Robot Vision:
Machine Learned Features

Prof. Friedrich Fraundorfer

SS 2024

Slides created by Emanuele Santellani and Friedrich Fraundorfer

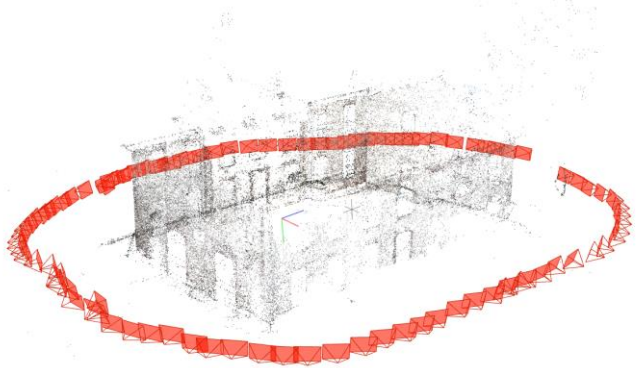
Outline

- Motivation
- Deep learning approach
- Performance comparison

Motivation

Establishing sparse sets of point correspondences between images is a fundamental task in many computer vision pipelines:

- 3D reconstruction (Structure from Motion)
- SLAM (Simultaneous Localization And Mapping)
- Visual Localization
- Object detection
- Object tracking



Deep Network approach

With the recent developments of deep neural networks, multiple local features extraction methods based on deep learning have been proposed in the past few years:

- LIFT [1] 2016
- HardNet [2] 2017
- SuperPoint [3] 2018
- R2D2 [4] 2019
- ASLFeat [5] 2020
- MD-Net [6] 2022

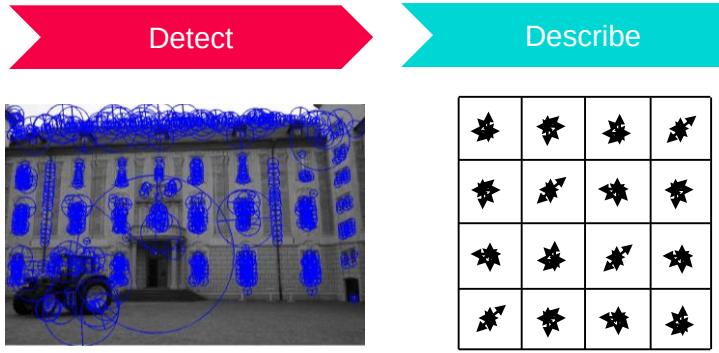
These approaches, in contrast with the hand-crafted classical methods, learn to find good keypoints and descriptors from the data directly.

After the local features are extracted (keypoints + descriptors) from each image, the descriptors are still commonly matched using the Mutual Nearest Neighbor strategy.

Classical approach vs Deep learning

Classic approaches: **detect then describe**

SIFT (1999), rootSIFT (2012), SURF (2006), ...



Pros:

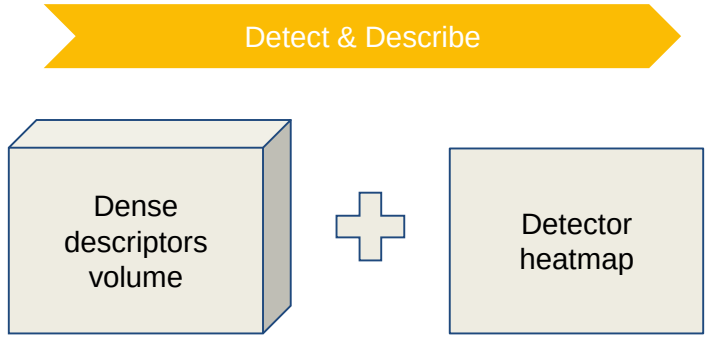
- ✓ General purpose
- ✓ Good point localization accuracy
- ✓ Easy to use
- ✓ Still competitive in non challenging scenarios

Cons:

- ✗ Keypoints are often unstable when changing view-point
- ✗ Keypoints on unreliable objects (vegetations, clouds, …)
- ✗ Do not handle repetitive patterns properly (similar descriptors)

Recent deep learning approaches: **detect&describe**

SuperPoint (2018), R2D2 (2019), ASLFeat (2020), MD-Net (2022) ...



Pros:

- ✓ Work well in specific difficult scenarios
- ✓ Their features benefits from higher level information
- ✓ Can avoid unstable points
- ✓ Can handle repetitive patterns better

Cons:

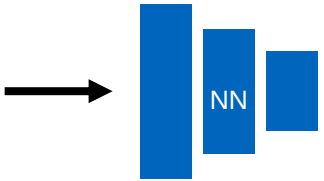
- ✗ Often provide less precise detections
- ✗ Can generalize poorly in unseen scenes

Deep Network approach

While for image classification is clear what the predicted class should be (supervised training), for the local feature extraction task there is not a clear definition of what a good keypoint is.

image classification

SUPERVISED LEARNING



Cat 0.8
Dog 0.2

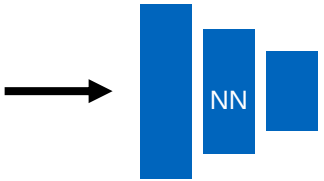
Ground truth

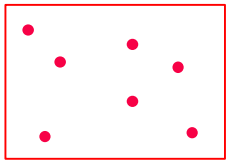
Cat

keypoint detection

?

LEARNING





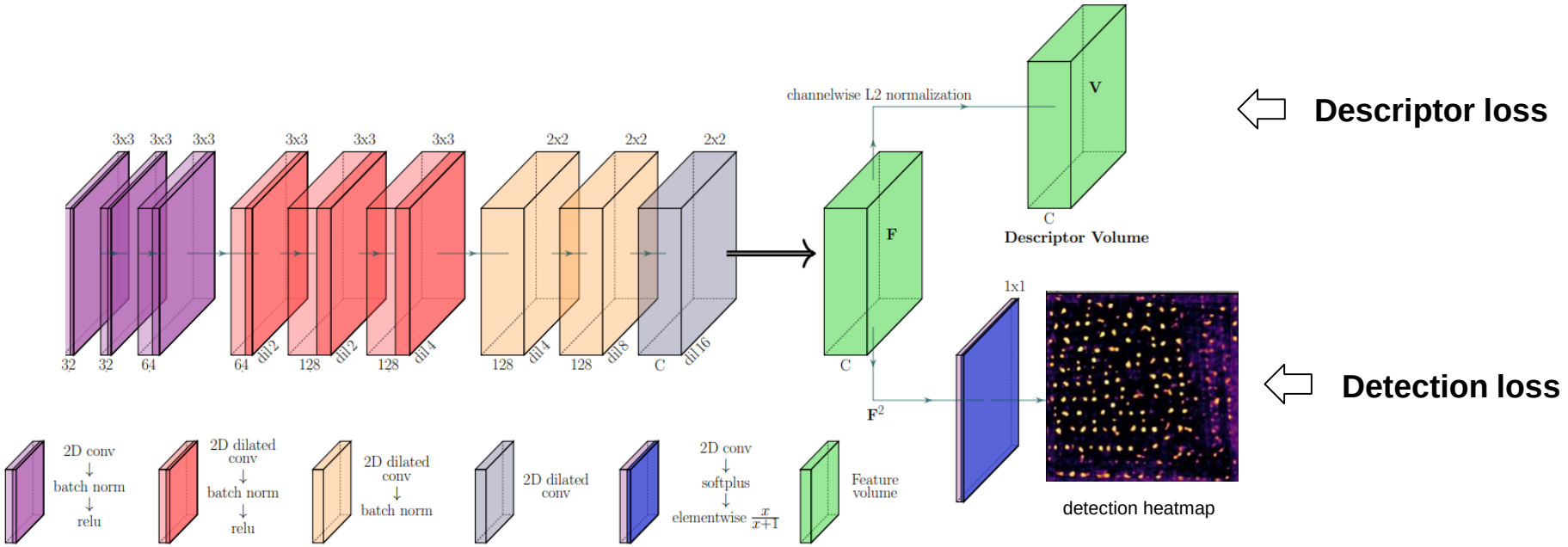
Ground truth

?

Deep methods: MD-Net [6]

MD-Net learns to detect keypoints **without requiring any GT keypoint**.

The training relies on pairs of images with known pixel-to-pixel transformation.



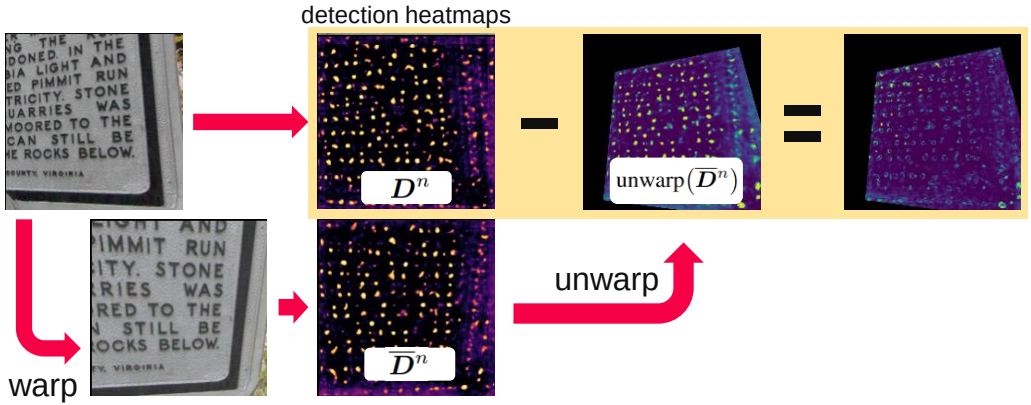
The network trains in 13h on a 1080Ti, consuming 710k images.

Image from [6]

Deep methods: MD-Net detection losses

In order to find keypoints that are **repeatable** (they are detected again in the same spot in a different image depicting the same scene) and well distributed in the image MD-Net employs a combination of two detection losses:

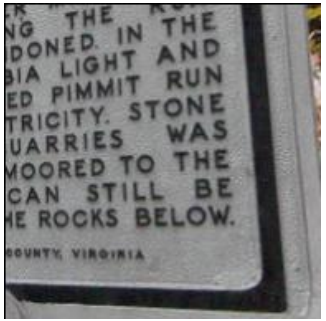
Similarity loss: the detection heatmaps should correspond



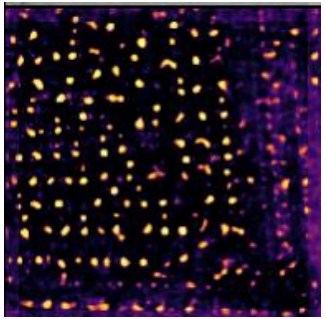
$$\mathcal{L}_{\text{sim}}(\boldsymbol{D}^n, \overline{\boldsymbol{D}}^n) = \left(\boldsymbol{D}^n - \text{unwarp}(\overline{\boldsymbol{D}}^n)\right)^2$$

Peaky loss: encourages local peaks

image



detection heatmap $\boldsymbol{D}^n$

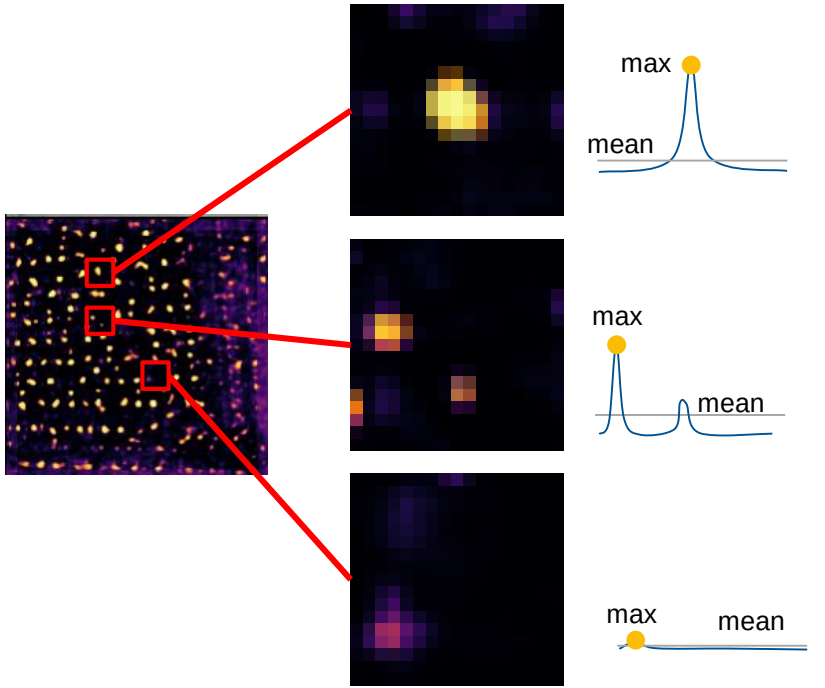


$$\mathcal{L}_{\text{peaky}}(\boldsymbol{D}^n) = 1 - \left(\max_{p \in \mathcal{P}} \boldsymbol{D}_p^n - \text{mean}_{p \in \mathcal{P}} \boldsymbol{D}_p^n\right)$$

where p is a 16x16px moving window

Deep methods: MD-Net peaky loss

$$\mathcal{L}_{\text{peaky}}(\boldsymbol{D}^n) = 1 - \left(\max_{p \in \mathcal{P}} \boldsymbol{D}_p^n - \operatorname{mean}_{p \in \mathcal{P}} \boldsymbol{D}_p^n \right)$$



1 - (0.9 - 0.1) = 0.2 **good**

1 - (0.7 - 0.2) = 0.5 **ok**

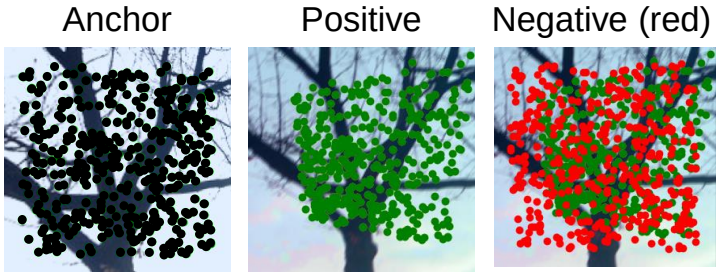
1 - (0.3 - 0.1) = 0.8 **bad**

Deep methods: MD-Net descriptors loss

Corresponding descriptors should be similar, and dissimilar from any other descriptor.

To achieve this, a set of anchor keypoints are randomly sampled from one image. The corresponding positives are obtained from a second image using the known pixel-to-pixel transformation.

A set of non matching descriptors can be sampled randomly from the second image.



One commonly used loss is the **Triplet loss**, which takes an *anchor-positive-negative* triplet and increase the *anchor-positive* score (dot product) while lowering the anchor-negative one.

$$\mathcal{L}_{Triplet} = [m - S(\mathbf{f}, \mathbf{f}_+) + S(\mathbf{f}, \mathbf{f}_-)]_+$$

The parameter m (e.g. 0.5) in this formulation sets the minimum margin we require from each triplet.

Each score **S** is a value between [-1,+1].

For each triplet, the negative can be chosen following different strategies. MD-Net samples the hardest negatives (i.e. the one with the highest *anchor-negative* score).

Deep methods: MD-Net triplet loss

$$\mathcal{L}_{Triplet} = [m - S(\mathbf{f}, \mathbf{f}_+) + S(\mathbf{f}, \mathbf{f}_-)]_+$$

$[*]_+ = \max(0, *)$

[0.5	- (1.0)	+	(-1.0)	]	=	0.0
[0.5	- (0.8)	+	(0.2)	]	=	0.0
[0.5	- (0.6)	+	(0.3)	]	=	0.2
[0.5	- (0.2)	+	(0.6)	]	=	0.9

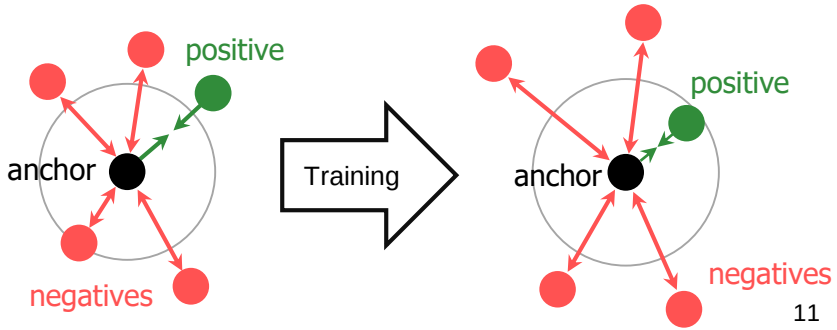
Perfect case, the positive score is 1.0 and the negative -1.0.

The difference between the positive and negative score is 0.6, which is greater than our chosen margin. The loss is 0.

Even if the positive score is greater than the negative one, the difference is smaller than the margin. The loss is low.

The positive score is smaller than the negative one. This is a wrong match! The loss is high.

Descriptor space 2D projection before and after training. The positive got closer to the anchor, while all the negatives has been pushed away.



Performances comparison

HPatches is a common benchmark to evaluate local feature performances. It is composed by two image sets:

- i: static images with different lighting conditions
- v: pictures from different viewpoints of planar scenes

COMPARISON ON HPATCHES

	Method	MMA ↑			MS ↑		
		@1px	@2p	@3px	@1px	@2px	@3px
v	MD-2-Net (ours)	0.316	**0.600**	**0.722**	0.171	0.313	0.393
	R2D2 [19]	0.280	0.568	0.700	0.118	0.228	0.273
	ASLFeat [20]	**0.332**	0.565	0.675	**0.203**	**0.338**	**0.398**
	Upright-SIFT [7]	0.313	0.472	0.533	0.167	0.247	0.277
i	MD-2-Net (ours)	**0.480**	0.658	0.765	0.242	0.323	0.368
	R2D2 [19]	0.377	0.660	**0.797**	0.170	0.285	0.336
	ASLFeat [20]	0.469	**0.664**	0.774	**0.290**	**0.398**	**0.456**
	Upright-SIFT [7]	0.344	0.475	0.528	0.161	0.216	0.238
overall	MD-2-Net (ours)	**0.398**	**0.630**	0.743	0.206	0.317	0.369
	R2D2 [19]	0.326	0.612	**0.747**	0.143	0.255	0.304
	ASLFeat [20]	**0.398**	0.613	0.723	**0.245**	**0.367**	**0.426**
	Upright-SIFT [7]	0.327	0.473	0.531	0.164	0.232	0.258

Mean Matching Accuracy: mean ratio between the number of correct matches and the total number of proposed matches

Matching Score: mean ratio between the number of correct matches and the number of keypoints extracted at one image in the area shared with the other.

All the deep methods outperform Upright-SIFT.

Upright-SIFT is the non-rotation invariant version which performs better than the original SIFT algorithm in this benchmark.

Table from [6]

Performances comparison

Image Matching Benchmark is a benchmark which evaluates local features for the stereo and multiview pose recovery.

IMAGE MATCHING BENCHMARK - RESTRICTED KEYPOINTS 2048

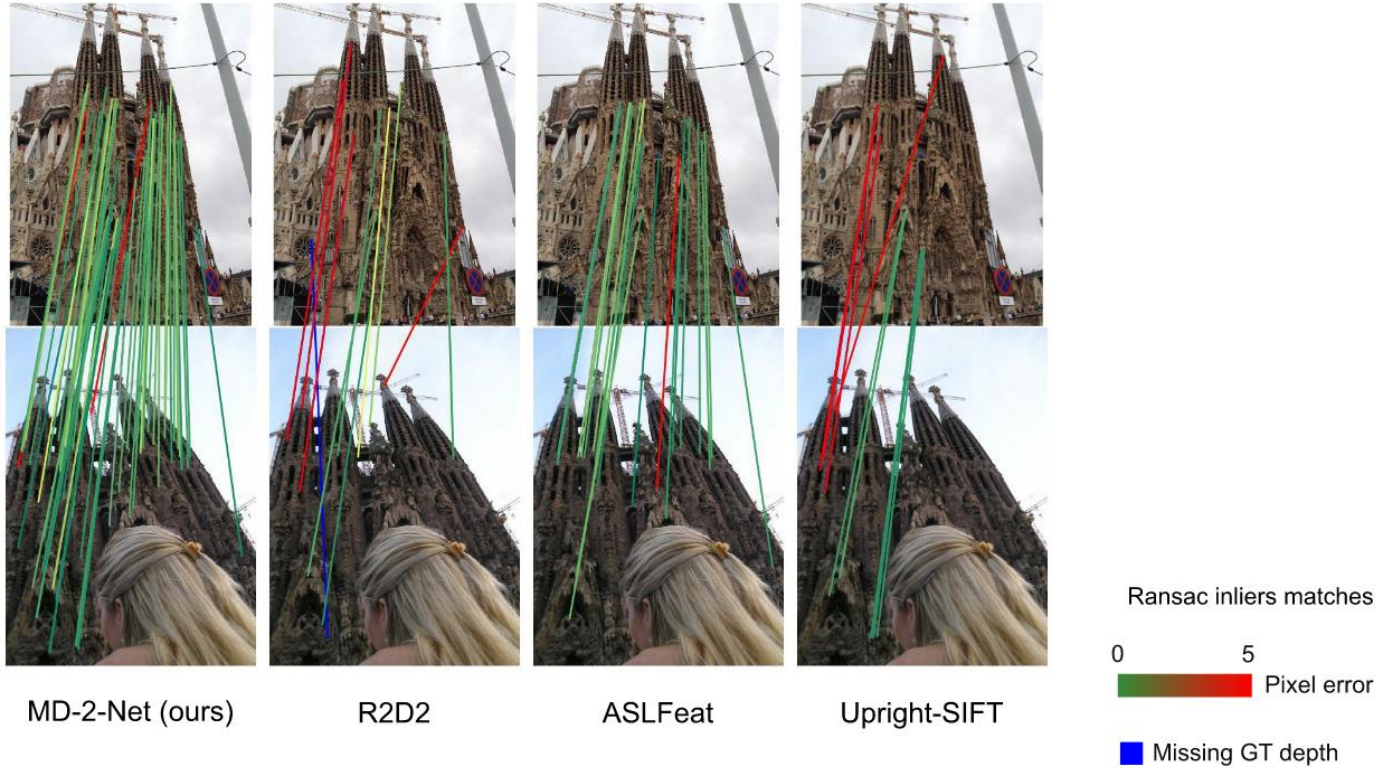
		Stereo					Multiview					Avg
	Method	NF	NI ↑	Rep@3px ↑	MS@3px ↑	mAA@10° ↑	NM ↑	NL ↑	TL ↑	ATE ↓	mAA@10° ↑	mAA@10° ↑
Phototourism	MD-2-net (ours)	2047.5	**233.0**	0.396	**0.792**	**0.455**	238.6	**1391.5**	**4.604**	**0.411**	**0.708**	**0.581**
	R2D2 [19]	2048.0	201.5	0.429	0.746	0.390	**294.3**	1225.9	4.280	0.478	0.640	0.515
	ASLfeat [20]	2042.6	126.0	**0.431**	0.749	0.337	157.5	1106.6	4.415	0.533	0.556	0.446
	Upright-SIFT [7]	1892.8	98.6	0.333	0.788	0.383	148.0	1165.7	4.118	0.524	0.555	0.469
PragueParks	MD-2-net (ours)	2048.0	**175.5**	0.039	0.027	**0.542**	236.3	**605.8**	**3.197**	6.753	**0.451**	**0.497**
	R2D2 [19]	2048.0	167.0	0.032	0.025	0.539	**338.9**	526.0	3.170	6.837	0.444	0.491
	ASLfeat [20]	2048.0	110.5	0.059	**0.029**	0.401	217.1	574.4	3.036	6.414	0.400	0.403
	Upright-SIFT [7]	2048.0	119.8	**0.060**	0.027	0.414	157.3	433.3	2.989	**5.666**	0.361	0.387

The metrics in the table are Number of Features (NF), Number of Inlier matches (NI), Repeatability (Rep), Matching Score (MS), Number of inlier Matches filtered by COLMAP, (NM), Number of triangulated Landmarks (NL), Track Length (TL), Absolute Trajectory Error (ATL), mean Average Accuracy (mAA) up to 10°.

Again, the deep methods outperform Upright-SIFT in most of the metrics, especially on the Mean Average Accuracy.

Table from [6]

Hand-crafted vs. Machine learned features



References

[1] LIFT: Learned Invariant Feature Transform. Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, Pascal Fua. ECCV 2016

[2] Working hard to know your neighbor's margins: Local descriptor learning loss. Anastasiya Mishchuk, Dmytro Mishkin, Filip Radenovic, Jiri Matas. NIPS 2017

[3] SuperPoint: Self-Supervised Interest Point Detection and Description. Daniel DeTone, Tomasz Malisiewicz, Andrew Rabinovich. CVPR workshop 2018

[4] R2D2: Repeatable and Reliable Detector and Descriptor. Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, Martin Humenberger. NIPS 2019.

[5] ASLFeat: Learning Local Features of Accurate Shape and Localization. Zixin Luo, Lei Zhou, Xuyang Bai, Hongkai Chen, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, Long Quan. CVPR 2020.

[6] MD-Net: Multi-Detector for Local Feature Extraction. Emanuele Santellani, Christian Sormann, Mattia Rossi, Andreas Kuhn, Friedrich Fraundorfer. ICPR 2022