.7 (2)
DVO SLAM [30]	2.1 (5)	1.8 (6)	3.5 (8)	6.3 (6)
ElasticFusion [69]	2.0 (4)	1.1 (3)	1.7 (2)	3.0 (4)
Kintinuous [68]	3.7 (8)	2.9 (9)	3.0 (6)	7.7 (8)
MRSMap [60]	4.3 (9)	2.0 (7)	4.2 (9)	8.3 (9)
ORB-SLAM2 [44]	**1.6 (1)**	**0.4 (1)**	**1.0 (1)**	**1.0 (1)**
PSM SLAM [70]	**1.6**	-	3.1	-
RGB-D SLAM [9]	2.3 (6)	0.8 (2)	3.2 (7)	5.0 (5)
VoxelHashing [47]	2.3 (6)	2.2 (8)	2.3 (5)	6.3 (6)
Ours (fixed intr.)	3.6	1.2	2.5	-
Ours	1.7 (3)	1.1 (3)	1.7 (2)	2.7 (2)

Results on the TUM RGB-D benchmark[4]

Observations:

- Optimizing **camera intrinstics** and **depth distortion** improves the result of BAD SLAM.

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Method – Dataset

	fr1/desk	fr2/xyz	fr3/office	avg. rank
BundleFusion [7]	**1.6 (1)**	1.1 (3)	2.2 (4)	2.7 (2)
DVO SLAM [30]	2.1 (5)	1.8 (6)	3.5 (8)	6.3 (6)
ElasticFusion [69]	2.0 (4)	1.1 (3)	1.7 (2)	3.0 (4)
Kintinuous [68]	3.7 (8)	2.9 (9)	3.0 (6)	7.7 (8)
MRSMap [60]	4.3 (9)	2.0 (7)	4.2 (9)	8.3 (9)
ORB-SLAM2 [44]	**1.6 (1)**	**0.4 (1)**	**1.0 (1)**	**1.0 (1)**
PSM SLAM [70]	**1.6**	-	3.1	-
RGB-D SLAM [9]	2.3 (6)	0.8 (2)	3.2 (7)	5.0 (5)
VoxelHashing [47]	2.3 (6)	2.2 (8)	2.3 (5)	6.3 (6)
Ours (fixed intr.)	3.6	1.2	2.5	-
Ours	1.7 (3)	1.1 (3)	1.7 (2)	2.7 (2)

Results on the TUM RGB-D benchmark[4]

Observations:

- Optimizing **camera intrinstics** and **depth distortion** improves the result of BAD SLAM.
- ORB-SLAM2 (**indirect**) outperforms all direct methods.

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Method – Dataset

	fr1/desk	fr2/xyz	fr3/office	avg. rank
BundleFusion [7]	**1.6 (1)**	1.1 (3)	2.2 (4)	2.7 (2)
DVO SLAM [30]	2.1 (5)	1.8 (6)	3.5 (8)	6.3 (6)
ElasticFusion [69]	2.0 (4)	1.1 (3)	1.7 (2)	3.0 (4)
Kintinuous [68]	3.7 (8)	2.9 (9)	3.0 (6)	7.7 (8)
MRSMap [60]	4.3 (9)	2.0 (7)	4.2 (9)	8.3 (9)
ORB-SLAM2 [44]	**1.6 (1)**	**0.4 (1)**	**1.0 (1)**	**1.0 (1)**
PSM SLAM [70]	**1.6**	-	3.1	-
RGB-D SLAM [9]	2.3 (6)	0.8 (2)	3.2 (7)	5.0 (5)
VoxelHashing [47]	2.3 (6)	2.2 (8)	2.3 (5)	6.3 (6)
Ours (fixed intr.)	3.6	1.2	2.5	-
Ours	1.7 (3)	1.1 (3)	1.7 (2)	2.7 (2)

Results on the TUM RGB-D benchmark[4]

Observations:

- Optimizing **camera intrinstics** and **depth distortion** improves the result of BAD SLAM.
- ORB-SLAM2 (**indirect**) outperforms all direct methods.

→ Depth distortions, rolling shutter and asynchronity hinder fair comparison.

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Method – Dataset

	fr1/desk	fr2/xyz	fr3/office	avg. rank
BundleFusion [7]	**1.6 (1)**	1.1 (3)	2.2 (4)	2.7 (2)
DVO SLAM [30]	2.1 (5)	1.8 (6)	3.5 (8)	6.3 (6)
ElasticFusion [69]	2.0 (4)	1.1 (3)	1.7 (2)	3.0 (4)
Kintinuous [68]	3.7 (8)	2.9 (9)	3.0 (6)	7.7 (8)
MRSMap [60]	4.3 (9)	2.0 (7)	4.2 (9)	8.3 (9)
ORB-SLAM2 [44]	**1.6 (1)**	**0.4 (1)**	**1.0 (1)**	**1.0 (1)**
PSM SLAM [70]	**1.6**	-	3.1	-
RGB-D SLAM [9]	2.3 (6)	0.8 (2)	3.2 (7)	5.0 (5)
VoxelHashing [47]	2.3 (6)	2.2 (8)	2.3 (5)	6.3 (6)
Ours (fixed intr.)	3.6	1.2	2.5	-
Ours	1.7 (3)	1.1 (3)	1.7 (2)	2.7 (2)

Results on the TUM RGB-D benchmark[4]

Observations:

- Optimizing **camera intrinstics** and **depth distortion** improves the result of BAD SLAM.
- ORB-SLAM2 (**indirect**) outperforms all direct methods.

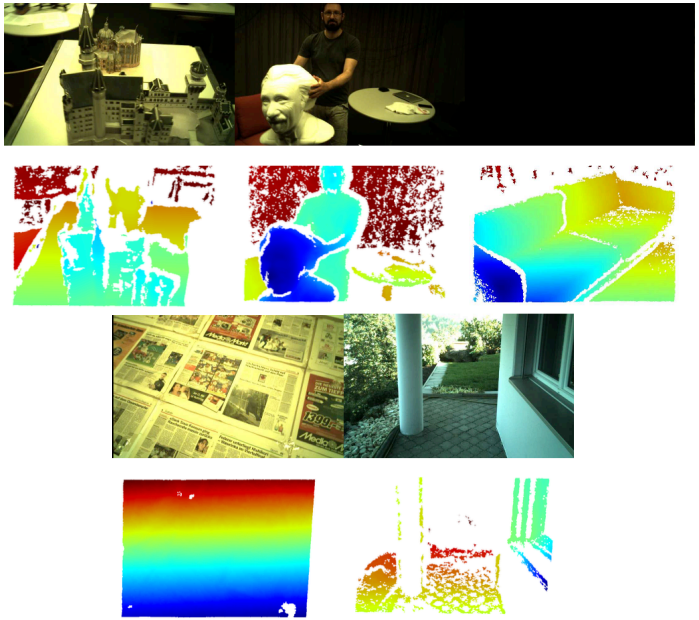
→ Depth distortions, rolling shutter and asynchronity hinder fair comparison.

→ Rather adress these problems in hardware.
→ Create a comparable benchmark for direct methods.

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Method – Dataset

Outline
- Frame synchronization (Depth & RGB)
- Active stereo for depth
- Ground truth poses through motion capturing
- Internally calibrated depth camera
- Non-public ground truth data



Dataset samples[4]

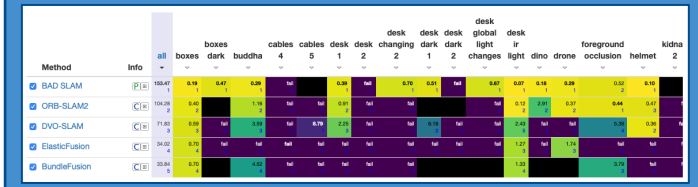
[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Method – Dataset

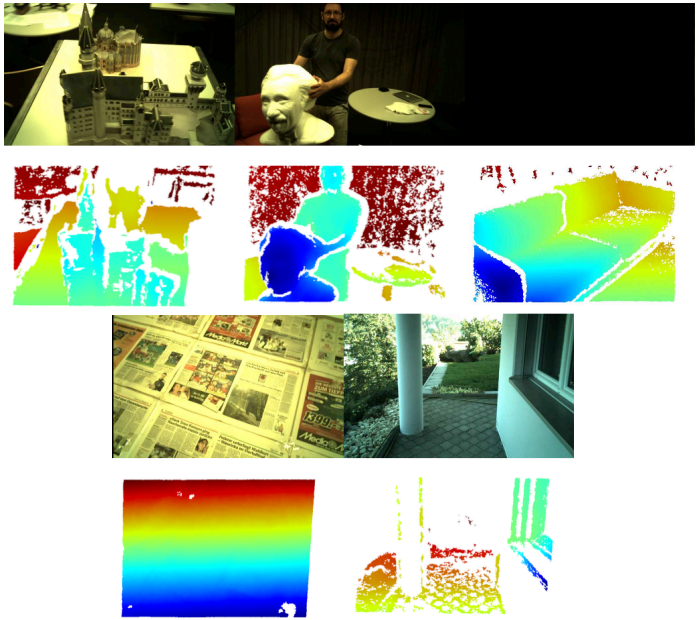
Outline

- Frame synchronization (Depth & RGB)
- Active stereo for depth
- Ground truth poses through motion capturing
- Internally calibrated depth camera
- Non-public ground truth data

Online leaderboard



https://www.eth3d.net/slam_benchmark



Dataset samples[4]

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Contribution

Real-time direct BA SLAM framework

- Cost function
- Alternating optimization

Contribution

Real-time direct BA SLAM framework

- Cost function
- Alternating optimization

Dataset

- No depth distortion, no rolling shutter and sync. frames
- Hidden ground truth

Contribution

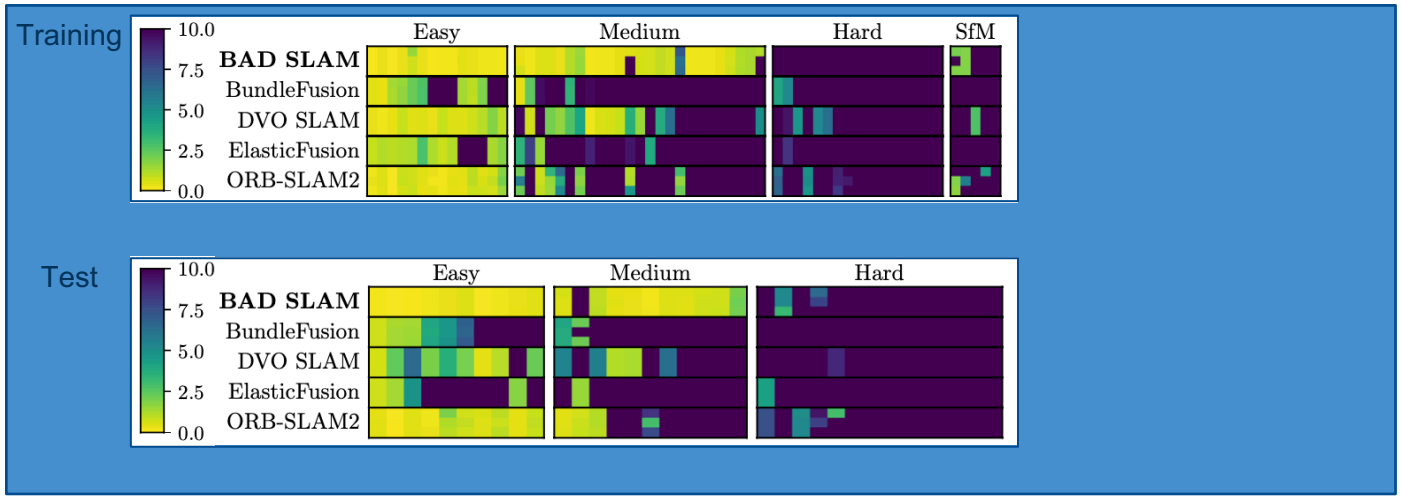
Real-time direct BA SLAM framework
- Cost function
- Alternating optimization

Dataset
- No depth distortion, no rolling shutter and sync. frames
- Hidden ground truth

Insights into susceptibility of direct methods to differnt disturbances

Results – Accuracy

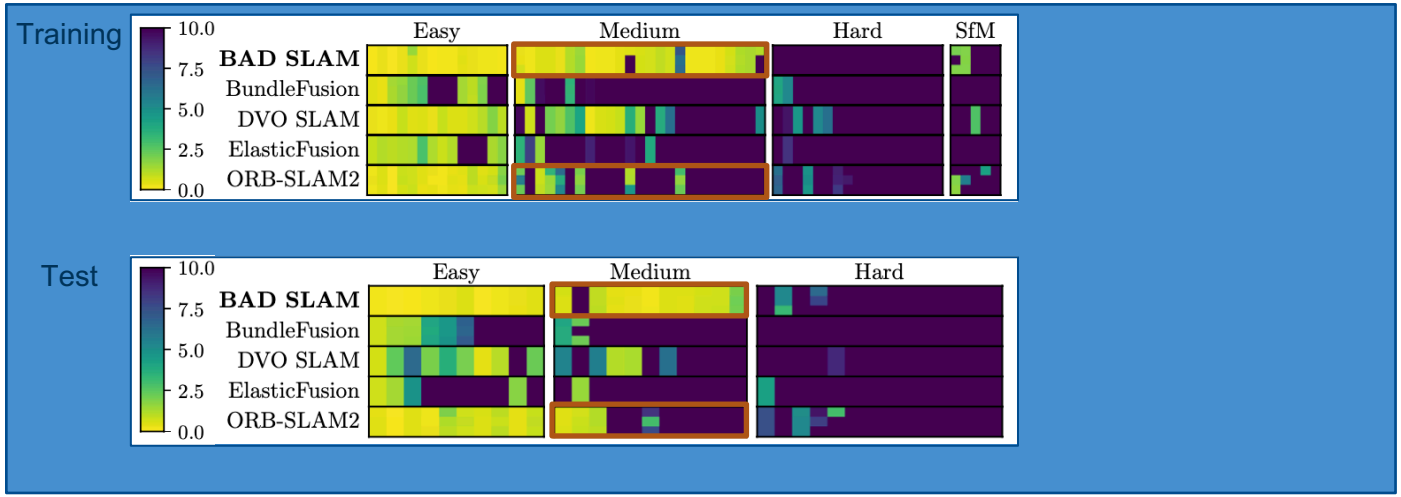
SE(3) ATE RMSE new dataset[4]



[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Results – Accuracy

SE(3) ATE RMSE new dataset[4]

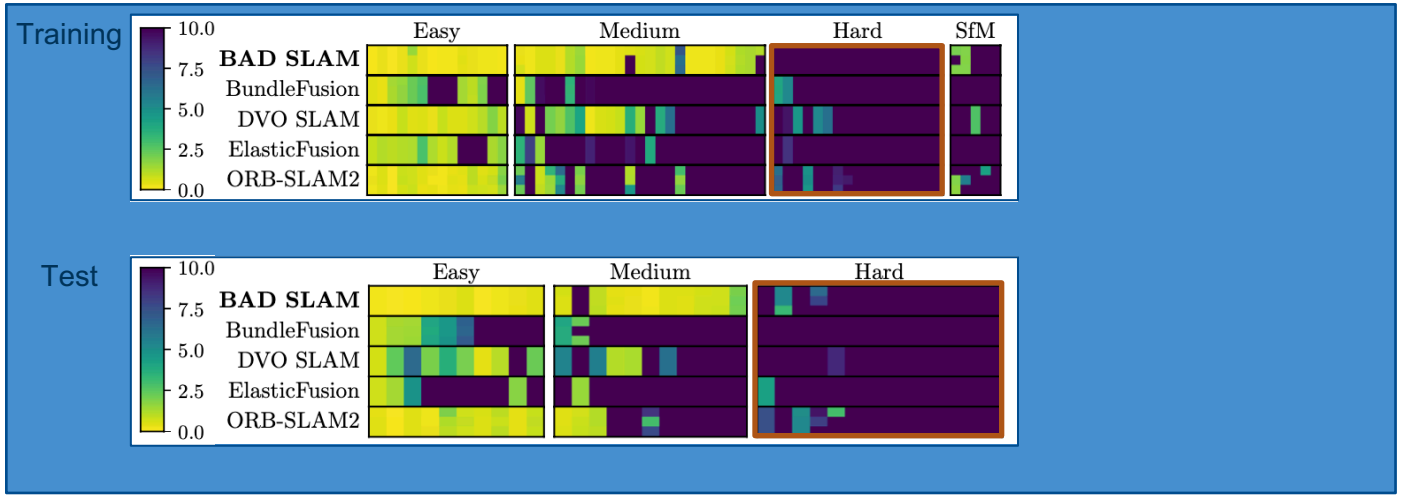


→BAD SLAM outperforms other methods on new dataset

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Results – Accuracy

SE(3) ATE RMSE new dataset[4]



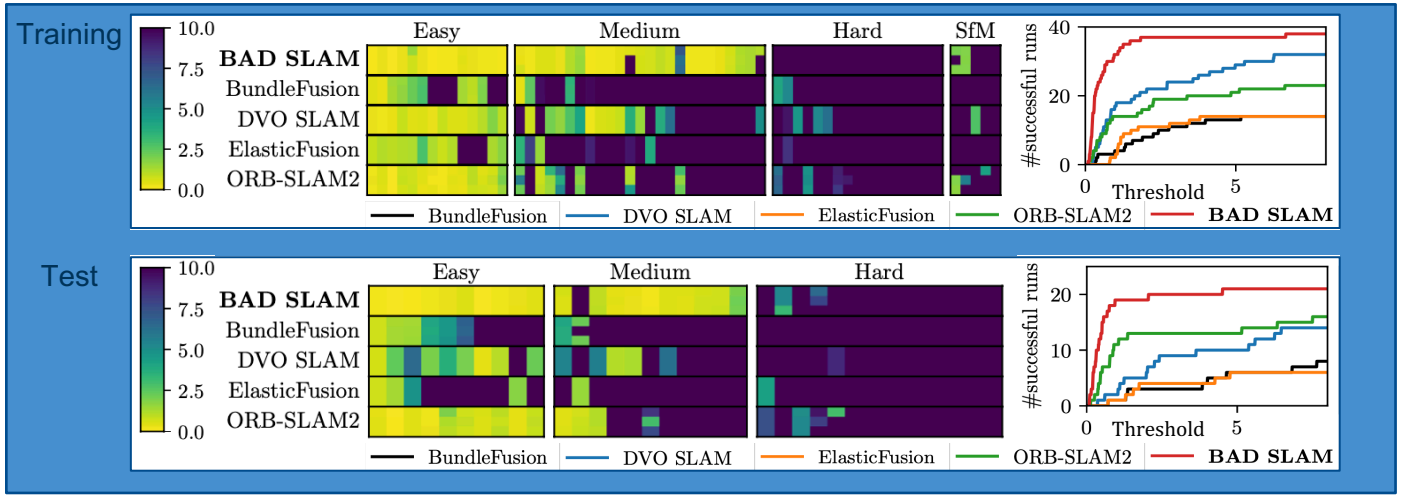
→BAD SLAM outperforms other methods on new dataset

→Hard cases remain an open challenge

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Results – Accuracy

SE(3) ATE RMSE new dataset[4]



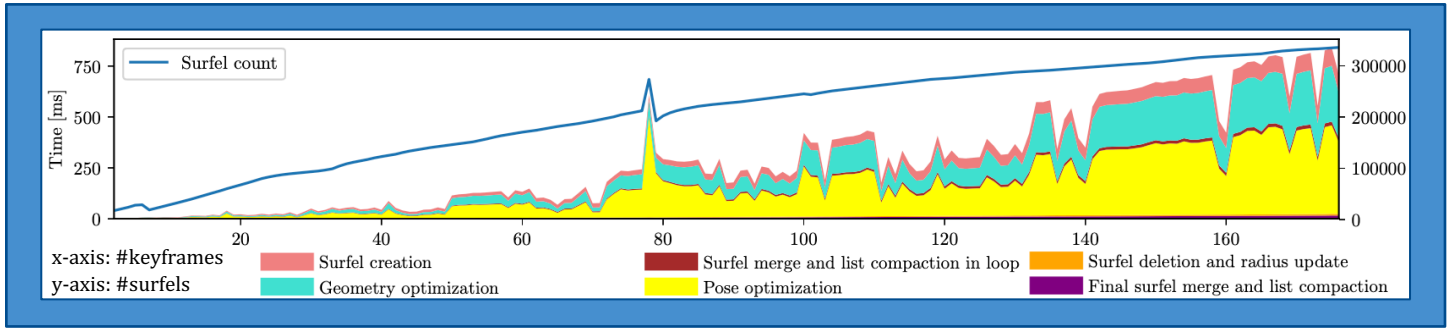
→BAD SLAM outperforms other methods on new dataset

→Hard cases remain an open challenge

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Results – Runtime

Runtime of back-end (not scipping BA)[4]

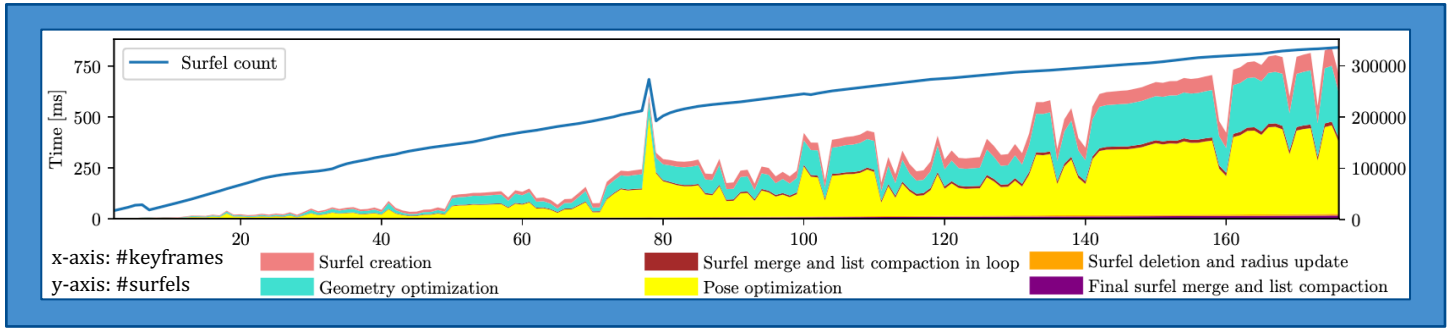


→Time for odometry is negligible

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Results – Runtime

Runtime of back-end (not scipping BA)[4]



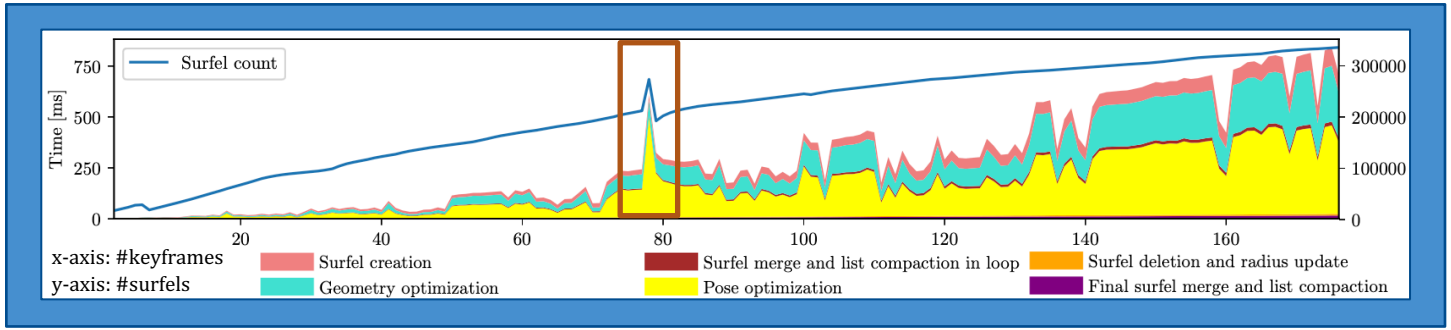
→Time for odometry is negligible

→Geometry and pose optimization take the most time

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Results – Runtime

Runtime of back-end (not scipping BA)[4]



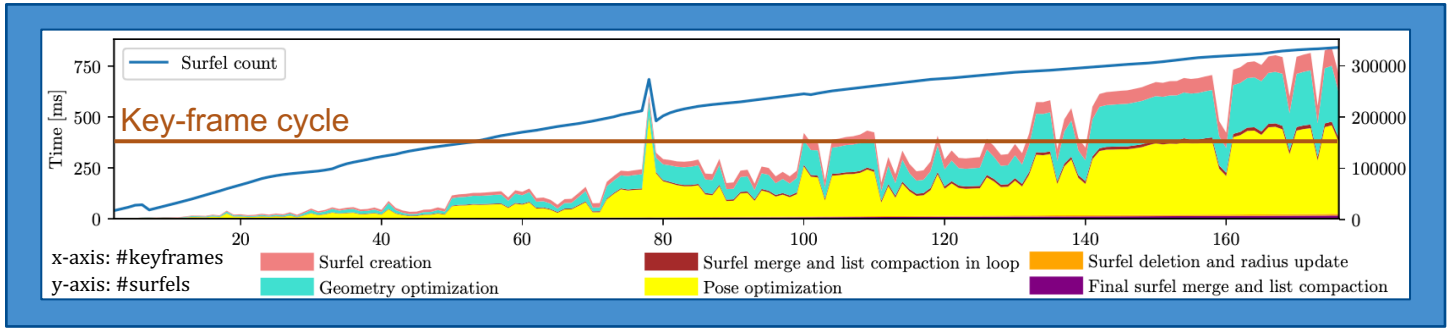
→Time for odometry is negligible

→Geometry and pose optimization take the most time

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Results – Runtime

Runtime of back-end (not scipping BA)[4]



→Time for odometry is negligible

→Geometry and pose optimization take the most time

→Real-time will start to degrade at keyframe #100

[4] BAD SLAM: Bundle Adjusted Direct RGB-D SLAM, Schöps et. al., 2019

Personal comments

Dislike

Questioning scalability

→How many BA iterations are acutally skipped?

→When does the system start to degrade and how much?

Personal comments

Dislike

Questioning scalability

→How many BA iterations are acutally skipped?

→When does the system start to degrade and how much?

Applications

→Is outperfomed on real/disturbed data

Personal comments

Dislike

Questioning scalability

→How many BA iterations are acutally skipped?

→When does the system start to degrade and how much?

Applications

→Is outperfomed on real/disturbed data

Like

Suplementary matterial

→In depth description on how they recorded their dataset

Personal comments

Dislike

Questioning scalability

→How many BA iterations are acutally skipped?

→When does the system start to degrade and how much?

Applications

→Is outperfomed on real/disturbed data

Like

Suplementary matterial

→In depth description on how they recorded their dataset

Open-source code

→With some changes to e.g. photometric residual

→Very transparent

Summary

- BAD-SLAM proposed a **new back-end** algorithm for realtime direct bundle adjustment.

Summary

- BAD-SLAM proposed a **new back-end** algorithm for realtime direct bundle adjustment.
- **Outperforms state of the art** methods on small scenes.

Summary

- BAD-SLAM proposed a **new back-end** algorithm for realtime direct bundle adjustment.
- **Outperforms state of the art** methods on small scenes.
- A new dataset enables **better comparison** of SLAM methods, especially direct ones.

Summary

- BAD-SLAM proposed a **new back-end** algorithm for realtime direct bundle adjustment.
- **Outperforms state of the art** methods on small scenes.
- A new dataset enables **better comparison** of SLAM methods, especially direct ones.
- **Unmodeled effects** such as rolling shutter, async. frames and geometric distortions drastically **degrade SLAM**.

Summary

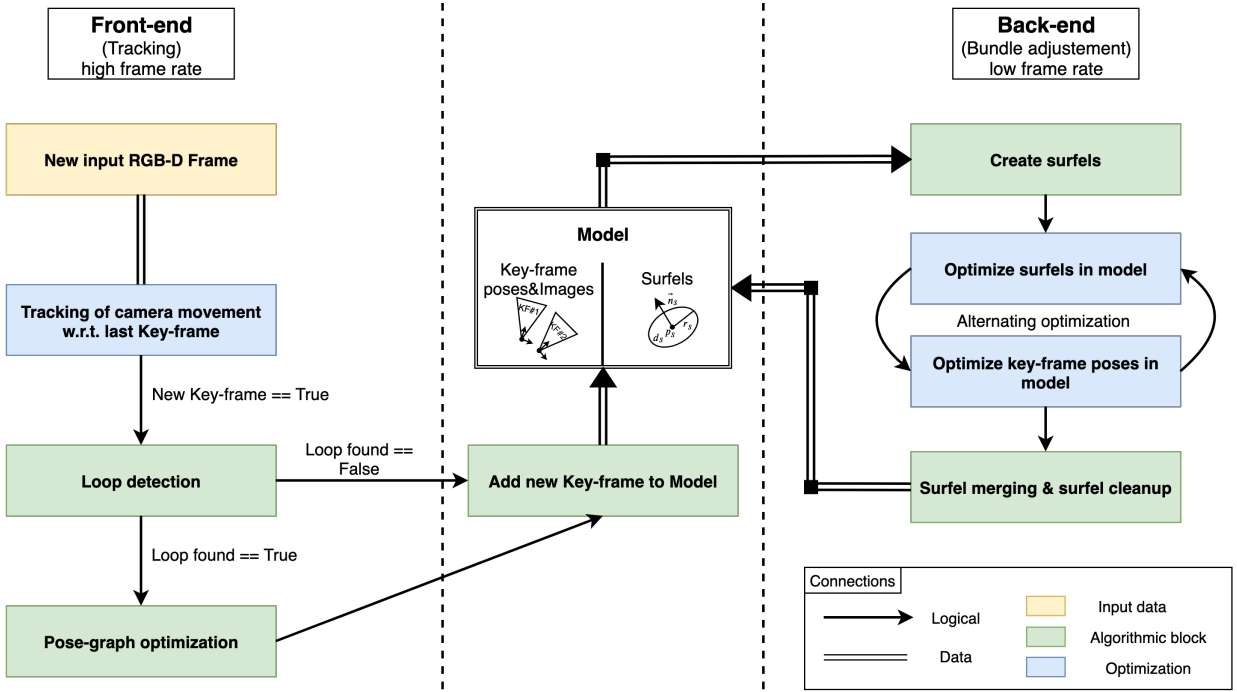
- BAD-SLAM proposed a **new back-end** algorithm for realtime direct bundle adjustment.
- **Outperforms state of the art** methods on small scenes.
- A new dataset enables **better comparison** of SLAM methods, especially direct ones.
- **Unmodeled effects** such as rolling shutter, async. frames and geometric distortions drastically **degrade SLAM**.

Further improvements:
→Key-frame selection
→Windowed BA
→Improving on hard cases

Thank you for your attention!

Michael Gentner (TUM)

Discussion


Front-end
(Tracking)
high frame rate
Back-end
(Bundle adjustement)
low frame rate
New input RGB-D Frame
Tracking of camera movement w.r.t. last Key-frame
New Key-frame == True
Loop detection
Loop found == False
Loop found == True
Pose-graph optimization
Add new Key-frame to Model
Model
Key-frame poses&Images
Surfels
Create surfels
Optimize surfels in model
Alternating optimization
Optimize key-frame poses in model
Surfel merging & surfel cleanup
Connections
Logical
Data
Input data
Algorithmic block
Optimization