LightGlue: Local Feature Matching at Light Speed

Supplementary Material

In the following pages, we present additional details on the experiments conducted in the main paper.

A. Image Matching Challenge

In this section, we present results obtained on the PhotoTourism dataset of the Image Matching Challenge 2020 (IMC) [26] in both stereo and multi-view tracks. The data is very similar to the MegaDepth [38] evaluation, exhibits similar statistics but different scenes. We follow the standardized matching pipeline of IMC with the setup and hyperparameters of SuperGlue [56]. We run the evaluation on the 3 validation scenes from the PhotoTourism dataset with LightGlue trained with two kinds of local features.

SuperPoint: For SuperPoint+SuperGlue and SuperPoint+LightGlue, we extract a maximum of 2048 keypoints and use DEGENSAC [11, 12, 43] with a threshold on the detection confidence of 1.1 in the stereo track (as suggested by SuperGlue). We do not perform any parameter tuning and reuse our model from the outdoor experiments with adaptive depth- and width, and use efficient self-attention [14] and mixed-precision during evaluation.

DISK: We also train LightGlue with DISK local features [73], a previous winner of the Image Matching Challenge. We follow the same training setup as for SuperPoint. For evaluation, we follow the guidelines from the authors for the restricted keypoint scenario (max 2048 features per image) and use mutual nearest neighbor matching with a ratio test of 0.95 as a baseline. We again use DEGENSAC for relative pose estimation with a threshold of 0.75.

Results: Table 7 reports the evaluation results. We also report the average matching speed over all 3 validation scenes. LightGlue is competitive with SuperGlue both in the stereo and multi-view track, while running 2.5× faster. Most of these run time improvements are due to the adaptive-depth, which largely reduces the run time for easy image pairs.

LightGlue trained with DISK [73] largely outperforms both the nearest-neighbor matching baseline with ratio test but also SuperPoint+LightGlue. On the smaller thresholds, DISK+LightGlue achieves +8%/+5% AUC in the stereo and multi-view tasks compared to our SuperPoint equivalent. With DISK, our model predicts 30% more matches than SP+LightGlue with an even higher epipolar precision. For DISK, the improvements are larger because DISK+NN+ratio only performs context aggregation within the image (from the U-Net), while LightGlue also aggregates information between images.

SfM features (2048 keypoints)	Task 1: Stereo AUC@K°		Task 2: Multiview AUC@5°@N			Pairs per second
	5°	10°	5	10	25	
SP+SuperGlue	58.64	71.07	61.88	78.97	86.75	16.2
SP+LightGlue	**59.03**	**71.13**	**62.87**	**79.36**	**86.98**	**43.4**
DISK+NN+ratio	57.76	68.73	59.91	78.95	87.54	**196.7**
DISK+LightGlue	**67.02**	**77.82**	**67.91**	**80.58**	**88.35**	44.5

Table 7. **Structure-from-Motion** with the Image Matching Challenge 2020. We evaluate the stereo track, at multiple error thresholds, and the multi-view track, for various numbers of images N. LightGlue yields better poses than SuperGlue on the multi-view track and significantly reduces the matching time. In combination with DISK, LightGlue improves over SuperPoint+SuperGlue and DISK+NN+ratio in both tracks by a large margin.

B. Additional results

Relative pose estimation:

Results reported in Section 5.2 were computed with a subset of the MegaDepth dataset [38] as introduced by previous works [9, 68, 78]. However, the images therein overlap with the training set of SuperGlue [56], the state-of-the-art sparse feature matcher and thus our main competitor.

For a more fair evaluation, we perform an extensive outdoor experiment on the test scenes of our MegaDepth [38] split, which covers 4 unique phototourism landmarks that SuperGlue was not trained with: Sagrada Familia, Lincoln Memorial Statue, London Castle, and the British Museum. To balance the difficulty of image pairs, we bin pairs into three categories based on their visual overlap score [19, 56], with intervals $[10, 30]\%$, $[30, 50]\%$, and $[50, 70]\%$. We sample 150 image pairs per bin per scene, totaling 1800 image pairs. We carefully rerun the experiment with the same setup that was used in Table 2. We report the precision as the ratio of matches with an epipolar error below 3px. With SIFT [41], we evaluate the ratio test and SGMNet [8] only, as the original SuperGlue model is not publicly available.

Table 8 confirms that LightGlue predicts more accurate correspondences than existing sparse matchers, at a fraction of the time. Detector-free feature matchers like LoFTR remain state-of-the-art on this task, although by a mere 2% AUC@5° with LO-RANSAC.

Outdoor visual localization: For completeness, we also report results on the Aachen v1.1 dataset [59] and compare our method to recent sparse and dense baselines. Table 9 shows that all methods perform similarly on this dataset, which is largely saturated, with insignificant variations in the results. LightGlue is however far faster than all approaches.

Indoor visual localization on InLoc: We report results for InLoc in Table 10. We use hloc and run SuperGlue again for fairness. For LoFTR and ASpanFormer, report existing results as no code is available. LightGlue is competitive

	features + matcher	#matches	P	pose estimation AUC @5°	@10°	@20°	time (ms)
dense	LoFTR	2231	89.8	66.4	79.1	87.6	181
	MatchFormer	2416	91.2	65.2	78.1	87.4	388
	ASPanFormer	4299	94.7	68.0	80.4	88.7	239
SIFT	NN+ratio	160	82.3	48.3	62.2	73.2	5.7
	SGMNet	405	82.5	50.7	66.6	76.5	71.7
	LightGlue	383	84.1	57.0	71.3	81.8	44.3
SuperPoint	NN+mutual	697	49.4	37.7	50.9	62.3	5.6
	SuperGlue	712	93.0	64.8	77.5	86.6	70.0
	SGMNet	**725**	89.8	61.7	74.3	83.4	74.0
	LightGlue	709	**94.5**	**65.5**	**77.8**	**86.9**	44.2

Table 8. **Relative pose estimation on Megadepth-1800.** This split is different from Table 2. In contrast to the split used by previous works [38, 68], this set of test images avoids training overlap with SuperGlue [56]. LightGlue predicts a similar amount of correspondences but with higher precision (P), pose accuracy (AUC), and speed than existing sparse matchers. It is competitive with dense matchers for a fraction of the inference time.

features + matcher	Day (0.25m,2°) / (0.5m,5°) / (1.0m,10°)	Night (0.25m,2°) / (0.5m,5°) / (1.0m,10°)	pairs per second
LoFTR	88.7 / 95.6 / 99.0	**78.5** / 90.6 / 99.0	-
ASpanFormer	89.4 / 95.6 / 99.0	77.5 / **91.6** / 99.5	-
SP+SuperGlue	89.8 / **96.1** / **99.4**	77.0 / 90.6 / **100**	6.4
SP+**LightGlue**	**90.2** / 96.0 / **99.4**	77.0 / 91.1 / **100**	**17.3**

Table 9. **Outdoor visual localization on Aachen v1.1.** LightGlue achieves similar accuracy with higher throughput.

features + matcher	DUC1 (0.25m,10°) / (0.5m,10°) / (1.0m,10°)	DUC2 (0.25m,10°) / (0.5m,10°) / (1.0m,10°)
LoFTR	47.5 / 72.2 / 84.8	54.2 / 74.8 / **85.5**
MatchFormer	46.5 / 73.2 / 85.9	**55.7** / 71.8 / 81.7
ASpanFormer	**51.5** / **73.7** / **86.4**	55.0 / 74.0 / 81.7
SP+SuperGlue	47.0 / 69.2 / 79.8	53.4 / **77.1** / 80.9
SP+**LightGlue**	49.0 / 68.2 / 79.3	55.0 / 74.8 / 79.4

Table 10. **Indoor visual localization on InLoc.** LightGlue performs similarly to SuperGlue (within the variability of the dataset).

with SuperGlue and more accurate at (0.25m,10°). Differences of <2% are insignificant because each split only has 205/151 queries (1.5% of difference ≡ 3 queries). Failures of LightGlue over SuperGlue (6/356 images @1m) are due to more matches on repeated objects (like trash cans), *i.e.* to better matching and weak retrieval – we show an example in Figure 8.

Large-scale visual localization on LaMAR: We perform another experiment on large-scale visual localization on the LaMAR dataset [57]. The benchmark evaluates single-image localization with queries from two devices (HoloLens2, Phone) on 3 scenes (indoor+outdoor), with strong illumination and viewpoint changes. We use hloc [55] and compare our method against SuperGlue [56]. For mapping, we use

Figure 8. **Failure cases on InLoc [70].** LightGlue sometimes matches repeated objects in the scene with strong texture, instead of the geometric structure.

SuperPoint + matcher	# pairs	HoloLens2 (0.1m, 1.0°)	Phone (1.0m, 5.0°)	seconds per query
SuperGlue	10	67.13 / 81.31	58.53 / 75.68	0.44
LightGlue	10	67.17 / 80.65	58.12 / 75.32	**0.13**
LightGlue	30	**70.53** / **83.38**	**63.93** / **79.32**	0.39

Table 11. **Large-scale visual localization on LaMAR.** We evaluate on the validation set of LaMAR [57], and report the pose recall under two thresholds and with two capture devices. Both on HoloLens2 and Phone, LightGlue achieves similar accuracy to SuperGlue, but much faster. By equalizing localization time through more retrieval pairs, LightGlue achieves state-of-the-art on this benchmark.

top-10 image retrieval for all methods. To illustrate the importance of matching speed in visual localization, we perform an additional experiment with LightGlue where we equalize the localization time by increasing the number of matched pairs between query and database images. The results on the validation set of LaMAR are reported in Table 10. LightGlue and SuperGlue achieve similar localization accuracy with the same setup, but LightGlue is almost 4x faster. By increasing the retrieval pairs from 10 to 30 per query image, LightGlue outperforms SuperGlue in both devices and under all thresholds.

C. Implementation details

C.1. Architecture

Positional Encoding. 2D image coordinates are normalized to a range [-1, 1] while retaining the image aspect ratio. We then project 2D coordinates into frequencies with a linear projection $\mathbf{W}_p \in \mathbb{R}^{2d/2h}$, where h is the number of attention heads. We cache the result for all layers. We follow the

efficient scheme of Roformer [67] to apply the rotations to query and key embeddings during self-attention, avoiding quadratic complexity to compute relative positional bias. We do not apply any positional encoding during cross-attention, but let the network learn spatial patterns by aggregating context within each image.

The proposed rotary encoding allows the network to attend to different frequencies, enabling the network to learn complex spatial patterns even under non-equivariant transformations. We believe that learning a positional encoding based on epipolar or 3D geometry in the cross-attention step could further improve the robustness of the method. This could be an exciting research direction in future work.

Graph Neural Network: The graph neural network consists of 9 transformer layers with both a self- and cross-attention unit. The update MLP (Eq. 1) has a single hidden layer of dimension $d_h = 2d$ followed by LayerNorm, GeLU activation and a linear projection $(2d, d)$ with bias.

Each attention unit has three projection matrices for query, key and value, plus an additional linear projection that merges the multi-head output. In bidirectional cross attention, the projections for query and key are shared. In practice we use an efficient self-attention [14] which optimizes IO complexity of the attention aggregation. This could also be extended for bidirectional cross attention. While training we use gradient checkpointing to significantly reduce the required VRAM.

Correspondences: The linear layers (Eq. 6) map from d to d and are not shared across layers. For all experiments we use the mutual check and a filter threshold $\tau = 0.1$.

Confidence classifier: The classifier predicts the confidence with a linear layer followed by a sigmoid activation. Confidences are predicted for each keypoint and only at layers $1, .., L-1$, since, by definition, the confidences of the final layer L are 1. Each prediction is supervised with a binary cross-entropy loss and its gradients are not propagated into the states to avoid impacting the matching accuracy. The state already encodes sufficient information since it is also supervised for matchability prediction.

Exit criterion and point pruning: During training we observed that the confidence predictions are less accurate in earlier layers. We therefore exponentially decay the confidence threshold:

$$\lambda_l = 0.8 + 0.1e^{-4\ell/L} \quad . \tag{12}$$

A state is deemed confident if $c_i^\ell > \lambda_\ell$. During inference, we halt the network if α=95% of states are deemed confident.

For point pruning, a point is deemed unmatchable when its predicted confidence is high and its matchability is low:

$$\text{unmatchable}(i) = c_i^l > \lambda_\ell \;\; \& \;\; \sigma_i^\ell < \beta \tag{13}$$

Method	#matches	P	pose estimation AUC @5°	@10°	@20°	time (%)
SP+LightGlue	**613**	**96.2**	**66.7**	**79.3**	**87.9**	100.0
↳ layer 7/9	705	96.0	66.2	79.1	88.0	82.4
↳ layer 5/9	702	94.5	65.0	77.8	87.0	60.0
↳ layer 3/9	687	90.0	64.0	76.7	85.8	**41.9**
↳ confidence 98%	**610**	**96.2**	66.6	79.3	88.0	80.5
↳ **confidence 95%**	608	95.4	66.3	79.0	87.9	70.6
↳ confidence 90%	607	94.5	65.9	78.5	87.2	61.5
↳ confidence 80%	605	92.6	65.2	77.8	86.7	48.4

Table 12. **Evaluation of early-stopping on MegaDepth.** Matches predicted by deeper layers are more accurate but require more computations with a higher inference time. Modeling confidences adaptively selects the model depth that yields a sufficient accuracy. A more conservative stopping, with a higher threshold α, yields a higher accuracy at the cost of higher inference time. α=95% yields the best trade-off.

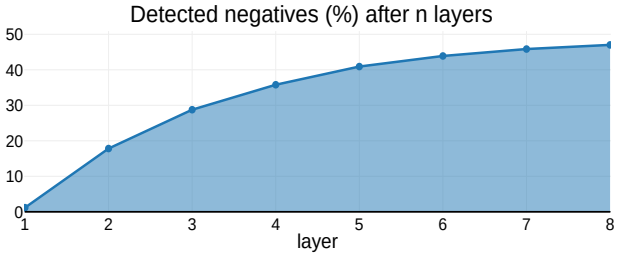

Figure 9. **Continuous detection of unmatchable points.** After just a few layers the network detects many points which are unmatchable, and we exclude them from context aggregation.

We report an ablation on the exit confidence α in Table 12 for relative pose estimation on MegaDepth. Lowering α to 80% reduces the inference time by almost 50% compared to our full model, while maintaining competitive accuracy compared to SuperGlue on this task. Reducing the confidence threshold is far more effective in terms of run time - accuracy tradeoff than trimming the model to fewer layers. Stopping the network early mainly sacrifices precision. For our experiments we chose 95% confidence, which yields on average 25% run time reduction with hardly any loss of accuracy on downstream tasks.

Here, $\beta = 0.01$ is a threshold on how matchable a point is. If Eq. 13 holds, we exclude the point from context aggregation in the following layers. This adds an overhead of gather and scatter per layer, but pruning becomes increasingly effective with more keypoints.

In Figure 9 we report the fraction of keypoints excluded in each layer. After just a few layers of context aggregation, LightGlue is confident to exclude $> 30\%$ of keypoints early on. Since the number of keypoints have a quadratic impact on run time, as shown in Fig. 7, this can largely reduce the number of computations in a forward pass and thus significantly reduce inference time.

C.2. Local features

We train LightGlue with three popular local feature detectors and descriptors: SuperPoint [16], SIFT [41] and DISK [73]. During training and evaluation, we discard the detection threshold for all methods and use the top-k keypoints according to the detection score. During training, if there are less than k detections available, we append random detections and descriptors. For SIFT [41] and DISK [73], we add a linear layer to project descriptors to d=256 before feeding them to the Transformer backbone.

SuperPoint: SuperPoint is a popular feature detector which produces highly repeatable points located at distinctive regions. We use the official, open-sourced version of SuperPoint from MagicLeap [16]. The detections are pixel-accurate, i.e. the keypoint localization accuracy depends on the image resolution.

SIFT: We use the excellent implementation of SIFT from vlfeat [75] when training on MegaDepth, and SIFTGPU from COLMAP [60] for fast feature extraction when pre-training on homographies. We observed that these implementations are largely equivalent during training and can be exchanged freely. Also, SIFT features from OpenCV can be used without retraining. Orientation and scale are not used in positional encoding.

DISK: DISK learns detection and description with a reinforcement learning objective. Its descriptors are more powerful than SIFT and SuperPoint and its detections are more repeatable, especially under large viewpoint and illumination changes.

C.3. Homography pre-training

Following Sarlin *et al*. [56], we first pre-train LightGlue on synthetic homographies of real-images.

Dataset: We use 170k images from the Oxford-Paris 1M distractors dataset [50], and split them into 150k/10k/10k images for training/validation/test.

Homography sampling: We generate homographies by randomly sampling four image corners. We split the image into four quarters, and sample a random point in each quarter. To avoid degenerates, we enforce that the enclosed area is convex. After, we apply random rotations and translations to the corners s.t. the corners remain inside the image. With this process, we can generate extreme perspective changes while avoiding border artifacts. This process is repeated twice, resulting in two largely skewed homographies. In interpolation, we then enforce the extracted images to be of size 640x480.

Photometric augmentation: The color images are then forwarded through a sequence of strong photometric augmentations, including blur, hue, saturation, sharpness, illumination, gamma and noise. Furthermore, we add random additive shades into the image to simulate occlusions and non-uniform illumination changes.

Figure 10. **Examples of synthetic homographies.** We show the original images (left) and two augmented examples (center and right) resulting from strong perspective transformations and extreme photometric augmentations.

Supervision: Correspondences with 3px symmetric reprojection error are deemed inliers, and points without any correspondence under this threshold are outliers.

Training details: We extract 512/1024/1024 keypoints for SuperPoint/SIFT/DISK, and a batch size of 64. The initial learning rate is 0.0001, and we multiply the learning rate by 0.8 each epoch after 20 epochs. We stop the training after 40 epochs (6M image pairs), or 2 days with 2 Nvidia RTX 3090 (for SuperPoint). Our network achieves $> 99\%$ recall and $> 90\%$ precision on the validation and test set. We also observed that, for fine-tuning, one can stop the pre-training after just one day with only minor losses.

We also experimented with sampling images from MegaDepth [38] for homography pre-training, and could not observe major differences. Strong photometric augmentations and perspective changes are crucial for training a robust model.

C.4. Finetuning on MegaDepth

We fine-tune our model on phototourism images with pseudo ground-truth camera poses and depth images.

Dataset: We use the MegaDepth dataset [38], which contains dense reconstructions of a large variety of popular landmarks all around the globe, obtained through COLMAP+MVS [60, 61]. Following Sun *et al*. [68], we bin each pair by its covisibility score [19], into ranges $[0.1, 0.3]$, $[0.3, 0.5]$ and $[0.5, 0.7]$. Scenes which are part of the validation and test set in the image matching challenge [26] are also excluded from training, resulting in 368/5/24 scenes for training/validation/test. At the beginning of each epoch, we sample 100 image pairs per scene.

Images are resized s.t. their larger edge is of size 1024, and zero-pad images to 1024×1024 resolution.

Supervision: Following SuperGlue [56], we reproject points using camera poses and depth to the other image. Correspondences with a maximum reprojection error of 3 pixels and which are mutually closest are labelled as inliers. A point

where the closest correspondence has a reprojection error larger than 5px are is labelled as outlier. Furthermore, we also declare points without depth and no correspondence with a Sampson Error smaller than 3 px outliers.

Training details: Weights are initialized from the pretrained model on homographies, Training starts with a learning rate of 1e-4 for 20 epochs and we exponentially decay it by a factor of 10 over 10 epochs, and stop training after 40 epochs (2 days on 2 RTX 3090). The top 2048 keypoints are extracted per image, and we use a batch size of 32. To speed-up training, we cache detections and descriptors per image, requiring around 200 GB of disk space.

C.5. Homography estimation

We validate the models capabilities on real homographies on the Hpatches dataset [2]. We follow the setup introduced in LoFTR [68] and resize images to a maximum edge length of 480.

For SuperPoint we extract the top 1024 keypoints with the highest detection score, and report precision (fraction of matches within 3px homography error) and recall (fraction of recovered mutual nearest-neighbour matches within 3px homography error). For LoFTR we only report epipolar precision. Furthermore, we evaluate the models in the downstream task of homography matrix estimation. Following SuperGlue [56], we report pose estimation results from robust estimation using RANSAC/MAGSAC [3] and the least squares solution with the weighted DLT algorithm. We evaluate the accuracy of estimated homography by their mean absolute corner distance towards the ground-truth homography.

We use OpenCV with USAC_MAGSAC for robust homography estimation, and tune the threshold for each method separately. Our reasoning behind this decision, which is in contrast to previous works in feature matching [56, 68] which fix the RANSAC parameters, is that we mainly use RANSAC as a tool to evaluate the low-level matches on a downstream task, and we want to minimize the variations introduced by its hyperparameters in order to obtain fair and representative evaluations. Different matches typically require different RANSAC thresholds, and thus a fixed threshold is suboptimal for comparison. For example on outdoor relative pose estimation, tuning the RANSAC threshold yields +7% AUC@5°on SuperGlue, skewing the reported numbers.

D. Timings

All experiments were conducted on a single RTX 3080 with 10GB VRAM. We report the timings of the matching process only, excluding sparse feature extraction (which is linear in the number of images) and robust pose estimation. We report the average over the respective datasets.

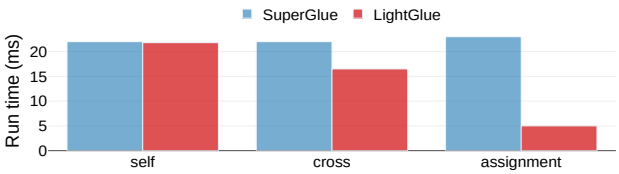

Figure 11. **Run time breakdown.** We evluate the runtime of self-, cross- and partial assignment layers on 1024 keypoints for SuperGlue and LightGlue. Most of LightGlue's default inference time improvements stem from a significantly faster partial assignment layer and reuse of computations in bidirectional cross-attention.

In Figure 11 we benchmark self-/cross-attention and solving the partial assignment problem against the respective counterparts in SuperGlue [56]. Bidirectional cross-attention reduces the run-time by 33% by only computing the similarity matrix once. However, the main bottleneck remains computing the softmax over both directions.

Our cheap double-softmax and the unary matchability predictions are significantly faster than solving it using optimal transport [66, 48], where 100 iterations are required during training to maintain stability.

In practice, we also use efficient self-attention [14] and mixed-precision to significantly reduce run time and memory requirements. However, for a fair comparison, we exclude these performance improvements from all experiments except where explicitly stated otherwise.

E. Qualitative Results

Figure 12 shows how LightGlue discards unmatched points and its early stopping mechanism on easy/medium/hard pairs. Figure 13 illustrates the matching output for LightGlue with SIFT [41], SuperPoint [16] and DISK [73] on some qualitative examples.

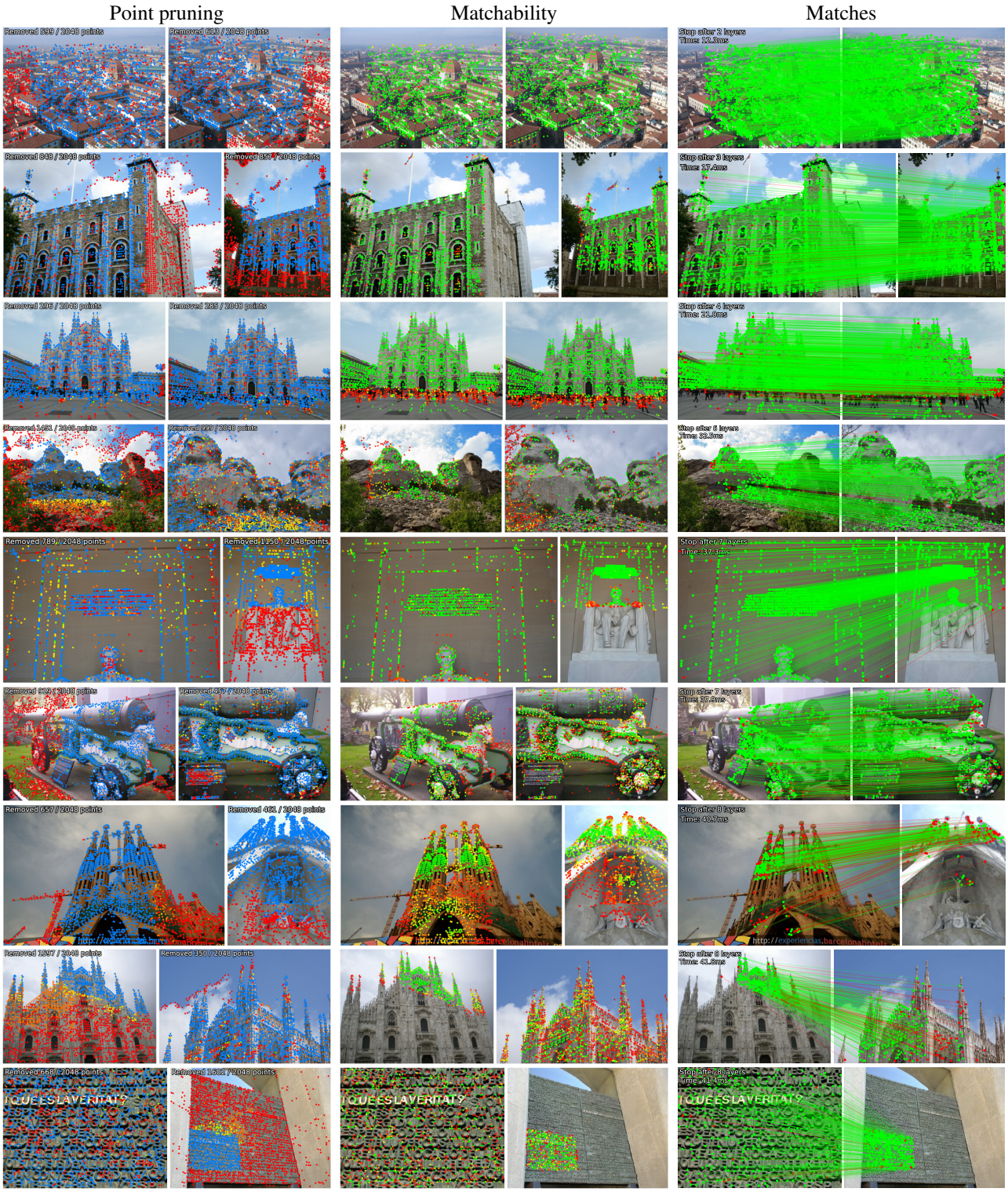

Figure 12. **Visualization of adaptive depth and width.** From top to bottom, we show three easy, medium and difficult image pairs. The left column shows how LightGlue reduces its width: it finds out early that some points (●) are unmatchable (mostly by visual overlap) and discards non-repeatable points in later layers: ● → ● → ●. This is very effective on difficult pairs. LightGlue looks for matches only in the reduced search space (●). The matchability scores (middle column, from non-matchable ● to likely matchable ●), help find accurate correspondences and are almost binary. On the right we visualize predicted matches as epipolar in- or outliers. We report the run time and stopping layer for each pair. On easy samples, LightGlue stops after only 2-3 layers, running with close to 100 FPS.

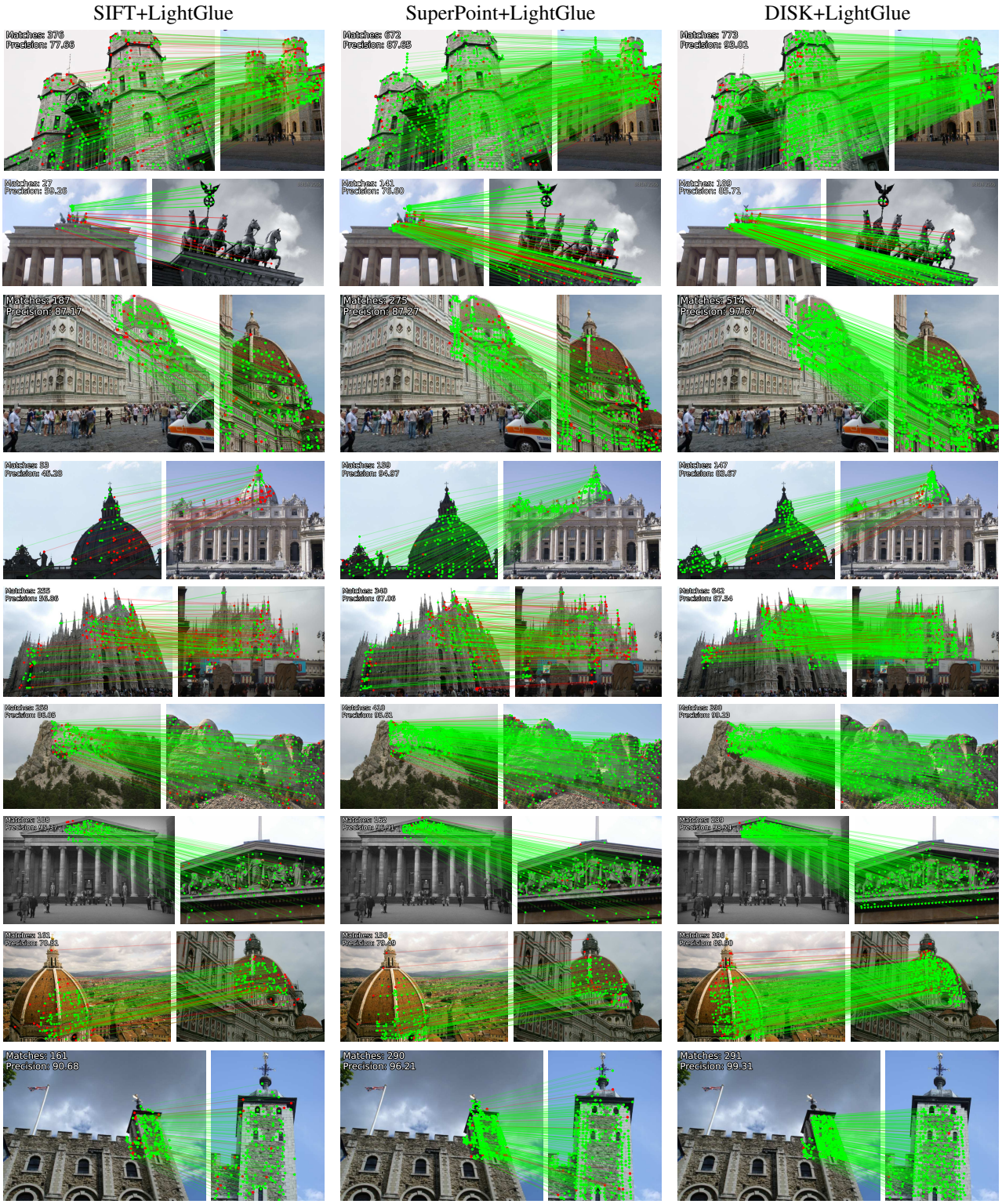

Figure 13. **Comparison of features produced by LightGlue for different local features.** We compare the outputs of SIFT+LightGlue (left), SuperPoint+LightGlue (middle) and DISK+LightGlue (right).

References

[1] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016. 7, 8

[2] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*, 2017. 6, 5

[3] Daniel Barath, Jiri Matas, and Jana Noskova. Magsac: marginalizing sample consensus. In *CVPR*, 2019. 5

[4] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. In *ECCV*, 2006. 2

[5] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *TRO*, 32(6):1309–1332, 2016. 2

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. In *ICCV*, 2021. 1

[7] Luca Cavalli, Viktor Larsson, Martin Ralf Oswald, Torsten Sattler, and Marc Pollefeys. Handcrafted outlier detection revisited. In *ECCV*, 2020. 2

[8] Hongkai Chen, Zixin Luo, Jiahui Zhang, Lei Zhou, Xuyang Bai, Zeyu Hu, Chiew-Lan Tai, and Long Quan. Learning to match features with seeded graph matching network. *ICCV*, 2021. 1, 2, 5, 6, 7, 8, 9

[9] 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 *ECCV*, 2022. 2, 6, 7, 1

[10] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training Deep Nets with Sublinear Memory Cost. *arXiv:1604.06174*, 2016. 6

[11] Ondřej Chum, Jiří Matas, and Josef Kittler. Locally optimized RANSAC. In *Joint Pattern Recognition Symposium*, pages 236–243. Springer, 2003. 7, 1

[12] Ondrej Chum, Tomas Werner, and Jiri Matas. Two-view geometry estimation unaffected by a dominant plane. In *CVPR*, 2005. 7, 1

[13] Aaron Daniel Cohen, Adam Roberts, Alejandra Molina, Alena Butryna, Alicia Jin, Apoorv Kulshreshtha, Ben Hutchinson, Ben Zevenbergen, Blaise Hilary Aguera-Arcas, Chung ching Chang, Claire Cui, Cosmo Du, Daniel De Freitas Adiwardana, Dehao Chen, Dmitry (Dima) Lepikhin, Ed H. Chi, Erin Hoffman-John, Heng-Tze Cheng, Hongrae Lee, Igor Krivokon, James Qin, Jamie Hall, Joe Fenton, Johnny Soraker, Kathy Meier-Hellstern, Kristen Olson, Lora Mois Aroyo, Maarten Paul Bosma, Marc Joseph Pickett, Marcelo Amorim Menegali, Marian Croak, Mark Díaz, Matthew Lamm, Maxim Krikun, Meredith Ringel Morris, Noam Shazeer, Quoc V. Le, Rachel Bernstein, Ravi Rajakumar, Ray Kurzweil, Romal Thoppilan, Steven Zheng, Taylor Bos, Toju Duke, Tulsee Doshi, Vincent Y. Zhao, Vinodkumar Prabhakaran, Will Rusch, YaGuang Li, Yanping Huang, Yanqi Zhou, Yuanzhong Xu, and Zhifeng Chen. LaMDA: Language Models for Dialog Applications. *arXiv:2201.08239*, 2022. 1

[14] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *NeurIPS*, 2022. 2, 6, 8, 1, 3, 5

[15] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal Transformers. In *ICLR*, 2019. 2

[16] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-supervised interest point detection and description. In *CVPR Workshop on Deep Learning for Visual SLAM*, 2018. 2, 6, 7, 4, 5

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019. 1, 3

[18] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*, 2021. 1

[19] 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 *CVPR*, 2019. 2, 1, 4

[20] Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. Depth-Adaptive Transformer. In *ICLR*, 2020. 2, 4

[21] Michael Figurnov, Maxwell D Collins, Yukun Zhu, Li Zhang, Jonathan Huang, Dmitry Vetrov, and Ruslan Salakhutdinov. Spatially Adaptive Computation Time for Residual Networks. In *CVPR*, 2017. 3

[22] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2, 6

[23] Christopher G Harris, Mike Stephens, et al. A combined corner and edge detector. In *Alvey vision conference*, 1988. 2

[24] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 6

[25] Jared Heinly, Johannes L Schonberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the World* in Six Days *(as Captured by the Yahoo 100 Million Image Dataset). In *CVPR*, 2015. 2

[26] CVPR 2020 Image Matching Challenge. https://www.cs.ubc.ca/research/image-matching-challenge/2020/. Accessed June 15, 2023. 1, 4

[27] CVPR 2021 Image Matching Challenge. https://www.cs.ubc.ca/research/image-matching-challenge/. Accessed June 15, 2023. 7

[28] CVPR 2023 Image Matching Challenge. https://www.kaggle.com/competitions/image-matching-challenge-2023/overview. Accessed June 15, 2023. 7

[29] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J Henaff, Matthew Botvinick, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. Perceiver IO: A general architecture for structured inputs & outputs. In *ICLR*, 2022. 1

[30] Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver: General Perception with Iterative Attention. In *ICML*, 2021. 2

[31] Yuhe Jin, Dmytro Mishkin, Anastasiia Mishchuk, Jiri Matas, Pascal Fua, Kwang Moo Yi, and Eduard Trulls. Image Matching across Wide Baselines: From Paper to Practice. *IJCV*, 2020. 6

[32] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020. 2

[33] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The Efficient Transformer. In *ICLR*, 2020. 2

[34] Viktor Larsson. PoseLib - Minimal Solvers for Camera Pose Estimation, 2020. 6, 7

[35] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks. In *ICML*, 2019. 2

[36] Xiaoxiao Li, Ziwei Liu, Ping Luo, Chen Change Loy, and Xiaoou Tang. Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade. In *CVPR*, 2017. 3

[37] Yang Li, Si Si, Gang Li, Cho-Jui Hsieh, and Samy Bengio. Learnable Fourier Features for Multi-dimensional Spatial Positional Encoding. In *NeurIPS*, 2021. 3

[38] Zhengqi Li and Noah Snavely. MegaDepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018. 5, 6, 1, 2, 4

[39] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-Perfect Structure-from-Motion with Featuremetric Refinement. In *ICCV*, 2021. 2

[40] Zhuang Liu, Zhiqiu Xu, Hung-Ju Wang, Trevor Darrell, and Evan Shelhamer. Anytime Dense Prediction with Confidence Adaptivity. In *ICLR*, 2022. 3, 4

[41] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 2, 6, 1, 4, 5

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

[43] Dmytro Mishkin, Jiri Matas, and Michal Perdoch. Mods: Fast and robust method for two-view matching. *Computer Vision and Image Understanding*, 2015. 7, 1

[44] Kwang Moo Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. Learning to find good correspondences. In *CVPR*, 2018. 2, 8

[45] Raúl Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *TRO*, 31(5):1147–1163, 2015. 2

[46] Rémi Pautrat, Viktor Larsson, Martin R. Oswald, and Marc Pollefeys. Online invariance selection for local feature descriptors. In *ECCV*, 2020. 2

[47] Malte Pedersen, Joakim Bruslund Haurum, Thomas B Moeslund, and Marianne Nyegaard. Re-identification of giant sunfish using keypoint matching. In *Northern Lights Deep Learning Workshop*, 2022. 1

[48] Gabriel Peyré and Marco Cuturi. Computational optimal transport. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019. 2, 5

[49] Markus N. Rabe and Charles Staats. Self-attention Does Not Need $O(n^2)$ Memory. *arXiv:2112.05682*, 2021. 2

[50] Filip Radenović, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Revisiting Oxford and Paris: Large-scale image retrieval benchmarking. In *CVPR*, 2018. 5, 4

[51] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018. 1, 3

[52] Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2D2: Repeatable and reliable detector and descriptor. In *NeurIPS*, 2019. 2

[53] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *ECCV*, 2006. 2

[54] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R Bradski. ORB: An efficient alternative to SIFT or SURF. In *ICCV*, 2011. 2

[55] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *CVPR*, 2019. 1, 7, 8, 2

[56] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 1, 2, 6, 7, 9, 4, 5

[57] Paul-Edouard Sarlin, Mihai Dusmanu, Johannes L. Schönberger, Pablo Speciale, Lukas Gruber, Viktor Larsson, Ondrej Miksik, and Marc Pollefeys. LaMAR: Benchmarking Localization and Mapping for Augmented Reality. In *ECCV*, 2022. 1, 2

[58] Paul-Edouard Sarlin, Ajaykumar Unagar, Måns Larsson, Hugo Germain, Carl Toft, Viktor Larsson, Marc Pollefeys, Vincent Lepetit, Lars Hammarstrand, Fredrik Kahl, and Torsten Sattler. Back to the Feature: Learning robust camera localization from pixels to pose. In *CVPR*, 2021. 1

[59] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF outdoor visual localization in changing conditions. In *CVPR*, 2018. 1, 8

[60] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 2, 7, 8, 4

[61] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 4

[62] Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Q. Tran, Yi Tay, and Donald Metzler. Confident Adaptive Language Modeling. In *NeurIPS*, 2022. 2, 4

[63] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-Attention with Relative Position Representations. In *NAACL-HTL*, 2018. 3

[64] Tianwei Shen, Zixin Luo, Lei Zhou, Runze Zhang, Siyu Zhu, Tian Fang, and Long Quan. Matchable image retrieval by learning from surface reconstruction. In *ACCV*, 2018. 6

[65] 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 *CVPR*, 2022. 1, 2, 5, 8

[66] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 1967. 5

[67] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *arXiv:2104.09864*, 2021. 3, 4

[68] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with Transformers. *CVPR*, 2021. 2, 6, 7, 1, 4, 5

[69] Jiaming Sun, Zihao Wang, Siyu Zhang, Xingyi He, Hongcheng Zhao, Guofeng Zhang, and Xiaowei Zhou. OnePose: One-shot object pose estimation without CAD models. In *CVPR*, 2022. 1

[70] Hajime Taira, Masatoshi Okutomi, Torsten Sattler, Mircea Cimpoi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, and Akihiko Torii. InLoc: Indoor Visual Localization with Dense Matching and View Synthesis. *TPAMI*, 2019. 2

[71] Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. BranchyNet: Fast inference via early exiting from deep neural networks. *ICPR*, 2016. 3, 4

[72] Yurun Tian, Xin Yu, Bin Fan, Fuchao Wu, Huub Heijnen, and Vassileios Balntas. SOSNet: Second Order Similarity Regularization for Local Descriptor Learning. In *CVPR*, 2019. 2

[73] Michał J Tyszkiewicz, Pascal Fua, and Eduard Trulls. DISK: Learning local features with policy gradient. In *NeurIPS*, 2020. 2, 7, 1, 4, 5

[74] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017. 1, 2, 3

[75] Andrea Vedaldi and Brian Fulkerson. VLFeat: An open and portable library of computer vision algorithms. In *ACM international conference on Multimedia*, 2010. 4

[76] Thomas Verelst and Tinne Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In *CVPR*, 2020. 3

[77] Phil Wang. Bidirectional cross attention. https://github.com/lucidrains/bidirectional-cross-attention. 4

[78] Qing Wang, Jiaming Zhang, Kailun Yang, Kunyu Peng, and Rainer Stiefelhagen. MatchFormer: Interleaving Attention in Transformers for Feature Matching. In *ACCV*, 2022. 2, 6, 7, 1

[79] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-Attention with Linear Complexity. *arXiv:2006.04768*, 2020. 2

[80] Yan Wang, Zihang Lai, Gao Huang, Brian H. Wang, Laurens van der Maaten, Mark E. Campbell, and Kilian Q. Weinberger. Anytime Stereo Image Depth Estimation on Mobile Devices. *ICRA*, 2018. 3, 4

[81] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. LIFT: Learned invariant feature transform. In *ECCV*, 2016. 2

[82] Jiahui Zhang, Dawei Sun, Zixin Luo, Anbang Yao, Lei Zhou, Tianwei Shen, Yurong Chen, Long Quan, and Hongen Liao. Learning two-view correspondences and geometry using order-aware network. In *ICCV*, 2019. 2, 8

[83] Lulin Zhang, Ewelina Rupnik, and Marc Pierrot-Deseilligny. Feature matching for multi-epoch historical aerial images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 182:176–189, 2021. 1