Supplementary Material *for* Adaptive Assignment for Geometry Aware Local Feature Matching

Anonymous CVPR submission

Paper ID 5298

1. Implementation Details

1.1. Architecture of Co-visible Area Segmentation Module

In order to obtain the co-visible area probability map, we borrowed the structure of DETR [3] and used a query to perform regression on the feature map. The specific network architecture is shown in the Fig 1. Firstly, the spatial attention map $F^i_{attn} \in \mathbb{R}^{1\times h\times w}$ by the dot product operation of $Q^i \in \mathbb{R}^{1\times 1\times C}$ and $F^{i_2}_{1/8} \in \mathbb{R}^{C\times h\times w}$, and then perform element-wise multiplication of F^i_{attn} and $F^{i_2}_{1/8}$, followed by a shortcut connection to obtain $F^i_{co} \in \mathbb{R}^{C\times h\times w}$. Finally, a simple block with two convolution layers are used to obtain the co-visible area probability map, where the first convolution is followed by a ReLU activation and the second convolution is followed by a Sigmoid function.

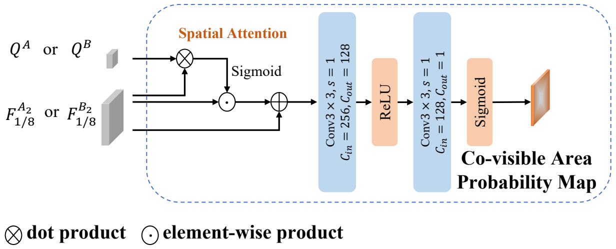
Q^A or Q^B
$F^{A_2}_{1/8}$ or $F^{B_2}_{1/8}$
Spatial Attention
Sigmoid
Conv3 × 3, $s = 1$
$C_{in} = 256, C_{out} = 128$
ReLU
Conv3 × 3, $s = 1$
$C_{in} = 128, C_{out} = 1$
Sigmoid
Co-visible Area Probability Map
⊗ dot product
⊙ element-wise product

Figure 1. **Architecture of co-visible area segmentation module.**

1.2. Refine Network for SuperGlue

As stated in the main text, our adaptive assignment and sub-pixel refine module can be treated as a refinement network for other matching methods, such as SuperGlue [12]. Here, we will present in detail how to use our method as a refinement network for SuperGlue. To perform adaptive assignment, we simply remove the mutual nearest neighbours constraint, and obtain proposal matches by applying $argmax$ operation on each of the two dimensions of the matching matrix generated from SuperGlue, and then filtering by adding a confidence threshold. The matching matrix can be the result of multiple iterations of the skinhorn algorithm, or the matrix obtained by decomposing the last iteration of the skinhorn algorithm. After obtaining the proposal matches, we sample the patch features $F^A, F^B \in \mathbb{R}^{n\times c\times w\times w}$ at the corresponding positions on the descriptor feature maps generated by SuperPoint [6] and feed them into our one-to-one refine module (Section 3.3 of the main text) to achieve scale alignment and sub-pixel location regression.

2. More Experiments

2.1. Additional Evaluation Metrics for HPatches

Metrics. Since the homography estimation accuracy contains the effect of the OpenCV-RANSAC, we use Mean Matching Accuracy (MMA) on the Hpatches [1] dataset to evaluate different methods. We use the ratio of correctly matched features within thresholds of 1, 2, and 3 pixels, respectively, and the maximum amount of matches is limited to 1024. **Results** in Tab.1 show that Adamatcher outperforms other methods in terms of matching accuracy. Since adaptive assignment eliminates the ambiguity of matching in supervision and inference, AdaMatcher is able to generate more accurate matches when the viewing angle changes.

Methods	**All**	**Viewpoint**
	MMA@1px / 2px / 3px	MMA@1px / 2px / 3px
SIFT [9]+HardNet [10]	0.460 / 0.718 / 0.828	0.383 / 0.671 / 0.800
KeyNet [2]+HardNet [10]	0.422 / 0.701 / 0.837	0.342 / 0.661 / 0.811
R2D2 [11]	0.334 / 0.611 / 0.751	0.273 / 0.566 / 0.699
SP [6]+SG [12]	0.367 / 0.685 / 0.827	0.315 / 0.654 / 0.815
SP [6]+SG [12]+Ada	**0.386 / 0.702 / 0.839**	**0.348 / 0.685 / 0.830**
LoFTR-OT [13]	0.593 / 0.814 / 0.893	0.461 / 0.731 / 0.836
LoFTR-DS [13]	0.613 / 0.830 / 0.902	0.495 / 0.759 / 0.853
AdaMatcher-LoFTR	0.628 / 0.845 / 0.914	0.540 / 0.798 / 0.882
QuadTree [14]	0.642 / 0.844 / 0.911	0.517 / 0.777 / 0.870
AdaMatcher-QuadTree	0.640 / 0.850 / 0.917	0.550 / 0.799 / 0.881
ASpanFormer [4]	**0.658** / 0.856 / 0.920	0.539 / 0.795 / 0.881
AdaMatcher-ASpan	0.657 / **0.857 / 0.920**	**0.552 / 0.802 / 0.887**

Table 1. MMA metrics on HPatches.

2.2. Results on YFCC100M

The YFCC100M [15] dataset is also used to conduct experiments to compare AdaMatcher with several baseline methods. To be fair, We use the same test pairs (a total of 4000 pairs) as in previous works [12, 16], using their evaluation metrics. The test set is derived from four selected landmark sequences, each sampling 1000 image pairs. All images are resized to 480 × 640 and all models are trained in MegaDepth [8]. The accuracy of pose estimation is measured by AUC under error thresholds (5°, 10° and 20°). The results are shown in Tab.2, where our methods all perform better than the corresponding baseline methods.

Methods	**Pose Estimation AUC**		
	@5°	@10°	@20°
LoFTR-DS [13]	43.06	62.21	77.26
AdaMatcher-LoFTR	**44.06**	**63.04**	**77.61**
ASpanFormer [4]	43.70	62.57	77.32
AdaMatcher-ASpan	**43.93**	**62.92**	**77.54**
QuadTree [14]	36.50	55.64	71.61
AdaMatcher-Quad	**44.20**	**63.09**	**77.59**

Table 2. The results of outdoor relative pose estimation on YFCC100M.

2.3. Indoor Pose Estimation

To validate the generalizability of different detector-free methods, we perform indoor pose estimation experiments on the ScanNet [5] dataset using models trained on the MegaDepth [8] dataset. We use the test split with 1500 image pairs following the experimental setting of [4, 12, 13]. To align with the existing methods [4, 13, 14], we resized all test images to 480×640. We use the same evaluation protocols as in Sec. 2.2. As presented in Tab.3, AdaMatcher has a significant performance improvement on different baselines [4, 13, 14].

Methods	**Pose Estimation AUC**		
	@5°	@10°	@20°
LoFTR-OT [13]	15.46	31.28	47.87
LoFTR-DS [13]	17.26	33.93	50.16
AdaMatcher-LoFTR	**18.60**	**35.00**	**50.75**
ASpanFormer [4]	20.64	39.34	56.61
AdaMatcher-ASpan	**21.33**	**39.93**	**56.69**
QuadTree [14]	19.83	37.86	55.03
AdaMatcher-Quad	**21.18**	**39.71**	**56.22**

Table 3. The results of indoor relative pose estimation on ScanNet. All models are trained on MegaDepth dataset.

2.4. Computational Costs of Feature Interaction

We evaluate the computation and parameters between LoFTR's feature interaction module [13] and our CFI module (using linear attention [7] as in LoFTR). The size of input tensor is $60 \times 80 \times 256$. As shown in Tab.4, compared to LoFTR's feature interaction module (consisting of four sets of self- and cross-attention layers), our CFI module reduces about 38.79% of the computational costs and 14.29% of the parameters.

Method	Flops(G)	Param(MB)
LoFTR module	51.82	5.25
CFI	31.74	4.50

Table 4. Computational complexity of feature interaction module

3. D. Qualitative Results

We present more qualitative comparisons of AdaMatcher and baselines on Hpatches [1] dataset and MegaDepth [8] dataset. In Fig.2, we display inlier and outlier matches using different projection thresholds to compare the matching accuracy of different methods on the Hpatches dataset. Fig.3 presents more qualitative results on the MegaDepth [8] dataset and Fig.4 shows more qualitative results of the co-visible area estimation.

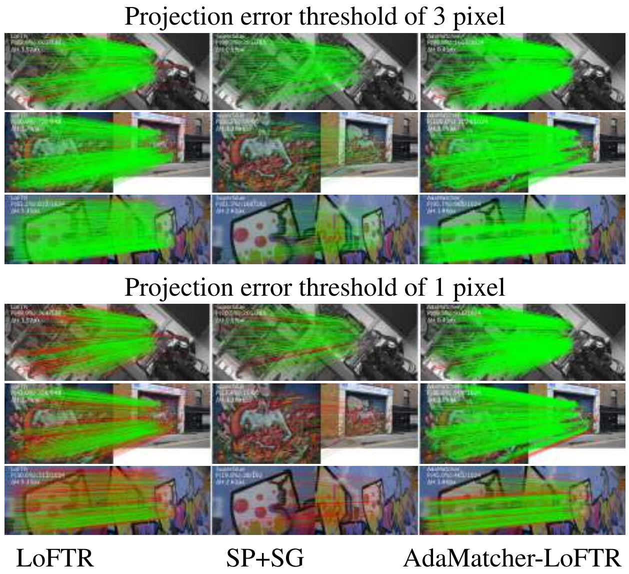
Projection error threshold of 3 pixel
Projection error threshold of 1 pixel
LoFTR
SP+SG
AdaMatcher-LoFTR

Figure 2. Qualitative image matches on Hpatches dataset. Matches with projection error less than the threshold are displayed in green, otherwise they are displayed in red.

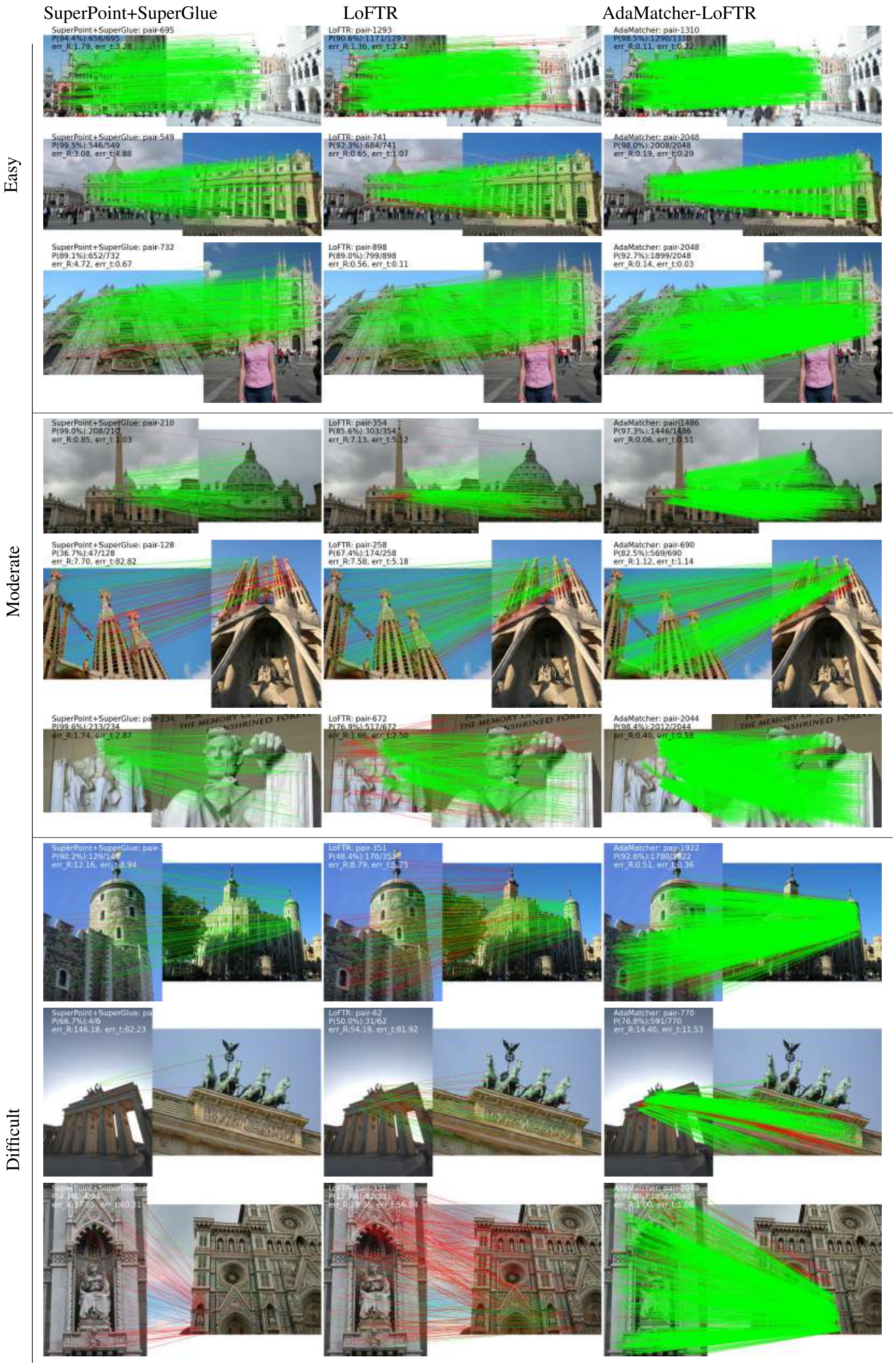

Figure 3. Qualitative image matches on MegaDepth dataset. Green indicates that epipolar error in normalized image coordinates is less than 1×10^{-4}, while red indicates that it is exceeded.

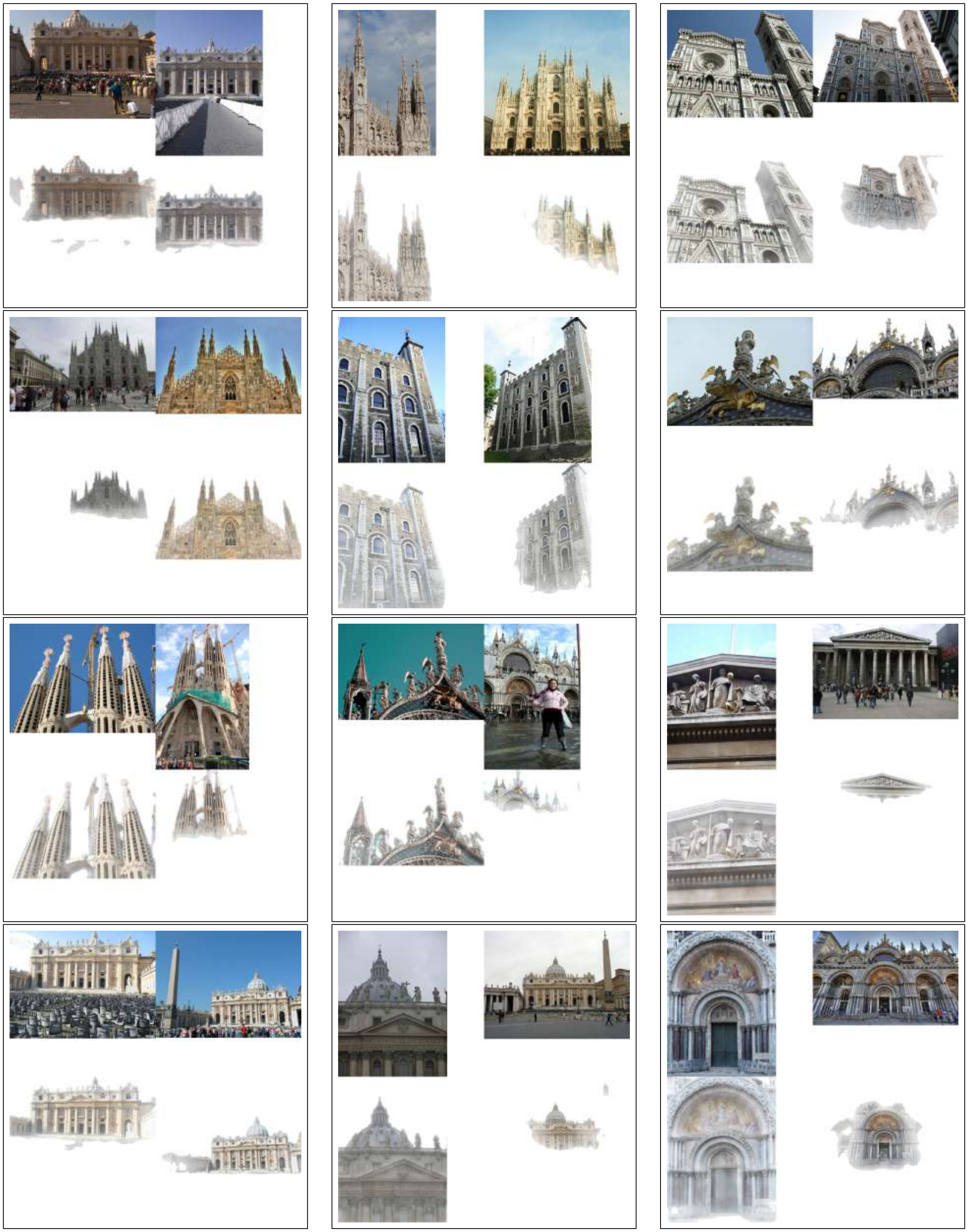

Figure 4. Qualitative co-visible area segmentation

References

[1] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*, pages 5173–5182, 2017. 1, 2

[2] Axel Barroso-Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Key. net: Keypoint detection by handcrafted and learned cnn filters. In *ICCV*, pages 5836–5844, 2019. 1

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020. 1

[4] 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*, pages 20–36, 2022. 1, 2

[5] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, pages 5828–5839, 2017. 2

[6] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPR*, pages 224–236, 2018. 1

[7] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*, pages 5156–5165, 2020. 2

[8] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *CVPR*, pages 2041–2050, 2018. 2

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

[10] Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. *NeurIPS*, 30, 2017. 1

[11] Jerome Revaud, Cesar De Souza, Martin Humenberger, and Philippe Weinzaepfel. R2d2: Reliable and repeatable detector and descriptor. *NeurIPS*, 32, 2019. 1

[12] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, pages 4938–4947, 2020. 1, 2

[13] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *CVPR*, pages 8922–8931, 2021. 1, 2

[14] Shitao Tang, Jiahui Zhang, Siyu Zhu, and Ping Tan. Quadtree attention for vision transformers. In *ICLR*, 2022. 1, 2

[15] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016. 2

[16] 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*, pages 5845–5854, 2019. 2