Improved Scene Landmark Detection for Camera Localization

Tien Do* Tesla

Sudipta N. Sinha Microsoft

Abstract

Camera localization methods based on retrieval, local feature matching, and 3D structure-based pose estimation are accurate but require high storage, are slow, and are not privacy-preserving. A method based on scene landmark detection (SLD) was recently proposed to address these limitations. It involves training a convolutional neural network (CNN) to detect a few predetermined, salient, scene-specific 3D points or landmarks and computing camera pose from the associated 2D–3D correspondences. Although SLD outperformed existing learning-based approaches, it was notably less accurate than 3D structure-based methods. In this paper, we show that the accuracy gap was due to insufficient model capacity and noisy labels during training. To mitigate the capacity issue, we propose to split the landmarks into subgroups and train a separate network for each subgroup. To generate better training labels, we propose using dense reconstructions to estimate visibility of scene landmarks. Finally, we present a compact architecture to improve memory efficiency. Accuracy wise, our approach is on par with state of the art structure-based methods on the INDOOR-6 *dataset but runs significantly faster and uses less storage. Code and models can be found at* https://github.com/microsoft/SceneLandmarkLocalization.

1. Introduction

In this paper, we study the task of estimating the 6-dof camera pose with respect to a reconstructed 3D model of a scene from a single image. This is an important task in robotics and augmented reality applications. The most common approach for solving the task is structure-based [26, 27, 29, 30], where typically, the local 2D image features are matched to 3D points in a scene model. Geometric constraints derived from the 2D–3D matches are then used to compute the camera pose. These methods can be quite accurate but the need to persistently store a lot of features and 3D points raises privacy issues [23] and also makes them less suitable for resource-constrained settings.

*work done while Tien Do was affiliated with Microsoft.

Learning-based localization methods [6, 7, 15, 21] can alleviate both the storage and privacy issues. However, despite much progress on learning-based localization, most of the methods are still not competitive with structure-based methods [26, 27]. Recently, Do *et al*. [11] proposed SLD, a localization framework that involves training CNNs for detecting pre-selected, scene landmarks (3D points) and regressing 3D bearing vectors (NBE) for the landmarks. The 2D detections and 3D bearing predictions are jointly used (SLD+NBE) to compute camera pose. Even though SLD+NBE outperforms learning-based methods [6, 15] on the challenging INDOOR-6 dataset, it is less accurate than hloc [26, 27] by a notable margin. It is also unclear to what extent the method can handle a large number of landmarks.

In this paper, we present important insights into what typically hurts SLD's accuracy and scalability. Our first finding is that insufficient model capacity is a key cause for a drop in performance when SLD is trained for a larger set of landmarks. We also find that the automatic structure from motion (SfM) processing phase which generates labeled training patches for landmarks from training images can produce erroneous training labels. Such outliers can sometimes affect the accuracy of models trained on the data.

To address the capacity issue, we propose to partition the set of scene landmarks into mutually exclusive subgroups, and train an ensemble of networks, where each network is trained on a different subgroup. Using an ensemble improves accuracy for scenes where a larger number of landmarks are present. To reduce the amount of erroneous labels in the training set, we propose using a dense scene reconstruction to recover more accurate visibility estimates of the scene landmarks in the training images, especially under strong lighting changes. We show that better training labels leads to more accurate landmark detections. We also propose SLD*, a variation of the SLD architecture that improves memory efficiency, and explore using output prediction scores as a confidence measure during pose estimation.

Incorporating all the proposed ideas leads to a dramatic improvement in pose estimation accuracy, making SLD* competitive with hloc on the INDOOR-6 dataset. At the same time, it is also *more than 40× faster* than hloc during localization and *20× more storage efficient*. Furthermore,

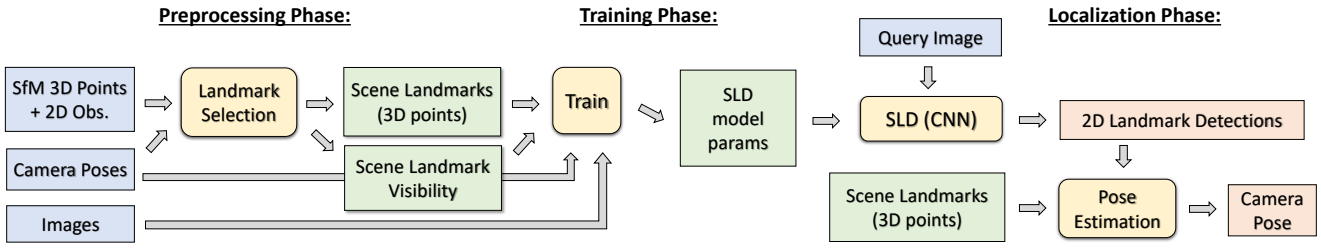


Figure 1. Key elements of the scene landmark detection-based localization approach [11]. The figure shows a single model (SLD) for brevity, but Do *et al*. [11] also proposed predicting landmark bearings using an additional model (NBE). This is discussed in the text.

SLD* is 20–30% more memory efficient than SLD.

2. Related Work

Structure-based Localization. Classical structure-based approaches use pre-computed 3D scene point clouds to compute camera pose by combining efficient visual retrieval [1, 22, 30, 38, 40], feature matching [10, 19, 25, 27, 29, 39], and geometric pose estimation[14]. hloc [26] is such a method with state-of-the-art performance on INDOOR-6 that uses learning for more accurate feature matching [10, 20, 25, 27]. While correspondences and pose is usually estimated independently, jointly refining deep multiscale features and camera pose has been shown to improve accuracy [28]. Alternatively, retrieval-based methods [1, 38, 39] can estimate the camera pose by interpolating poses of retrieved database images[40]. Efficient and scalable alternatives for large-scale location classification and place recognition have also been studied [3, 12, 44].

Learning-based Localization. Learning-based techniques do not require storing 3D scene models. A popular approach is to train models to regress the camera pose directly from the query image, which is called absolute pose regression (APR). PoseNet [15] first proposed end-to-end trainable CNN architectures, which have been extended for leveraging attention mechanisms [41] and to use transformer architectures [33]. However, APR methods rely on training sets with homogeneous camera pose distributions. When the pose distribution is highly heterogeneous, performance can suffer on such datasets [31], as was reported on INDOOR-6 in [11]. Unlike APR approaches, relative pose regression (RPR) approaches predict the relative pose with respect to stored database images [2, 16]. They usually generalize better but have higher storage costs.

Scene Coordinate Regression. In contrast to APR and RPR methods, scene coordinate regression (SCR) [34] approaches involve training model that predict dense 3D coordinates for points in the query image and computing pose from the dense 2D–3D correspondences. DSAC[7] was amongst the earliest works to propose an end-to-end differentiable SCR architecture. Subsequently, the framework was extended for improved efficiency during inference [17], removing the need for RGBD ground truth during training [4], and improving the accuracy and robustness of the underlying method [6]. Other ideas have been explored, such as, the use of ensembles to improve scalability [5], design of hierarchical scene representations [18] and scene agnostic approaches[45], and ideas to make the models amenable to continual updates [42] and faster training [8].

Finally, we review methods for privacy-preserving localization, and storage efficiency. Speciale et al [35] explored new geometric scene and query representations [35, 36] and proposed pose estimation techniques for those representations. GoMatch [47] is a storage efficient method for geometric matching of 2D keypoints and 3D points that does not require local descriptors. SegLoc [21] achieves storage efficiency by leveraging semantic segmentation-based map and query representations. Approaches leveraging objects of interests in the scene [43] have also been studied.

3. Proposed Methodology

We now present a brief review of SLD before describing our proposed ideas for improvements, in the following sections.

3.1. Background: Scene Landmark Detection

Do *et al*. [11] proposed a localization approach where given the SfM reconstruction of the mapping images, a few salient, scene-specific 3D points are first selected from the SfM point cloud. Then, two CNNs (SLD, NBE) are trained using the mapping images and their associated poses. While SLD detects the landmarks visible in images, NBE regresses 3D bearing vectors for all the landmarks in the scene. Finally, the 2D–3D landmark constraints are used to recover the camera pose. Figure 1 provides an overview.

Landmark Selection. 3D scene points with discriminative appearance that are associated with permanent scene structures can serve as good scene landmarks. Do *et al*. [11] proposed a greedy method to select landmarks, given SfM camera poses, 3D points and the associated 2D image observations. Their method heuristically selects groups of 3D points that are well distributed within the scene. We use the

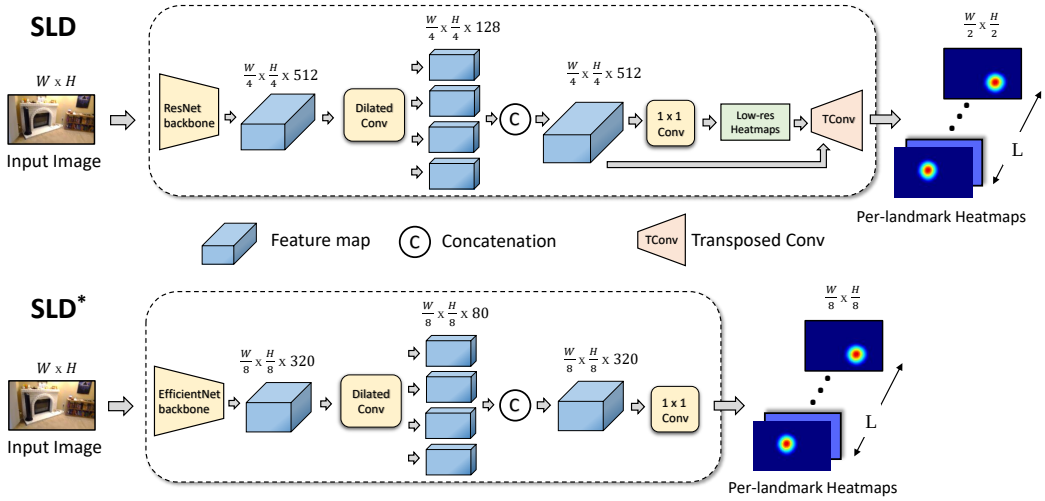


Figure 2. [Top] The original SLD architecture [11]. [Bottom] An illustration of the proposed SLD* architecture (see text for details).

same method, but experiment with up to 1500 landmarks, in contrast to the use of 200–400 points in prior work [11].

Model. The SLD architecture is fully convolutional and inspired by existing neural architectures for keypoint prediction in images using heatmaps. Do *et al*. [11] implemented SLD using both ResNet-18 [13] and EfficientNet [37] backbones. The features from the backbone network are then passed into a dilated convolution layer [46] followed by a 1×1 convolution layer to produce low-resolution heatmaps. Finally, the heatmaps are upsampled using a transposed convolution layer. The architecture is illustrated in the upper part of Figure 2. In contrast to SLD, the NBE network uses fully connected layers after a ResNet-18 backbone that output the final bearing predictions. SLD and NBE models were trained on the same scene and the authors proposed running inference using both models on every query image.

Training. The SLD and NBE architectures are trained using ground truth 2D landmark detections (and 3D bearing vectors) derived from associated camera poses in the training data. Training SLD also requires knowledge about which images each landmark is visible in. The visibilities are recovered from 2D data association of SfM 3D points in the training images. SLD is then trained using mean squared loss with respect to the ground truth heatmaps, while NBE is trained with a robust angular loss.

Datasets and Metrics. SLD and NBE was evaluated on INDOOR-6 [11], a challenging indoor localization dataset with six scenes, where images captured over multiple days have strong lighting changes. Pseudo ground truth (pGT) camera poses were recovered with COLMAP [32]. Given, camera pose estimates, the standard rotational error ΔR and position error Δt is computed as follows.

$$\Delta R = \arccos \frac{\mathrm{Tr}(\mathbf{R}^\top \hat{\mathbf{R}})-1}{2}, \quad \Delta t = \|\mathbf{R}^\top \mathbf{t} - \hat{\mathbf{R}}^\top \hat{\mathbf{t}}\|_2.$$

given $(\mathbf{R}, \mathbf{t})$ and $(\hat{\mathbf{R}}, \hat{\mathbf{t}})$, the estimated and ground truth poses respectively. The final metric is recall at 5cm/5°, the fraction of test images where $\Delta R \leq 5°$ and $\Delta t \leq 5$cm.

3.2. SLD* Architecture

In this section, we introduce SLD*, a more compact and memory efficient architecture, and an improved pose solver. Next, we highlight the four key differences with SLD+NBE. Figure 2 compares the SLD and SLD* architectures.

NBE not used. Do *et al*. [11] proposed using NBE to directly regress bearing vectors of the landmarks even when they were not visible in the image. These bearing predictions were complementary to SLD's heatmap detections. As SLD's typical budget of landmarks is quite small, sometimes enough landmarks are not visible in a test image. However, the steps to merge the two sets of predictions is adhoc. SLD* does not use NBE, as it uses a larger landmark budget to directly address the underlying issue.

Absence of an upsampling layer. SLD first predicts a set of low-resolution heatmaps and then spatially upsamples them using transposed convolutions to produce the final heatmaps. In contrast, SLD* directly predicts the output heatmaps using 1×1 convolution without any spatial upsampling. Without the upsampling layer, SLD* has fewer parameters to learn and has a smaller memory footprint. Yet, this change does not adversely affect the accuracy of landmark prediction in our experience. This is because, for each detected landmark, the associated 2D position is estimated by computing a weighted mean of all the heatmap samples from a 17×17 patch centered at the location of the peak in each heatmap. We observe that the weighted averaging step provided sufficient sub-pixel precision in the 2D landmark coordinates and thus predicting heatmaps at a

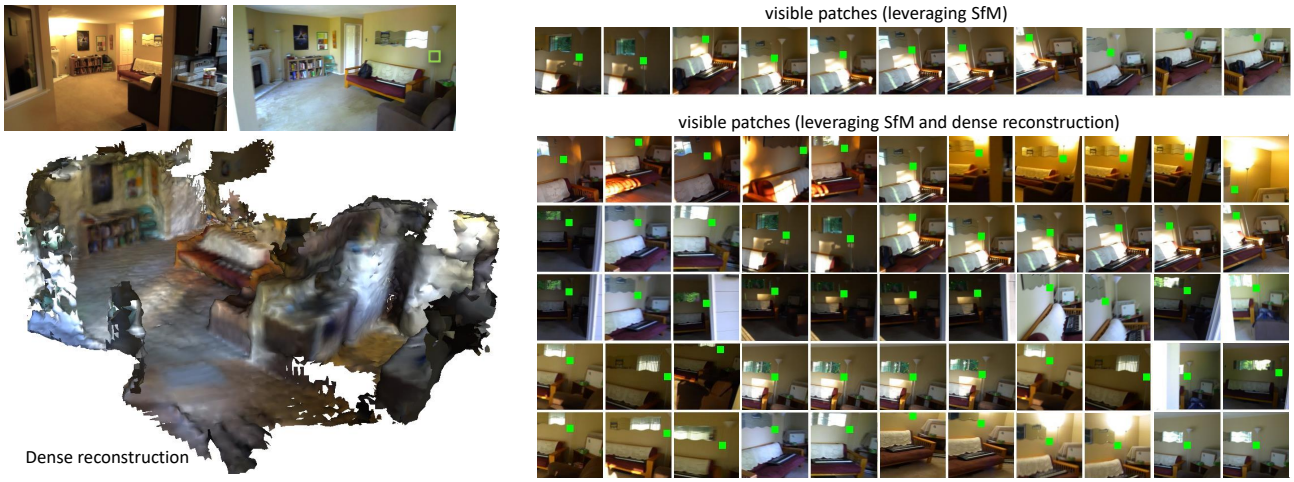


Figure 3. **Better Visibility Estimation**. [Left] Two images from scene1 in the INDOOR-6 dataset taken at different times of day and a rendering of the dense 3D mesh reconstruction of the scene. [Right] On the top right, we show a single row of patches depicting a scene landmark (indicated by the green square) in different images where the landmark was found to be visible. The original method leveraged data association from only structure from motion. On the lower right, we show patches for the same landmark based the proposed visibility estimation approach that also uses the dense mesh reconstruction (see text for details). The high appearance diversity in the observed patches under varying illumination makes the trained landmark detector more robust.

high output resolution appears to be unnecessary.

Memory footprint reduction. Do *et al*. [11] experimented with both ResNet-18 [13] and EfficientNet [37] backbones. In our implementation, we focus only on EfficientNet, as we aim to reduce the storage size and the memory footprint of the architecture. Furthermore, we use fewer feature map channels and more aggressive downsampling than SLD. SLD* has 320 channels unlike SLD which has 512 channels. SLD*'s feature maps have $8\times$ downsampling in contrast to SLD, where the downsampling factor is $4\times$.

Weighted pose estimation. We implemented a weighted pose estimation scheme using weights derived from the heatmap values associated with SLD*'s output predictions. Denoting peak heatmap values per detection as v, we first prune detections for which $v \leq 0.3$. Next, we compute a per-landmark weight $w = v^e$ where e is a parameter. We propose using the weights w in two different steps. First, for PROSAC [9] (RANSAC variant) used for robust estimation and also as weights during the PnP pose optimization.

3.3. Landmark Visibility Estimation

In this section, we discuss a limitation of how training data is generated for SLD [11] and propose methods for addressing the limitation. While SfM pipelines such as COLMAP [32] can produce 3D points with accurate 2D data association in multiple images, they often fail to detect all the potential observations (true positives) of the point. This can happen when the illumination varies dramatically. To alleviate this issue, Do *et al*. [11] proposed an ad-hoc augmentation strategy where they assumed that a landmark is visible in images whose camera poses estimated by SfM are nearby to the pose of images where the point is known to be visible. However, this strategy can corrupt the training data with outliers (false positives) by including views where the landmark is occluded. We propose to mitigate this issue using geometry and explicit occlusion reasoning.

Dense Reconstruction. We reconstruct a dense 3D mesh for each scene as follows. First, dense monocular depth maps for all map images are estimated using the dense depth vision transformer [24]. The dense 3D point clouds from these depth maps are then robustly registered to the sparse SfM 3D point cloud (which is computed by COLMAP [32]). The registration involves robustly estimating an affine transformation from 3D point-to-point matches. We first prune 3D points observed in less than 50 images and remove images which did not observe a sufficient number of 3D points. We also check residuals after aligning the depth maps to the SfM points and prune out images for which the mean depth residual exceeded 5cm. Finally, we use truncated signed distance function based depth-map fusion and isosurface extraction to compute the mesh. Figure 3 shows the reconstructed mesh for scene1.

Occlusion Reasoning. For every pair of a selected landmark $\mathbf{p}$ and an image $\mathcal{I}$ with its pose $\mathbf{T}_\mathcal{I}$ and the estimated dense depth $d_\mathcal{I}$ we determine whether the landmark is visible in the image by checking the following conditions:

- The 3D point $\mathbf{p}$ is in front of the camera for $\mathcal{I}$ (i.e., $(\mathbf{T}_\mathcal{I}\mathbf{p}_l)_z > 0$) and the point projects within the image

	Num. Landmarks			
	100	200	300	400
scene1	34.7	39.8	41.8	17.6
scene2a	31.5	46.3	45.9	28.8
scene3	34.3	43.2	55.2	42.5
scene4a	46.2	63.3	65.8	42.4
scene5	28.5	31.4	35.1	29.7
scene6	43.3	58.2	56.4	40.3
avg.	36.4	47.0	50.0	33.6

(a) **Pose recall at 5cm/5°** (in %) ↑

	Num. Landmarks			
	100	200	300	400
scene1	0.29	0.39	0.41	0.53
scene2a	0.24	0.25	0.34	0.43
scene3	0.27	0.29	0.36	0.45
scene4a	0.23	0.28	0.29	0.44
scene5	0.25	0.39	0.40	0.53
scene6	0.25	0.28	0.30	0.46
avg.	0.26	0.31	0.35	0.47

(b) **angular error (in deg.)** ↓

	Num. Landmarks			
	100	200	300	400
scene1	0.29	0.34	0.35	0.49
scene2a	0.24	0.26	0.31	0.37
scene3	0.27	0.29	0.31	0.39
scene4a	0.23	0.26	0.28	0.41
scene5	0.25	0.32	0.35	0.41
scene6	0.25	0.29	0.34	0.45
avg.	0.26	0.29	0.32	0.42

(c) **angular error (in deg.) (first 100)** ↓

Table 1. **Analyzing Model Capacity:** (a) The table reports averge camera pose estimation accuracy according to the 5cm/5° recall metric for four SLD* models trained with 100, 200, 300 and 400 landmarks respectively for all the scenes in INDOOR-6. (b) The median angular error in degrees for the same four models averaged across the six scenes. The median is computed over the set of all 2D SLD detections obtained using the trained models on all the test images. (c) In our implementation, the elements in the selected set of landmarks are stored in the order they were selected. Therefore, the fist 100 landmarks in the ordered sets for the models trained on 100, 200, 300 and 400 landmarks are identical. The median errors for the first 100 landmarks averaged on all scenes, are reported in the table.

(i.e., $\Pi(\mathbf{T}_{\mathcal{I}}\mathbf{p}_l)) \in \mathcal{R}(\mathcal{I})$ where $\Pi(.)$ is the 2D projection operator and $\mathcal{R}(.)$ denotes the image extent.

- The depth of the 2D projected point is not too far from the depth at that pixel, computed using the reconstructed mesh, i.e., $d_{\mathcal{I}}(\Pi(\mathbf{T}_{\mathcal{I}}\mathbf{p})) \approx (\mathbf{T}_{\mathcal{I}}\mathbf{p})_z$.
- The surface normal of the 2D projected point is not too far from the normal vector estimated using the reconstructed mesh, i.e, $\nabla d_{\mathcal{I}}(\Pi(\mathbf{T}_{\mathcal{I}}\mathbf{p})) \approx \nabla(\mathbf{T}_{\mathcal{I}}\mathbf{p})_z$

3.4. Landmark Partitioning For Scalability

In this section, we discuss what prevents SLD from accurately scaling to a large number of landmarks and present a simple solution that does not add computational overhead.

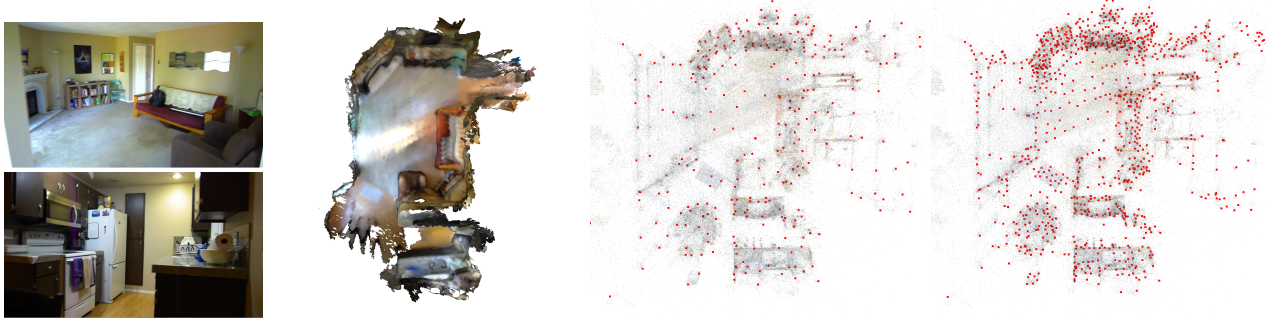
Insufficient Capacity. Do *et al*. [11] evaluated SLD (with ResNet-18) models with 200, 300 and 400 landmarks per scene on INDOOR-6 and reported that 300 landmarks worked best. When evaluating SLD* with different number of landmarks, we observed that accuracy increases from 100 to 300 but falls with 400 landmarks (see the recall at 5cm/5° metrics in Table 1(a)). It is worth noting that the smaller sets of landmarks are strictly contained within the larger landmark sets. The results imply that insufficient model capacity in the network could be hurting accuracy.

To confirm our hypothesis, we analyzed the angular errors of the predicted 2D landmarks from the SLD* models trained on 100, 200, 300 and 400 landmarks. The median angular errors reported in Table 1(b) increased as the number of landmarks increased. The angular errors depend only on the network, and are not affected by pose estimation or other factors. We also analyzed the angular error of the first 100 landmarks (defined with respect to an ordering defined by landmark ids) for the four SLD* models trained on 100, 200, 300 and 400 landmarks. Since the first 100 landmarks are identical in all four cases, comparing the median errors on these 100 points in the four models is the best way to compare them. Indeed as Table 1(c) shows, the predictions for the first 100 landmarks get worse as the model is trained for 200, 300 and 400 landmarks. This confirms our hypothesis that the models have insufficient capacity.

Training network ensembles. Instead of modifying the architecture, we address the insufficient capacity issue by choosing a divide and conquer strategy for scaling to a higher number of landmarks. We propose to simply partition the set of landmarks into non-overlapping subsets where the subsets are relatively small and their size is selected by keeping the typical capacity of the SLD* architecture under consideration. Then, we independently train multiple identical networks, one for each subset. We refer to the networks together as an ensemble. The networks in the ensemble can be trained independently and each is aware only of its own associated subset of landmarks. Training a SLD* ensemble is thus trivially parallelizable.

Parallel vs. Sequential Inference. At test time, there are two ways to run inference using the ensemble. When GPU memory is abundant, all SLD* networks could be initialized in GPU memory, allowing parallel inference on multiple networks. Despite having multiple networks, the total memory footprint can still be quite reasonable as each SLD* network is quite memory efficient (< 0.99 GB). In this setting, inference can be extremely fast and real-time processing is quite viable. However, on GPUs with smaller memory budgets, inference must be done sequentially. Even though, the processing time grows, localization can still run at 3–5 images/sec for practical ensemble sizes. In this paper, all reported timings are for the sequential inference setting.

Partioning Criteria. We compare four different criteria for partitioning the landmark set – (1) Default: sorting land-



Scene1 images | Scene1 mesh | 300 landmarks | 1000 landmarks

Figure 4. The top view of the mesh and 3D SfM point cloud from scene1, shown with the overlaid scene landmarks (red points). The sets of 300 and 1000 landmarks respectively are both computed by the existing selection method [11]. The image on the right shows that a higher number of landmarks provides denser scene coverage. We show later that it leads to an improvement in camera pose accuracy.

	200×1	300×1	100×3	100×4	125×6	125×8	125×12
R @ 5cm/5° ↑	46.0	50.8	61.1	63.0	66.6	**70.1**	69.1
Time (sec.) ↓	**0.05**	0.11	0.16	0.19	0.23	0.3	0.5
Size (MB) ↓	**15**	**15**	45	60	90	120	180

Table 2. **Ablation study.** Recall at 5cm/5° for a×b ensembles where a is the number of landmarks in each subset and b is the number of networks in the ensemble. The 125×8 ensemble has the best performance. As expected, ensembles containing more networks and dealing with more scene landmarks have slightly higher storage requirements and running times.

	~~w~~	w = v	w = v$\sqrt{v}$	w = v^2	w = v$^2\sqrt{v}$
R @ 5cm/5° ↑	68.0%	68.4%	69.4%	**70.1**%	69.6

Table 3. **Evaluating weighted pose estimation schemes.** Recall at 5cm/5° of a 1000 landmark SLD* ensemble on INDOOR-6 for non-weighted (~~w~~) and weighted pose estimates. Four schemes for deriving the weights (w) from heatmap values (v) are compared.

marks by the saliency score and then splitting the sorted list into equal sized partitions; (2) Random: randomly assigning landmarks to partitions; (3) Spatial clustering: grouping landmarks by k-means clustering and then rebalancing points in adjoining clusters to get equal sized partitions; (4) Farthest-point sampling: iteratively selecting the point farthest from the points already in existing partitions and adding it to the best partition until all partitioned reached the specified size. We compared the four criteria using 1000 landmarks and 8 partitions and found that the recall at 5cm/5° pose metric was similar (within 1-2 % points) in the four cases. We conclude that the partitioning criteria is not crucial on the dataset and thus used the default strategy thereafter. However, when a coarse location prior is available in large scenes, clustering-based partitioning can improve computational efficiency by enabling locality-based pruning of redundant inference passes.

4. Experimental Results

In this section, we report ablation studies and a quantitative comparison of SLD* and other methods on INDOOR-6. We then study in detail, the accuracy and speed tradeoff of SLD* and hloc [26]. Finally, we present visual examples to show the benefit of using a larger number of landmarks.

Ablation: Ensemble Size. Table 2 shows 5cm/5° recall for a variety of ensemble sizes that we have evaluated. We empirically found that 125×8 (8 networks with 125 landmarks each) works the best on INDOOR-6. We also report how storage and running times increase proportional to the ensemble size and the total number of landmarks.

Ablation: Weighted Pose Estimation. Table 3 reports 5cm/5° recall for non-weighted and weighted pose estimation using a 125×8 SLD* ensemble. For the weighted case, the effect of setting values of the parameter e to 1, 1.5, 2 and 2.5 is reported. The setting e = 2 gave the best results and was used in all the other experiments.

Quantitative Evaluation. In Table 4, we compare recall at 5cm/5° for several methods. For **DSAC*** [6], **SegLoc** [21], **NBE+SLD** [11] we present results reported in prior work [11, 21]. The **SLD** column in the table shows the results of the public EfficientNet-based SLD implementation. The table also includes results of hloc [26]. Previously reported results are shown in column hloc-A whereas our results obtained with hloc's public implementation are shown in column hloc-B. Finally, hloc-lite (hloc-l) results from prior work [11] are also included.

The accuracy metric for SLD* and SLD (both with 300 landmarks) is 50.8% and 44.9% respectively. This 6% accuracy improvement of SLD* can be attributed to better training labels generated using the proposed visibility estimation method. However, the best results of SLD* is 70.1% obtained using a 125×8 ensemble trained on 1000 landmarks which is competitive with hloc [26] at 71.4%.

	Scene	DSAC* [6]	NBE+SLD [11]	SLD [11]	SegLoc [21]	SLD* **ours**	hloc-l_{1000} [11]	hloc-l_{3000} [11]	hloc-A [11, 26]	hloc-B [26]	SLD* **ours**
#landmarks		n/a	300	300	n/a	300	1000	3000	n/a	n/a	1000
R@5cm/5° ↑	scene1	18.7	38.4	35.0	51.0	47.2	33.3	48.1	64.8	**70.5**	68.5
	scene2a	28.0	–	34.6	56.4	48.2	12.5	17.1	51.4	52.1	**62.6**
	scene3	19.7	53.0	50.8	41.8	56.2	48.3	61.9	81.0	**86.0**	76.2
	scene4a	60.8	–	56.3	33.8	67.7	34.8	39.2	69.0	75.3	**77.2**
	scene5	10.6	40.0	43.6	43.1	33.7	21.9	31.1	42.7	**58.0**	57.8
	scene6	44.3	50.5	48.9	34.5	52.0	47.4	59.1	79.9	**86.7**	78.0
R@5cm/5° ↑	avg.	30.4	45.5	44.9	43.4	50.8	33.0	42.8	64.8	**71.4**	70.1
Size (GB) ↓		0.027	0.135	0.020	0.161	0.015	0.17–0.21	0.2–0.5	0.7–2.4	0.7–2.4	0.120
Mem. (GB) ↓		0.85	1.35	1.2	–	0.99	1.3	1.3	1.3	1.3	0.99

Table 4. **Quantitative Evaluation on INDOOR-6.** We report the recall at 5cm/5° (in %), storage used (Size), and in-memory footprint (Mem.) of several methods. For the SLD [11] baseline, we report previously published results in the column NBE+SLD and results from the public EfficientNet-based code in the SLD column. For hloc [26], we first present published results in Do *et al*. [11] in the column hloc-A, and then, the best results we obtained using hloc's public codebase in the column hloc-B. Finally, we present results for SLD* (denoted "ours") with 300 and 1000 landmarks respectively. The best method (per row) is highlighted in bold and the second-best in blue.

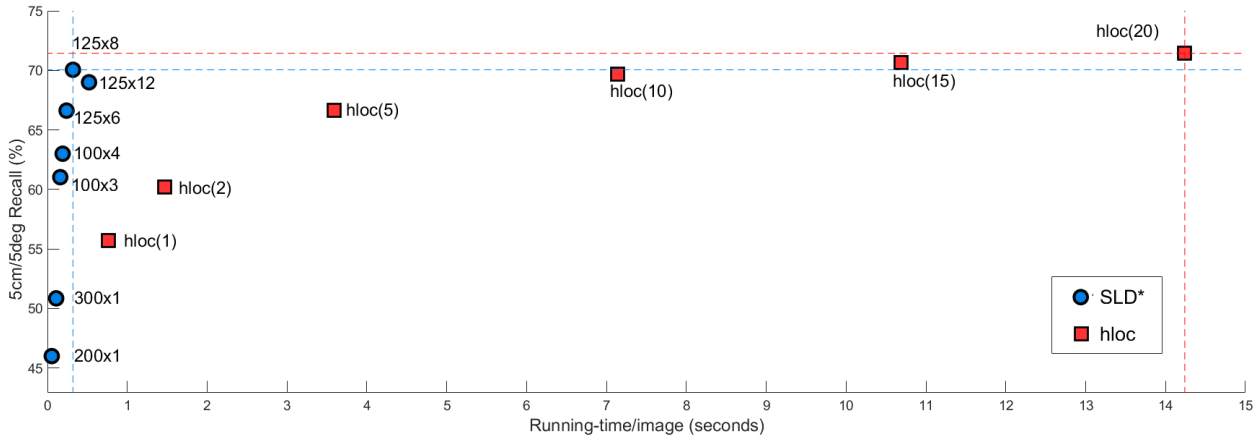


Figure 5. **Accuracy/speed tradeoff of SLD* and hloc.** The plot shows how hloc's performance varies with the number of matched image pairs. Tthe number of pairs were set to 1, 2, 5, 10, 15 and 20 respectively, as denoted by the text labels). hloc's best accuracy was 71.4% with 20 image pairs for which the timing was 14.2 seconds/image. Similarly, seven SLD* configurations were evaluated. The text label a × b next to the blue dots indicate the SLD* configuration, where a is the number of landmarks in each partition and b represents the number of partitions. SLD*'s best result was 70.1% using 125 × 8 = 1000 landmarks with a running time of 0.3 seconds/image. The plot shows that accuracy wise, SLD*'s best configuration is competitive with hloc but more than 40X faster.

Accuracy Speed Trade-off. Accuracy wise, SLD* and hloc have similar performance on INDOOR-6. Thus, we report a detailed accuracy and speed trade-off analysis for them. Figure 5 shows that the two hloc configurations (where 15 and 20 matching pairs are used respectively) beats SLD* by a small accuracy margin. However, these two hloc configurations are quite slow. The best SLD* setting outperforms all other hloc configurations (which use 1, 2, 5 and 10 matching pairs). Moreover, even though smaller ensembles are slightly worse accuracy wise, they also run significantly faster. Note that, the reported timings are for sequential inference. In the parallel inference setting, SLD* runs extremely fast because all the models are preloaded in GPU memory and all networks run inference in parallel. However, the memory footprint of the ensemble linearly increase with its size. Nonetheless, parallel inference may

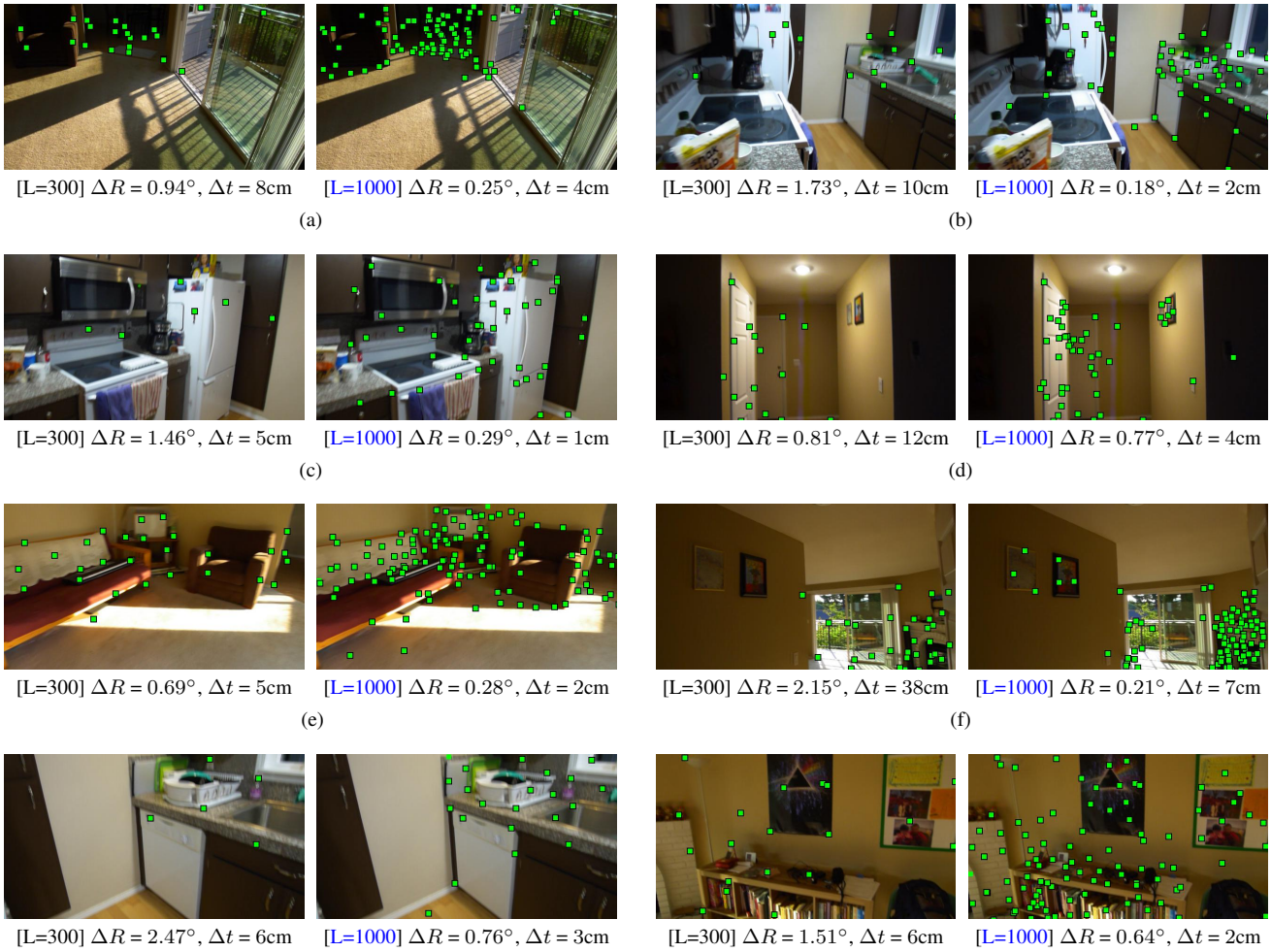


Figure 6. **Qualitative results on INDOOR-6.** Detected scene landmarks are shown as green points on the images from scene1, and the rotation and translation errors in the SLD* pose estimate are also reported below each image. (a)–(h) In all eight examples, the result on the left is for 300 landmarks, whereas the result on the right is for 1000 landmarks. Using 1000 landmarks instead of 300 landmarks produces more 2D–3D point constraints and the 2D locations are spatially better distributed in most images. which later yields a more accurate pose.

still be practical when sufficient GPU memory is available.

Qualitative Results. Finally, we present test images from scene1 in Figure 6 that were localized using two SLD* models trained on 300 and 1000 landmarks respectively and also report the associated pose errors. These examples clearly demonstrate the benefit of scaling up the number of scene landmarks. The model for 1000 landmarks consistently produces more accurate results, due to more pose inliers being present and a better distribution of those inliers. All SLD* models were trained using NVIDIA V100 GPUs whereas queries were processed on a laptop with a RTX 2070 GPU.

5. Conclusion

In this paper, we proposed SLD*, an extension of the existing SLD framework for scene landmark detection-based camera localization. SLD* is memory and storage efficient like SLD but it shows a dramatic improvement in performance (accuracy). The improvement makes SLD* competitive with structure-based methods such as hloc [26, 27] while being about 40X faster. The improved accuracy can be attributed to two ideas proposed in the paper First, we proposed a new processing pipeline to generate more accurate training labels for training the detector. Secondly, we showed that partitioning the landmarks into smaller groups and training independent networks for each subgroup dramatically boosts accuracy when a large number of scene landmarks are present. SLD* is currently trained from scratch for each scene which is time consuming and expensive. Exploring ideas similar to those proposed recently for accelerating scene coordinate regression [8] could lead to faster training and is an important avenue for future work.

References

[1] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016. 2

[2] Vassileios Balntas, Shuda Li, and Victor Prisacariu. RelocNet: Continuous metric learning relocalisation using neural nets. In *ECCV*, 2018. 2

[3] Alessandro Bergamo, Sudipta N Sinha, and Lorenzo Torresani. Leveraging structure from motion to learn discriminative codebooks for scalable landmark classification. In *CVPR*, 2013. 2

[4] Eric Brachmann and Carsten Rother. Learning less is more - 6D camera localization via 3D surface regression. In *CVPR*, 2018. 2

[5] Eric Brachmann and Carsten Rother. Expert sample consensus applied to camera re-localization. In *ICCV*, 2019. 2

[6] Eric Brachmann and Carsten Rother. Visual camera relocalization from rgb and rgb-d images using DSAC. *TPAMI*, 2021. 1, 2, 6, 7

[7] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC-differentiable ransac for camera localization. In *CVPR*, 2017. 1, 2

[8] Eric Brachmann, Tommaso Cavallari, and Victor Adrian Prisacariu. Accelerated coordinate encoding: Learning to relocalize in minutes using rgb and poses. In *CVPR*, 2023. 2, 8

[9] Ondrej Chum and Jiri Matas. Matching with PROSAC - progressive sample consensus. In *CVPR*, 2005. 4

[10] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPR workshops*, 2018. 2

[11] Tien Do, Ondrej Miksik, Joseph DeGol, Hyun Soo Park, and Sudipta N. Sinha. Learning to detect scene landmarks for camera localization. In *CVPR*, pages 11132–11142, 2022. 1, 2, 3, 4, 5, 6, 7

[12] Petr Gronat, Guillaume Obozinski, Josef Sivic, and Tomas Pajdla. Learning and calibrating per-location classifiers for visual place recognition. In *CVPR*, 2013. 2

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 4

[14] Tong Ke and Stergios I. Roumeliotis. An efficient algebraic solution to the perspective-three-point problem. In *CVPR*, 2017. 2

[15] Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, 2015. 1, 2

[16] Zakaria Laskar, Iaroslav Melekhov, Surya Kalia, and Juho Kannala. Camera relocalization by computing pairwise relative poses using convolutional neural network. In *ICCV*, 2017. 2

[17] Xiaotian Li, Juha Ylioinas, and Juho Kannala. Full-frame scene coordinate regression for image-based localization. In *RSS*, 2018. 2

[18] Xiaotian Li, Shuzhe Wang, Yi Zhao, Jakob Verbeek, and Juho Kannala. Hierarchical scene coordinate classification and regression for visual localization. In *CVPR*, 2020. 2

[19] Hyon Lim, Sudipta N Sinha, Michael F Cohen, Matt Uyttendaele, and H Jin Kim. Real-time monocular image-based 6-dof localization. *IJRR*, 2015. 2

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

[21] Maxime Pietrantoni, Martin Humenberger, Torsten Sattler, and Gabriela Csurka. Segloc: Learning segmentation-based representations for privacy-preserving visual localization. In *CVPR*, 2023. 1, 2, 6, 7

[22] Noé Pion, Martin Humenberger, Gabriela Csurka, Yohann Cabon, and Torsten Sattler. Benchmarking image retrieval for visual localization. In *3DV*, 2020. 2

[23] Francesco Pittaluga, Sanjeev J Koppal, Sing Bing Kang, and Sudipta N Sinha. Revealing scenes by inverting structure from motion reconstructions. In *CVPR*, 2019. 1

[24] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, 44(3):1623–1637, 2020. 4

[25] Jerome Revaud, Philippe Weinzaepfel, César Roberto de Souza, and Martin Humenberger. R2D2: repeatable and reliable detector and descriptor. In *NeurIPS*, 2019. 2

[26] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *CVPR*, 2019. 1, 2, 6, 7, 8

[27] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 1, 2, 8

[28] Paul-Edouard Sarlin, Ajaykumar Unagar, Måns Larsson, Hugo Germain, Carl Toft, Victor 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. 2

[29] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Improving image-based localization by active correspondence search. In *ECCV*, 2012. 1, 2

[30] Torsten Sattler, Tobias Weyand, B. Leibe, and Leif P. Kobbelt. Image retrieval for image-based localization revisited. In *BMVC*, 2012. 1, 2

[31] Torsten Sattler, Qunjie Zhou, Marc Pollefeys, and Laura Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *CVPR*, 2019. 2

[32] Johannes Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *CVPR*, 2016. 3, 4

[33] Yoli Shavit, Ron Ferens, and Yosi Keller. Learning multi-scene absolute pose regression with transformers. In *ICCV*, 2021. 2

[34] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *CVPR*, 2013. 2

[35] Pablo Speciale, Johannes Schonberger, Sing Bing Kang, Sudipta N Sinha, and Marc Pollefeys. Privacy preserving image-based localization. In *CVPR*, 2019. 2

[36] Pablo Speciale, Johannes Schonberger, Sudipta N Sinha, and Marc Pollefeys. Privacy preserving image queries for camera localization. In *ICCV*, 2019. 2

[37] Mingxing Tan and Quoc V. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. *arXiv*, 2019. 3, 4

[38] Giorgos Tolias, Yannis Avrithis, and Hervé Jégou. To aggregate or not to aggregate: Selective match kernels for image search. In *ICCV*, 2013. 2

[39] Akihiko Torii, Relja Arandjelović, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. 24/7 place recognition by view synthesis. In *CVPR*, 2015. 2

[40] Akihiko Torii, Hajime Taira, Josef Sivic, Marc Pollefeys, Masatoshi Okutomi, Tomas Pajdla, and Torsten Sattler. Are large-scale 3d models really necessary for accurate visual localization? *IEEE transactions on pattern analysis and machine intelligence*, 43(3):814–829, 2019. 2

[41] Bing Wang, Changhao Chen, Chris Xiaoxuan Lu, Peijun Zhao, Niki Trigoni, and Andrew Markham. Atloc: Attention guided camera localization. In *AAAI*, 2020. 2

[42] Shuzhe Wang, Zakaria Laskar, Iaroslav Melekhov, Xiaotian Li, and Juho Kannala. Continual learning for image-based camera localization. In *ICCV*, 2021. 2

[43] Philippe Weinzaepfel, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. Visual localization by learning objects-of-interest dense match regression. In *CVPR*, 2019. 2

[44] Tobias Weyand, Ilya Kostrikov, and James Philbin. Planet-photo geolocation with convolutional neural networks. In *ECCV*, 2016. 2

[45] Luwei Yang, Ziqian Bai, Chengzhou Tang, Honghua Li, Yasutaka Furukawa, and Ping Tan. SANet: Scene agnostic network for camera localization. In *ICCV*, 2019. 2

[46] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2015. 3

[47] Qunjie Zhou, Sérgio Agostinho, Aljoša Ošep, and Laura Leal-Taixé. Is geometry enough for matching in visual localization? In *ECCV*, 2022. 2