BundleFusion:

Real-time Globally Consistent 3D Reconstruction using Online Surface Re-integration

Angela Dai[1]
Matthias Nießner[1,2]
Michael Zollhöfer[4]
Shahram Izadi[3]
Christian Theobalt[4]

[1]*Stanford University*
[2]*Technical University of Munich*
[3]*PerceptiveIO*
[4]*Max-Planck-Institute for Informatics*

SIGGRAPH2017

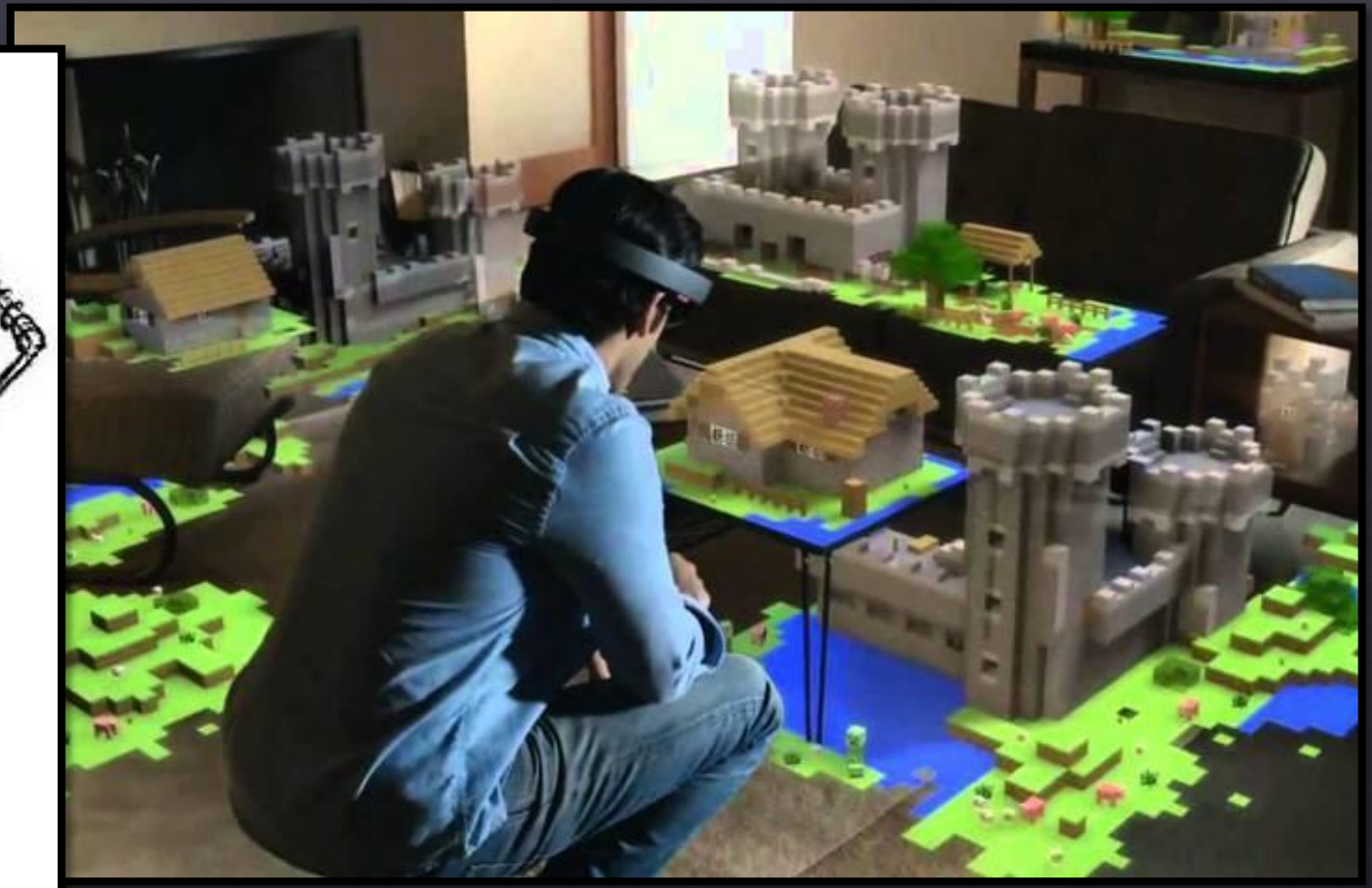
Scanning the World

robotics

autonomous navigation

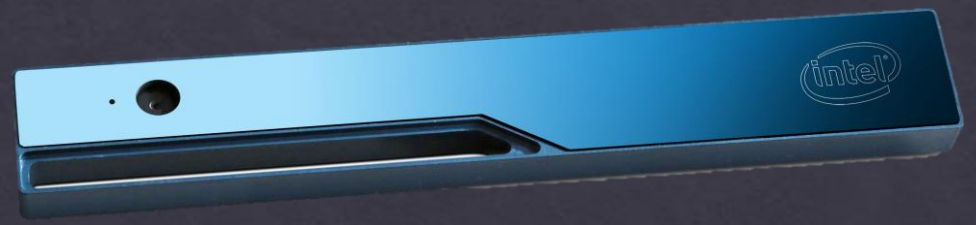
Scanning the World

content creation



mixed reality

Commodity RGB-D Sensors

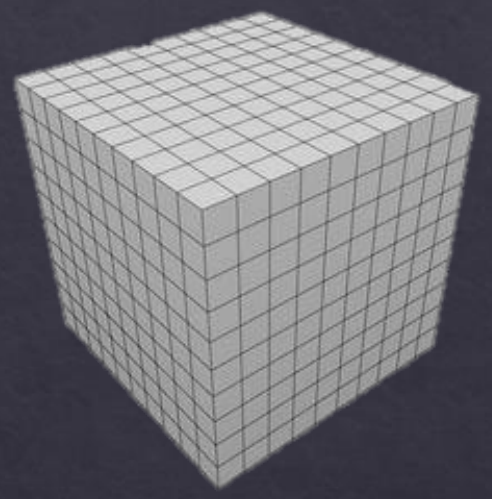


Online 3D Reconstruction

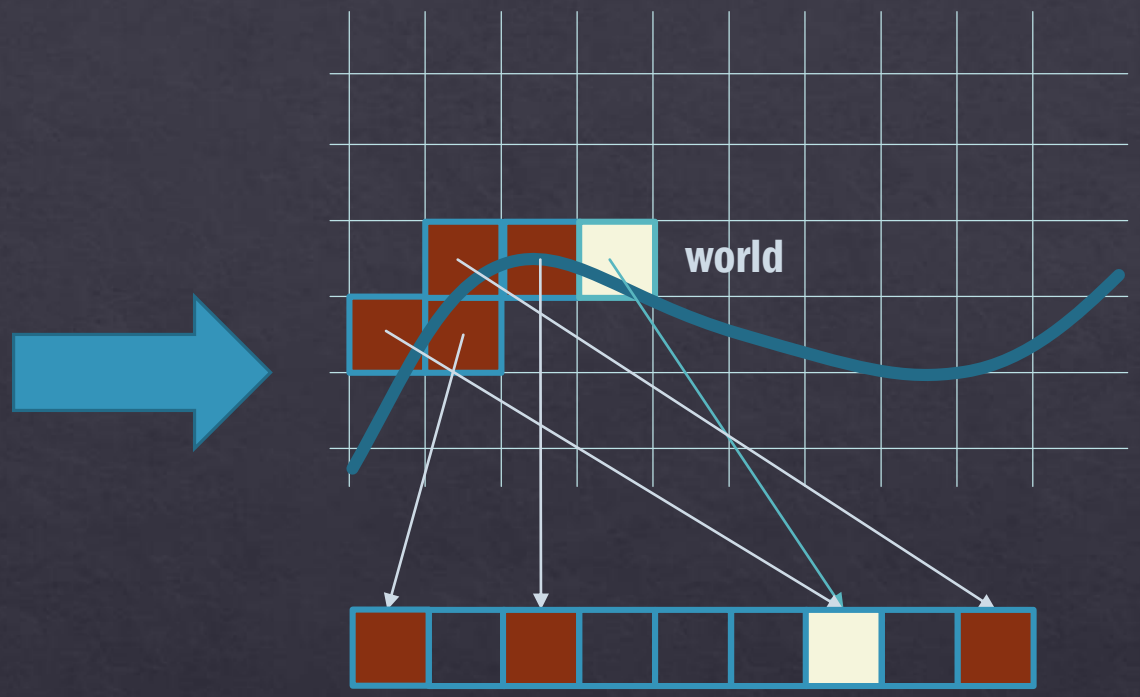
KinectFusion:
first real-time volumetric fusion

Online 3D Reconstruction: Challenges

- Spatial representation



Volumetric Fusion [Curless and Levoy 96]
KinectFusion [Newcombe et al. 11; Izadi et al. 11]



No spatial limits!

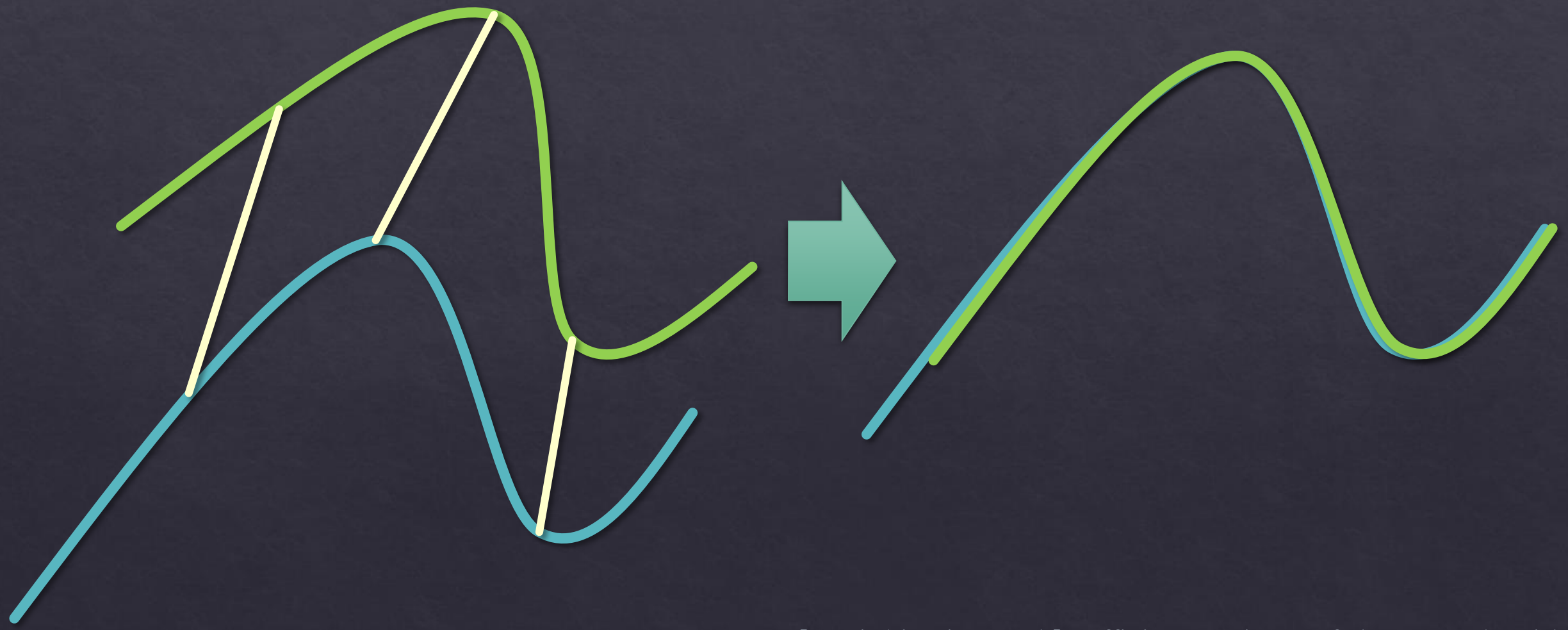
Spatial hashing:

$$H(x, y, z) = (x \cdot p_1 \oplus y \cdot p_2 \oplus z \cdot p_3) \bmod n$$

TOG'13 [Nießner et al.]: VoxelHashing

Online 3D Reconstruction: Challenges

- Tracking: frame-to-model ICP for real-time reconstruction



3DIM'01 [Rusinkiewicz et al.]: Efficient Variants of the ICP Algorithm

Online 3D Reconstruction: Challenges

- Tracking: want global consistency, loop closure, re-localization

TOG'13 [Nießner et al.]: VoxelHashing

Online 3D Reconstruction: Challenges

- Tracking: want global consistency, loop closure, re-localization

RSS'15 [Whelan et al.]: ElasticFusion

Online 3D Reconstruction: Challenges

- Tracking: want global consistency, loop closure, re-localization

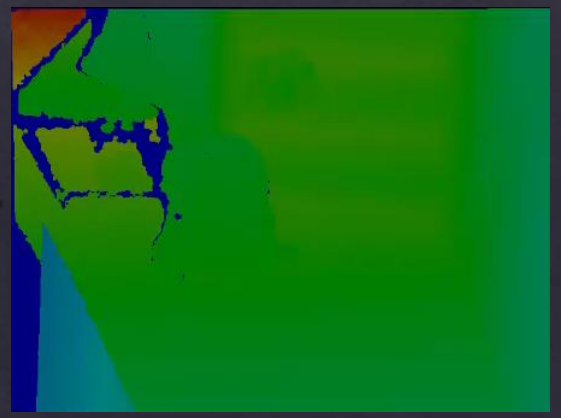
ElasticFusion: surfel representation

TSDF implicit surface

Online 3D Reconstruction: Real-time Bundling

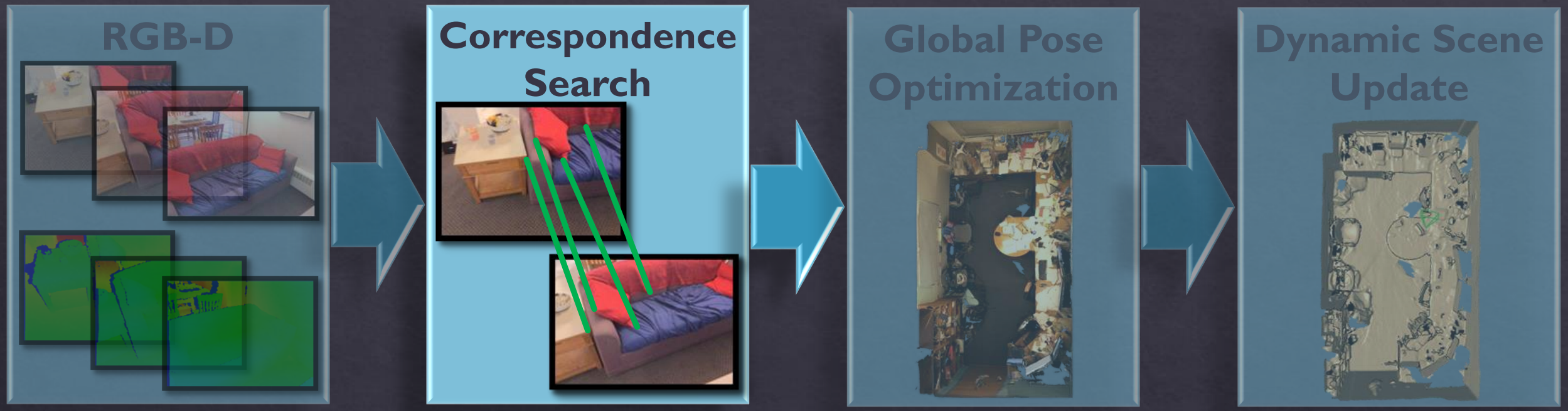
Overview

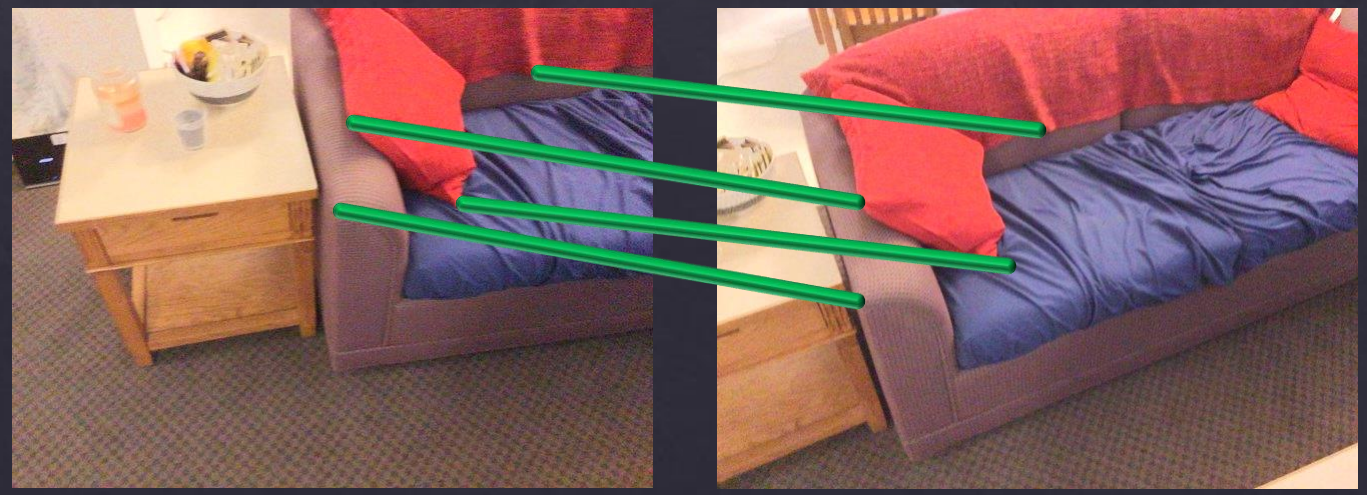
Depth



Color

Overview


RGB-D
Correspondence Search
Global Pose Optimization
Dynamic Scene Update

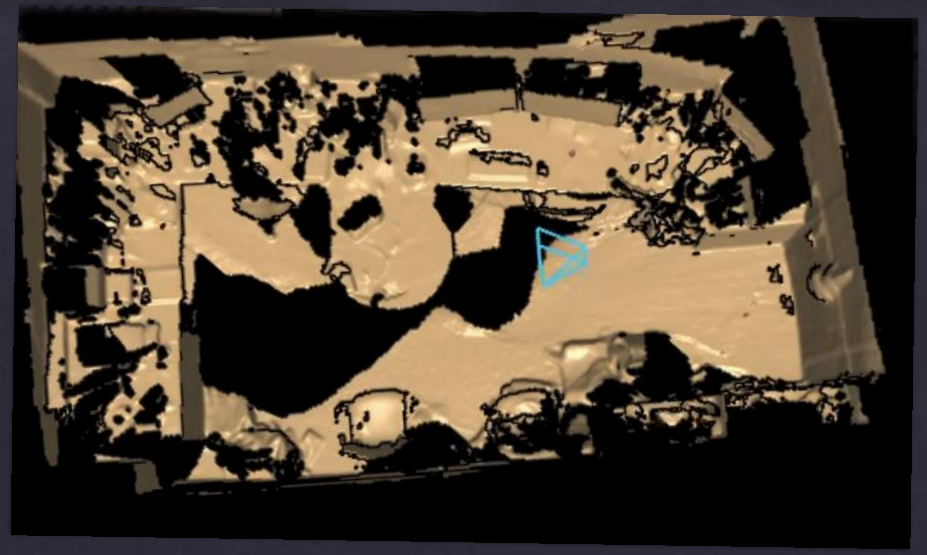


Overview

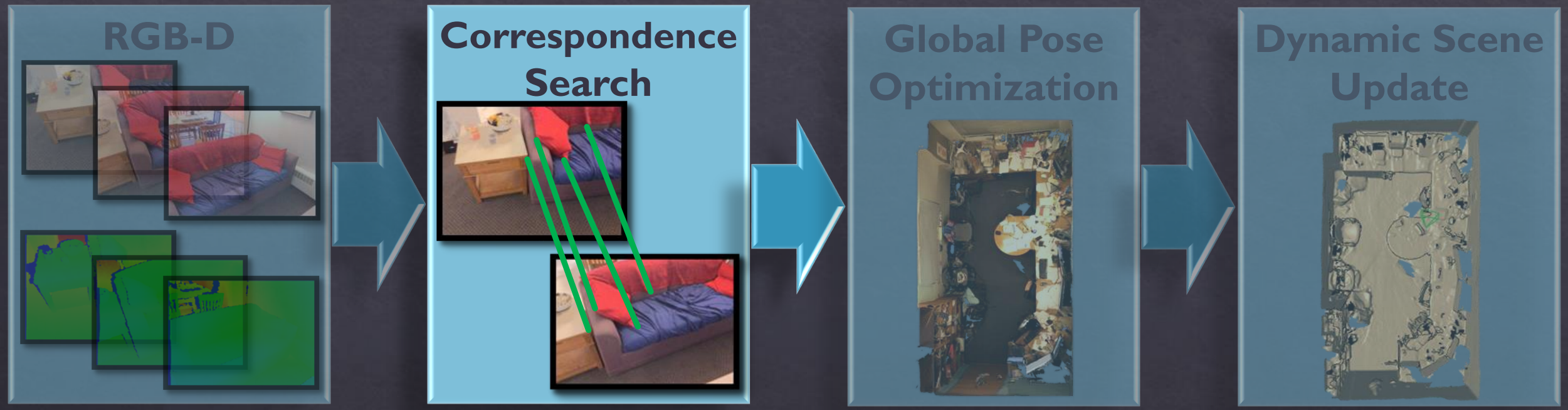

RGB-D
Correspondence Search
Global Pose Optimization
Dynamic Scene Update

Overview


RGB-D
Correspondence Search
Global Pose Optimization
Dynamic Scene Update

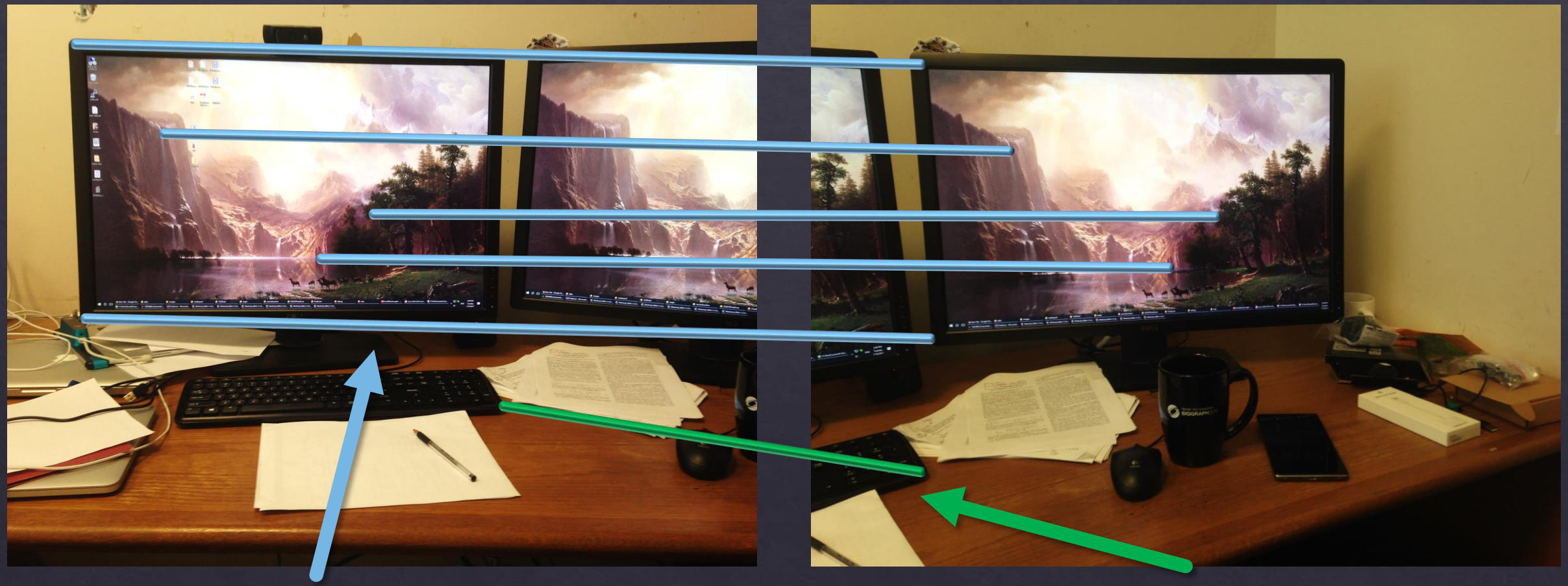


Correspondence Search



Correspondence Filtering

- Find outlier matches

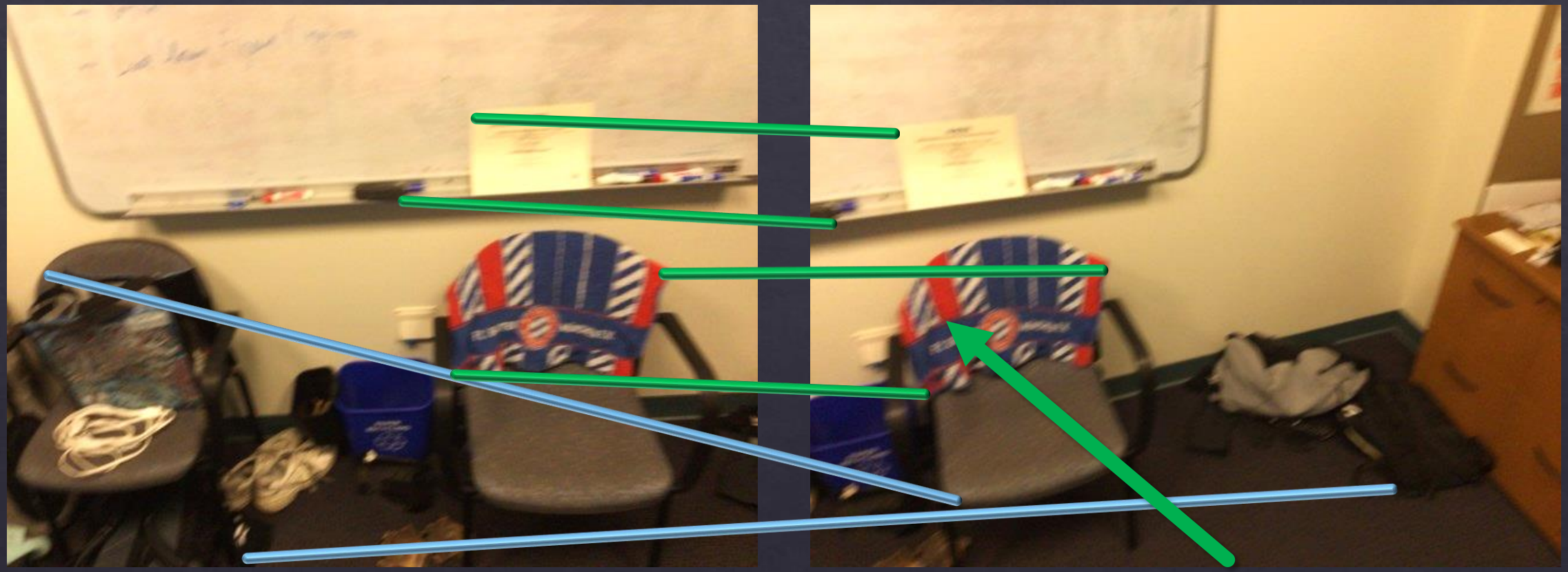


Correspondence Filtering

- Key point correspondence filter: find consistent transform

Correspondence Filtering

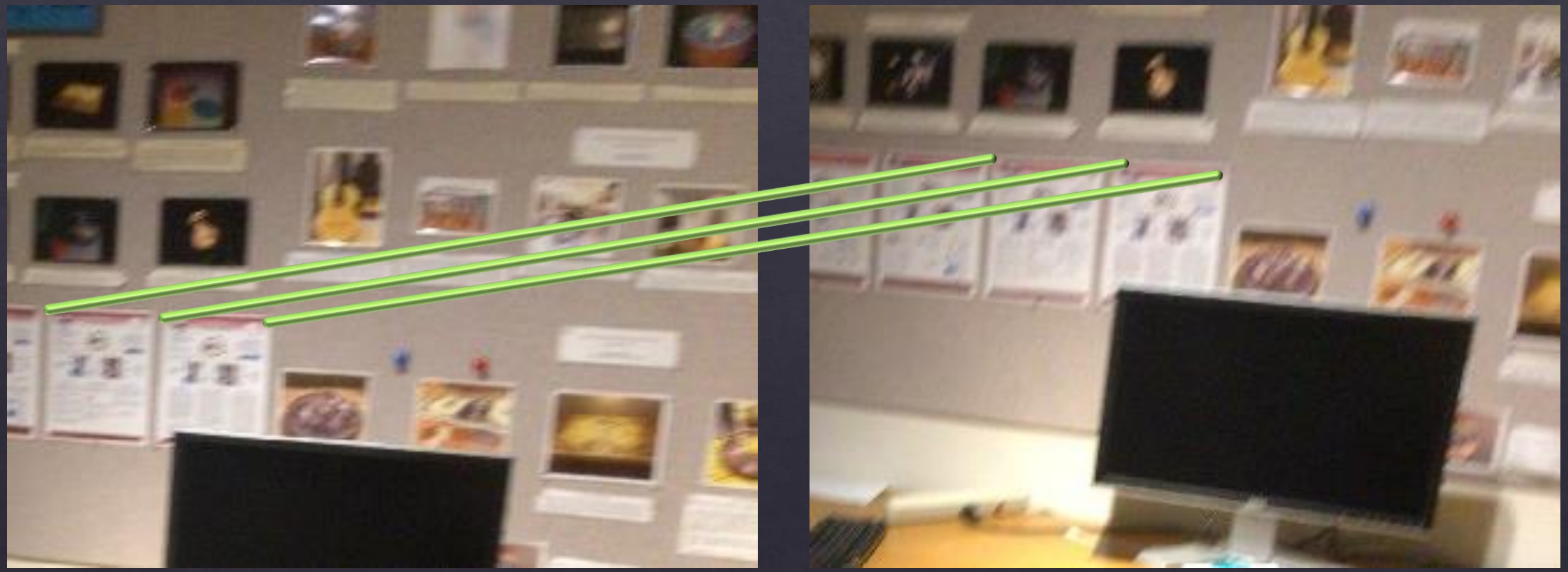
- Key point correspondence filter: find consistent transform



Consistent Correspondences

Correspondence Filtering

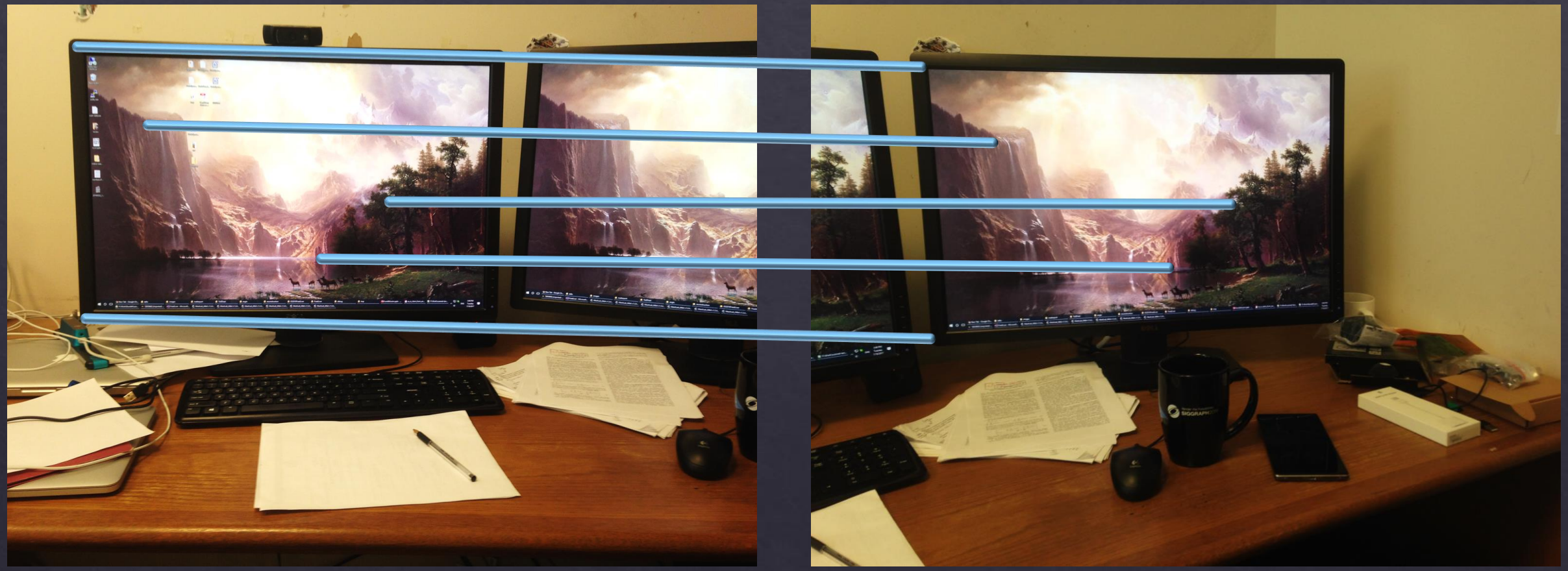
- Prune badly conditioned correspondence sets



Unstable Transform

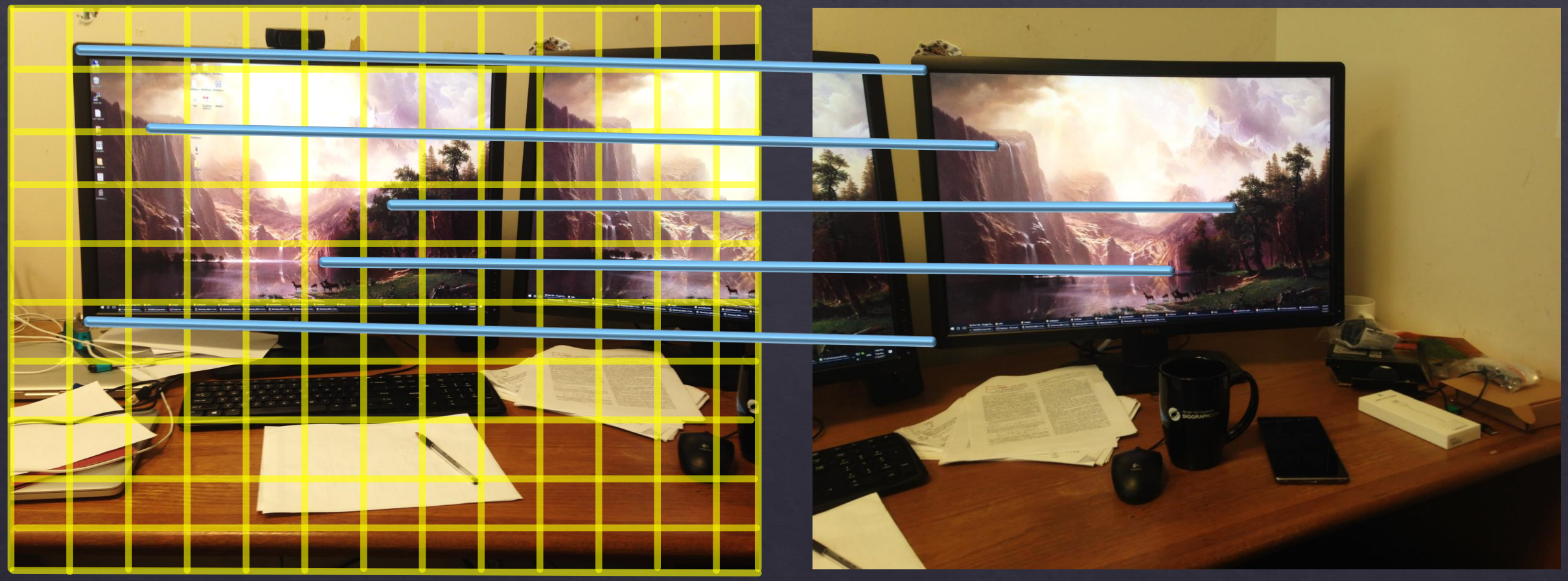
Correspondence Filtering

- Dense verification: prune high re-projection error



Correspondence Filtering

- Dense verification: prune high re-projection error

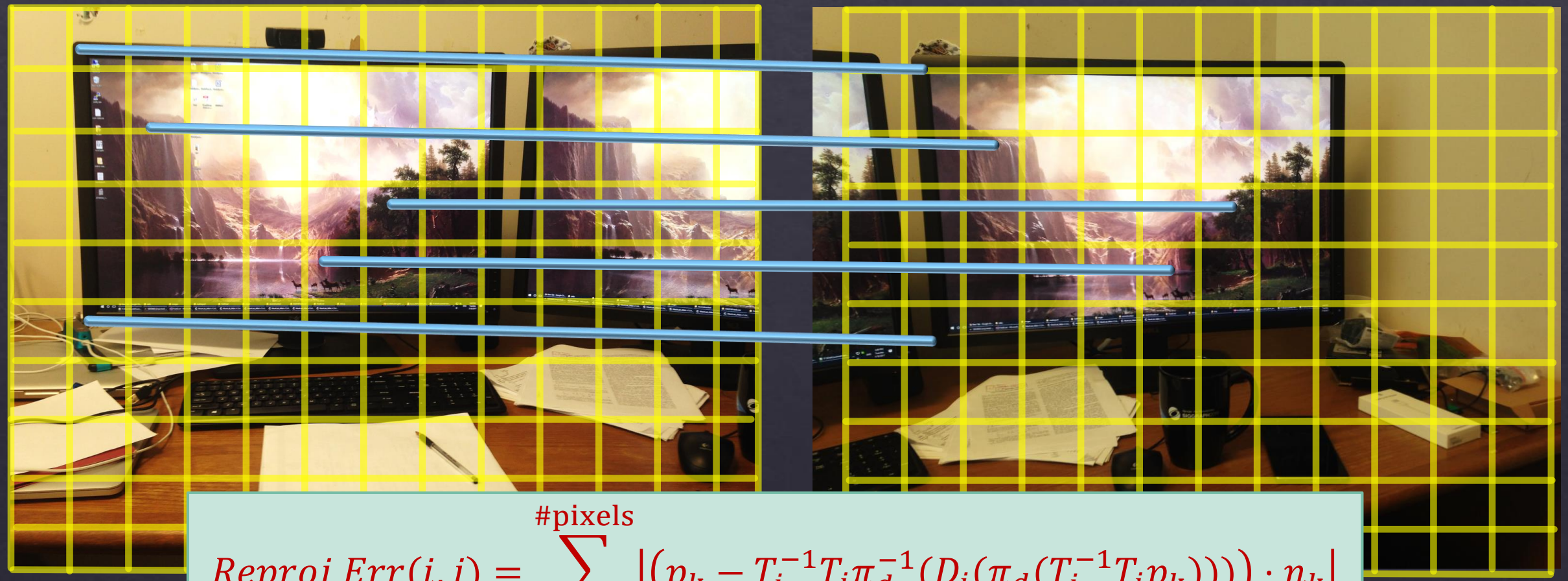


Correspondence Filtering

- Dense verification: prune high re-projection error

Correspondence Filtering

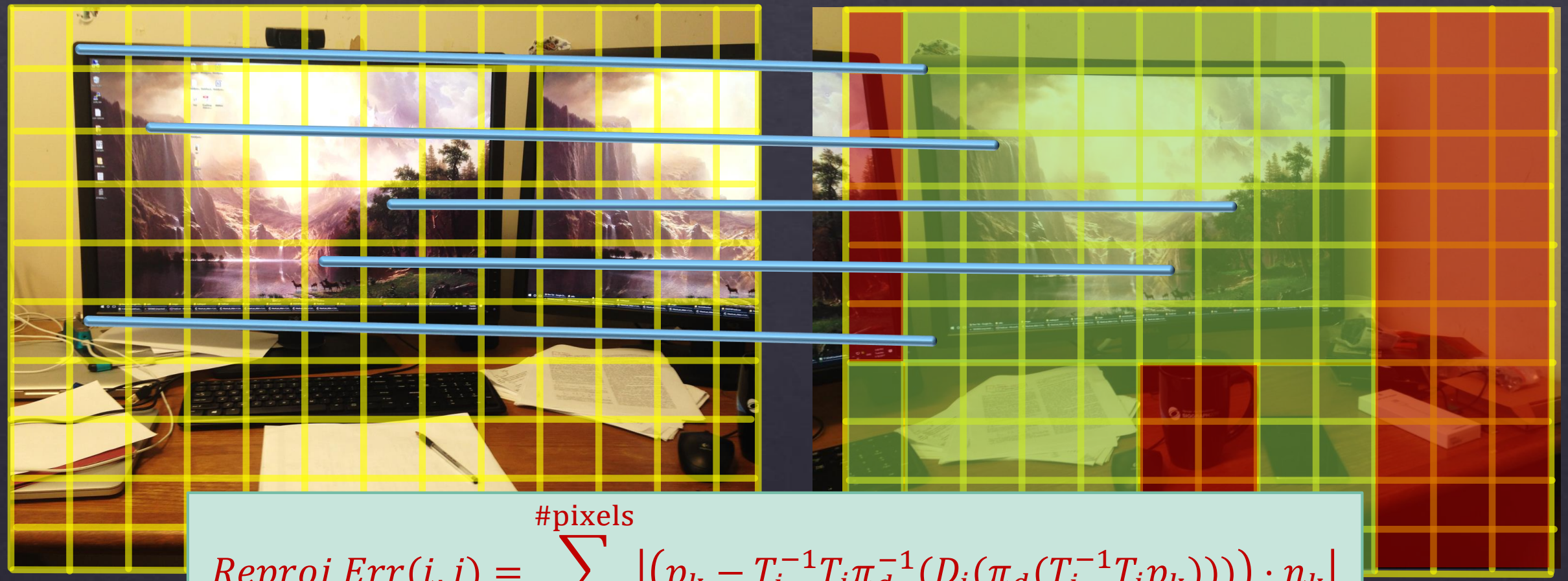
- Dense verification: prune high re-projection error



$$Reproj\ Err(i,j) = \sum_{k}^{\#\text{pixels}} \left| \left(p_k - T_i^{-1} T_j \pi_d^{-1} \left(D_j \left(\pi_d \left(T_j^{-1} T_i p_k \right) \right) \right) \right) \cdot n_k \right|$$

Correspondence Filtering

- Dense verification: prune high re-projection error



$$Reproj\ Err(i,j) = \sum_{k}^{\#\text{pixels}} \left| \left(p_k - T_i^{-1} T_j \pi_d^{-1} \left(D_j \left(\pi_d \left(T_j^{-1} T_i p_k \right) \right) \right) \right) \cdot n_k \right|$$

Pose Optimization

Sparse-to-Dense Optimization

- Initialize with sparse features, continue dense

$$E(T) = w_{sparse} E_{sparse}(T) + w_{dense} E_{dense}(T)$$

Sparse-to-Dense Optimization

- Initialize with sparse features, continue dense

$$E(T) = w_{sparse} E_{sparse}(T) + w_{dense} E_{dense}(T)$$

Sparse-to-Dense Optimization

- Initialize with sparse features, continue dense

$$E(T) = w_{sparse} E_{sparse}(T) + w_{dense} E_{dense}(T)$$

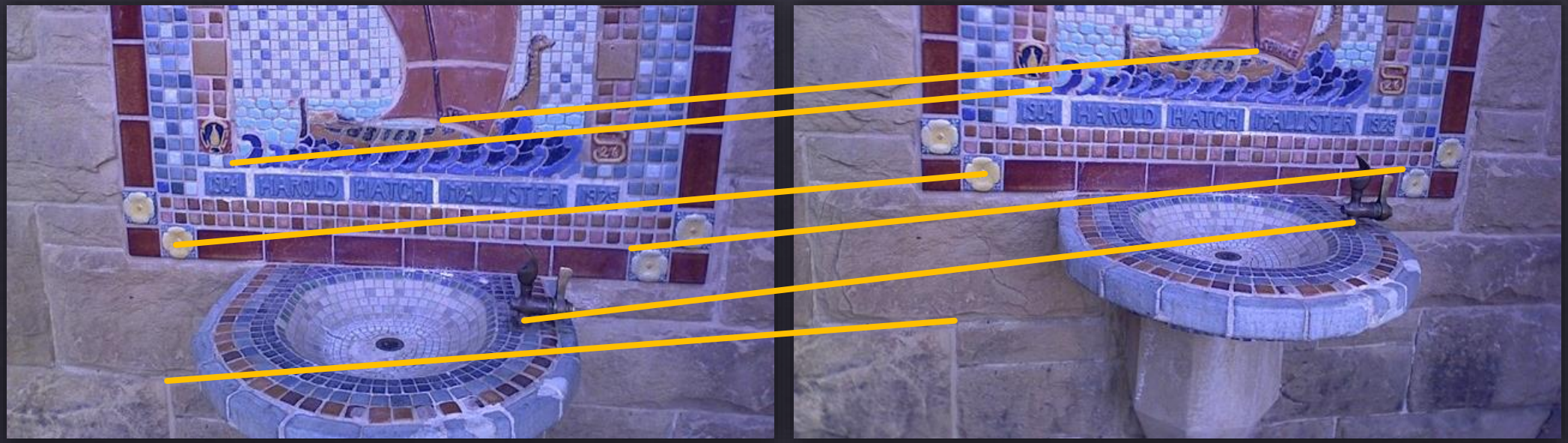
$$E_{sparse}(T) = \sum_{i,j}^{\#frames} \sum_{k}^{\#corresp.} \left\| T_i p_{ik} - T_j p_{jk} \right\|_2^2$$

Sparse-to-Dense Optimization

- Initialize with sparse features, continue dense

$$E(T) = w_{sparse} E_{sparse}(T) + w_{dense} E_{dense}(T)$$

$$E_{sparse}(T) = \sum_{i,j}^{\#frames} \sum_{k}^{\#corresp.} \left\| T_i p_{ik} - T_j p_{jk} \right\|_2^2$$



Sparse-to-Dense Optimization

- Initialize with sparse features, continue dense

$$E(T) = w_{sparse} E_{sparse}(T) + w_{dense} E_{dense}(T)$$

Sparse-to-Dense Optimization

- Initialize with sparse features, continue dense

$$E(T) = w_{sparse} E_{sparse}(T) + w_{dense} E_{dense}(T)$$

$$E_{dense}(T) = w_{depth} E_{depth}(T) + w_{color} E_{color}(T)$$

Sparse-to-Dense Optimization

- Initialize with sparse features, continue dense

$$E(T) = w_{sparse} E_{sparse}(T) + w_{dense} E_{dense}(T)$$

$$E_{dense}(T) = w_{depth} E_{depth}(T) + w_{color} E_{color}(T)$$

$$E_{depth}(T) = \sum_{i,j}^{\#\text{frames}} \sum_{k}^{\#\text{pixels}} \left\| \left(p_k - T_i^{-1} T_j \pi_d^{-1} \left(D_j \left(\pi_d \left(T_j^{-1} T_i p_k \right) \right) \right) \right) \cdot n_k \right\|_2^2$$

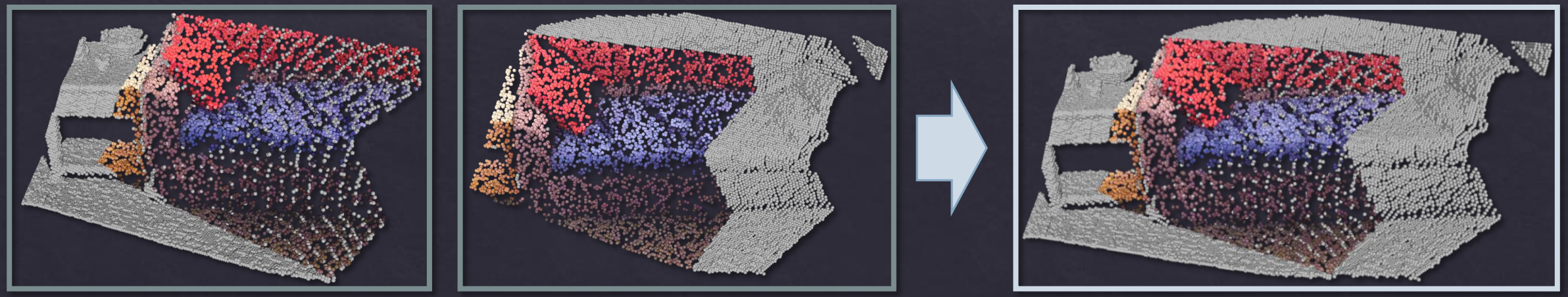
Sparse-to-Dense Optimization

- Initialize with sparse features, continue dense

$$E(T) = w_{sparse} E_{sparse}(T) + w_{dense} E_{dense}(T)$$

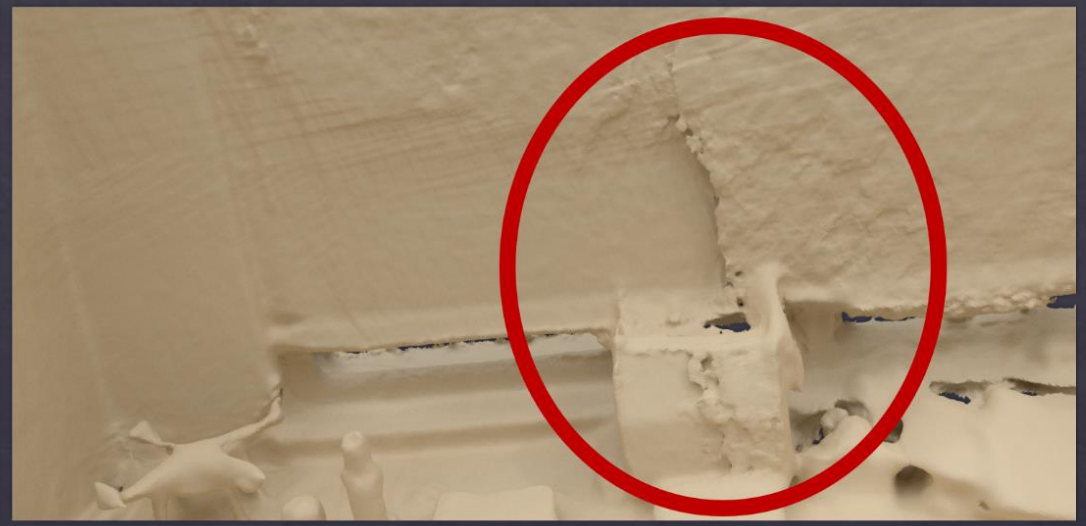
$$E_{dense}(T) = w_{depth} E_{depth}(T) + w_{color} E_{color}(T)$$

$$E_{color}(T) = \sum_{i,j}^{\#\text{frames}} \sum_{k}^{\#\text{pixels}} \left\| \nabla I(\pi_c(p_k)) - \nabla I(\pi_c(T_j^{-1} T_i p_k)) \right\|_2^2$$

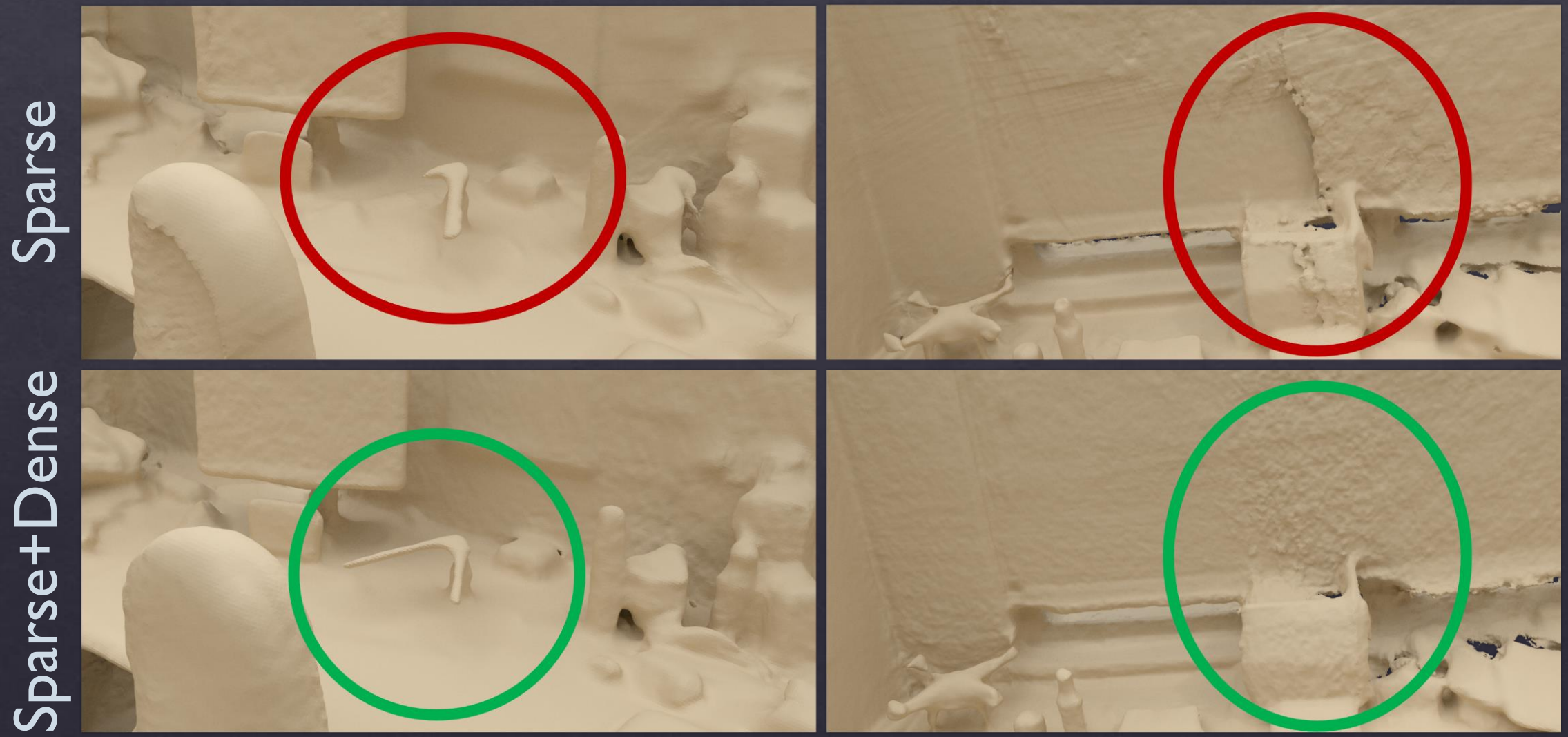


Sparse-to-Dense Optimization

Sparse



Sparse-to-Dense Optimization



Local-to-Global Strategy

Local-to-Global Strategy

Optimize local chunks of frames first

10 frames with identity pose
Chunk 1

10 frames with identity pose
Chunk 2

Local-to-Global Strategy

Optimized local chunks

10 aligned frames
Chunk 1

10 aligned frames
Chunk 2

Local-to-Global Strategy

Chunk 1

Chunk 2

Global optimization over chunks

Chunk 1+2

Sparse-to-Dense Optimization

	kt0	kt1	kt2	kt3
DVO SLAM	10.4cm	2.9cm	19.1cm	15.2cm
RGB-D SLAM	2.6cm	0.8cm	1.8cm	43.3cm
MRSMap	20.4cm	22.8cm	18.9cm	109cm
Kintinuous	7.2cm	0.5cm	1.0cm	35.5cm
Elastic Fusion	0.9cm	0.9cm	1.4cm	10.6cm
Redwood (rigid)*	25.6cm	3.0cm	3.3cm	6.1cm
Ours (s)	0.9cm	1.2cm	1.3cm	1.3cm
Ours (sd)	0.8cm	0.5cm	1.1cm	1.2cm
Ours	**0.6cm**	**0.4cm**	**0.6cm**	**1.1cm**

sparse only

sparse + local dense

full

ATE RMSE on the synthetic ICL-NUIM Dataset by Handa et al.

On-the-fly Scene Updates

On-the-fly Scene Updates

- Surface Integration *[Curless and Levoy 96]*

```
Voxel {
    distance;
    color;
    weight;
}
```

$$D(v) = \frac{\sum_i w_i(v) d_i(v)}{\sum_i w_i(v)}$$

$$W(v) = \sum_i w_i(v)$$

On-the-fly Scene Updates

- Surface Integration *[Curless and Levoy 96]*

```
Voxel {
    distance;
    color;
    weight;
}
```

$$D'(v) = \frac{W(v)D(v) + w_k(v)d_k(v)}{W(v) + w_k(v)}$$

$$W'(v) = W(v) + w_k(v)$$

On-the-fly Scene Updates

- Surface De-integration: remove d_k from weighted average

```
Voxel {
    distance;
    color;
    weight;
}
```

$$D'(v) = \frac{W(v)D(v) - w_k(v)d_k(v)}{W(v) - w_k(v)}$$

$$W'(v) = W(v) - w_k(v)$$

On-the-fly Scene Updates

On-the-fly Scene Updates

Integrated
(wrong pose)

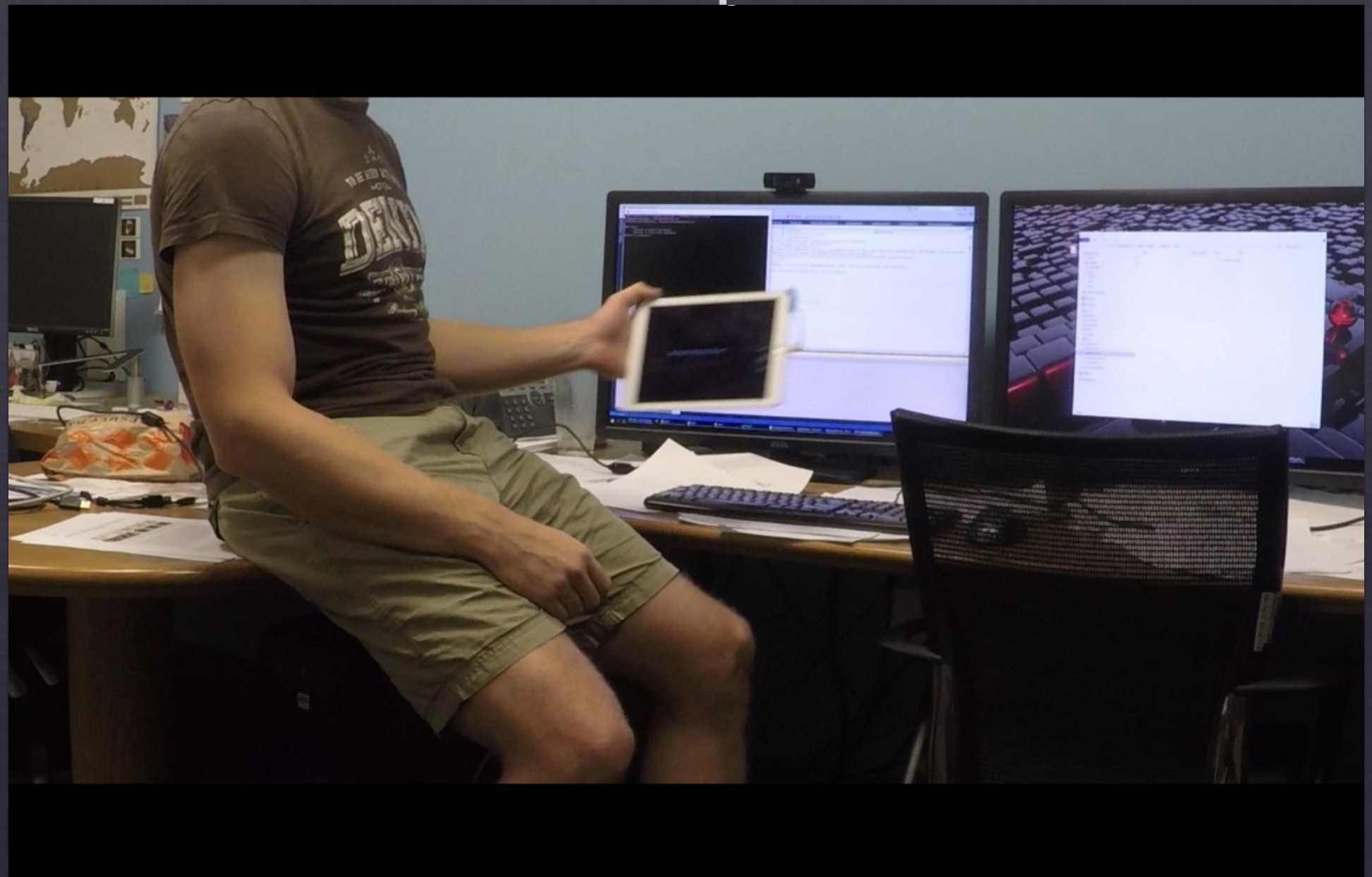
On-the-fly Scene Updates

De-integrated

On-the-fly Scene Updates

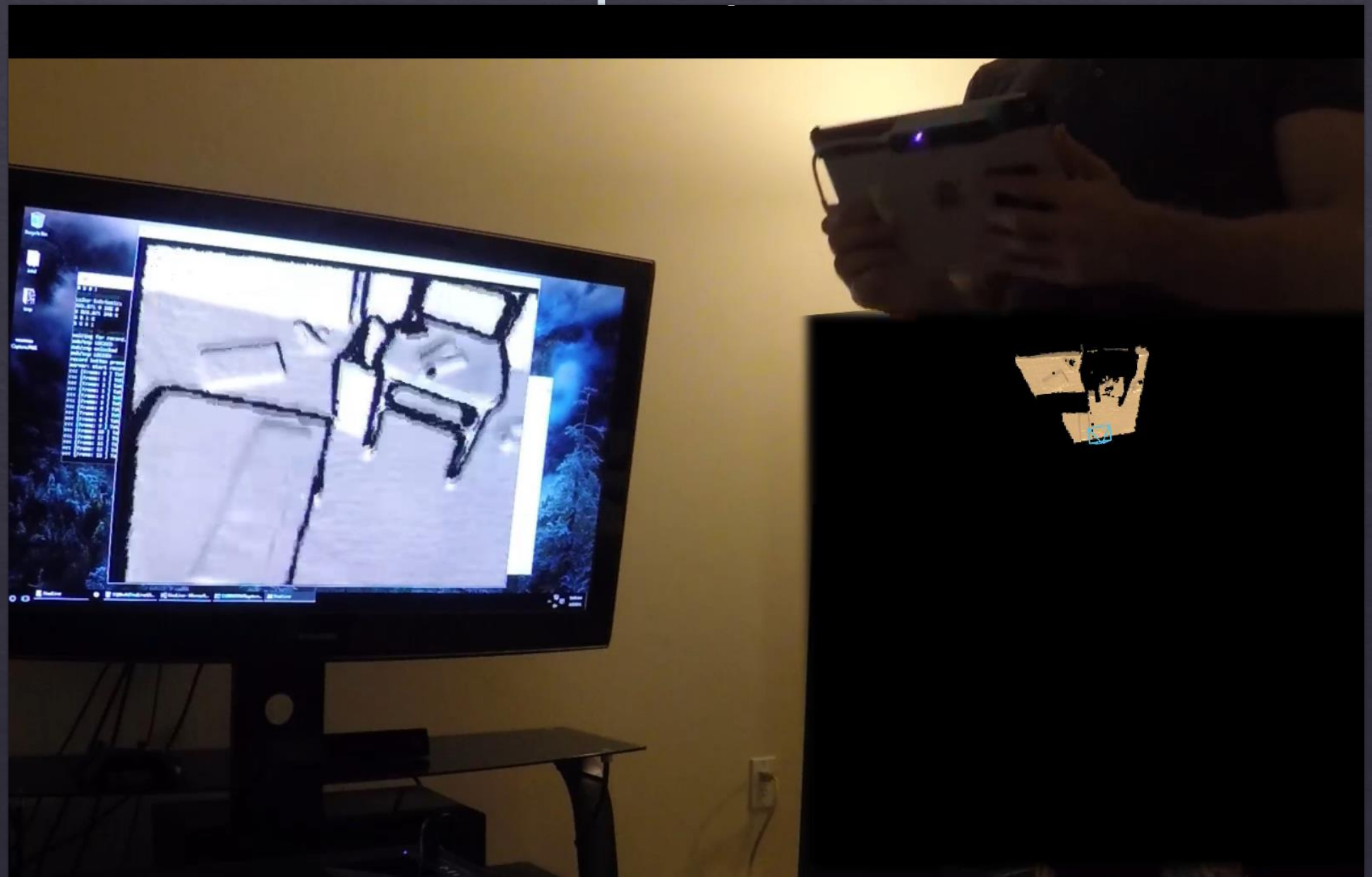
Re-integrated with updated pose

Live Capture



Loop Closure

Loop Closure

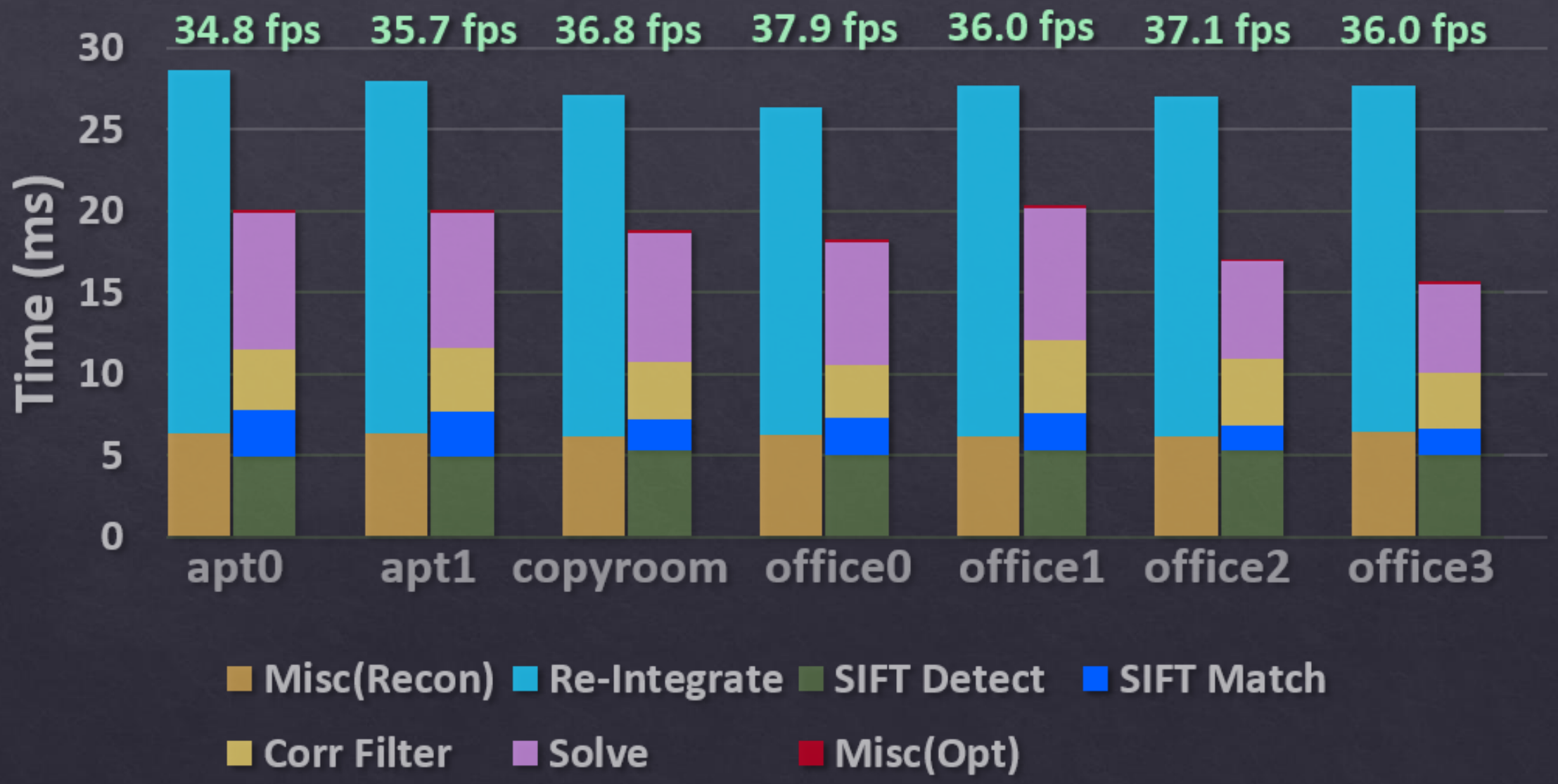


Re-Localization

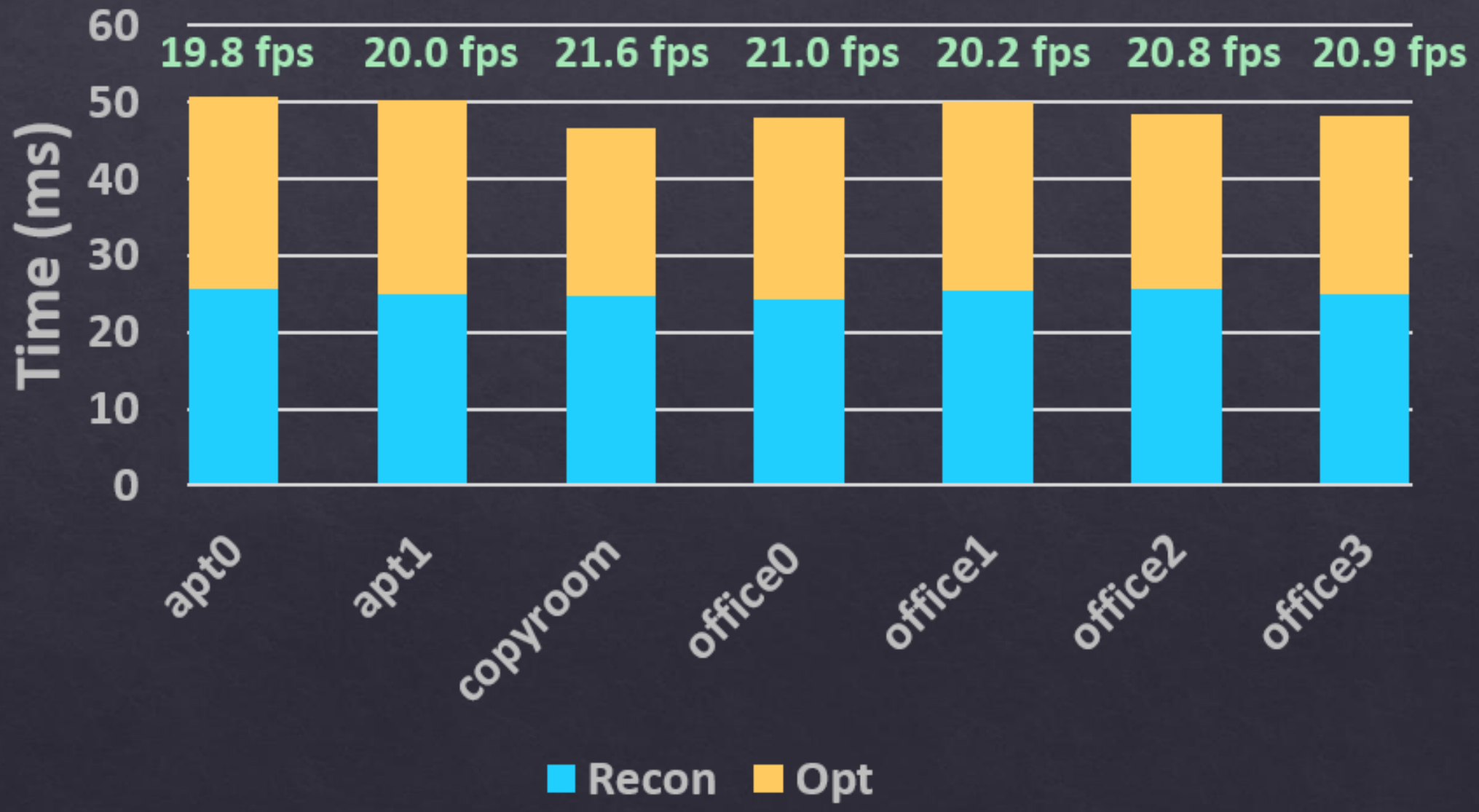
Re-Localization

Performance

Dual GPU Performance

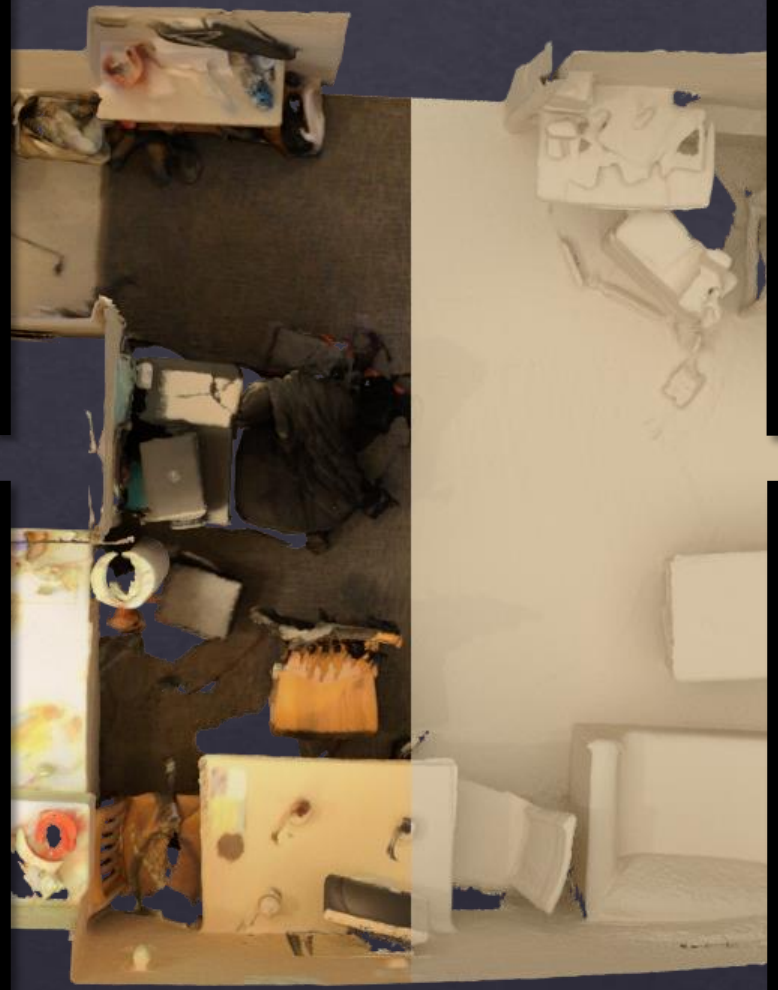


Single GPU Performance



Results

Results



Results

Results

Results

Results

ScanNet: Reconstructed with BundleFusion

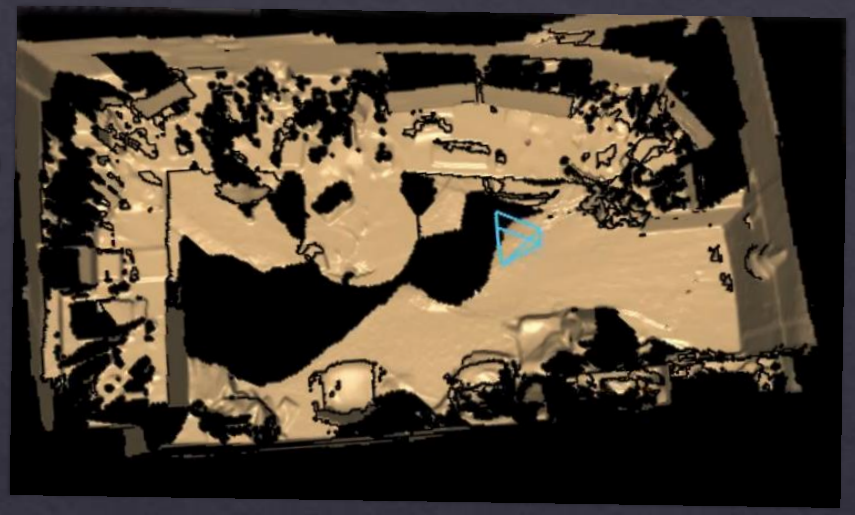
1513 reconstructed scenes

CVPR'17 (spotlight) [Dai et al.]: ScanNet

Conclusion

- Real-time reconstruction with commodity rgb-d sensors
- Global pose optimization
- Online loop closures
- Real-time re-localization
- On-the-fly scene updates

Loop Closing



Online Re-localizatoin

Thank You!

code and data online:

https://graphics.stanford.edu/projects/bundlefusion