DeDoDe v2: Analyzing and Improving the DeDoDe Keypoint Detector

Johan Edstedt[1] Georg Bökman[2] Zhenjun Zhao[3,4]
[1]Linköping University, [2]Chalmers University of Technology
[3]The Chinese University of Hong Kong, [4]Texas A&M University

Figure 1. **DeDoDe (left) vs DeDode v2 (right).** We propose DeDoDe v2, an improved keypoint detector following the *detect don't describe* approach, whereby the detector is descriptor agnostic. We improve the DeDoDe detector, as demonstrated in the figure. DeDoDe struggles with *clustering*, whereby keypoints are overly detected in distinct regions. This, in turn, causes it to underdetect in other regions, causing performance to degrade. In contrast, our proposed detector produces diverse but repeatable keypoints for the entire scene.

Abstract

In this paper, we analyze and improve into the recently proposed DeDoDe keypoint detector. We focus our analysis on some key issues. First, we find that DeDoDe keypoints tend to cluster together, which we fix by performing non-max suppression on the target distribution of the detector during training. Second, we address issues related to data augmentation. In particular, the DeDoDe detector is sensitive to large rotations. We fix this by including 90-degree rotations as well as horizontal flips. Finally, the decoupled nature of the DeDoDe detector makes evaluation of downstream usefulness problematic. We fix this by matching the keypoints with a pretrained dense matcher (RoMa) and evaluating two-view pose estimates. We find that the original long training is detrimental to performance, and therefore propose a much shorter training schedule. We integrate all these improvements into our proposed detector DeDoDe v2 and evaluate it with the original DeDoDe descriptor on the MegaDepth-1500 and IMC2022 benchmarks. Our proposed detector significantly increases pose estimation results, notably from 75.9 to 78.3 mAA on the IMC2022 challenge. Code and weights are available at github.com/Parskatt/DeDoDe.

1. Introduction

Obtaining corresponding pixels in multiple images of the same scene is a cornerstone task in computer vision, being an integral part of most structure-from-motion pipelines. Classically, finding correspondences between two images is split into three sub-tasks: keypoint detection, description and matching. This paper focuses on keypoint detection, building on the cutting-edge DeDoDe detector [12]. A keypoint is a local image feature distinct enough to be recognized under strong viewpoint and illumination variations. However, it is difficult to precisely define into what this means (*e.g.*, how strong variations should be allowed?), and we argue that the community still poorly understands what exactly a keypoint should be. As such, it is of interest to understand the training process of keypoint detectors and thoroughly investigate what components matter. In this paper, we conduct such an analysis, resulting in several simple modifications to the training pipeline of the DeDoDe detector, significantly improving its performance. We hope our analysis will further encourage research into finding new and better objectives for keypoint detection.

Our main contributions are as follows.

1. We introduce a series of training enhancements, including non-max suppression and improved data augmentation. See Section 3.1, Section 3.3.
2. We shorten the training time of the DeDoDe detector to 20 *minutes* on a single A100 GPU while improving performance. See Section 3.2.
3. We detail a multitude of tested modifications that did not work. See Section 3.4.

2. Related Work

The classical approach to finding correspondences between two images is to rely on a keypoint detector and a keypoint descriptor and match keypoint descriptions by some variant of nearest neighbours. Influential works include the Harris and Shi-Tomasi corner detectors [15, 23] as well as the SIFT detector and descriptor [19] based on finding local extrema in scale space. These classical methods have been improved and refined over the years, *e.g.* through RootSIFT [1], HarrisZ [5] and HarrisZ+ [4]. Recently, however, deep learning approaches have become increasingly popular for both keypoint detection and description, *e.g.* Tilde [31], SuperPoint [9], AffNet [20] and later works [3, 10, 21, 30, 33, 34], demonstrating impressive performance boosts compared to the hand-crafted classical methods. Furthermore, improving on the detector-descriptor pipeline, graph neural network-based descriptor matching such as SuperGlue and follow-up work [18, 22, 24] as well as end-to-end semi-dense methods starting with LoFTR [6, 8, 25, 26, 32] and dense image matchers like GLU-Net [11, 13, 27–29] have started to see increasing use, however at non-negligible computational expense. DeDoDe [12] showed that the more straightforward detector-descriptor pipeline remains competitive with the newer end-to-end approaches. Edstedt et al. [12] suggested decoupling the detector and descriptor training to reduce the reliance of the detector on a simultaneously trained descriptor. Decoupling the two further enables research to focus on only the keypoint detector, which is the aim of this paper. In the next section, we recap how the DeDoDe detector was trained in [12].

2.1. The DeDoDe Detector

The main idea in DeDoDe is to train the detector with 3D tracks from structure-from-motion (SfM) reconstructions as a prior. The idea is that keypoints from an available detector (in this case, SIFT) are filtered by the SfM process to obtain *good* keypoints. To this end, the large MegaDepth [17] dataset of SfM reconstructions from internet images of tourist locations is used as the source for SfM tracks and hence keypoint priors.

In more detail, the training objective for the DeDoDe detector consists of 1) the cross entropy between a predicted keypoint probability map over the input image and a target probability map, and 2) a coverage regularization encouraging probability spread over the MVS from MegaDepth per image. The following procedure generates the target probability map. During training, pairs of overlapping views are sampled. As the base target for the probability map of each view, the 2D projections of the 3D tracks visible in both views are used. This base target is smoothened by convolving with a Gaussian. The probability maps of each view are then multiplied after being warped to the other view. To add a degree of self-supervision to the training, the probability maps are multiplied by the predicted probability map in each view. Finally, the target probability maps are binarized by thresholding at an adaptive value, giving k keypoints per batch.

3. Analysis and Improvements

In this section, we analyze issues of the DeDoDe detector and propose a set of improvements to solve them.

3.1. Preventing Clustering

We observe that the original DeDoDe detector tends to detect clusters (*cf* Figure 1, Figure 2). This is undesired, as it reduces the diversity and coverage of the keypoints. However, we found that naively enforcing NMS during test time did not work well.

NMS During Training. We are inspired by the peakiness loss from, *e.g.*, R2D2 [21] and the built-in soft-NMS in detectors such as ALIKED [34], and propose a train-time NMS objective. To this end, we do top-k after performing a $h \times h$ NMS on the posterior detection distribution. That

Figure 2. **Clusters in DeDoDe Detections.** The DeDoDe detection objective does not explicitly enforce sparsity in the detections. This has the side-effect of the network producing so-called *clusters* of detections in particularly salient areas of the image. This is problematic in downstream tasks, as it means that many keypoints must be sampled to ensure repeatability.

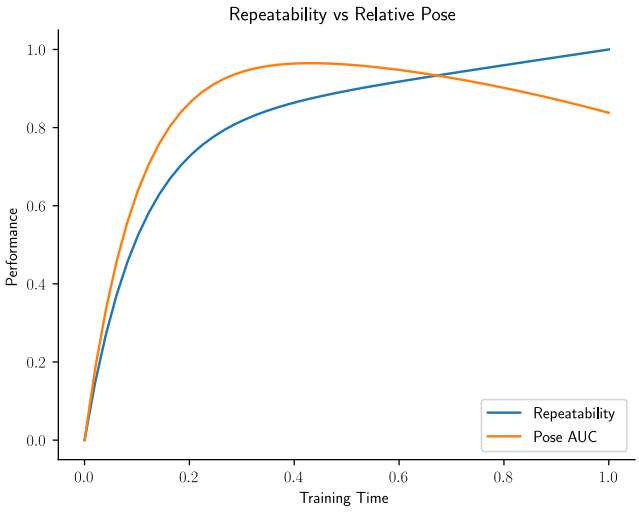

Figure 3. **Overfit to repeatability objective.** We qualitatively illustrate the tension between repeatability and downstream relative pose objectives. We found that during the course of training, while the keypoints tended to become more distinct and repeatable, this resulted in less distinct regions getting almost no keypoints, in particular outside regions with COLMAP MVS, resulting in worse relative pose estimates.

is, after combining the detection prior with the logit predictions of the detector, we additionally enforce that the score be a local maximum to be set as a target. In practice, we set $h = 3$, as it worked the best empirically. This simple change alleviated many issues in the original detector, *cf*. Figure 1.

3.2. Training Time

The original DeDoDe detector is trained with 800,000 image pairs on the MegaDepth dataset. We find that while the repeatability of the keypoints keeps increasing during the training, even on the test set, this does not transfer to the downstream objective of two-view relative pose estimation. We qualitatively illustrate this in Figure 3.

However, it is not entirely obvious how to measure this downstream objective as the detector is decoupled from the descriptor. Measuring the AUC would seemingly require recoupling the objectives, which is problematic [12].

Instead, we propose using RoMa [13] to match the keypoints to estimate downstream usability. When evaluating the original DeDoDe detector in this way, it quickly overfits the repeatability metric and that performance on pose estimation drops during training. To ensure that the detector did not overfit the scenes, we conducted an experiment where we included the test scenes in the training data. Perhaps surprisingly, we saw no major difference in performance and a similar downward trending curve over time. This indicates that there is a more fundamental issue between the detection objective and pose estimation, the investigation of which we leave for future study.

No matter the cause, we thus choose to drastically reduce the training time of the detector, setting it to 10,000 image pairs, which significantly increases performance. Furthermore, the decrease in training time has the additional benefit of requiring significantly less compute to train, with training of DeDoDe v2 taking $\approx$ 20 minutes on a single A100 GPU.

3.3. Minor Improvements

Here we discuss some minor changes and improvements we make to the training of the detector.

Top-k Computation. DeDoDe computes the top-k over a minibatch[1] instead of per-pair. While this relaxes the assumption that each pair must contain a certain number of matching keypoints, it is problematic as difficult pairs may receive very few keypoints. We change this computation to be independent between the pairs.

Augmentation. We follow the approach of Steerers [7] and train the detector using random rotations in $\{0, 90, 180, 270\}$. We additionally include random horizontal flips. This makes the detector more robust to large rotations.

[1]This is not explicitly stated in the paper but can be observed in the released code: github.com/Parskatt/DeDoDe.

Figure 4. **Sensitivity to large rotations.** The original DeDoDe detector (left) is sensitive to large in-plane rotations. This was first noted by Bökman et al. [7]. We extend their ideas and additionally include joint horizontal flips. DeDoDe v2 produces more consistent keypoints under rotation of the input image (right). We plot the top 5,000 keypoints in all images.

3.4. Changes with no Effect

Here, we describe a set of different hypotheses that turned out to have a negative or negligible effect on the detector's performance. We include these experiments for the curious reader.

Diversity at Inference. DeDoDe has a local density estimate post-hoc during inference. This is controlled by a parameter α. In DeDoDe $\alpha = 1/2$. We experimented with setting it to other values $\in [0.5, 1]$. We found no significant improvement in pose estimation results.

Smoothness of Detection Prior. The detection prior smoothing in DeDoDe is assumed to be Normal with standard deviation $\sigma = 0.5$. We experimented with setting it to other values (lower and higher). We found that both lower and higher values performed slightly worse.

Annealing Prior Strength. The prior strength in DeDoDe is set to 50, which in practice means that the prior detections will always end up in the top-k target. We found that decaying the strength over training significantly increases the repeatability of the keypoints. However, it simultaneously significantly decreases the downstream pose AUC. We hy-

pothesize that this is due to the network disregarding less repeatable keypoints that nonetheless are important for accurate pose estimation.

Changing k in top-k. We experimented with setting different $k \in [512, 2048]$. We found that while there was a slight difference in performance, the setting of $k = 1024$ from DeDoDe was seemingly optimal.

Changing Regularizer. DeDoDe uses a coverage regularization:

$$\mathcal{L}_{\text{coverage}} = \text{CE}(\mathcal{N}(0, \sigma^2) * p_{f_\theta}, \mathcal{N}(0, \sigma^2) * p_{\text{MVS}}). \quad (1)$$

with $\sigma = 12.5$ pixels. We investigated changing σ as well as removing the regularization, as well as replacing it with a uniform regularizer. We found that these changes had either negligible or negative effects.

Learning Rate. The default learning rate in DeDoDe is 10^{-4} for the decoder and $2 \cdot 10^{-5}$ for the encoder. Changing these had a negligible effect on performance.

Training Resolution. Since DeDoDe is trained on 512×512 resolution and tested on 784×784 resolution, we believed that using a random crop strategy during training (where the crops come from 784×784 images) would alleviate potential train-test resolution gaps. This, however, turned out to have little effect on performance.

NMS During Inference. We found that even when trained with NMS, our detector still produces worse detections when NMS is applied post-hoc. However, the reduction in performance is significantly lower than for the baseline model.

4. Experiments

We train the detector for 10,000 image pairs on the MegaDepth dataset, using the same training split as DeDoDe [12]. We use a fixed image size of 512×512. We use a batch size of 7. Training is done on a single A100 GPU and takes ≈ 20 minutes.

We run all SotA experiments at a resolution of 784×784 and sample top-k keypoints. We use the descriptors from [12].

4.1. SotA Comparison

MegaDepth-1500 Relative Pose: MegaDepth-1500 is a relative pose benchmark proposed in LoFTR [25] and consists of 1500 pairs of images in two scenes of the MegaDepth dataset, which are non-overlapping with our training set. We mainly compare our approach to DeDoDe and previous detector descriptor methods. As in DeDoDe, we tune the methods for the preferred number of keypoints, and let both SiLK [14] and DeDoDe [12] detect up to 30,000 keypoints, as we find that they benefit from it. We call this *unlimited*, as any number of keypoints can be used. We present results in Table 1. We show clear gains on all performed benchmarks. We additionally evaluate our proposed detector with the RoMa matcher using 8,000 keypoints with the original DeDoDe detector in Table 2. Again, we observe a clear boost in performance.

Table 1. **SotA comparison on the Megadepth-1500-Unlimited benchmark**. We follow DeDoDe [12] and evaluate each detector with its optimal number of keypoints. In the case of SiLK and DeDoDe, we cap the number to 30k. Measured in AUC (higher is better).

Method ↓ AUC →	@5°	@10°	@20°
SuperPoint [9] CVPRW'18	31.7	46.8	60.1
DISK [30] NeurIps'20	36.7	52.9	65.9
ALIKED [34] TIM'23	41.9	58.4	71.7
SiLK [14] ICCV'23	39.9	55.1	66.9
DeDoDe v1-L — v1-B [12] 3DV'24	49.4	65.5	77.7
DeDoDe **v2**-L — v1-B	**52.5**	**67.4**	**78.7**
DeDoDe C4-L — C4-B [7] CVPR'24	51	67	79
DeDoDe **v2**-L — C4-B	**52.6**	**67.9**	**79.5**
DeDoDe v1-L — v1-G [12] 3DV'24	52.8	69.7	82.0
DeDoDe **v2**-L — v1-G	**54.6**	**70.7**	**82.4**
DeDoDe v1-L — RoMa [13] CVPR'24	55.1	71.6	83.5
DeDoDe **v2**-L — RoMa	**57.6**	**73.3**	**84.4**

Table 2. **SotA comparison on the Megadepth-1500-8k benchmark**. We investigate the effect of reducing the number of keypoints when using the SotA RoMa matcher. Measured in AUC (higher is better).

Method ↓ AUC →	@5°	@10°	@20°
DeDoDe v1-L — RoMa	52.9	69.9	82.2
DeDoDe **v2**-L — RoMa	**54.9**	**71.4**	**83.1**

Image Matching Challenge 2022: The Image Matching Challenge 2022 [16] comprises challenging uncalibrated relative pose estimation pairs. Different from MegaDepth-1500, the test set is hidden and does not derive from MegaDepth, and may, therefore, better indicate generalization performance, especially for models such as DeDoDe that train on MegaDepth. We follow the setup in SiLK [14] and DeDoDe [12] and use 30,000 keypoints, and MAGSAC++ [2] with a threshold of 0.2 pixels. We use a fixed image size of 784×784. We follow the approach in RoMa [13] and report results on the hidden test set. We present results in Table 3. We improve by $+2.5$ mAA compared to the DeDoDe baseline.

4.2. Qualitative Examples

We provide qualitative examples of DeDoDe v2 keypoints in Figure 5 and matches in Figure 6. As can be seen in the figures, our proposed detector produced diverse but repeat-

Figure 5. **Qualitative comparison of DISK [30] (left), DeDoDe [12] (middle), DeDoDe v2 (right).** Best viewed in high resolution. DISK (left) produces diverse, but non-discriminative keypoints. DeDoDe, in contrast, produces discriminative kepoints, but tends to cluster. Our proposed DeDoDe v2 has the benefit of both approaches, yielding both diverse and discriminative keypoints.

Table 3. **SotA comparison on the IMC2022 benchmark.** Relative pose estimation results on the IMC2022 [16] hidden test set, measured in mAA (higher is better).

Method ↓	mAA →	@10
DISK [30]	Neurips'20	64.8
ALIKED [34]	IEEE-TIM'23	64.9
SiLK [14]	ICCV'23	68.5
DeDoDe v1-L — v1-B [12]	3DV'24	72.9
DeDoDe v2-L — v1-B		**74.7**
DeDoDe v1-L — v1-G [12]	3DV'24	75.8
DeDoDe **v2**-L — v1-G		**78.3**

able keypoints (Figure 5), that are matchable (Figure 6).

5. Conclusion

We have presented DeDoDe v2, an improved keypoint detector. We analyzed issues with the original detector and proposed several improvements to solve these. Our proposed detector sets a new state-of-the-art on the challenging IMC2022 and MegaDepth-1500 relative pose estimation benchmarks.

Limitations. We have empirically analyzed and improved the DeDoDe detector. However, we still lack a theoretical understanding of some of the underlying issues. In particular, formulating an objective that is not in tension with relative pose (*cf*. Figure 3) is of future interest.

Acknowledgements. We thank the reviewers for the constructive feedback. This work was supported by the Wallenberg Artificial Intelligence, Autonomous Systems and

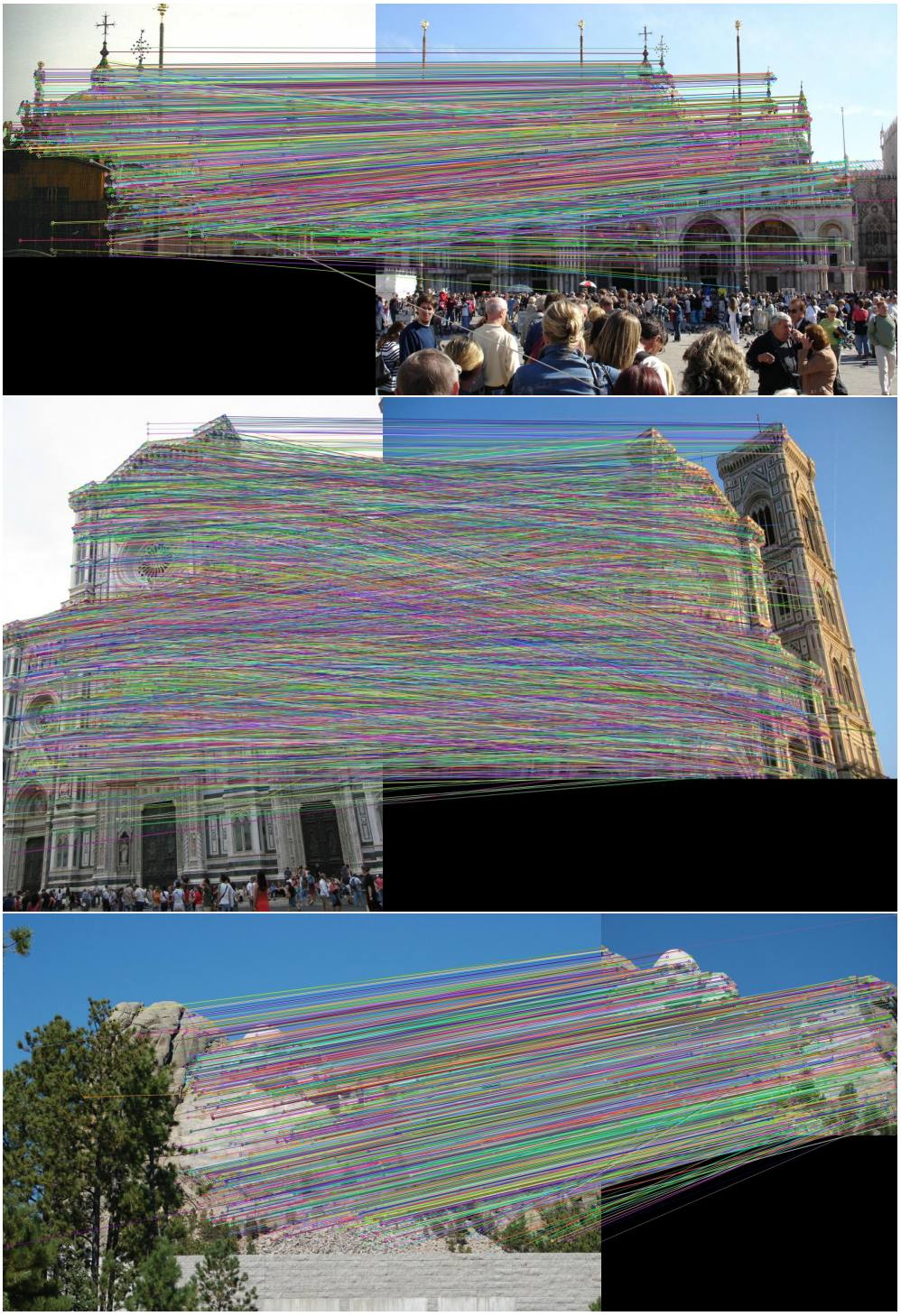

Figure 6. **Qualitative example of DeDoDe v2 matches.**

Software Program (WASP), funded by the Knut and Alice Wallenberg Foundation and by the strategic research environment ELLIIT funded by the Swedish government. The computational resources were provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) at C3SE, partially funded by the Swedish Research Council through grant agreement no. 2022-06725, and by the Berzelius resource, provided by the Knut and Alice Wallenberg Foundation at the National Supercomputer Centre.

References

[1] Relja Arandjelović and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2911–2918. IEEE, 2012. 2

[2] Daniel Barath, Jana Noskova, Maksym Ivashechkin, and Jiri Matas. Magsac++, a fast, reliable and accurate robust estimator. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1304–1312, 2020. 5

[3] Axel Barroso-Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Key.Net: Keypoint detection by handcrafted and learned cnn filters. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5836–5844, 2019. 2

[4] Fabio Bellavia and Dmytro Mishkin. Harrisz+: Harris corner selection for next-gen image matching pipelines. *Pattern Recognition Letters*, 158:141–147, 2022. 2

[5] Fabio Bellavia, D Tegolo, and Cf Valenti. Improving harris corner selection strategy. *IET Computer Vision*, 5(2):87–96, 2011. 2

[6] Georg Bökman and Fredrik Kahl. A case for using rotation invariant features in state of the art feature matchers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5110–5119, 2022. 2

[7] Georg Bökman, Johan Edstedt, Michael Felsberg, and Fredrik Kahl. Steerers: A framework for rotation equivariant keypoint descriptors. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2024. 3, 4, 5

[8] Hongkai Chen, Zixin Luo, Lei Zhou, Yurun Tian, Mingmin Zhen, Tian Fang, David Mckinnon, Yanghai Tsin, and Long Quan. ASpanFormer: Detector-free image matching with adaptive span transformer. In *Proc. European Conference on Computer Vision (ECCV)*, 2022. 2

[9] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018. 2, 5

[10] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-Net: A Trainable CNN for Joint Detection and Description of Local Features. In *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 2

[11] Johan Edstedt, Ioannis Athanasiadis, Mårten Wadenbäck, and Michael Felsberg. DKM: Dense kernelized feature matching for geometry estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2023. 2

[12] Johan Edstedt, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. DeDoDe: Detect, Don't Describe – Describe, Don't Detect for Local Feature Matching. In *2024 International Conference on 3D Vision (3DV)*. IEEE, 2024. 2, 3, 5, 6

[13] Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. RoMa: Robust dense feature matching. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2024. 2, 3, 5

[14] Pierre Gleize, Weiyao Wang, and Matt Feiszli. SiLK: Simple Learned Keypoints. In *ICCV*, 2023. 5, 6

[15] Chris Harris, Mike Stephens, et al. A combined corner and edge detector. In *Alvey vision conference*, pages 10–5244. Citeseer, 1988. 2

[16] Addison Howard, Eduard Trulls, Kwang Moo Yi, Dmitry Mishkin, Sohier Dane, and Yuhe Jin. Image matching challenge 2022, 2022. 5, 6

[17] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 2041–2050, 2018. 2

[18] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. LightGlue: Local Feature Matching at Light Speed. In *IEEE Int'l Conf. Computer Vision (ICCV)*, 2023. 2

[19] David G Lowe. Distinctive image features from scale-invariant keypoints. *Int'l J. Computer Vision (IJCV)*, 60:91–110, 2004. 2

[20] Dmytro Mishkin, Filip Radenović, and Jiři Matas. Repeatability is not enough: Learning affine regions via discriminability. In *European Conf. Computer Vision (ECCV)*, page 287–304, 2018. 2

[21] Jerome Revaud, Cesar De Souza, Martin Humenberger, and Philippe Weinzaepfel. R2d2: Reliable and repeatable detector and descriptor. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019. 2

[22] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4938–4947, 2020. 2

[23] Shi and Tomasi. Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE, 1994. 2

[24] Yan Shi, Jun-Xiong Cai, Yoli Shavit, Tai-Jiang Mu, Wensen Feng, and Kai Zhang. Clustergnn: Cluster-based coarse-to-fine graph neural network for efficient feature matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12517–12526, 2022. 2

[25] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8922–8931, 2021. 2, 5

[26] Shitao Tang, Jiahui Zhang, Siyu Zhu, and Ping Tan. Quadtree attention for vision transformers. In *International Conference on Learning Representations*, 2022. 2

[27] Prune Truong, Martin Danelljan, and Radu Timofte. GLU-Net: Global-local universal network for dense flow and correspondences. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6258–6268, 2020. 2

[28] Prune Truong, Martin Danelljan, Fisher Yu, and Luc Van Gool. Warp Consistency for Unsupervised Learning of Dense Correspondences. In *IEEE/CVF International Conference on Computer Vision, ICCV*, 2021.

[29] Prune Truong, Martin Danelljan, Radu Timofte, and Luc Van Gool. PDC-Net+: Enhanced Probabilistic Dense Correspondence Network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 2

[30] Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. Disk: Learning local features with policy gradient. *Advances in Neural Information Processing Systems (NeurIPS)*, 33: 14254–14265, 2020. 2, 5, 6

[31] Yannick Verdie, Kwang Yi, Pascal Fua, and Vincent Lepetit. Tilde: A temporally invariant learned detector. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5279–5288, 2015. 2

[32] Qing Wang, Jiaming Zhang, Kailun Yang, Kunyu Peng, and Rainer Stiefelhagen. MatchFormer: Interleaving attention in transformers for feature matching. In *Asian Conference on Computer Vision*, 2022. 2

[33] Xiaoming Zhao, Xingming Wu, Jinyu Miao, Weihai Chen, Peter CY Chen, and Zhengguo Li. Alike: Accurate and lightweight keypoint detection and descriptor extraction. *IEEE Transactions on Multimedia*, 2022. 2

[34] Xiaoming Zhao, Xingming Wu, Weihai Chen, Peter C. Y. Chen, Qingsong Xu, and Zhengguo Li. Aliked: A lighter keypoint and descriptor extraction network via deformable transformation. *IEEE Transactions on Instrumentation & Measurement*, 72:1–16, 2023. 2, 5, 6