SACReg: Scene-Agnostic Coordinate Regression for Visual Localization

Jerome Revaud[†] Yohann Cabon[†] Romain Brégier[†] JongMin Lee[*] Philippe Weinzaepfel[†]

[†]Naver Labs Europe
firstname.lastname@naverlabs.com

[*]Seoul University
sdrjseka96@naver.com

https://sacreg.europe.naverlabs.com/public/index.html

Abstract

Scene coordinates regression (SCR), i.e.*, predicting 3D coordinates for every pixel of a given image, has recently shown promising potential. However, existing methods remain limited to small scenes memorized during training, and thus hardly scale to realistic datasets and scenarios. In this paper, we propose a generalized SCR model trained once to be deployed in new test scenes, regardless of their scale, without any finetuning. Instead of encoding the scene coordinates into the network weights, our model takes as input a database image with some sparse 2D pixel to 3D coordinate annotations, extracted from* e.g. *off-the-shelf Structure-from-Motion or RGB-D data, and a query image for which are predicted a dense 3D coordinate map and its confidence, based on cross-attention. At test time, we rely on existing off-the-shelf image retrieval systems and fuse the predictions from a shortlist of relevant database images w.r.t. the query. Afterwards camera pose is obtained using standard Perspective-n-Point (PnP). Starting from self-supervised CroCo pretrained weights, we train our model on diverse datasets to ensure generalizabilty across various scenarios, and significantly outperform other scene regression approaches, including scene-specific models, on multiple visual localization benchmarks. Finally, we show that the database representation of images and their 2D-3D annotations can be highly compressed with negligible loss of localization performance.*

1. Introduction

Image-based scene coordinate regression (*SCR*) consists in predicting the 3D coordinates of the point associated to each pixel of a given query image. *SCR* methods have numerous applications in computer vision, and previous work has shown promising potential over the last few years. Such methods have for instance been proposed for visual localization [9, 60, 66, 82] in combination with a Perspective-n-Point (*PnP*) solver [34]. Other applications include object pose estimation [5, 85], depth completion [15, 27, 44, 47, 73], augmented reality or robotics [60, 80].

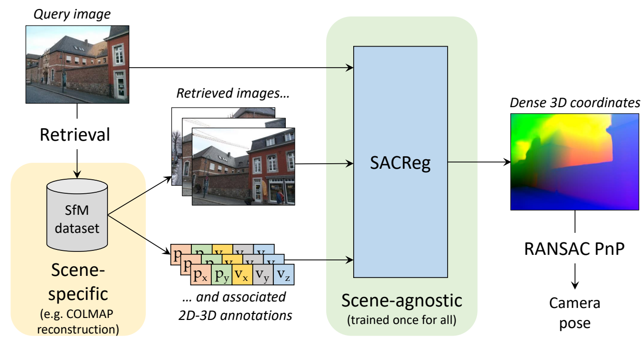

Figure 1. **Scene-agnostic coordinate regression (SACReg) for visual localization.** Given a query image and a set of related views with sparse 2D-3D annotations retrieved from a database (bottom left), SACReg predicts absolute 3D coordinates for each pixel of the query image (right). These can be used for visual localization using a robust *PnP* algorithm (bottom right). Importantly, SACReg is scene-agnostic: it can be used in any novel scene without re-training, only the images and 2D-3D annotations that serve as input are scene-specific.

Unfortunately, existing *SCR* approaches pose significant scalability issues and end up being rather impractical. Most of the time, 3D scene coordinates are directly embedded into the parameters of the learned model, being it a random forest [60] or a neural network [6, 8, 9, 80], hence *de facto* limiting one model to a specific, generally small, scene for which it was trained on. Some recent attempts to mitigate this issue, such as training different experts [8], sharing scene-agnostic knowledge between scenes [67], or heavily relying on dense 3D reconstructions at test time [66, 82], improve by some aspects but still require scene-specific finetuning, can be limited to small scenes, and do not offer scene-agnostic solutions yet. In essence, there is no *universal* SCR model that can seamlessly function *as-is* on any given test scene.

In this paper, we propose a new paradigm for scene coordinates regression that allows to train a generic model once, and deploy it to novel scenes of arbitrary scale. As illustrated in Figure 1, our scene-agnostic coordinate regression (SACReg) model takes as input a query image as well as a set of relevant database images for which 3D scene coordinates are available at sparse 2D locations. SACReg predicts dense 3D coordinates, for each pixel of the query image. From this output, the camera pose can be obtained by solving a Perspective-n-Point (PnP) problem. Note that all inputs of SACReg can be obtained via off-the-shelf methods: relevant database images can be obtained using image retrieval techniques [29, 49, 78], while the sparse 2D-3D correspondences are a by-product of map construction procedures, *i.e*., obtained using dedicated sensors or Structure-from-Motion (SfM) pipelines [58].

In summary, our first contribution is to introduce a generic model for scene-agnostic coordinate regression. It uses a Vision Transformer (ViT) [20] to encode query and a database image, as illustrated in Figure 2. Database image tokens are augmented with their provided sparse 2D-3D correspondences, using a transformer decoder. Afterward, another transformer decoder combines these augmented tokens with those extracted from the query image, which are further processed by a convolutional head to regress dense 3D scene coordinates and an associated pixelwise confidence map. Finally, predictions made separately for each database image are fused based on the confidence values.

As a second contribution, we propose to regress an encoding of the 3D coordinates rather than the raw 3D coordinates. Doing so solves a major limitation of existing scene-agnostic approaches which assume small scenes with zero-centered coordinate systems and cannot generalize to unbounded scenes [66, 82]. To that aim, we introduce an invertible and noise-resistant cosine-based encoding of 3D coordinates. We show that it can generalize effortlessly to arbitrary coordinate ranges at test time.

As a third contribution, we show that the augmented database tokens (combining image and associated 2D-3D correspondences) can be pre-computed and compactly stored for faster inference. Specifically, using simple product quantization (PQ) [30], we achieve compression rates over 30 for VGA images with no loss of performance, reducing the storage needs from 3.7MB to 115kB per image. This simple scheme significantly outperforms recent compression approaches for visual localization and sets a new state of the art of database footprint.

Lastly, we report on par or better performance than existing state-of-the-art SCR approaches on multiple benchmarks without any finetuning. To ensure generalization, we initialize the network weights with cross-view completion pretraining (CroCo) [77, 79] and train on diverse sources: outdoor buildings with the MegaDepth dataset [38], indoor environments from the ARKitScenes [3] dataset and synthetic data generated using the Habitat-Sim simulator [57]. In particular, we find that CroCo pretraining is a key ingredient to the success of our approach. On the Aachen Day-Night [56] and Cambridge-Landmarks [33] benchmarks, SACReg outperforms current scene-specific and dataset-specific *SCR* methods, while being competitive with state-of-the-art structure-based methods [29].

2. Related work

Scene-specific coordinates regression. Several methods have been proposed to estimate dense 3D coordinates for a query image in a scene known at training time. Early approaches [25, 60, 72] used regression forest models to predict the correspondence of a pixel in a RGB-D frame to its 3D world coordinate. More recent works [6–9, 19, 28, 36, 66, 80, 82, 87] have replaced regression forests with CNN-based models that only require an RGB image. For example, Brachmann *et al*. [6–9] train neural networks for this task and combine them with a differentiable RANSAC strategy for camera relocalization. Dong *et al*. [19] and Li *et al*. [36] later introduce region classification into their pipelines for effective scene memorization. Huang *et al*. [28] propose to add a segmentation branch to obtain segmentation on scene-specific landmarks, which can then be associated with 3D landmarks in the scene to estimate camera pose. These methods are designed to memorize specific scenes, making them hard to scale and impractical in many scenarios where the test scene is unknown at training time. In contrast, our method can adjust at test time to any environment for which a database of images is available, by relying on external image retrieval techniques.

Scene-agnostic coordinates regression with dense database 3D points. More related to our work are the scene-agnostic methods of [66, 82]. They regress dense scene coordinates given some reference views for which dense coordinates are already available. Their methods are also limited to small scenes with unit-normalized world coordinates. In contrast, our approach only requires sparse annotations and imposes no restriction on coordinate range, making it better suited to large-scale environments.

Image-based localization consists in estimating 6-DoF camera pose from a query image. Different approaches can be used towards that goal, and *SCR* is one of them. Recently, learning-based methods in which the pose of a query image is directly regressed with a neural network have been proposed [4, 10, 31, 33, 74, 75]. By training the network with database images and their known ground-truth poses as training set, they learn and memorize the relationship between RGB images and associated camera pose. These direct approaches however need to be trained specifically for each scene. This issue was somehow solved by relative pose regression models [1, 2, 18, 88], which train a neural network to predict the relative pose between the query image and similar database image found by image retrieval. However, their performance tends to be inferior to structure-based methods [26, 29, 58, 62]. Structure-based visual localization frameworks use sparse feature matching to estimate the pose of a query image relative to a 3D map

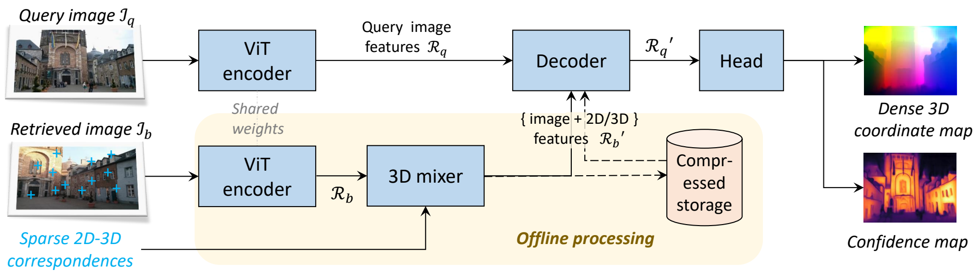

Figure 2. **Overview of the SACReg architecture** for a given pair of query and relevant database image. Both images are first encoded by a vision transformer, then sparse 2D-3D correspondences are used to augment the encoded tokens of the database image with geo-spatial information. A decoder jointly processes both sets of tokens and outputs a dense 3D coordinate map and an associated confidence map using a specific head. Database images can be encoded offline with their 2D-3D annotations and compressed for better test-time efficiency.

constructed from database images using SfM techniques, such as those employed in [58]. This involves extracting 2D features from images using interest point detectors and descriptors [17, 21, 35, 40–42, 45, 50, 52, 69, 71, 76, 84], and establishing 2D-3D correspondences. A PnP problem is solved using variants of RANSAC [23], which then returns the estimated pose. However, the structure-based methods have to store not only 3D points but also keypoints descriptors, and maintaining the overall localization pipeline is complex. Our approach in contrast does not require to store keypoints descriptors, is arguably simpler, and can use highly compressed database representations, thus reducing the storage requirement.

Database compression for visual localization. Compressing the database while maintaining localization accuracy is important for scalable localization systems. For structured-based methods, most techniques rely on selecting a compact but expressive subset of the 3D points. K-cover method and its follow-up works [11, 12, 14, 37] reduce the number of 3D scene points, maintaining even spatial distribution and high visibility. Some other methods [22, 43, 46, 83] formulate the problem with quadratic programming (QP) to optimize good spatial coverage and visual distinctiveness. Another approach for compression is feature quantization: descriptors associated to 3D points can be compressed into binary representation [13] or using quantized vocabularies [54]. In the field of SCR, the recent NeuMap [67] approach leverages a latent code per voxel and applies code-pruning to remove redundant codes.

3. The SACReg model

After describing our scene-agnostic coordinate regression model (Section 3.1), we then detail our robust coordinate encoding and associated training loss (Section 3.2). We finally present the application to visual localization (Section 3.3) and training details (Section 3.4).

3.1. Model architecture

Our model takes as input a query image $\mathcal{I}_q$ and a mapped database images $\mathcal{I}_b$ for which sparse 2D-3D annotations are available, denoted as $\mathcal{V} = \{(\mathbf{p}_j, \mathbf{v}_j)\}$ where $\mathbf{v}_j \in \mathbb{R}^3$ is a 3D point expressed in a world coordinate system visible at pixel $\mathbf{p}_j$. It then predicts a 3D coordinate point for every pixel in the query image. In the more realistic case where multiple database images are relevant to the query, we perform independent predictions between the query and each database image with the model described below, and fuse predictions afterward (see Section 3.3).

Overview. Figure 2 shows an overview of the model architecture. First, the query image is encoded into a set of token features with a Vision Transformer [20] (ViT) encoder. The same encoder is used to encode the database image, but this time the resulting database features are augmented with geo-spatial information from the sparse 2D-3D annotations. This is achieved using a transformer decoder referred to as *3D Mixer* in the following. The next step consists in transferring geo-spatial information from the augmented database features to the query features using a transformer decoder. Finally, a prediction head outputs dense 3D coordinates for each pixel of the query image, see Figure 3. We now detail each module: the image encoder, the 3D mixer, the decoder and the prediction head.

Image encoder. We use a vision transformer [20] to encode the query and database images. In more details, each image is divided into non-overlapping patches, and a linear projection encodes them into patch features. A series of transformer blocks is then applied on these features: each block consists of multi-head self-attention and an MLP. In practice, we use a ViT-Base model, *i.e.*, 16×16 patches with 768-dimensional features, 12 heads and 12 blocks. Following [79, 81], we use RoPE [64] relative position embeddings. As a result of the ViT encoding, we obtain sets of token features denoted $\mathcal{R}_q$ for the query and $\mathcal{R}_b$ for the database image respectively.

3D mixer. We then augment the database tokens $\mathcal{R}_b$ with geo-spatial information encoded by the sparse 2D-3D correspondence set $\mathcal{V}$, yielding the augmented tokens $\mathcal{R}_b' =$ 3Dmixer$(\mathcal{R}_b, \mathcal{V})$. To that aim, we encode the 3D coordinates using a cosine point encoding ϕ before feeding them to an MLP (see next Section 3.2 for details on ϕ). We

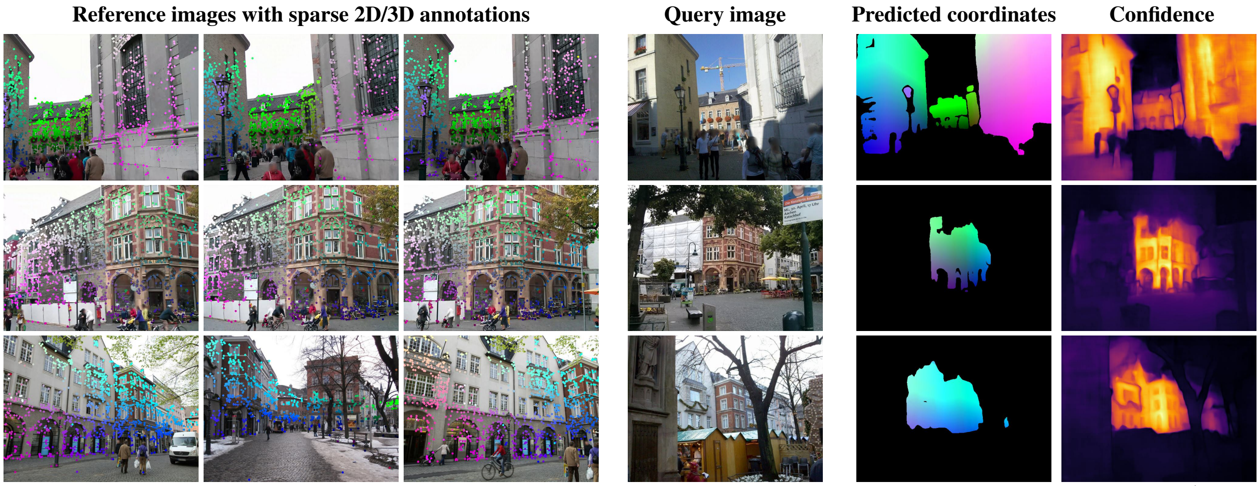

Figure 3. **Regression examples on Aachen-Day.** Our model predicts a dense 3D coordinates point map and a confidence map (5th and 6th columns) for a given query image (4th column) using reference images retrieved from a *SfM* database (1st, 2nd, 3rd columns). Only the first 3 reference images (out of $K = 8$) are depicted. 3D coordinates and confidence are colorized for visualization purposes, and areas with a confidence below $\tau = \exp(0)$ are not displayed. Best viewed in color.

then use a series of transformer decoder blocks, where each block consists of a self-attention between image tokens, a cross-attention that shares information from all point tokens with these image tokens, and an MLP. We find that alternating between image-level and point-level decoder blocks improves the performance, and we refer to the supplementary material for more details and ablative studies on the 3D mixer architecture.

Decoder. The next step is to transfer information from the database, *i.e.*, from $\mathcal{R}'_b$, into the query features $\mathcal{R}_q$. We again rely on the cross-attention mechanism of a generic transformer decoder, *i.e.*, a series of blocks, each composed of self-attention between the token features, cross-attention with the database tokens $\mathcal{R}'_b$ and an MLP, yielding augmented query features $\mathcal{R}'_q$.

Prediction head. We finally reshape $\mathcal{R}'_q$ from the last transformer decoder block into a dense feature map and apply a convolutional head. Specifically, we first linearly project the features to 1024 dimensions, then apply a sequence of 6 ConvNeXt blocks [39], with a PixelShuffle [59] operation every two blocks to increase the resolution while halving the channel dimension. For a 224×224 input image, we get a $14^2 \times 1024$ token map after the initial projection, which is gradually expanded to $28^2 \times 512$, $56^2 \times 256$ and finally $224^2 \times d$, d being the output dimension.

3.2. Generalization and training loss

Output space. A naive approach consists in setting $d = 3$, *i.e.*, trying to directly regress dense 3D points $\{\hat{\mathbf{v}}\} \in \mathbb{R}^3$ from the regression head. This is possible and could be trained with a standard ℓ_1 or ℓ_2 regression loss, but is subject to a major limitation. At test time, the network is typically unable to regress coordinates outside the range seen during training. Thus, except for small scenes, it cannot generalize to new datasets (see Section 4.2).

Instead, we propose to regress a higher-dimensional 3D point encoding $\phi(\mathbf{v}) \in (\mathbb{S}^1)^{d/2} \subset [-1,1]^d$, with $d \gg 3$. We design ϕ with several desirable properties holding for any given $\mathbf{v} \in \mathbb{R}^3$: (i) ϕ is an injective mapping, with an inverse projection ϕ^{-1} such that $\phi^{-1}(\phi(\mathbf{v})) = \mathbf{v}$; (ii) the input space of ϕ^{-1} is the unit-circle product $(\mathbb{S}^1)^{d/2} \subset [-1,1]^d$, whose high dimension enables error-correcting mechanisms in ϕ^{-1}. Thanks to these properties, our method can handle any coordinate at test time.

Point encoding. Assuming uncorrelated x, y and z coordinates, we can decompose $\phi(\mathbf{v}) = [\psi(x), \psi(y), \psi(z)]$ and define $\psi(x)$ as:

$$\psi(x) = [\cos(f_1 x), \sin(f_1 x), \cos(f_2 x), \sin(f_2 x), \dots] \quad (1)$$

where the f_i's are frequencies defined as $f_i = f_0 \gamma^{i-1}$, $i \in \{1, \dots, d/6\}$, with $f_0 > 0$ and $\gamma > 1$. In practice, we set f_0 and γ such that the periods of the lowest and highest frequencies f_1 and $f_{d/6}$ approximately correspond to the maximum scale of a query scene (*e.g.* 300 meters) and the desired spatial resolution (*e.g.* 0.5 meter). The encoding dimension d then becomes a parameter that controls the level of redundancy. d must be carefully chosen, as too small encodings are not noise-resistant enough, while too large encodings may demand too much capacity for the decoder. The inverse mapping ψ^{-1} efficiently solves a least-square problem of the form $\psi^{-1}(y) = \text{argmin}_x \|y - \psi(x)\|^2$, see the supplementary material and [51].

Regression loss. As for the naive regression case, we apply a standard ℓ_1 regression loss to train the network:

$$\mathcal{L}_{\text{reg}}(\mathbf{v}, \hat{\mathbf{y}}) = |\phi(\mathbf{v}) - \hat{\mathbf{y}}|, \quad (2)$$

where $\hat{\mathbf{y}} \in \mathbb{R}^d$ is the network output and $\mathbf{v}$ is the corresponding ground-truth 3D point. We further exploit the relation between pairs of adjacent components of $\phi(\mathbf{v})$, based on the equality $\cos^2(f_i x)+\sin^2(f_i x) = 1$. Before applying the $\mathcal{L}_{\text{reg}}$ loss, we thus ℓ_2-normalize each pairs of consecutive components of $\hat{\mathbf{y}}$. We empirically find that this helps the training significantly.

Pixelwise confidence. Regressing coordinates is inevitably harder, or even impossible, for some parts of the query image such as the sky or objects not visible in database images. We therefore jointly predict a per-pixel confidence $\tau > 0$ that modulates the regression loss (2), following [32]:

$$\mathcal{L}_{\text{SCR}}(\mathbf{v}, \hat{\mathbf{y}}, \tau) = \tau \mathcal{L}_{\text{reg}}(\mathbf{v}, \hat{\mathbf{y}}) - \log \tau. \quad (3)$$

τ can be interpreted as the confidence of the prediction: if τ is low for a given pixel, the corresponding $\mathcal{L}_{\text{reg}}$ loss at this location will be down-weighted. The second term of the loss incites the model to avoid being under-confident. The estimated confidence can also serve to fuse predictions from multiple database images, as well as for the PnP pose estimation step, see Section 3.3.

3.3. Application to visual localization

We now present how our model can be applied to predict the camera pose of a given query image from a small set of relevant database images with sparse 2D-3D point correspondences. An overview of our visual localization pipeline is shown in Figure 1.

Image retrieval. Given a query image, we first follow the same retrieval step than for standard feature-matching-based localization approaches [29, 53, 55]. Namely, we utilize off-the-shelf image retrieval methods such as HOW [70], AP-GeM [49] or FIRe [78] to obtain a shortlist of K relevant database images for a given query image.

Sparse 2D-3D annotations. Our model takes as input sparse 2D-3D correspondences for each database image. To get them, we randomly subsample 2D points from the dense RGB-D data and reproject them in 3D using the known camera poses, when available. If not, we rely on standard Structure-from-Motion pipelines [58] during which 2D keypoint matches between images are used to recover the corresponding 3D point locations and the camera poses. This process directly yields a set of 2D-3D correspondences for each database image. In practice, we use the output of COLMAP [58] with SIFT [40] keypoints.

Multi-image fusion strategy. To mitigate the potential presence of outliers returned by the image retrieval module, we fuse the predictions from the top-K relevant database images. We first compute the augmented database features $\mathcal{R}'_b$ for each image $\mathcal{I}_b$ separately, with $b = 1 \dots K$. We then feed each $(\mathcal{R}_q, \mathcal{R}'_b)$ pair to the decoder, gathering each time the dense coordinate and confidence output maps. The final aggregation is then simply done pixelwise. We fuse all results by keeping, for each pixel i, the most confident prediction according to the estimated confidence $\{\tau_b^i\}_{b=1\dots K}$.

Predicting camera poses. The output of our model is a dense 3D coordinate map and corresponding confidence map, see Figure 2. To perform visual localization, we first filter out all unconfident predictions, *i.e.*, points for which the confidence is inferior to the median confidence. We then use an off-the-shelf PnP solver to obtain the predicted camera pose. Specifically, we rely on SQ-PnP [68] with 4096 2D-3D correspondences sampled randomly, 10,000 iterations and a reprojection error threshold of 5 pixels.

Database compression. Since spatially-augmented database features $\mathcal{R}'_b$ do not depend on the query image (see Figure 2), they can thus be computed offline once and stored. Raw representations require a few megabytes (MB) of storage per database image, similar to standard feature-based localization methods. We find however that they can be significantly compressed with negligible loss of performance. Namely, we employ Product Quantization (PQ) [30], which is a simple and effective technique consisting of splitting vectors into multiple sub-vectors and vector-quantizing [24] them into byte codes (see the supplementary material for more details). Note that all the compression parameters (*e.g.* codebooks) are scene-agnostic as well, *i.e.*, trained once and for all.

3.4. Training details

We initialize the weights of the encoder and the decoder with CroCo v2 pretraining [79], which we find crucial for the success of our approach. We train our model on 512×384 images, but perform a first training stage with 224×224 images while freezing the encoder, *i.e.*, fine-tuning only the 3D mixer and the decoder for 100 epochs with a fixed learning rate of 10^{-4} to reduce overall training costs. Training is then performed at higher resolution for 40 epochs with a cosine decay learning rate schedule.

Data. We train our model on datasets that cover various scenarios for robustness: MegaDepth [38] contains SfM reconstruction of 275 (mainly) outdoor scenes, ARKitScenes [3] consists of indoor house scenes, and Habitat of synthetic indoor scenes derived from HM3D [48], ScanNet [16], Replica [63] and ReplicaCAD [65] rendered using Habitat-Sim [57]. These three datasets provide dense depth estimates and camera poses, thus allowing to train our model in a fully-supervised manner. We use 100K query from each dataset (300K in total). For each query, we use FIRe [78] to retrieve beforehand a shortlist of K similar images.

Augmentation. We apply standard random crop and color jitter during training. For robustness to possible triangulation noise, we augment 5% of the sparse 3D points with simulated depth noise. We also apply random geometric 3D transformation to scene coordinates for better generalization. Namely, we apply random 3D rotation followed by

Point encoding	Aug	Camb. ↓	7scenes ↓	Aachen-Night ↑
$(x, y, z) \in \mathbb{R}^3$		1.69	**0.11**	0.0 / 0.0 / 0.0
$(x, y, z) \in \mathbb{R}^3$	✓	14.43	2.89	0.0 / 2.1 / 44.5
$\phi(\cdot) \in [-1, 1]^{24}$	✓	0.47	**0.11**	22.0 / 46.6 / 89.5
$\phi(\cdot) \in [-1, 1]^{36}$	✓	**0.43**	**0.11**	22.0 / **47.1** / **90.6**
$\phi(\cdot) \in [-1, 1]^{48}$	✓	0.55	**0.11**	**23.6** / 40.8 / 87.4

Table 1. **Ablation on 3D point encoding.** Aug=Augmentation.

Pretraining	Frozen	Camb. ↓	7scenes ↓	Aachen-Night ↑
-	-	1.14	0.19	5.2 / 20.4 / 66.0
CroCo v2	-	0.54	0.14	18.3 / 37.7 / 85.3
CroCo v2	Encoder	**0.43**	**0.11**	**22.0 / 47.1 / 90.6**

Table 2. **Ablation on pretraining and encoder freezing.**

Regression head	Channels	Camb. ↓	7scenes ↓	Aachen-Night ↑
Linear	x, y, z, τ	0.94	0.12	11.0 / 31.9 / 84.3
ConvNeXt	$xyz\tau$	0.64	**0.11**	19.9 / 39.8 / 88.0
ConvNeXt	xyz, τ	0.61	**0.11**	15.7 / 41.4 / 87.4
ConvNeXt	x, y, z, τ	**0.43**	**0.11**	**22.0 / 47.1 / 90.6**

Table 3. **Ablation on regression head.**

Train res.	Test res.	Camb. ↓	7scenes ↓	Aachen-Night ↑
224×224	224×224	0.43	0.11	22.0 / 47.1 / 90.6
512×384	512×384	0.21	0.10	39.3 / 63.4 / **94.8**
512×384	640×480	**0.20**	**0.07**	**45.5** / 68.6 / **94.8**
512×384	768×512	0.24	**0.07**	**45.5** / **70.2** / 93.7

Table 4. **Impact of training and test image resolution.**

random scaling in the range $[1/2, 2]$ and random translation in $[-1000m, 1000m]^3$.

4. Experiments

After describing the test datasets (Section 4.1), we present ablations in Section 4.2 and provide visualizations of the attention in Section 4.3. We then compare our approach to the state of the art in visual localization without (Section 4.4) and with compression (Section 4.5), and finally evaluate the accuracy of the regressed coordinates (Section 4.6).

4.1. Datasets and metrics

Cambridge-Landmarks [33] consists of 6 outdoor scenes with RGB images from videos and small-scale landmarks.
7 Scenes [61] consists of 7 indoor scenes with RGB-D images from videos. Each scene has a limited size, and the images contain repeating structures, motion blur, and textureless surfaces.We do not use the depth data of the query image during inference.
Aachen Day-Night v1.1 [56, 86] contains 6,697 database images captured at day time, and 1015 query images including 824 taken during daytime (Aachen-Day) and 191 during nighttime (Aachen-Night).
Metrics. For Cambridge and 7-Scenes, we report the median translation error. For Aachen, we report the percentage of successfully localized images within three thresholds: (0.25m, 2°), (0.5m, 5°) and (5m, 10°).

4.2. Ablative study

We now ablate the main design, architectural and and training choices of our approach. We perform all ablations using a lower image resolution of 224×224 with a single retrieved image ($K = 1$). For each ablation table, we put a gray background color on the row with default settings.
Validation sets and metrics. We report the visual localization performance on a selected subset of 5 diverse and relatively challenging datasets: 7scenes-stairs, 7scenes-pumpkin, Cambridge-GreatCourt, Cambridge-OldHospital and Aachen-Night. For 7scenes and Cambridge-Landmarks, we report the averaged median translation error, while for Aachen-Night we report the localization accuracy for the 3 standard thresholds.
Robust coordinate encoding. We first study in Table 1 the impact of different point encoding schemes. Notably, we observe that direct coordinate regression is only successful when the train and test output distributions are aligned. This is the case for 7-scenes, or Cambridge to a lesser extent, as they are small and well-centered around the origin. For larger scenes with unconstrained coordinates (like Aachen), direct regression utterly fails. One way to mitigate this issue is to augment 3D coordinates at training time, *e.g.* using random translations (see Section 3.4). Augmentations somehow improve the situation for Aachen-Night, but the performance overall strongly degrades for Cambridge and 7scenes. In contrast, the cosine-based encoding ϕ proposed in Section 3.2 effectively deals with indoor and outdoor scenes in any coordinate ranges. We find optimal to use 6 frequencies, yielding $d = 36$-dimensional outputs.
Impact of CroCo pretraining. Table 2 shows that pretraining the ViT encoder and decoder with CroCo v2 [79] self-supervised objective is key to the success of our approach. Without CroCo pretraining, the performance significantly drops, which is explained by the fact that CroCo essentially learns to compare and implicitly match images, which is empirically verified in Section 4.3. We hypothesize that CroCo pretraining also ensures generalization, since the pretraining set (7M pairs) is much larger than our training dataset. Another illustration of this benefit is that the performance further improves when we *freeze* the ViT encoder during this training step, meaning that pretraining with CroCo effectively learns image representations already fit for our coordinate regression task.
Separate heads. We experiment with different architectures for the regression head, this time aiming at exploiting priors of the output space. Recall that for each pixel, we ultimately predict 4 values: 3 spatial components (x, y and z) and a confidence τ. A priori, these four components have no reason to be correlated. In fact, predicting them jointly could turn detrimental if there is a risk for the network to

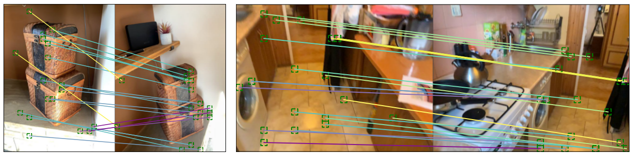

Figure 4. **Two example visualizations of the cross attention between query and reference (left and right image, resp.) images in the decoder.** We plot the top-20 cross-attention scores as red lines between 16x16 image patches of their corresponding tokens.

learn false correlations. Therefore, we compare: (i) as a baseline, a simple linear head, which is the same as 4 independent linear heads (one per component); (ii) regressing the 4 components jointly using the same head; (iii) regressing the spatial and confidence components separately; (iv) regressing all 4 components separately, in which case we still use the same prediction head with shared weight for all spatial x, y and z components after an independent linear projection. From Table 3, option (iv) clearly yields the best performance, while the linear heads is the worst option.

Image resolution can have a strong impact on the test performance. Table 4 shows that test performance generally increases as a function of image resolution. Interestingly, the model is able to generalize to higher resolution at test time, as training in 512×384 and testing on higher resolutions consistently yields better results. In the following, we always test on 640×480 images.

4.3. Visualization of internal attention

To better understand how the network is able to perform the coordinate regression task, we visualize in Figure 4 the highest cross-attention scores in the decoder, displayed as patch correspondences between corresponding tokens. Interestingly, we observe that the decoder implicitly performs image matching under the hood. In a sense, this is expected since to solve the task, the model has to essentially perform a matching-guided interpolation/extrapolation of the known reference coordinates to the query image. Note that it learns to implicitly perform matching without any explicit supervision for this task (*i.e.*, only from the regression signal).

4.4. Visual localization benchmarking

We compare our approach to the state of the art for visual localization on indoor (7-scenes) and outdoor datasets (Cambridge-Landmarks, Aachen-DayNight). We compare to learning-based approaches as well as a few representative keypoint-based methods such as Active Search [55] and HLoc [53]. Results are presented in Table 5, Table 6 and Figure 5, with *SACReg* and *SACReg-L* denoting the proposed method using a ViT-Base or ViT-Large encoder backbone respectively. On the indoor 7-Scenes dataset, our method obtains similar or slightly worse performance compared to other approaches, but overall still performs well with a median error of a few centimeters. On outdoor datasets, the proposed methods strongly outperforms other learning-based methods, in particular other scene-specific or scene-agnostic coordinate regression approaches like [6, 8, 9, 66, 67, 80, 82]. This is remarkable because, in contrast to any other learning-based approaches, SACReg is directly applied to each test set without any finetuning. In other words, our approach works out of the box on test data that were never seen during training. Interestingly, it even reaches the performance of keypoints-based approaches such as Active Search [55] or HLoc [53].

		Aachen-Day ↑	Aachen-Night ↑
Kpts	Active Search [55]	57.3 / 83.7 / 96.6	28.6 / 37.8 / 51.0
	HLoc [53]	**89.6 / 95.4** / 98.8	**86.7 / 93.9 / 100**
Learning-based	DSAC [6]	0.4 / 2.4 / 34.0	-
	ESAC (50 experts) [8]	42.6 / 59.6 / 75.5	-
	HSCNet [36]	65.5 / 77.3 / 88.8	22.4 / 38.8 / 54.1
	NeuMap [67]	76.2 / 88.5 / 95.5	37.8 / 62.2 / 87.8
	SACReg, $K = 20$	85.3 / 93.7 / **99.6**	64.9 / 90.1 / **100.0**
	SACReg-L, $K = 20$	85.8 / 95.0 / **99.6**	67.5 / 90.6 / **100.0**

Table 5. **Comparison to the state of the art on Aachen.**

		ShopFacade↓	OldHospital↓	College↓	Church↓	Court↓
Kpts	Active search [55]	0.12, 1.12	0.52, 1.12	0.57, 0.70	0.22, 0.62	1.20, 0.60
	HLoc [53]	**0.04, 0.20**	0.15, 0.3	0.12, 0.20	0.07, 0.21	0.11, 0.16
Learning-based	DSAC++ [7]	0.06, 0.3	0.20, 0.3	0.18, 0.3	0.13, 0.4	0.20, 0.4
	DSAC* [9]	0.05, 0.3	0.21, 0.4	0.15, 0.3	0.13, 0.4	0.49, 0.3
	KFNet [87]	0.05, 0.35	0.18, 0.28	0.16, 0.27	0.12, 0.35	0.42, 0.21
	HSCNet [36]	0.06, 0.3	0.19, 0.3	0.18, 0.3	0.09, 0.3	0.28, 0.2
	SANet [82]	0.1, 0.47	0.32, 0.53	0.32, 0.54	0.16, 0.57	3.28, 1.95
	DSM [66]	0.06, 0.3	0.23, 0.38	0.19, 0.35	0.11, 0.34	0.19, 0.43
	SC-wLS [80]	0.11, 0.7	0.42, 1.7	0.14, 0.6	0.39, 1.3	1.64, 0.9
	NeuMap [67]	0.06, 0.25	0.19, 0.36	0.14, 0.19	0.17, 0.53	**0.06**, 0.1
	SACReg, K=20	0.05, 0.29	**0.13**, **0.25**	0.13, **0.18**	**0.06**, 0.22	0.12, **0.08**
	SACReg-L, K=20	0.05, 0.28	**0.13**, **0.24**	**0.11**, **0.18**	**0.06**, **0.20**	0.13, **0.08**

Table 6. **Comparison to the state of the art on Cambridge** with the median translation (m) and angular (°) errors.

4.5. Database compression

One important limitation of the proposed method so far is the large volume of the pre-computed database image representations, if stored uncompressed. Indeed, considering an input resolution $HW \triangleq 640 \times 480$, and a ViT-Base architecture with a patch size of 16px, an encoded image $\mathcal{R}'_b$ requires 3.69MB of storage. Product quantization (Section 3.3) allows a significant storage reduction with negligible loss of performance. We use a codebook of 256 features per block, and vary the number of blocks for reaching different compression rates.

Results. During an offline phase, we compute, compress and store the representations of all database images. At test time, we reconstruct the full token features from the stored codebook indices and the corresponding codebooks. To alleviate the performance drop due to quantization, we slightly finetune the model for one additional epoch using compressed database features as inputs (considered as frozen), with a learning rate of 10^{-4} and a cosine-decay

		Chess ↓	Fire ↓	Heads ↓	Office ↓	Pumpkin ↓	Kitchen ↓	Stairs ↓
Kpts	Active search [55]	0.04, 1.96	0.03, 1.53	0.02, 1.45	0.09, 3.61	0.08, 3.10	0.07, 3.37	**0.03**, 2.22
	HLoc [53]	**0.02**, 0.79	**0.02**, 0.87	0.02, 0.92	**0.03**, 0.91	0.05, 1.12	0.04, 1.25	0.06, 1.62
Learning-based	RelocNet [2]	0.12, 4.14	0.26, 10.4	0.14, 10.5	0.18, 5.32	0.26, 4.17	0.23, 5.08	0.28, 7.53
	CamNet [18]	0.04, 1.73	0.03, 1.74	0.05, 1.98	0.04, 1.62	**0.04**, 1.64	0.04, 1.63	0.04, 1.51
	DSAC++ [7]	**0.02**, **0.5**	**0.02**, 0.9	**0.01**, **0.8**	**0.03**, 0.7	**0.04**, 1.1	0.04, **1.1**	0.09, 2.6
	KFNet [87]	**0.02**, 0.65	**0.02**, 0.9	**0.01**, 0.82	**0.03**, **0.69**	**0.04**, 1.02	0.04, 1.16	**0.03**, 0.94
	HSCNet [36]	**0.02**, 0.7	**0.02**, 0.9	**0.01**, 0.9	**0.03**, 0.8	**0.04**, **1.0**	0.04, 1.2	**0.03**, **0.8**
	SANet [82]	0.03, 0.88	0.03, 1.12	0.02, 1.48	**0.03**, 1.00	**0.04**, 1.21	0.04, 1.40	0.16, 4.59
	DSM [66]	**0.02**, 0.68	**0.02**, **0.80**	**0.01**, **0.8**	**0.03**, 0.78	**0.04**, 1.11	**0.03**, 1.11	0.04, 1.16
	SC-wLS [80]	0.03, 0.76	0.05, 1.09	0.03, 1.92	0.06, 0.86	0.08, 1.27	0.09, 1.43	0.12, 2.80
	NeuMaps [67]	**0.02**, 0.81	0.03, 1.11	0.02, 1.17	**0.03**, 0.98	**0.04**, 1.11	0.04, 1.33	0.04, 1.12
	SACReg, K=20	0.03, 0.94	0.03, 1.12	0.02, 1.08	0.04, 1.10	0.05, 1.38	0.05, 1.36	0.05, 1.44
	SACReg-L, K=20	0.03, 0.94	0.03, 1.03	0.02, 1.16	**0.03**, 1.06	0.05, 1.41	0.04, 1.35	0.06, 1.62

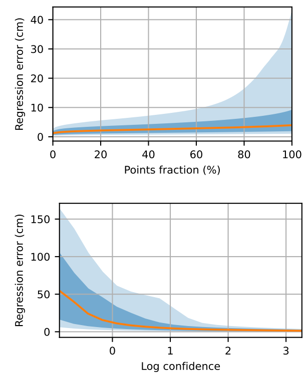

Figure 5. **Evaluations on 7-Scenes.** *Left: visual localization.* Comparison with the state of the art in terms of median translation (m) and angular (°) errors. *Right: SCR.* Distribution of coordinate prediction errors (first and last quartiles, deciles, and median) w.r.t. relative (top plot where we keep the $x\%$ most confident predicted points for each image) and absolute confidence (bottom plot where we show statistics for all points with a confidence above a given threshold). Errors are typically below 10cm, and correlate well with the predicted confidence.

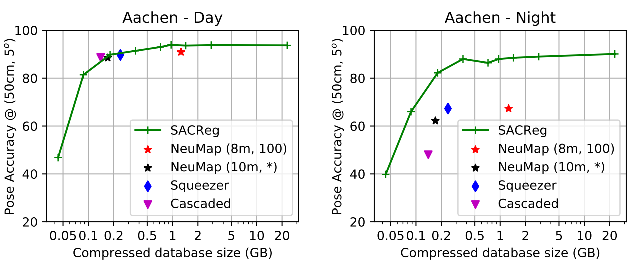

Figure 6. **Compression experiments.** We report the localization performance on Aachen as a function of the database storage size.

Method	Size ↓	Aachen-Day ↑	Aachen-Night ↑
SACReg (3.69MB/img)	30.16	**85.3 / 93.7 / 99.6**	**64.9 / 90.1 / 100.0**
NeuMap (8m, 100) [67]	1.26	80.8 / 90.9 / 95.6	48.0 / 67.3 / 87.8
SACReg+PQ (154kB/img)	0.96	**85.3 / 93.9 / 99.6**	**62.8 / 88.0 / 100.0**
SACReg+PQ (58kB/img)	0.36	**81.1 / 91.4 / 99.5**	**59.7 / 88.0 / 100.0**
Squeezer [83]	0.24	75.5 / 89.7 / 96.2	50.0 / 67.3 / 78.6
SACReg+PQ (29kB/img)	0.18	76.6 / **89.8 / 98.9**	**53.9 / 82.2 / 100.0**
NeuMap (10m, *) [67]	0.17	76.2 / 88.5 / 95.5	37.8 / 62.2 / 87.8
Cascaded [13]	0.14	**76.7 / 88.6** / 95.8	33.7 / 48.0 / 62.2
SACReg+PQ (14kB/img)	0.09	**61.8 / 81.4 / 98.2**	**41.9 / 66.0 / 98.4**

Table 7. **Results with compression compared to the state of the art on Aachen Day-Night.** The 'Size' column represents the compressed dataset size in gigabytes. We highlight in **bold** optimal values lying on an accuracy-versus-compression Pareto front.

scheduler. This step is still scene-agnostic and is performed once for all. In Figure 6, we report the performance on Aachen while varying the number of blocks of PQ quantization. We observe that the performance remains similar with a compression factor up to 32, *i.e.*, effectively reducing the database storage size from about 30GB (3.69MB/img) to 0.96GB (154kB/img). Beyond this point, the performance gracefully degrades, such that for a compression factor of 128, our method is still able to obtain more than 80% accuracy at 50cm&5° on Aachen-Night.

Comparison with the state of the art. In Figure 6 and Table 7, we compare our approach on the Aachen dataset with other scene-compression methods such as NeuMap [67], which directly regress the 3D coordinates of a given set of 2D keypoints using learned neural codes, and other scene compression methods such as Cascaded [13] and Squeezer [83], which are based on feature matching. Our approach achieves similar or better results compared to all other methods under similar compression ratios. Additionally, it is noteworthy to point out that, unlike NeuMap, we did not train our model on the Aachen dataset at all.

4.6. Scene coordinates regression

Lastly, to evaluate the regression performances of SACReg, we apply our model on *7-Scenes*, which provides dense ground-truth annotations. Using a shortlist size of $K = 8$, we predict the 3D coordinates and corresponding confidence for each pixel of the test images. We obtain a median and mean error of 4.2cm and 13.2cm respectively. Results furthermore validate that confidence predictions are meaningful, as errors tends to get smaller when the confidence increases (Figure 5, top right). Confidence can thus be used as a proxy to filter out regions where errors are likely to be large (Figure 5, bottom right, and black regions in Figure 3).

5. Conclusion

We introduce a novel paradigm for Scene Coordinates Regression with a model predicting pixelwise coordinates for a query image based on database images with sparse 2D-3D correspondences. Our single model can be applied for visual localization in novel scenes of arbitrary scale without re-training, and outperforms other learning-based approaches that are trained for a single or a few small specific scenes. Its database representations can be pre-computed offline for greater efficiency, and we furthermore show they can be highly compressed with negligible loss of visual localization performance.

References

[1] Map-free Visual Relocalization: Metric Pose Relative to a Single Image. In *ECCV*, 2022. 2

[2] Vassileios Balntas, Shuda Li, and Victor Prisacariu. RelocNet: Continuous Metric Learning Relocalisation Using Neural Nets. In *ECCV*, 2018. 2, 8

[3] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, and Elad Shulman. ARKitscenes - a diverse real-world dataset for 3d indoor scene understanding using mobile RGB-d data. In *NeurIPS*, 2021. 2, 5

[4] Hunter Blanton, Connor Greenwell, Scott Workman, and Nathan Jacobs. Extending absolute pose regression to multiple scenes. In *CVPRW*, 2020. 2

[5] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6D Object Pose Estimation Using 3D Object Coordinates. In *ECCV*, 2014. 1

[6] 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, 7

[7] Eric Brachmann and Carsten Rother. Learning Less is More - 6D Camera Localization via 3D Surface Regression. In *CVPR*, 2018. 7, 8

[8] Eric Brachmann and Carsten Rother. Expert Sample Consensus Applied to Camera Re-Localization. In *ICCV*, 2019. 1, 7

[9] Eric Brachmann and Carsten Rother. Visual camera relocalization from RGB and RGB-D images using DSAC. *IEEE Trans. PAMI*, 2021. 1, 2, 7

[10] Samarth Brahmbhatt, Jinwei Gu, Kihwan Kim, James Hays, and Jan Kautz. Geometry-Aware Learning of Maps for Camera Localization. In *CVPR*, 2018. 2

[11] Federico Camposeco, Andrea Cohen, Marc Pollefeys, and Torsten Sattler. Hybrid scene compression for visual localization. In *CVPR*, 2019. 3

[12] Song Cao and Noah Snavely. Minimal scene descriptions from structure from motion models. In *CVPR*, 2014. 3

[13] Wentao Cheng, Weisi Lin, Kan Chen, and Xinfeng Zhang. Cascaded parallel filtering for memory-efficient image-based localization. In *ICCV*, 2019. 3, 8

[14] Wentao Cheng, Weisi Lin, Xinfeng Zhang, Michael Goesele, and Ming-Ting Sun. A data-driven point cloud simplification framework for city-scale image-based localization. *IEEE Trans. Image Processing*, 2017. 3

[15] Xinjing Cheng, Peng Wang, Chenye Guan, and Ruigang Yang. Cspn++: Learning context and resource aware convolutional spatial propagation networks for depth completion. In *AAAI*, 2020. 1

[16] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *CVPR*, 2017. 5

[17] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPRW*, 2018. 3

[18] Mingyu Ding, Zhe Wang, Jiankai Sun, Jianping Shi, and Ping Luo. CamNet: Coarse-to-Fine Retrieval for Camera Re-Localization. In *ICCV*, 2019. 2, 8

[19] Siyan Dong, Shuzhe Wang, Yixin Zhuang, Juho Kannala, Marc Pollefeys, and Baoquan Chen. Visual localization via few-shot scene region classification. In *3DV*, 2022. 2

[20] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2, 3

[21] 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. 3

[22] Marcin Dymczyk, Simon Lynen, Michael Bosse, and Roland Siegwart. Keep it brief: Scalable creation of compressed localization maps. In *IROS*, 2015. 3

[23] 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*, 1981. 3

[24] Robert M. Gray and David L. Neuhoff. Quantization. *IEEE Trans. Inf. Theory*, 1998. 5

[25] Abner Guzman-Rivera, Pushmeet Kohli, Ben Glocker, Jamie Shotton, Toby Sharp, Andrew Fitzgibbon, and Shahram Izadi. Multi-output learning for camera relocalization. In *CVPR*, 2014. 2

[26] Jared Heinly, Johannes L. Schönberger, 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

[27] Mu Hu, Shuling Wang, Bin Li, Shiyu Ning, Li Fan, and Xiaojin Gong. Penet: Towards precise and efficient image guided depth completion. In *ICRA*, 2021. 1

[28] Zhaoyang Huang, Han Zhou, Yijin Li, Bangbang Yang, Yan Xu, Xiaowei Zhou, Hujun Bao, Guofeng Zhang, and Hongsheng Li. VS-Net: Voting with segmentation for visual localization. In *CVPR*, 2021. 2

[29] Martin Humenberger, Yohann Cabon, Nicolas Guerin, Julien Morat, Jérôme Revaud, Philippe Rerole, Noé Pion, Cesar de Souza, Vincent Leroy, and Gabriela Csurka. Robust image retrieval-based visual localization using kapture. *arXiv preprint arXiv:2007.13867*, 2020. 2, 5

[30] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Trans. PAMI*, 2011. 2, 5

[31] Alex Kendall and Roberto Cipolla. Geometric Loss Functions for Camera Pose Regression with Deep Learning. In *CVPR*, 2017. 2

[32] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, 2018. 5

[33] Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: a Convolutional Network for Real-Time 6-DOF Camera Relocalization. In *ICCV*, 2015. 2, 6

[34] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o(n) solution to the pnp problem. *IJCV*, 2009. 1

[35] Kunhong Li, Longguang Wang, Li Liu, Qing Ran, Kai Xu, and Yulan Guo. Decoupling makes weakly supervised local feature better. In *CVPR*, 2022. 3

[36] Xiaotian Li, Shuzhe Wang, Yi Zhao, Jakob Verbeek, and Juho Kannala. Hierarchical scene coordinate classification and regression for visual localization. In *CVPR*, 2020. 2, 7, 8

[37] Yunpeng Li, Noah Snavely, and Daniel P. Huttenlocher. Location recognition using prioritized feature matching. In *ECCV*, 2010. 3

[38] Zhengqi Li and Noah Snavely. MegaDepth: Learning Single-View Depth Prediction from Internet Photos. In *CVPR*, 2018. 2, 5

[39] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *CVPR*, 2022. 4

[40] David G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 3, 5

[41] Zixin Luo, Tianwei Shen, Lei Zhou, Siyu Zhu, Runze Zhang, Yao Yao, Tian Fang, and Long Quan. Geodesc: Learning local descriptors by integrating geometry constraints. In *ECCV*, 2018.

[42] Zixin Luo, Lei Zhou, Xuyang Bai, Hongkai Chen, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. Aslfeat: Learning local features of accurate shape and localization. In *CVPR*, 2020. 3

[43] Marcela Mera-Trujillo, Benjamin Smith, and Victor Fragoso. Efficient scene compression for visual-based localization. In *3DV*, 2020. 3

[44] Danish Nazir, Alain Pagani, Marcus Liwicki, Didier Stricker, and Muhammad Zeshan Afzal. Semattnet: Toward attention-based semantic aware guided depth completion. *IEEE Access*, 2022. 1

[45] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. Lf-net: Learning local features from images. In *NeurIPS*, 2018. 3

[46] Hyun Soo Park, Yu Wang, Eriko Nurvitadhi, James C. Hoe, Yaser Sheikh, and Mei Chen. 3d point cloud reduction using mixed-integer quadratic programming. In *CVPR Workshop*, 2013. 3

[47] Jinsun Park, Kyungdon Joo, Zhe Hu, Chi-Kuei Liu, and In So Kweon. Non-local spatial propagation network for depth completion. In *ECCV*, 2020. 1

[48] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-Matterport 3D Dataset (HM3D): 1000 Large-scale 3D Environments for Embodied AI. In *NeurIPS datasets and benchmarks*, 2021. 5

[49] Jerome Revaud, Jon Almazán, Rafael S Rezende, and Cesar Roberto de Souza. Learning with average precision: Training image retrieval with a listwise loss. In *ICCV*, 2019. 2, 5

[50] Jerome Revaud, Cesar De Souza, Martin Humenberger, and Philippe Weinzaepfel. R2D2: Reliable and repeatable detector and descriptor. In *NeurIPS*, 2019. 3

[51] Jérôme Revaud, Guillaume Lavoué, and Atilla Baskurt. Improving zernike moments comparison for optimal similarity and rotation angle retrieval. *IEEE trans. PAMI*, 2008. 4

[52] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, 2011. 3

[53] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *CVPR*, 2019. 5, 7, 8

[54] Torsten Sattler, Michal Havlena, Filip Radenovic, Konrad Schindler, and Marc Pollefeys. Hyperpoints and fine vocabularies for large-scale location recognition. In *ICCV*, 2015. 3

[55] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE trans. PAMI*, 2017. 5, 7, 8

[56] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, et al. Benchmarking 6dof outdoor visual localization in changing conditions. In *CVPR*, 2018. 2, 6

[57] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *ICCV*, 2019. 2, 5

[58] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion Revisited. In *CVPR*, 2016. 2, 3, 5

[59] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016. 4

[60] 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. 1, 2

[61] 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. 6

[62] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Modeling the World from Internet Photo Collections. *IJCV*, 2008. 2

[63] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 5

[64] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021. 3

[65] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan,

Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training Home Assistants to Rearrange their Habitat. In *NeurIPS*, 2021. 5

[66] Shitao Tang, Chengzhou Tang, Rui Huang, Siyu Zhu, and Ping Tan. Learning Camera Localization via Dense Scene Matching. In *CVPR*, 2021. 1, 2, 7, 8

[67] Shitao Tang, Sicong Tang, Andrea Tagliasacchi, Ping Tan, and Yasutaka Furukawa. Neumap: Neural coordinate mapping by auto-transdecoder for camera localization. In *CVPR*, 2023. 1, 3, 7, 8

[68] George Terzakis and Manolis Lourakis. A consistently fast and globally optimal solution to the perspective-n-point problem. In *ECCV*, 2020. 5

[69] 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. 3

[70] Giorgos Tolias, Tomas Jenicek, and Ondřej Chum. Learning and aggregating deep local descriptors for instance-level recognition. In *ECCV*, 2020. 5

[71] Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. Disk: Learning local features with policy gradient. In *NeurIPS*, 2020. 3

[72] Julien Valentin, Matthias Nießner, Jamie Shotton, Andrew Fitzgibbon, Shahram Izadi, and Philip Torr. Exploiting uncertainty in regression forests for accurate camera relocalization. In *CVPR*, 2015. 2

[73] Wouter Van Gansbeke, Davy Neven, Bert De Brabandere, and Luc Van Gool. Sparse and noisy lidar completion with RGB guidance and uncertainty. In *ICMVA*, 2019. 1

[74] Florian Walch, Caner Hazirbas, Laura Leal-Taixé, Torsten Sattler, Sebastian Hilsenbeck, and Daniel Cremers. Image-based localization with spatial lstms. In *ICCV*, 2017. 2

[75] Bing Wang, Changhao Chen, Chris Xiaoxuan Lu, Peijun Zhao, Niki Trigoni, and Andrew Markham. AtLoc: Attention Guided Camera Localization. In *AAAI*, 2020. 2

[76] Qianqian Wang, Xiaowei Zhou, Bharath Hariharan, and Noah Snavely. Learning feature descriptors using camera pose supervision. In *ECCV*, 2020. 3

[77] Philippe Weinzaepfel, Vincent Leroy, Thomas Lucas, Romain Brégier, Yohann Cabon, Vaibhav Arora, Leonid Antsfeld, Boris Chidlovskii, Gabriela Csurka, and Jérôme Revaud. CroCo: Self-Supervised Pre-training for 3D Vision Tasks by Cross-View Completion. In *NeurIPS*, 2022. 2

[78] Philippe Weinzaepfel, Thomas Lucas, Diane Larlus, and Yannis Kalantidis. Learning super-features for image retrieval. In *ICLR*, 2022. 2, 5

[79] Philippe Weinzaepfel, Thomas Lucas, Vincent Leroy, Yohann Cabon, Vaibhav Arora, Romain Brégier, Gabriela Csurka, Leonid Antsfeld, Boris Chidlovskii, and Jérôme Revaud. Croco v2: Improved cross-view completion pre-training for stereo matching and optical flow. In *ICCV*, 2023. 2, 3, 5, 6

[80] Xin Wu, Hao Zhao, Shunkai Li, Yingdian Cao, and Hongbin Zha. Sc-wls: Towards interpretable feed-forward camera relocalization. In *ECCV*, 2022. 1, 2, 7, 8

[81] Tao Xie, Kun Dai, Ke Wang, Ruifeng Li, and Lijun Zhao. Deepmatcher: A deep transformer-based network for robust and accurate local feature matching. *arXiv preprint arXiv:2301.02993*, 2023. 3

[82] Luwei Yang, Ziqian Bai, Chengzhou Tang, Honghua Li, Yasutaka Furukawa, and Ping Tan. SANet: Scene Agnostic Network for Camera Localization. In *ICCV*, 2019. 1, 2, 7, 8

[83] Luwei Yang, Rakesh Shrestha, Wenbo Li, Shuaicheng Liu, Guofeng Zhang, Zhaopeng Cui, and Ping Tan. Scenesqueezer: Learning to compress scene for camera relocalization. In *CVPR*, 2022. 3, 8

[84] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. Lift: Learned invariant feature transform. In *ECCV*, 2016. 3

[85] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. DPOD: 6D Pose Object Detector and Refiner. In *ICCV*, 2019. 1

[86] Zichao Zhang, Torsten Sattler, and Davide Scaramuzza. Reference Pose Generation for Long-term Visual Localization via Learned Features and View Synthesis. *IJCV*, 2020. 6

[87] Lei Zhou, Zixin Luo, Tianwei Shen, Jiahui Zhang, Mingmin Zhen, Yao Yao, Tian Fang, and Long Quan. Kfnet: Learning temporal camera relocalization using kalman filtering. In *CVPR*, 2020. 2, 7, 8

[88] Qunjie Zhou, Torsten Sattler, Marc Pollefeys, and Laura Leal-Taixe. To Learn or Not to Learn: Visual Localization from Essential Matrices. In *ICRA*, 2020. 2