ElasticFusion: Real-Time Dense SLAM and Light Source Estimation

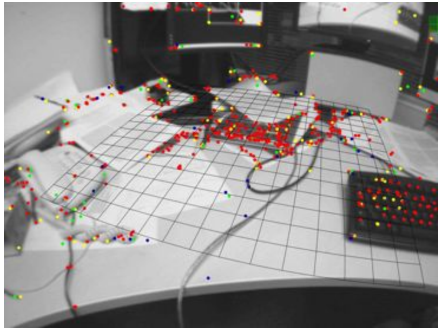
Fedor Fedoseev

ElasticFusion: Real-Time Dense SLAM and Light Source Estimation

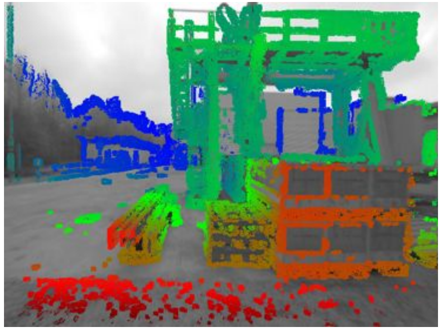
Aim: capturing comprehensive dense *globally consistent* surfel-based maps of room scale environments

- using an RGB-D camera
- in an incremental online fashion
- without any post-processing step

Terminology



sparse

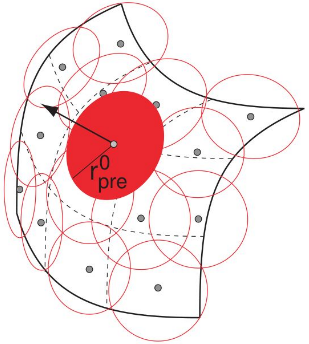


semi-dense

⟹ ***dense***

Surfel-based: "Surface Element", a rendering primitive. Information about:

- position
- normal
- color
- other (e.g. weight, radius, init time stamp, last updated timestamp)



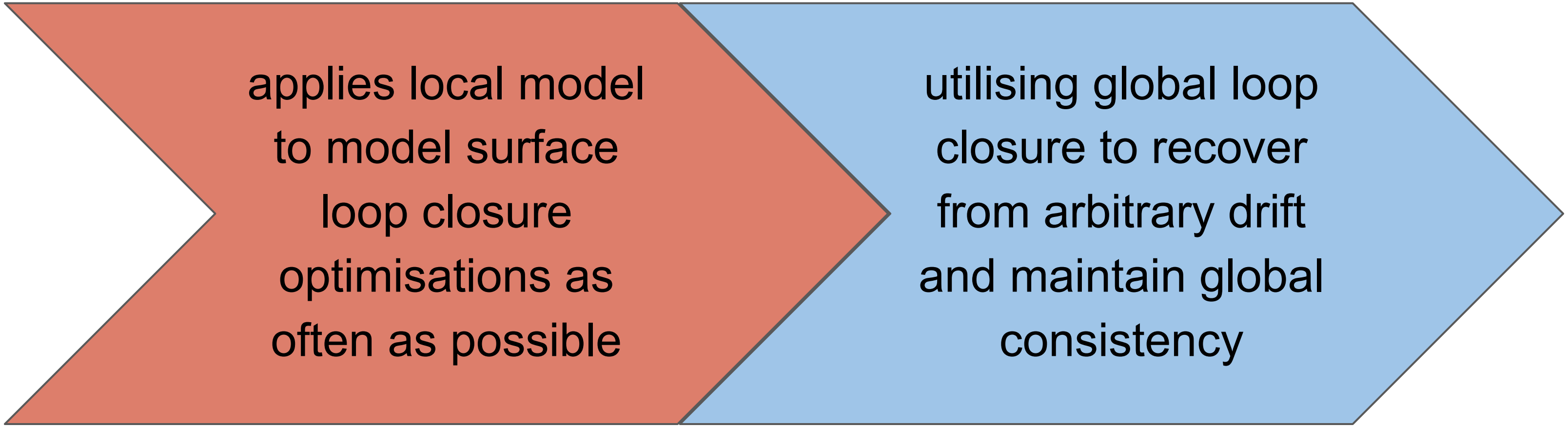
Motivation

Problem: Real-time operation struggles when the sensor makes movements which are both:

1. of extended duration and
2. often criss-cross loop back on themselves

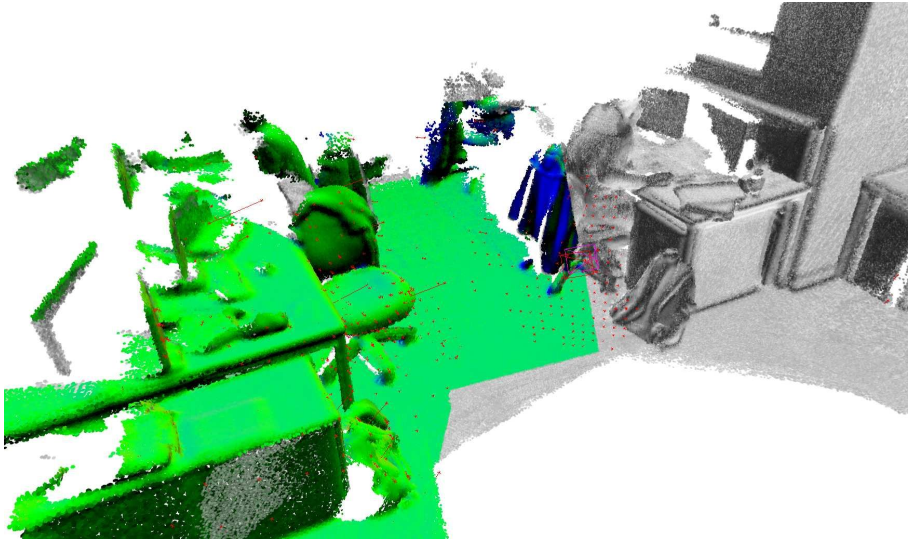
For dense vision the number of points matched and measured at each sensor frame is much higher than in feature-based systems

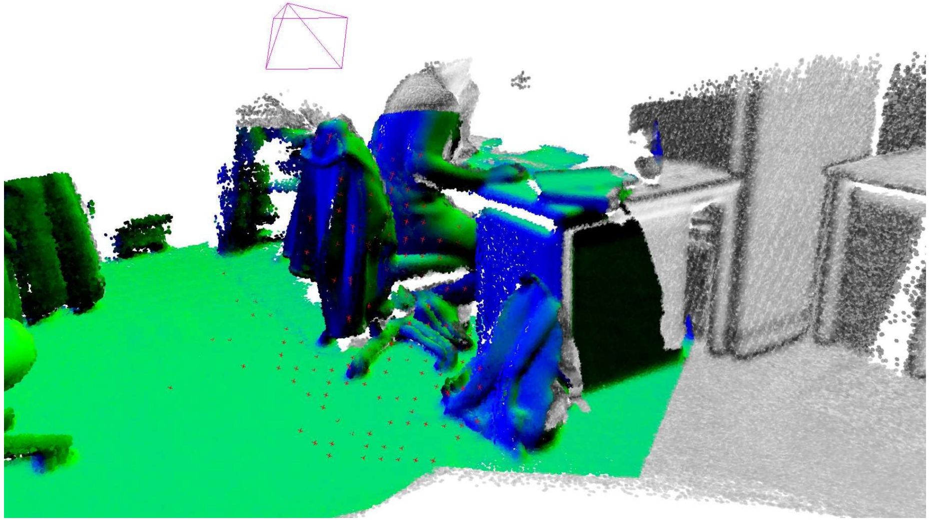
Map-centric approach



Approach

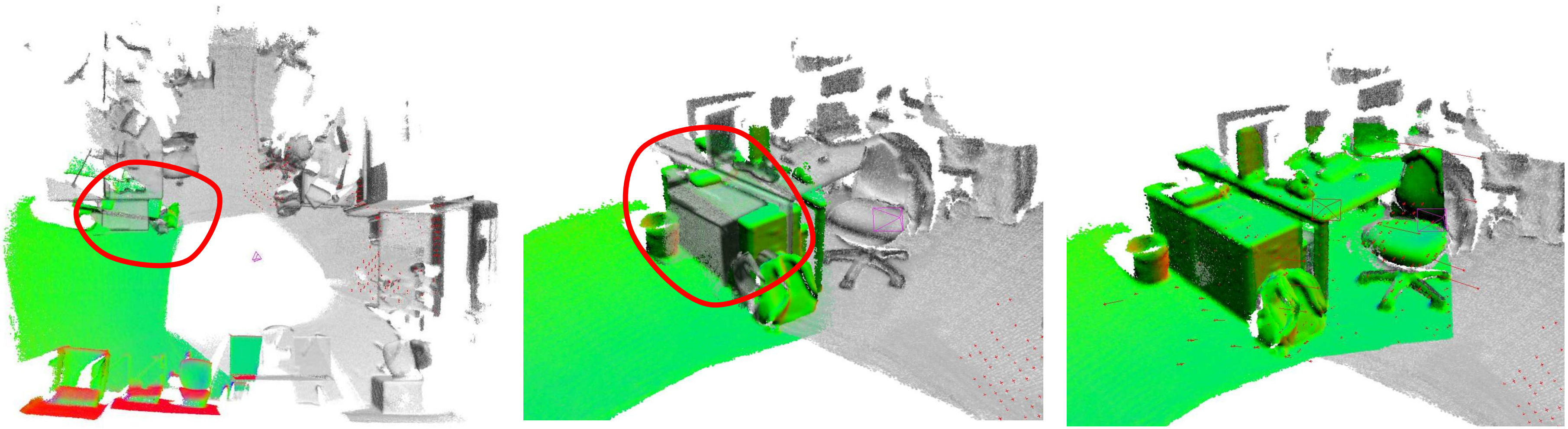
estimate model → split model into active/inactive areas → local loop closure → global loop closure





Approach

global loop closure



Fused Predicted Tracking

- Geometric Pose Estimation: point to plane ICP

$$E_{icp} = \sum_{k} \left(\left(\mathbf{v}^k - \exp(\hat{\boldsymbol{\xi}}) \mathbf{T} \mathbf{v}_t^k \right) \cdot \mathbf{n}^k \right)^2$$

- Photometric Pose Estimation

$$E_{rgb} = \sum_{\mathbf{u} \in \Omega} \left(I(\mathbf{u}, \mathcal{C}_t^l) - I\left(\boldsymbol{\pi}(\mathbf{K} \exp(\hat{\boldsymbol{\xi}}) \mathbf{T} \mathbf{p}(\mathbf{u}, \mathcal{D}_t^l)), \hat{\mathcal{C}}_{t-1}^a \right) \right)^2$$

- Joint Cost Function

$$E_{track} = E_{icp} + w_{rgb} E_{rgb}$$

Deformation Graph

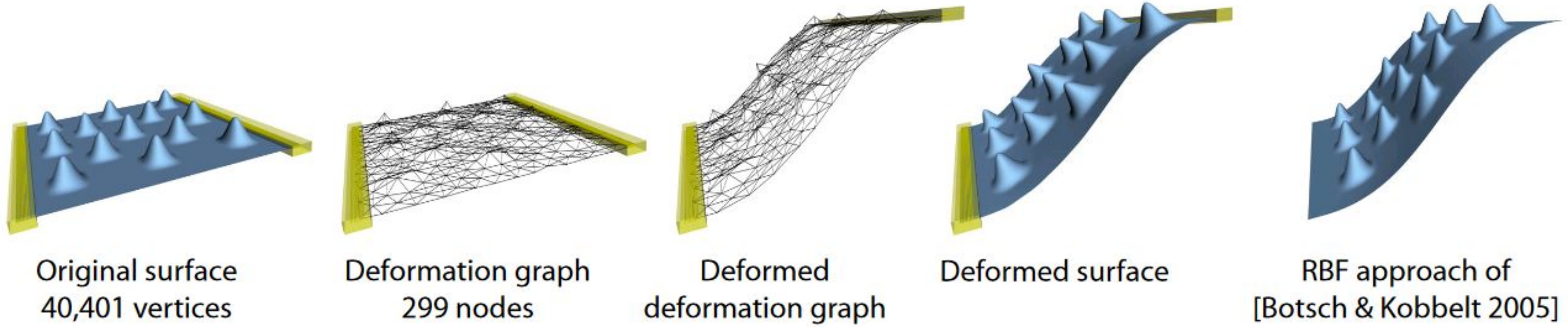
Nodes: {*timestamp, position, set of neighbours, affine transformation* $\mathbf{G_R}$*, 3x1 vector* $\mathbf{G_t}$ *initialized to identity and zeroes*}
Edges: connecting neighbours up to count k

these parameters of each node are optimized when deforming surface

- ❖ For non-rigid deformations in loop closures
- ❖ New deformation graph each frame for the set of surfels
- ❖ Each Surfel is deformed by influencing Deformation Nodes
  - ➢ Search for the closest node in time
  - ➢ Take k-nearest (influencing) nodes based on Euclidean distance
  - ➢ Weights computation
  - ➢ Transformation applied to the surfels

Deformation Graph Optimization

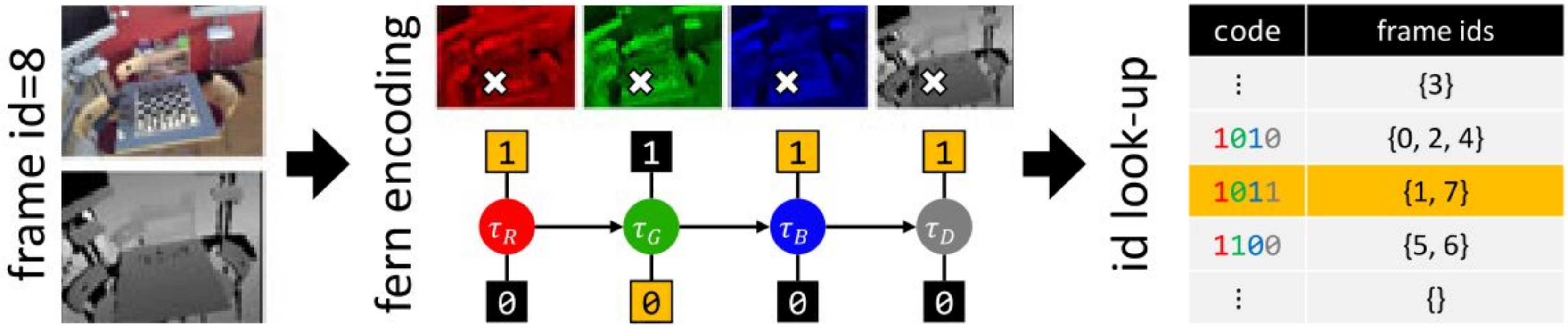
- Preserve Details - Affine transformations should be rotations
- Smooth overlapping transformations
- Minimize error of position constraints
- Deform active area into inactive. Pin the inactive area
- Prevent surface registrations from being pulled apart



Local loop closure

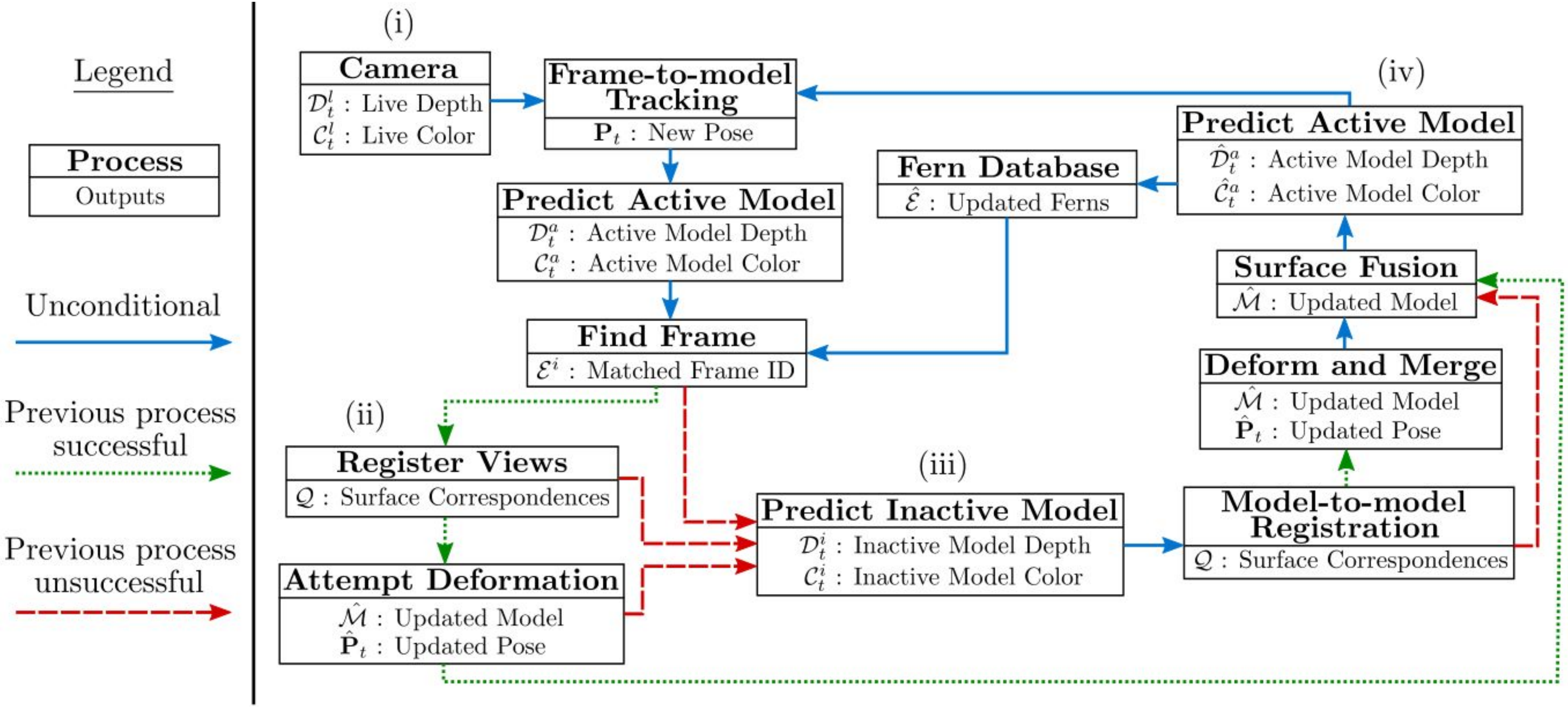
- For each frame divide set of surfels into 2 disjoint sets: Θ and Ψ (active and inactive)
- In each frame if global loop was not detected, attempt to compute match between Θ and Ψ by registering their predicted surface renderings from the latest pose estimate using fused predicted tracking approach
- Output is relative transformation matrix $H \in SE3$ from Θ to Ψ which brings 2 predicted surface renderings into alignment
- To decide on the quality of registration and whether or not to carry out deformation look at final cost of tracking optimization E_{track}

Global loop closure



- Surfaces constraints are added to the Deformation Graph

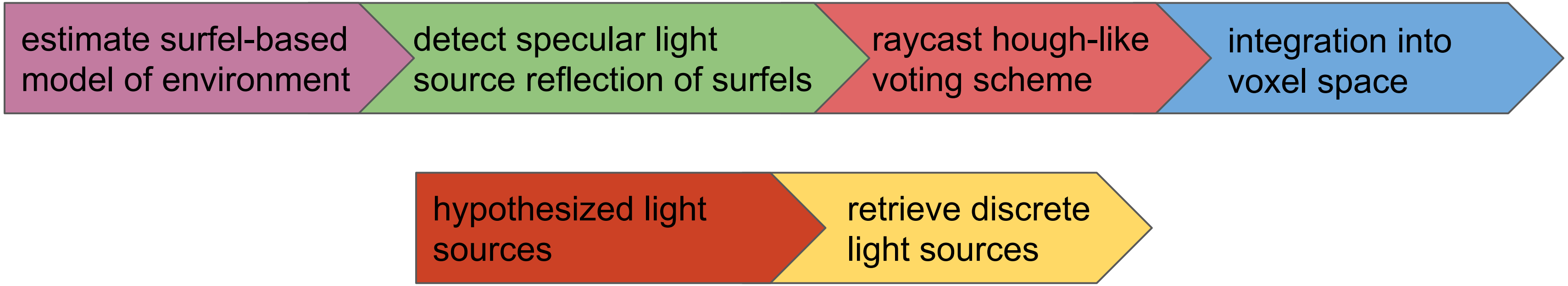
$$Q = (Q^d, Q^s, Q_t^d, Q_t^s)$$


Legend
Process
Outputs
Unconditional
Previous process successful
Previous process unsuccessful
(i)
Camera
$\mathcal{D}_t^l$: Live Depth
$\mathcal{C}_t^l$: Live Color
Frame-to-model Tracking
$\mathbf{P}_t$: New Pose
Predict Active Model
$\mathcal{D}_t^a$: Active Model Depth
$\mathcal{C}_t^a$: Active Model Color
Find Frame
$\mathcal{E}^i$: Matched Frame ID
Fern Database
$\hat{\mathcal{E}}$: Updated Ferns
(iv)
Predict Active Model
$\hat{\mathcal{D}}_t^a$: Active Model Depth
$\hat{\mathcal{C}}_t^a$: Active Model Color
Surface Fusion
$\hat{\mathcal{M}}$: Updated Model
Deform and Merge
$\hat{\mathcal{M}}$: Updated Model
$\hat{\mathbf{P}}_t$: Updated Pose
(ii)
Register Views
$\mathcal{Q}$: Surface Correspondences
Attempt Deformation
$\hat{\mathcal{M}}$: Updated Model
$\hat{\mathbf{P}}_t$: Updated Pose
(iii)
Predict Inactive Model
$\mathcal{D}_t^i$: Inactive Model Depth
$\mathcal{C}_t^i$: Inactive Model Color
Model-to-model Registration
$\mathcal{Q}$: Surface Correspondences

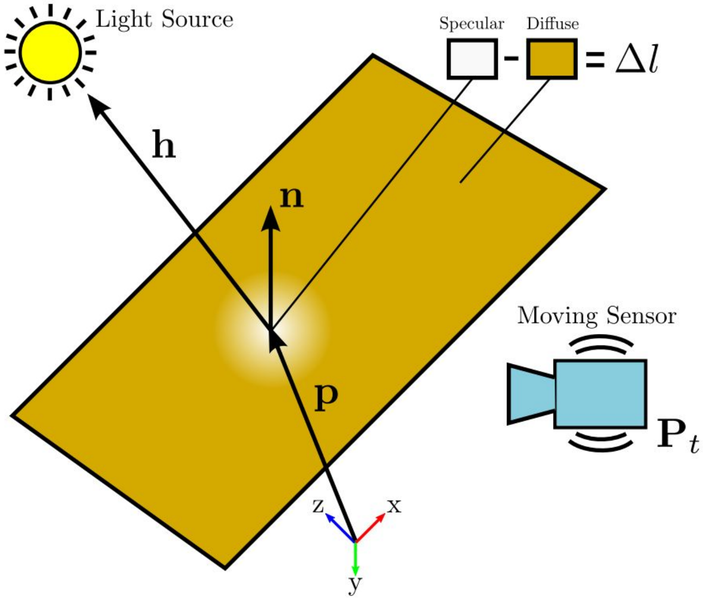
Light Source Estimation

Discrete point light source estimation usability:

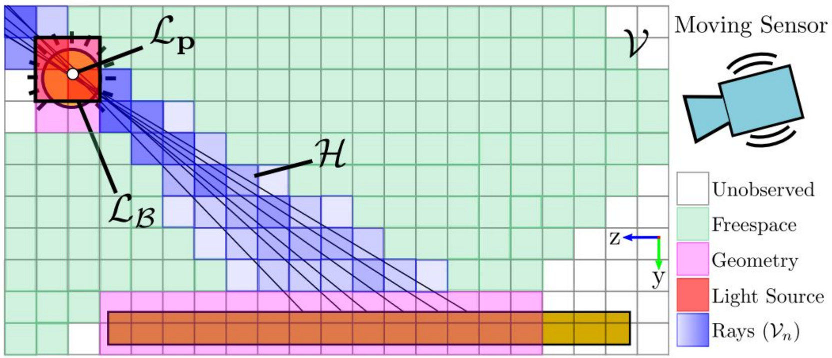
- ❖ predictive tracking
- ❖ path planning
- ❖ real-time augmented reality effects



Specular reflection ray detection



Ray measurement integration into the voxel grid



Summary of results

Improved trajectory estimate of camera

datasets

System	fr1/desk	fr2/xyz	fr3/office	fr3/nst
DVO SLAM	0.021m	0.018m	0.035m	0.018m
RGB-D SLAM	0.023m	**0.008m**	0.032m	0.017m
MRSMap	0.043m	0.020m	0.042m	2.018m
Kintinuous	0.037m	0.029m	0.030m	0.031m
Frame-to-model	0.022m	0.014m	0.025m	0.027m
ElasticFusion	**0.020m**	0.011m	**0.017m**	**0.016m**

real world datasets of Sturm et al. (2012)

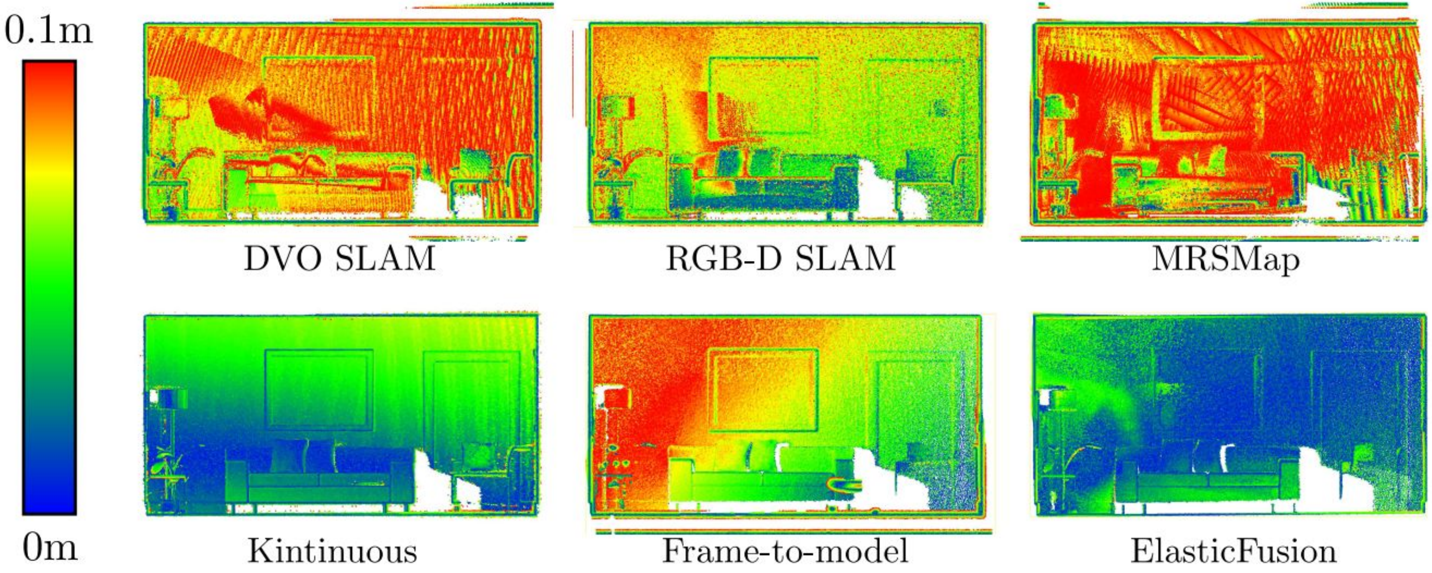
System	kt0	kt1	kt2	kt3
DVO SLAM	0.104m	0.029m	0.191m	0.152m
RGB-D SLAM	0.026m	0.008m	0.018m	0.433m
MRSMap	0.204m	0.228m	0.189m	1.090m
Kintinuous	0.072m	**0.005m**	**0.010m**	0.355m
Frame-to-model	0.497m	0.009m	0.020m	0.243m
ElasticFusion	**0.009m**	0.009m	0.014m	**0.106m**

synthetic datasets of Handa et al. (2012)

ATE RMSE

Summary of results

Improved surface reconstruction quality



heat maps showing reconstruction error on the kt0 dataset

Summary of results

Novel method for detecting multiple discrete point light sources in a scene in real time

Example of application in AR

Experimental Demonstration

https://youtu.be/-dz_VauPjEU?t=295

Questions

Deformation Graph Optimization

- ❖ Preserve Details - Affine transformations should be rotations

$$E_{rot} = \sum_{l} \left\| {\mathcal{G}_{\mathbf{R}}^{l}}^{\top} \mathcal{G}_{\mathbf{R}}^{l} - \mathbf{I} \right\|_F^2$$

- ❖ Smooth overlapping transformations

$$E_{reg} = \sum_{l} \sum_{n \in \mathcal{N}(\mathcal{G}^l)} \left\| \mathcal{G}_{\mathbf{R}}^{l}(\mathcal{G}_{\mathbf{g}}^{n} - \mathcal{G}_{\mathbf{g}}^{l}) + \mathcal{G}_{\mathbf{g}}^{l} + \mathcal{G}_{\mathbf{t}}^{l} - (\mathcal{G}_{\mathbf{g}}^{n} + \mathcal{G}_{\mathbf{t}}^{n}) \right\|_2^2$$

- ❖ Minimize error of position constraints

$$E_{con} = \sum_{p} \| \phi(\mathcal{Q}_{\mathbf{s}}^{p}) - \mathcal{Q}_{\mathbf{d}}^{p} \|_2^2$$

- ❖ Deform active area into inactive. Pin the inactive area

$$E_{pin} = \sum_{p} \| \phi(\mathcal{Q}_{\mathbf{d}}^{p}) - \mathcal{Q}_{\mathbf{d}}^{p} \|_2^2$$

- ❖ Prevent surface registrations from being pulled apart

$$E_{rel} = \sum_{p} \| \phi(\mathcal{R}_{\mathbf{s}}^{p}) - \phi(\mathcal{R}_{\mathbf{d}}^{p}) \|_2^2$$

$$E_{glo} = w_f E_{rot} + w_r E_{reg} + w_c(E_{con} + E_{pin} + E_{rel})$$

$$E_{loc} = w_f E_{rot} + w_r E_{reg} + w_c(E_{con} + E_{pin})$$