Adapting Depth to Relocalization

Mitsubishi-A team

Keigo Horikoshi   Ikkei Sato   Zachary Fendler   Sam Scheuerman

Academic Mentor: Xiwen Jiang
Industry Mentors: Akinobu Sasada, Takuya Saeki

2025/8/6 G-RIPS Final Presentation

Our Sponsors

This work is supported by the following:

MITSUBISHI ELECTRIC

AIMR

ipam

TOHOKU FORUM for CREATIVITY

MathCCS

東北大学 数理科学共創社会センター

Mathematical Science Center for Co-creative Society, Tohoku University

Outline

1. **Introduction:**
   - **The Kidnapped robot problem**
2. Background:
   - Relocalization terminology
3. Our Method:
   - Concrete pipeline
4. Method Validation:
   - Experiments, data and result
5. Conclusion:
   - Summarization and future work

Introduction: Kidnapped Robot

I have a camera with LiDAR

A robot is navigating,

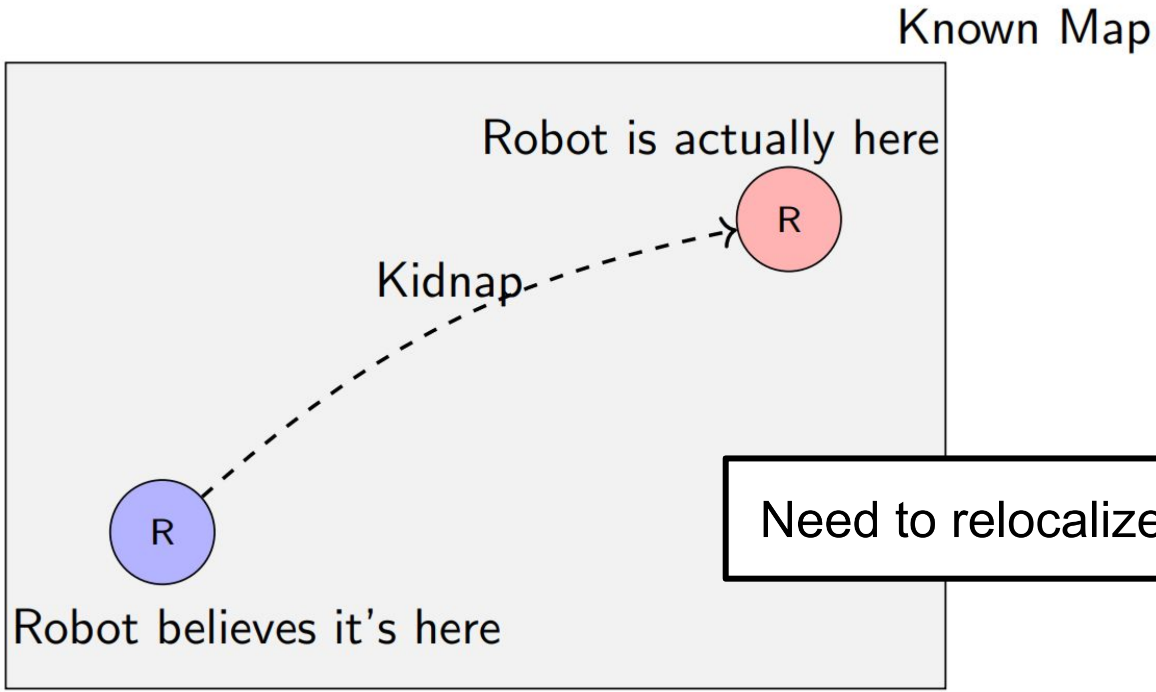
Introduction: Kidnapped Robot

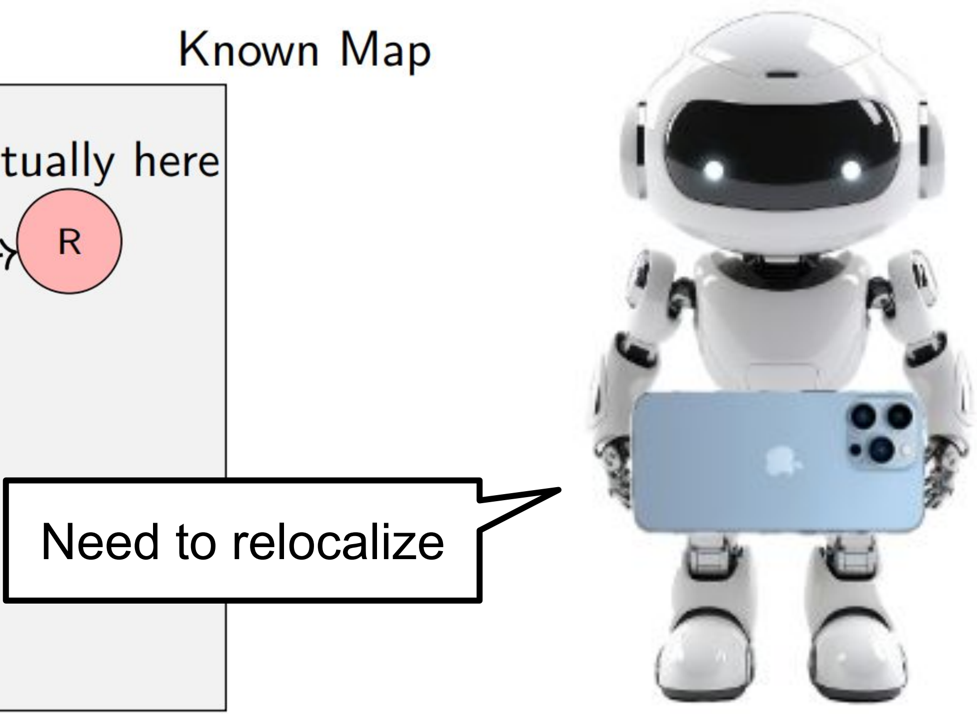
I have a camera with LiDAR

A robot is navigating, but gets “kidnapped!”

Mr. Lost Robot

Introduction: Kidnapped Robot





Mr. Lost Robot

When Do We Use Relocalization?

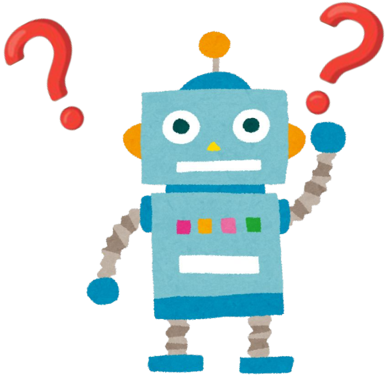
Systems lose tracking due to:

- GPS dropout
- “Kidnap robot” problem
- Visual occlusion or motion blur
- Algorithm drift or failure.

When Do We Use Relocalization?

Without relocalization, the system:

- Poorly interacts with the environment
- Becomes unsafe
- Fails to resume after failure or interruption.



Why Does This Matter?

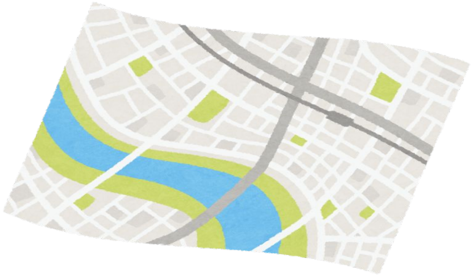
Several industries need reliable relocalization devices such as:

- Robotics & Automation
- Self-Driving Vehicles
- Augmented Reality (AR) & Mobile Apps
- Drones

Kidnapped Robot

It has

- A camera can capture color 2D image
- LiDAR measure the depth data (3D)
- Map (2D image and 3D data)



Mr. Lost Robot

Kidnapped Robot

If **humans** get lost, they can relocalize by comparing landmarks with a map.

Kidnapped Robot

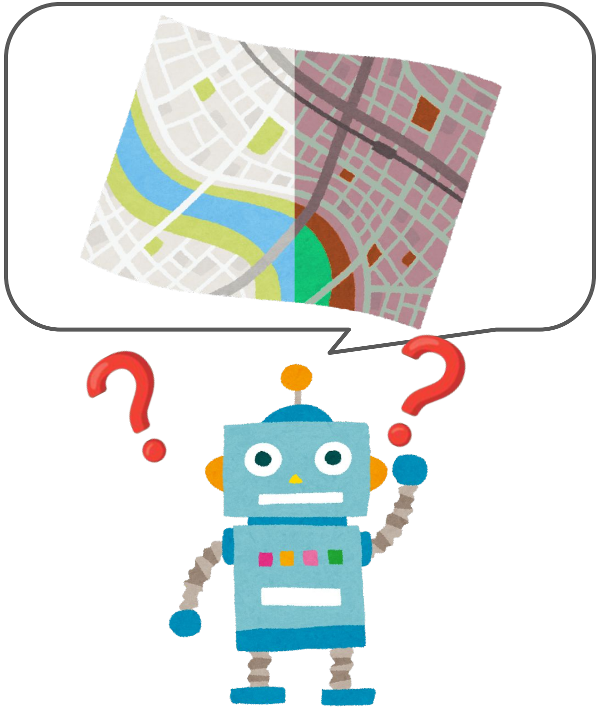
Robots try to do the same as humans

Mr. Lost Robot

Why did we focus on this problem?

Existing methods still struggle with large changes in image composition

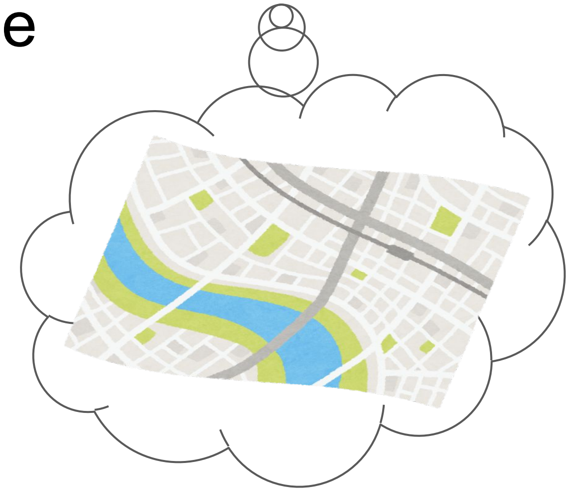
Ex: Lighting changes



Our Achievement

We propose **a novel pipeline** for relocalization.

- Focused on **accuracy**
- Potentially **robust** to large lighting change
- Used **2D image** with **depth** data



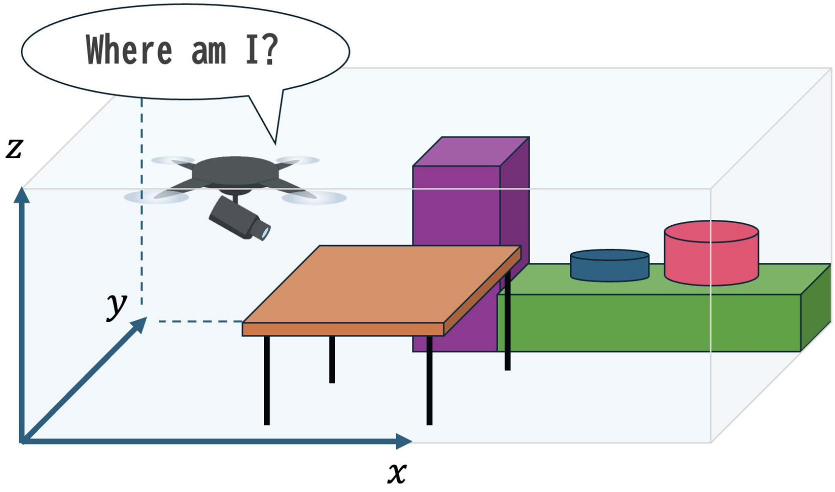
*Pipeline: A sequence of steps for relocalization

Outline

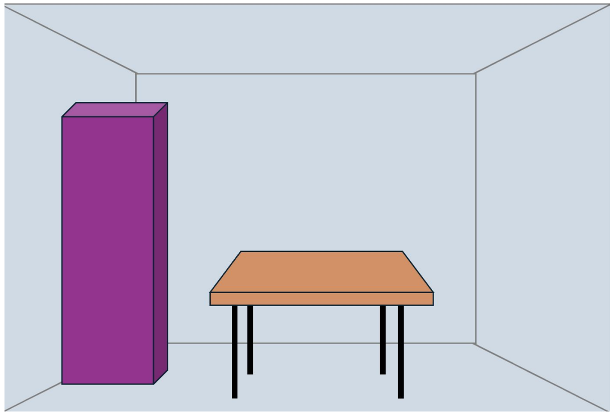
1. Introduction:
    ○ The Kidnapped robot problem
2. **Background:**
    ○ **Relocalization terminology**
3. Our Method:
    ○ Concrete pipeline
4. Method Validation:
    ○ Experiments, data and result
5. Conclusion:
    ○ Summarization and future work

Relocalization

Attempt by a robot to relocate itself once it has become lost.



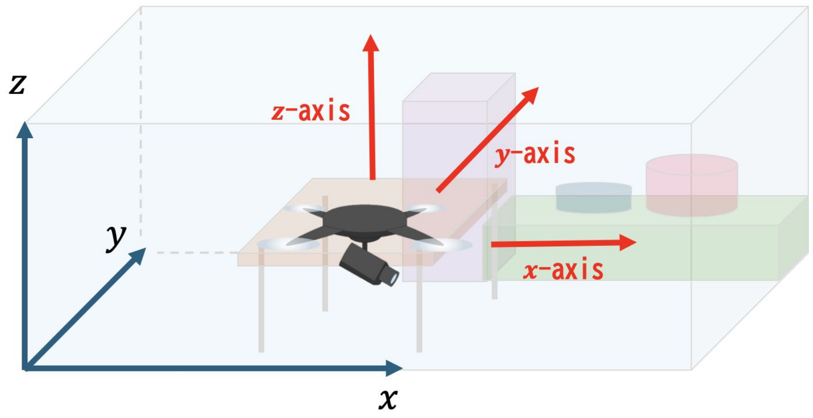
Mr. Lost Robot



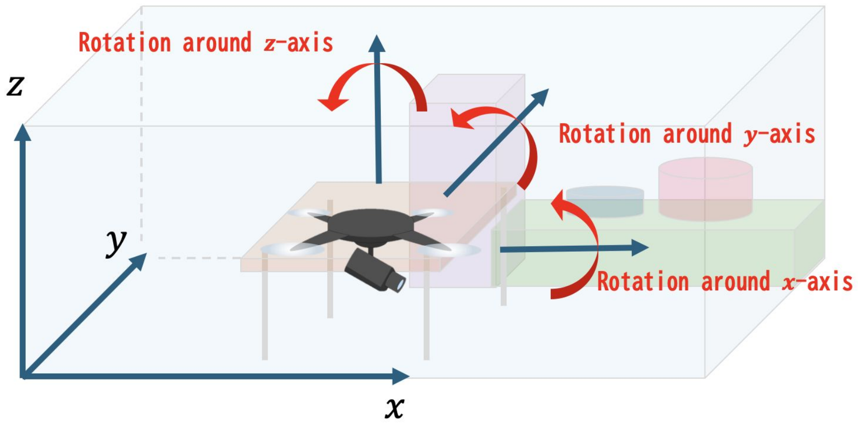
The view of Mr. Lost Robot

6 Degrees of Freedom (6DoF)

Degrees of freedom of position and orientation that an object has in 3D space.



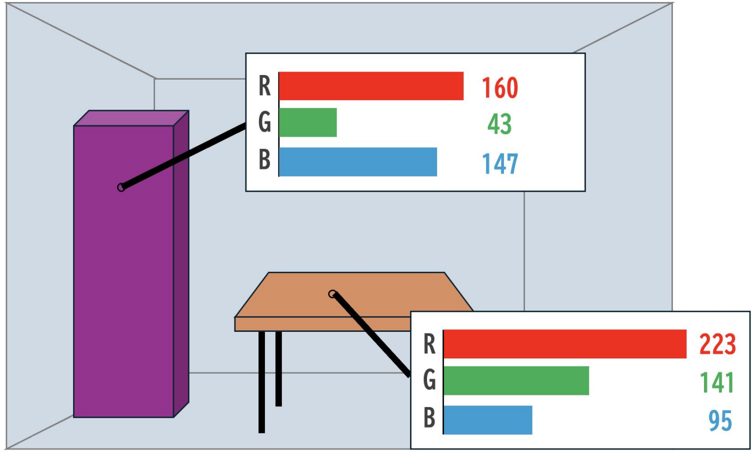
Translation (Position)



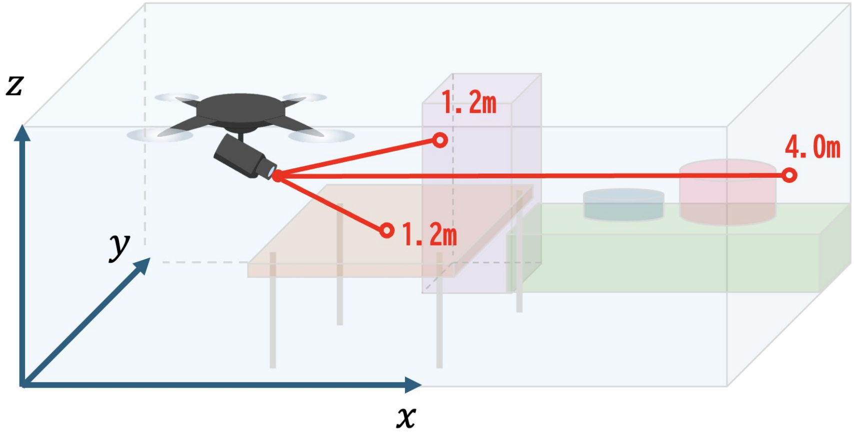
Rotation (Orientation)

RGB-D

Image information with the D (depth) parameter in addition to the RGB (red, green, blue) parameter that a normal camera has.



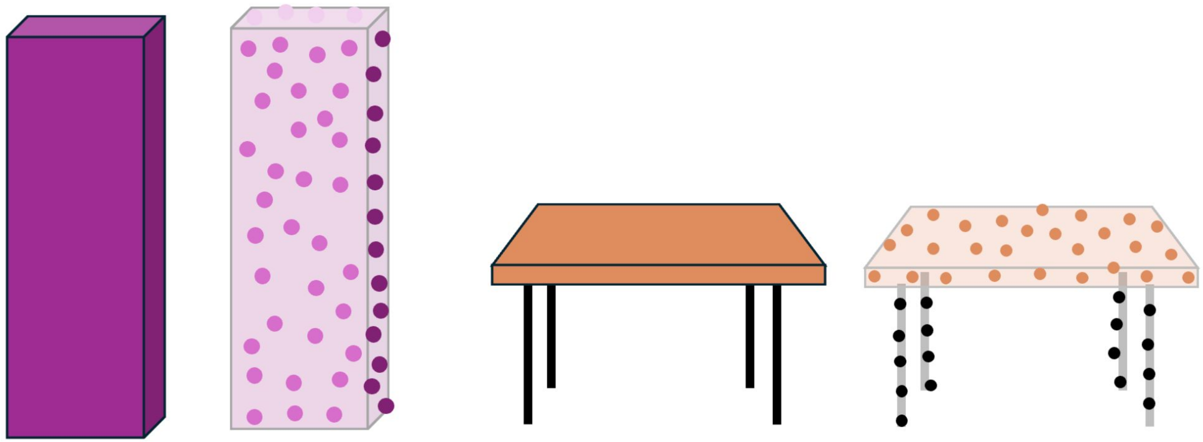
RGB-Image



Depth-Image

3D point cloud

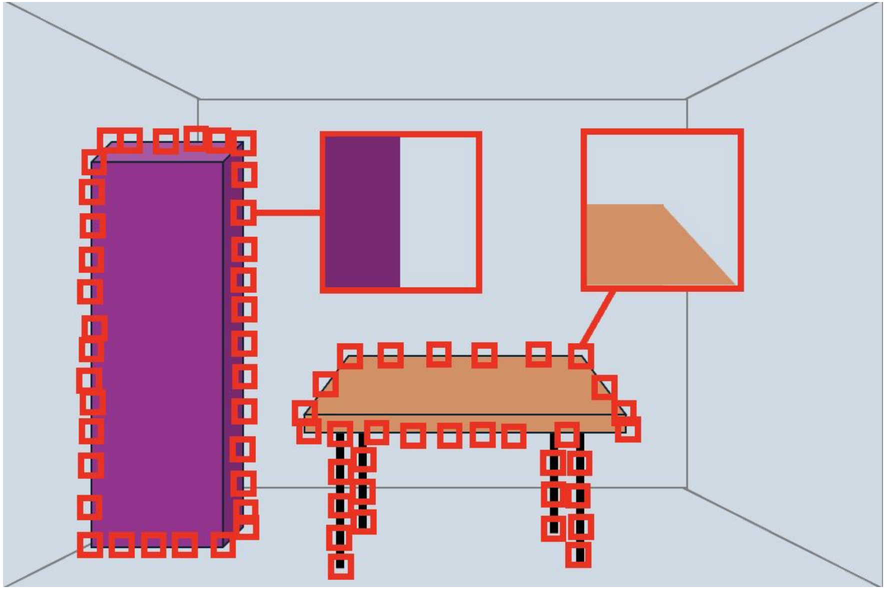
A collection of points on 3D-spaces. It is a discrete representation of the shape of a figure.



Examples of 3D-point clouds

Keypoint

Distinctive points in images

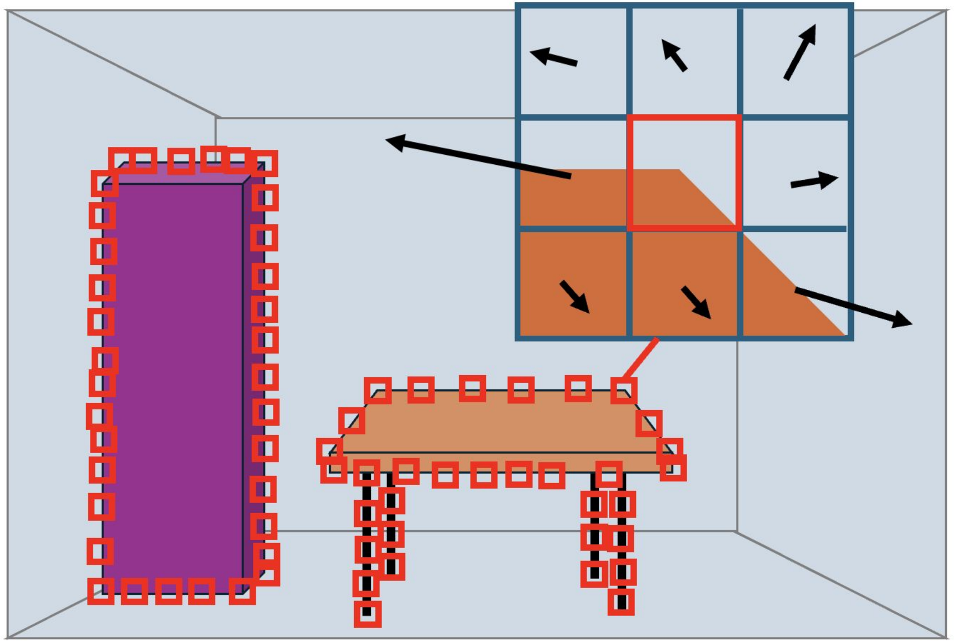


The red-circled area indicates a keypoint.

Local feature

Local Feature: Structural information extracted from keypoint

Feature extractor: Detects keypoints and corresponding features



Coarse to Fine Method

Global feature: A method that represents an image with a single feature vector

Coarse to Fine Method:

- In **global feature matching**, the similarity between images is compared.
- In **local feature matching**, identical objects in different images are matched.

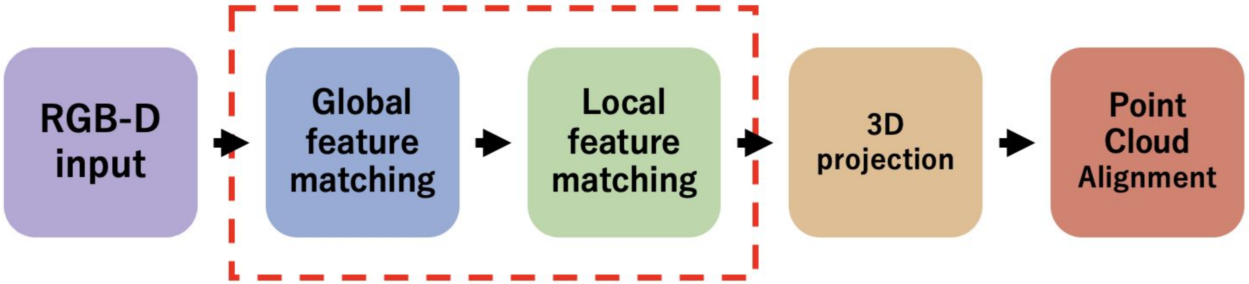
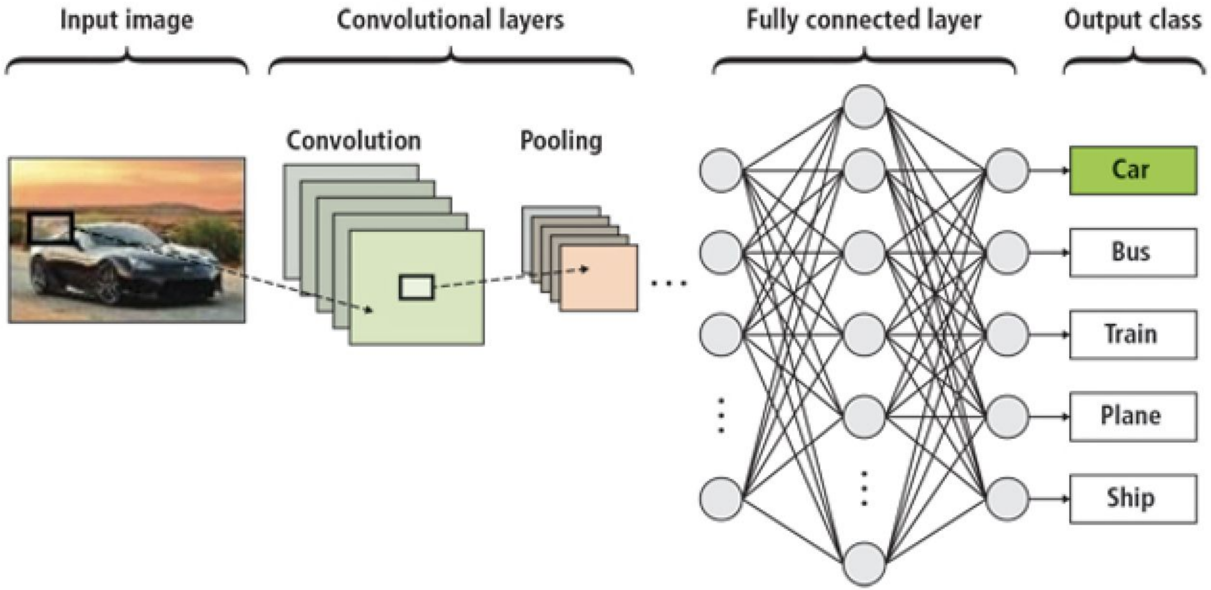


Figure: Our relocalization pipeline

CNN: A Common Backbone for Both Matchings

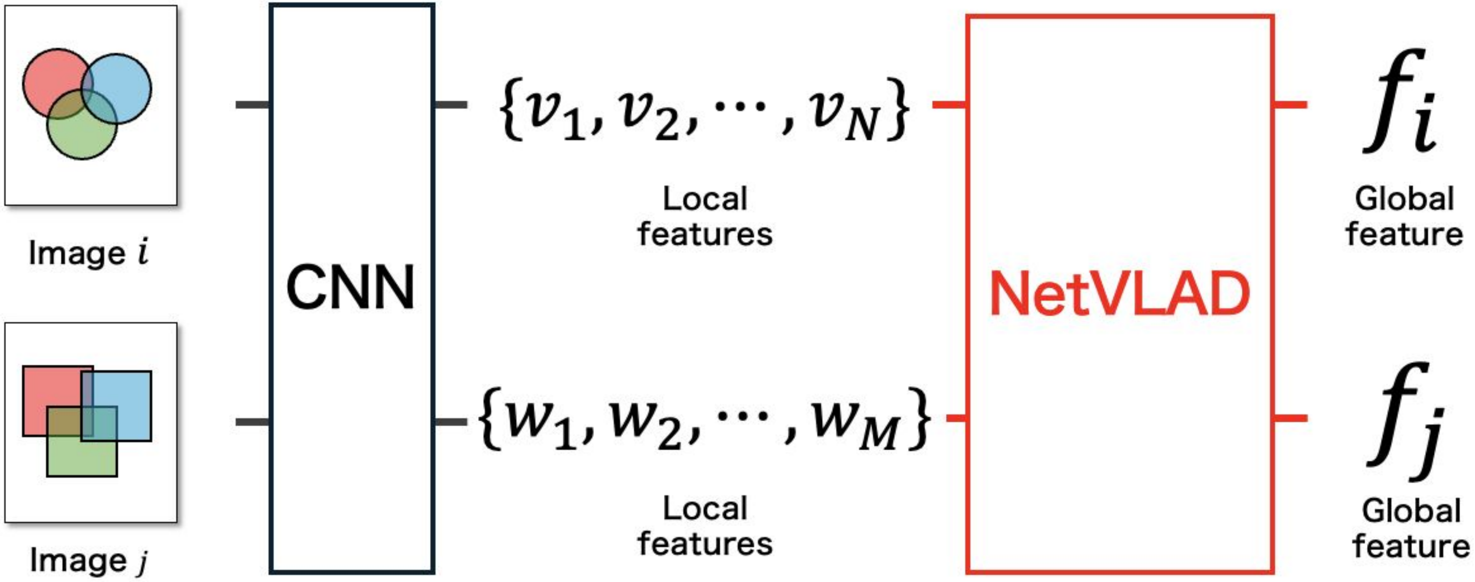
A deep learning model for image recognition and classification that automatically learns features from input images.



Source: NVIDIA, "Convolutional Neural Networks", https://developer.nvidia.com/convolutional-neural-networks

NetVLAD

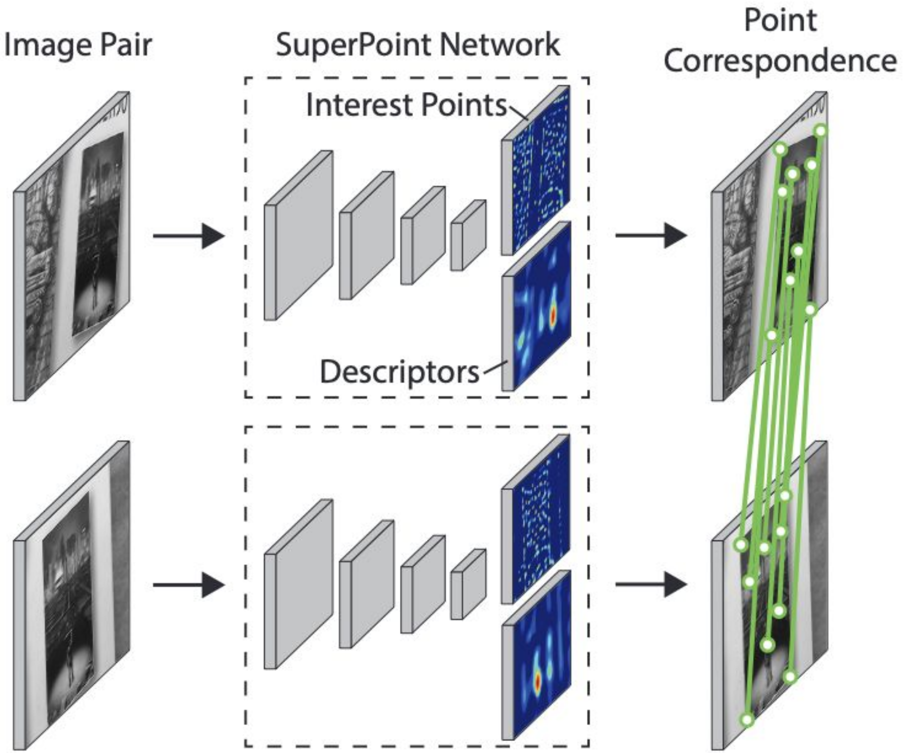
A novel pooling layer that aggregates image features from CNNs, enabling end-to-end training for place recognition.



RGB-D input → Global feature matching → **Local feature matching** → 3D projection → Point Cloud Alignment

Superpoint

A self-supervised, intelligent system that uses CNNs to efficiently detect "keypoints" and numerically describe their "descriptor" in an image.

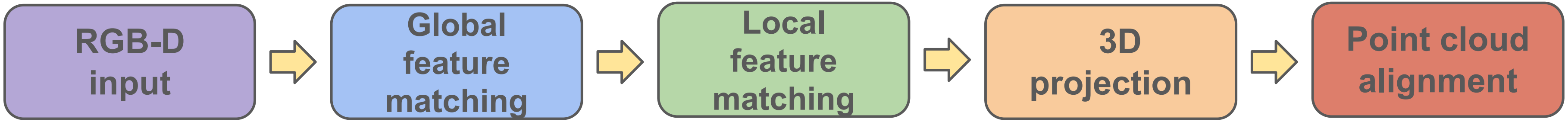


Source: Detone et al., "SuperPoint: Self-Supervised Interest Point Detection and Description," CVPR 2018.

Outline

1. Introduction:
    - The Kidnapped robot problem
2. Background:
    - Relocalization terminology
3. **Our Method:**
    - **Concrete pipeline**
4. Method Validation:
    - Experiments, data and result
5. Conclusion:
    - Summarization and future work

Our Method



- Coarse-to-fine paradigm
- α-Fusion

Figure: (left) Red-Green-Blue image and depth image (right).

RGB-D input → **Global feature matching** → Local feature matching → 3D projection → Point Cloud Alignment

Purpose:

- Image to global feature vector
- Global look up (quicker comparison)

Method:

- NetVLAD (deep learning)

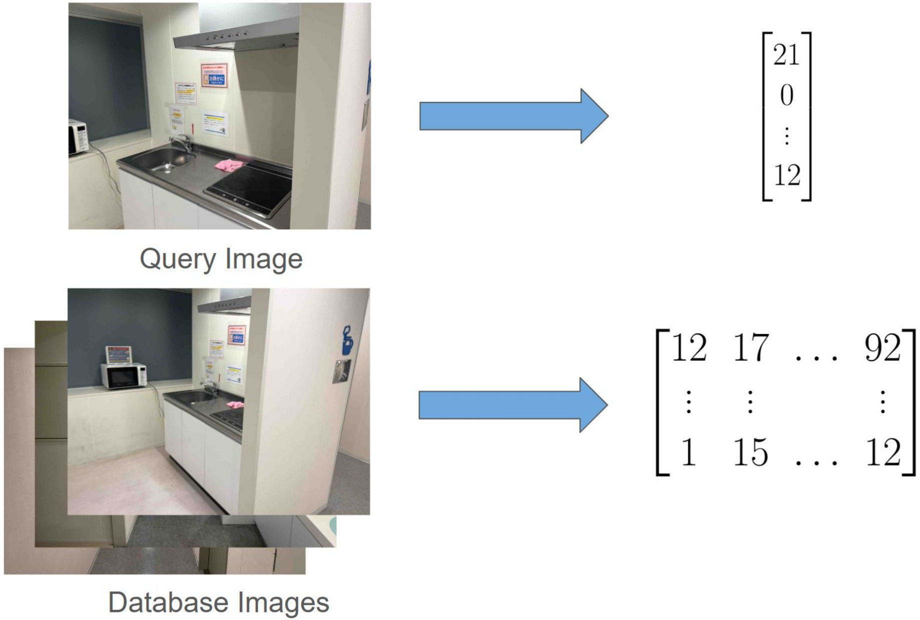
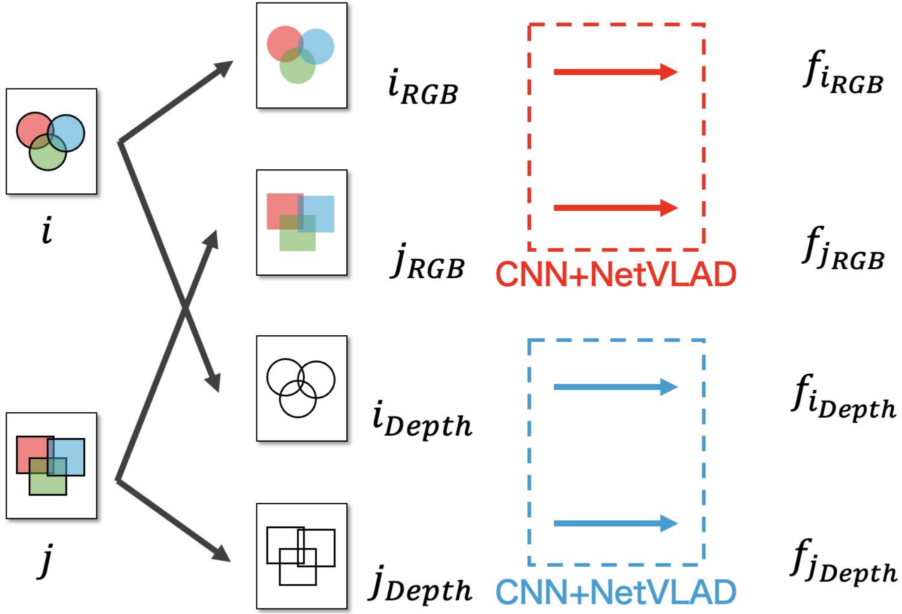


Figure: Global feature extraction.



$$s_{\mathrm{RGB}}(i,j) = \left\| f_{i_{\mathrm{RGB}}} - f_{j_{\mathrm{RGB}}} \right\|,$$
$$s_{\mathrm{Depth}}(i,j) = \left\| f_{i_{\mathrm{Depth}}} - f_{j_{\mathrm{Depth}}} \right\|$$

$$s(i,j) := \alpha \cdot s_{\mathrm{RGB}}(i,j) + (1-\alpha) \cdot s_{\mathrm{Depth}}(i,j)$$

α-Fusion

Figure: global feature extraction for late α-fusion.

RGB-D input → Global feature matching → **Local feature matching** → 3D projection → Point Cloud Alignment

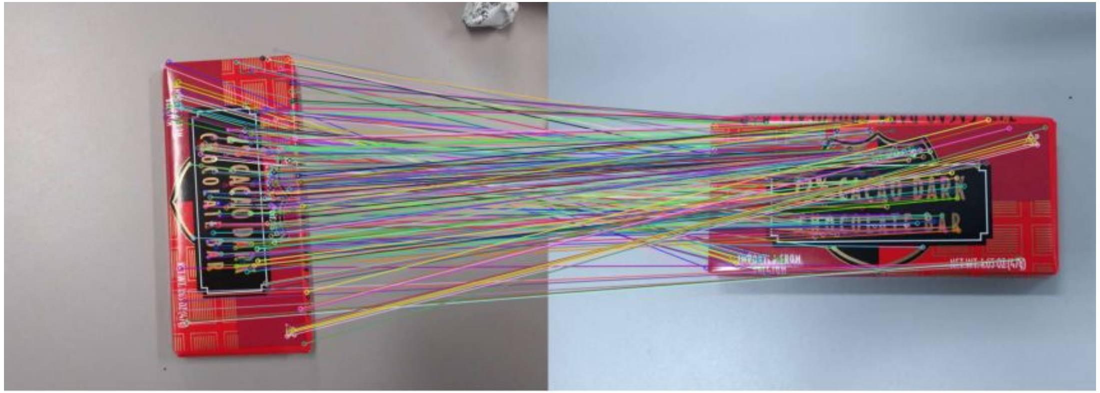


Figure: ASIFT matching

Purpose:

- More accurate matching
- Increased resolution for alignment

Methods:

- ASIFT (handcrafted)
- SuperPoint (deep learning)

Purpose:

- High resolution RGB
- Low resolution depth
- Localize detailed RGB key points to 3D points

Method:

- Pinhole camera model

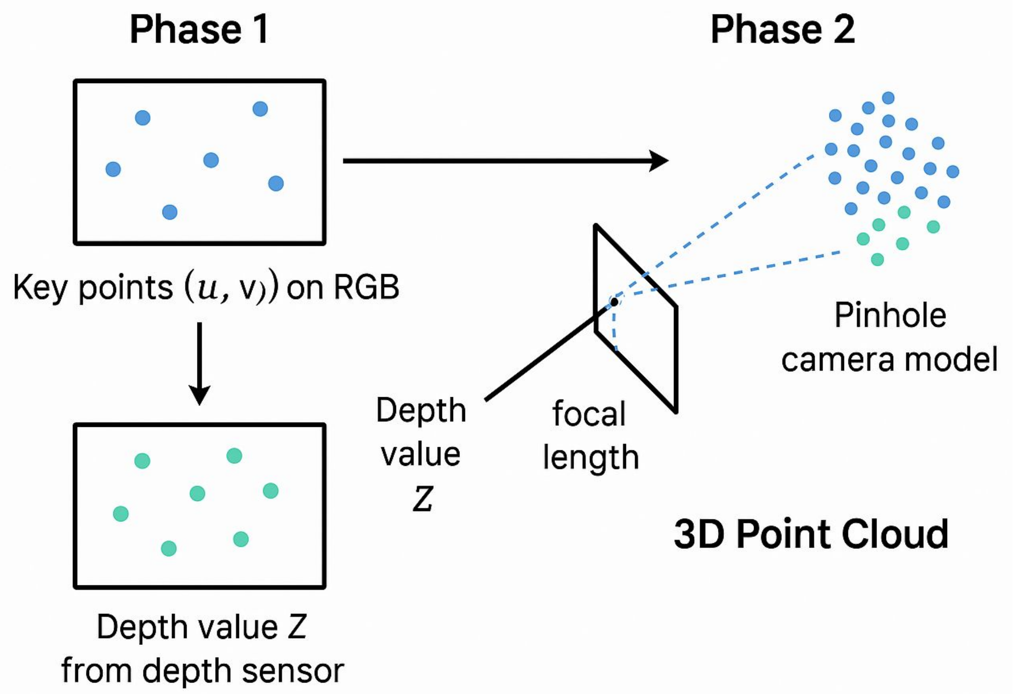
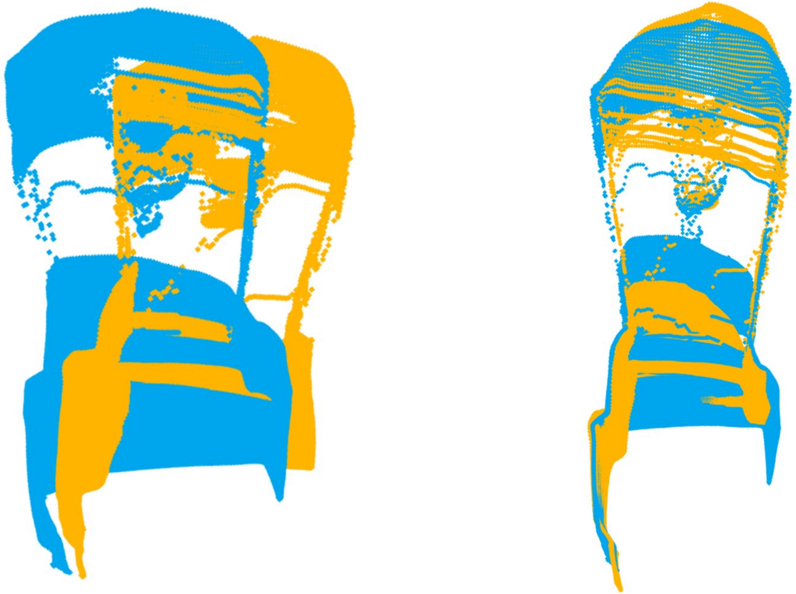


Figure: 3D projection using the pinhole camera model.



Unaligned

Aligned

Figure: Initial and aligned point clouds.

Purpose:

- Exploit 3D data
- Rigid transform to relative 6DoF

Methods:

- Kabsch Algorithm

Outline

1. Introduction:
    - The Kidnapped robot problem
2. Background:
    - Relocalization terminology
3. Our Method:
    - Concrete pipeline
4. **Method Validation:**
    - **Experiments, data and result**
5. Conclusion:
    - Summarization and future work

Our Results

RGB-D input

3D projection

Point cloud alignment

Global feature matching

Local feature matching

Validation:

- Small scale, each component
- Preliminary, whole pipeline

Datasets:

- Our own
- Stanford 2D/3D/Semantics

RGB-D input → Global feature matching → Local feature matching → 3D projection → Point cloud alignment

RGB-D input → Global feature matching → **Local feature matching** → 3D projection → Point Cloud Alignment

Local Matching

Should match with

- different camera position
- visually similar clutter
- different illumination

Conclusion:

- A-SIFT ✅

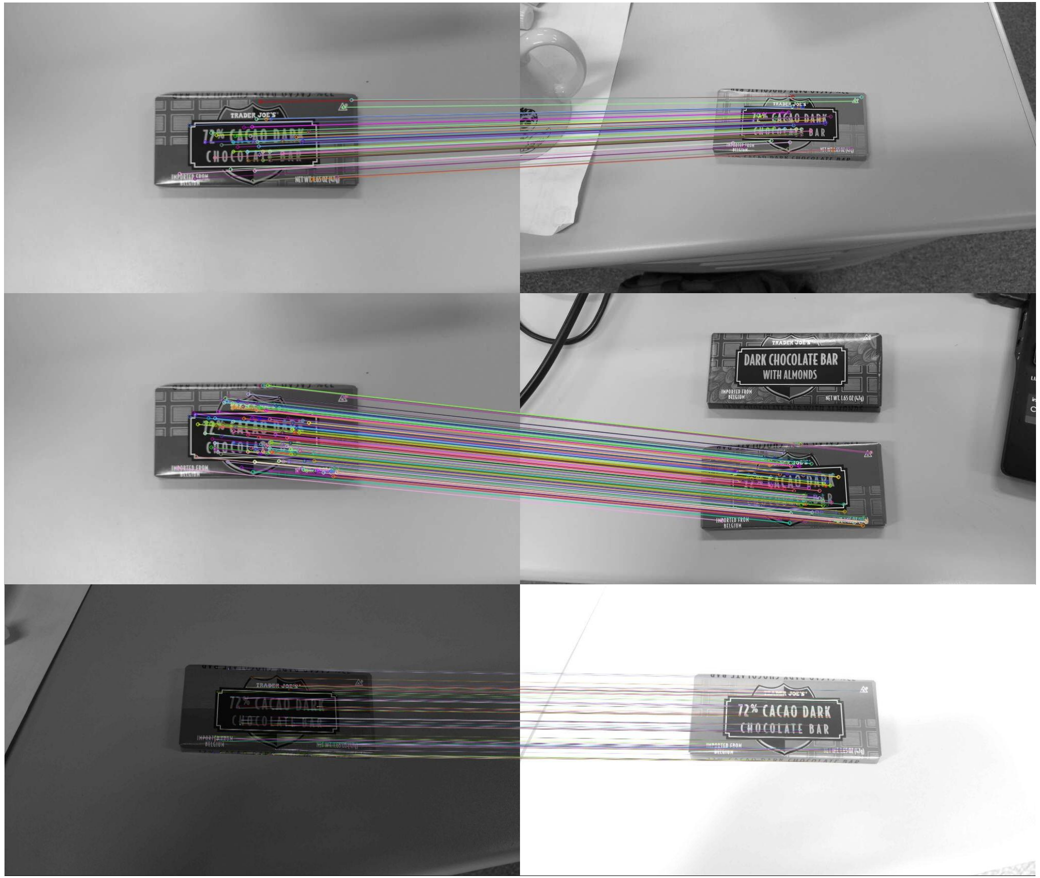


Figure: A-SIFT Results

RGB-D input → Global feature matching → **Local feature matching** → 3D projection → Point Cloud Alignment

Local Matching

Should match with

- different camera position
- visually similar clutter
- different illumination

Conclusion:

- A-SIFT ✅
- SuperPoint ✅

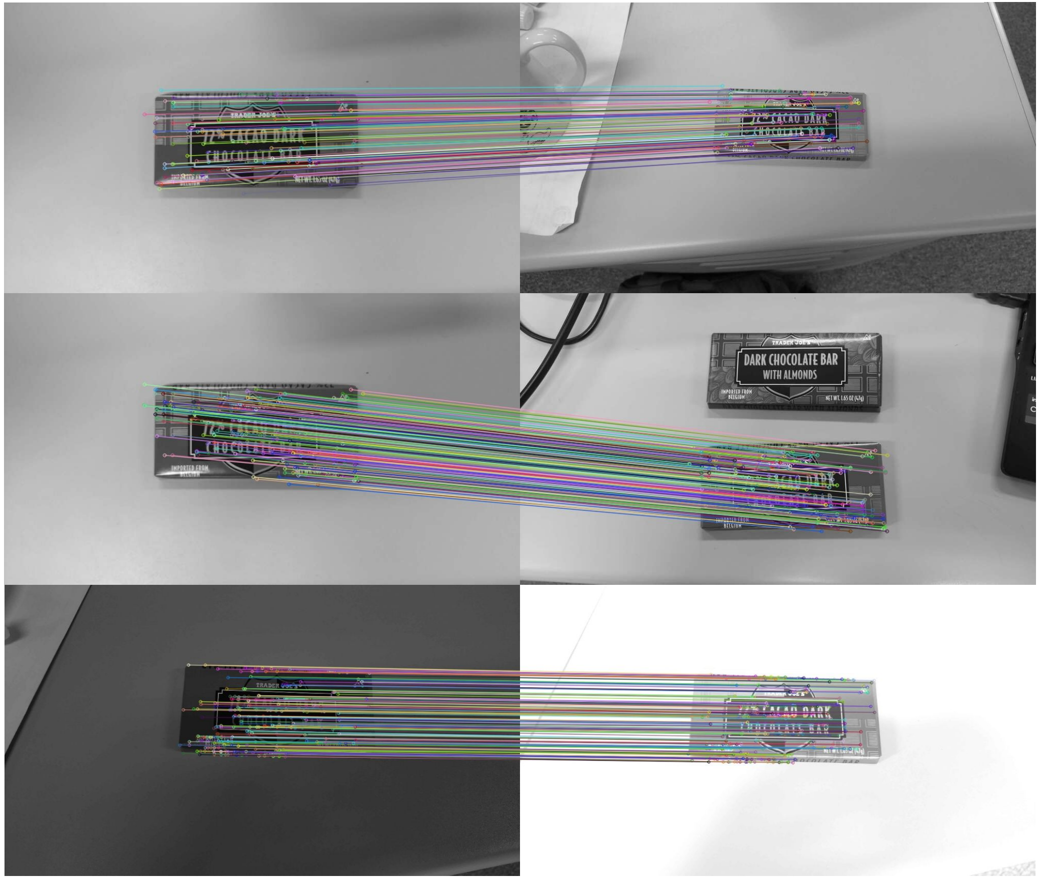


Figure: Superpoint Results

RGB-D input → Global feature matching → Local feature matching → 3D projection → **Point Cloud Alignment**

Point Cloud Alignment

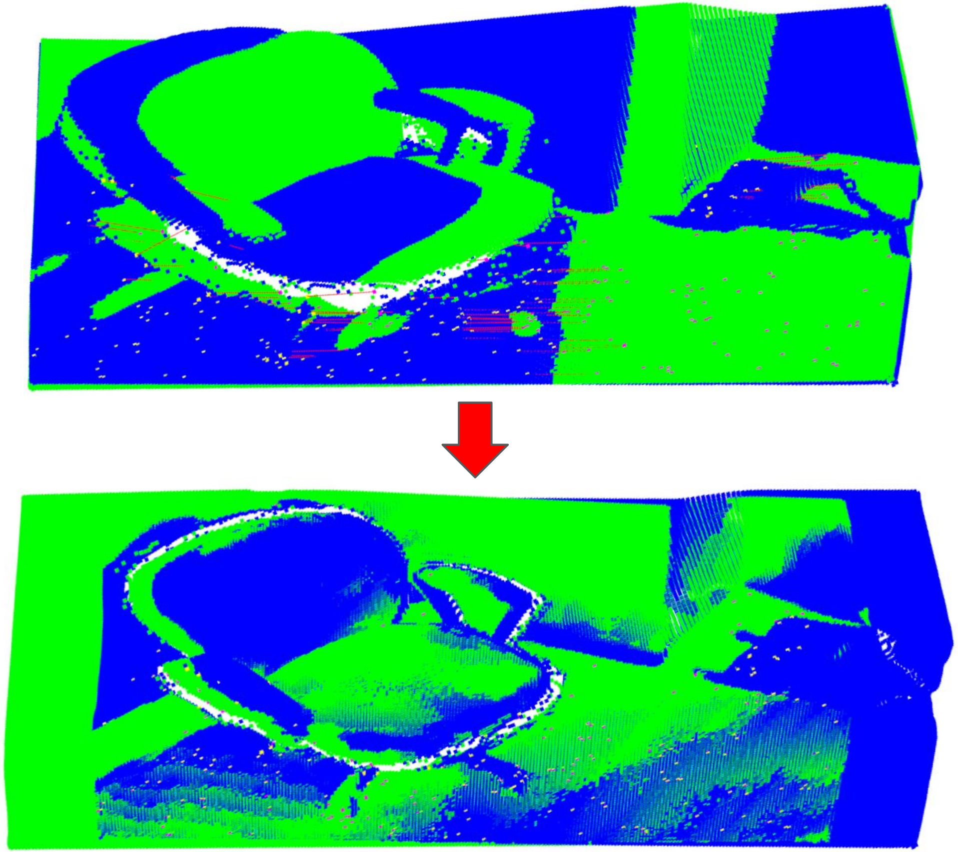
Criteria: do they look aligned

Procedure:

- Match RGB features
- Correlate 3D keypoints
- Use Kabsch (+RANSAC)

Results:

- Looks good ✅



RGB-D input → Global feature matching → Local feature matching → 3D projection → **Point Cloud Alignment**

6DoF estimation

Procedure:

- Align query to reference (known close)
- Compute relative 6DoF
- Compose with reference
- Compare to ground truth

Results:

- Good error ✅

Office 3, Frame 0

Office 3, Frame 12

Translation Error	<0.1 cm
Rotation Error	~3.75°

RGB-D input → **Global feature matching** → Local feature matching → 3D projection → Point Cloud Alignment

Global Retrieval

Criteria:

- Is the retrieved image close to the query image?
- Recall @ 1: Percentage of time first suggested image close
- Recall @ 3: Percentage first 3 suggested images close

Base NetVLAD, Stanford Area 3		
	Recall @ 1	Recall @ 3
Success	3670	3691
Total	3704	3704
Rate (%)	99.08	99.65

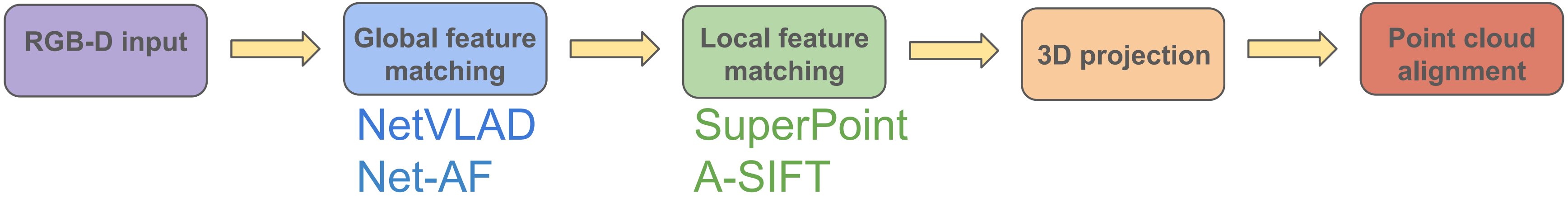
Results:

- Base NetVLAD ✅
- Trained NetVLAD ❌

Training:

- Triplet Loss
- Stanford Area 3

Trained Net_AF, Stanford Area 3		
	Recall @ 1	Recall @ 3
Success	1651	2400
Total	3704	3704
Rate (%)	44.57	64.79



Large Scale Testing: Base Dataset

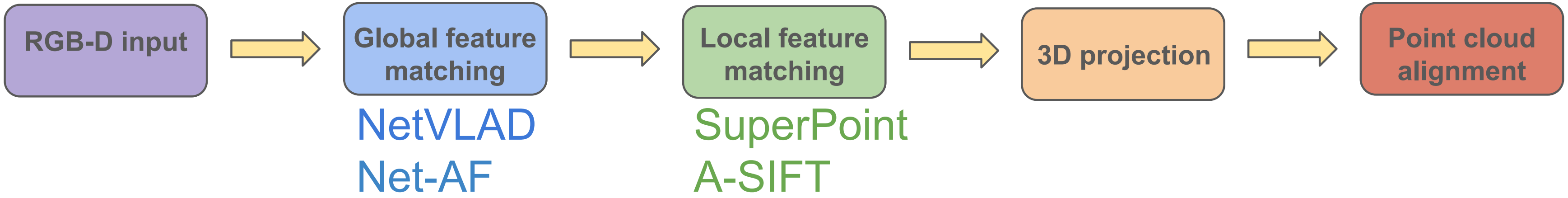
Dataset:

- Stanford 2D/3D/S Area 3
- IM Stanford 2D/3D/S Area 3

Success:

- <5cm
- <5°

Configuration	Trained	Dataset	Translation <5cm	Rotation <5°	Full Success
NetVLAD + SuperPoint	❌	Area 3	95.17%	86.69%	84.75%
NetVLAD + ASIFT	❌	Area 3	**97.15%**	**86.92%**	86.16%
Net-AF + SuperPoint	✅	Area 3	6.76%	23.31%	6.76%
ACE (SOTA Method)	✅	7-Scenes	Not Reported	Not Reported	**97.1%**



Large Scale Testing: Artificial Lighting Variance

Dataset:

- Stanford 2D/3D/S Area 3
- IM Stanford 2D/3D/S Area 3

Success:

- <5cm
- <5°

Configuration	Trained	Dataset	Translation <5cm	Rotation <5°	Full Success
NetVLAD + SuperPoint	❌	IM Area 3	TBD	TBD	TBD
NetVLAD + ASIFT	❌	IM Area 3	TBD	TBD	TBD
Net-AF + SuperPoint	❌	IM Area 3	TBD	TBD	TBD
Net-AF + ASIFT	❌	IM Area 3	TBD	TBD	TBD

Outline

1. Introduction:
    - The Kidnapped robot problem
2. Background:
    - Relocalization terminology
3. Our Method:
    - Concrete pipeline
4. Method Validation:
    - Experiments, data and result
5. **Conclusion:**
    - **Summarization and future work**

Conclusion

What we've done:

- Proposed novel algorithm for global relocalization
- Verified components work
- Tested untrained method

What still needs to be done:

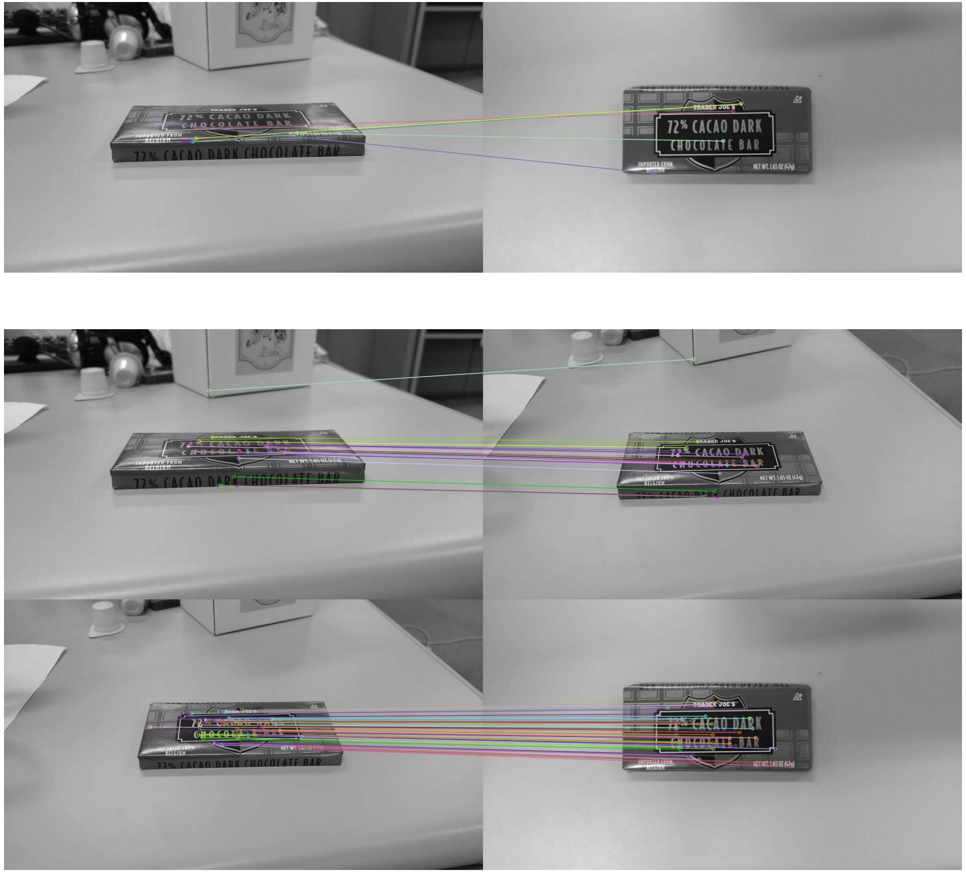
- Model training
- Larger scale testing
- More sophisticated methods

Future Work

- Extremely distorted images

Figure:
(Top) Failed matching,

(Middle, Bottom) Successful sequence.



- Color information
    - Color SIFT (CSIFT)

Future Work

- PointNetVLAD
  - Extracts local features directly from 3D point clouds.
- Dynamically determined α
  - Learn α as a function of (brightness and contrast)
- Training
  - Increasing descriptor dimension to 1024 with deeper backbones.
  - Larger batch sizes
  - More data

Thank You!

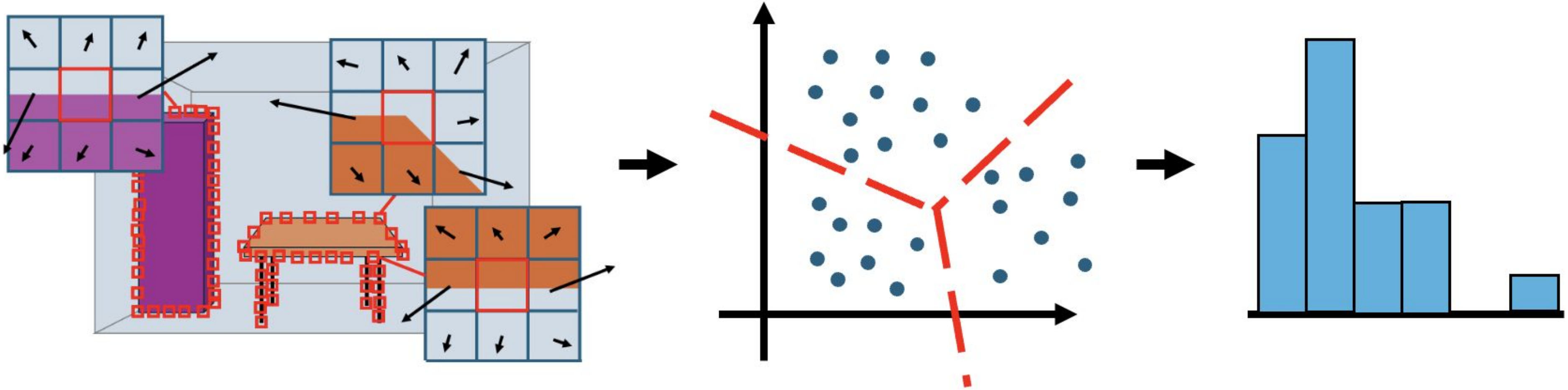
Questions?

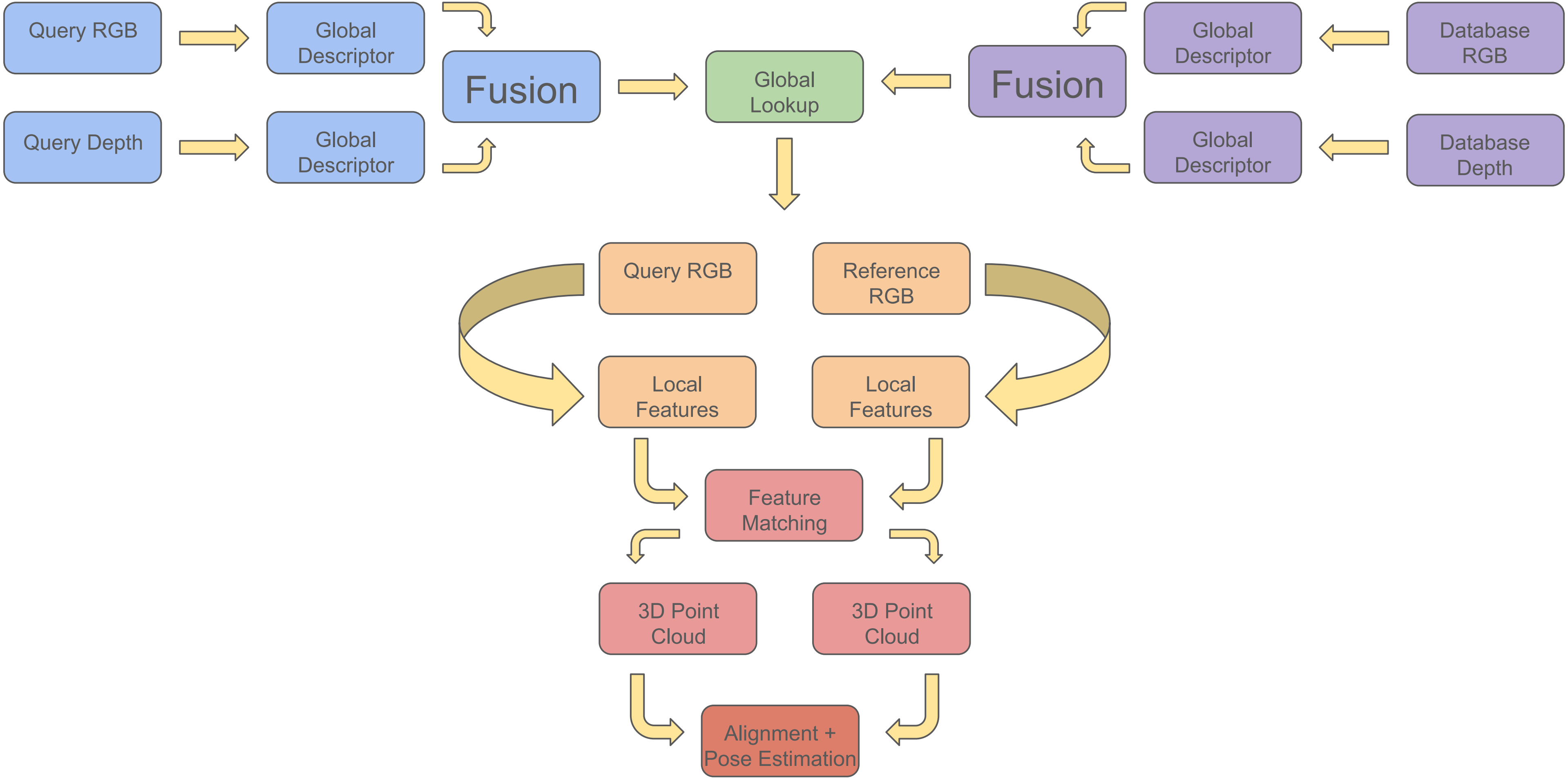
Global feature

Global feature: A method that represents an image with a single feature vector

Clustering: Groups similar local features together

Histogram: A vector that counts how many local features fall into each group




Query RGB
Global Descriptor
Query Depth
Global Descriptor
Fusion
Global Lookup
Fusion
Global Descriptor
Database RGB
Global Descriptor
Database Depth
Query RGB
Reference RGB
Local Features
Local Features
Feature Matching
3D Point Cloud
3D Point Cloud
Alignment + Pose Estimation

An Illumination Problem

Robots detect landmarks by its camera

I can detect that

An Illumination Problem

But if lighting condition is changed…

Is that a same object?

An Illumination Problem

Challenge: Visual relocalization methods often fail under lighting changes.

Solution

- **Objective**: Enhance global descriptor robustness to illumination changes using structural depth cues.
- **Solution**: Construct a new method with depth aware features.
- **Advantages**:
  - Illumination robustness
  - Geometry-aware retrieval
- **Takeaway**: A depth-aware method improves indoor global localization under variable illumination.

How hard is illumination change?

Table 2: Results on the Oxford RobotCar relocalization tracking benchmark [36]. We compare LM-Net (Ours) against other state-of-the-art methods (SuperGlue, R2D2, SuperPoint, and D2-Net). As can be seen from the results, our method almost consistently outperforms other SOTA approaches in terms of rotation AUC whilst achieving comparable results on translation AUC.

Sequence	Ours		SuperGlue [32]		R2D2 [26]		SuperPoint [8]		D2-Net [10]	
	t_{AUC}	R_{AUC}	t_{AUC}	R_{AUC}	t_{AUC}	R_{AUC}	t_{AUC}	R_{AUC}	t_{AUC}	R_{AUC}
Sunny-Overcast	79.83	**55.48**	**81.01**	52.83	80.86	53.57	78.95	50.03	71.93	39.0
Sunny-Rainy	71.54	**43.7**	**75.58**	40.59	74.84	41.23	69.76	37.12	65.63	27.5
Sunny-Snowy	59.69	**44.06**	**63.57**	43.64	62.92	41.78	60.85	40.02	55.65	30.86
Overcast-Rainy	80.54	**63.7**	79.99	61.64	**81.29**	61.23	80.36	61.56	75.66	51.06
Overcast-Snowy	55.38	47.88	57.67	47.16	**57.68**	**48.41**	55.39	44.96	51.17	34.54
Rainy-Snowy	68.57	**41.67**	69.91	39.87	**71.79**	39.86	67.7	38.05	61.91	27.74

Source: Lukas von Stumberg et al. “LM-Reloc: Levenberg-Marquardt Based Direct Visual Relocalization”. In: CoRR abs/2010.06323 (2020).

RGB-D input → Global feature matching → Local feature matching → 3D projection → **Point Cloud Alignment**

6DoF estimation

Procedure:

- Align query to reference (known close)
- Compute relative 6DoF
- Compose with reference
- Compare to ground truth

Results:

- Good error ✅

Office 3, Frame 0

Office 3, Frame 12

```
x error:  0.000
y error:  0.000
z error:  0.000
translation error:  0.000
theta error:  0.419
psi error:  0.532
phi error:  356.432
Best error reported by RANSAC:  1.329
Total keypoint alignment error:  1.329
Predicted direction vector:    [-0.04646283 -0.99877676  0.0169182 ]
Actual direction vector:    [ 0.01842026 -0.99949688  0.02582032]
Predicted translation:  [21.31999152  2.32363976  1.37246506]
Actual translation:     [21.320011  2.323802  1.372698]
Angle between two directions: 3.753252634798042
```