Map-Relative Pose Regression for Visual Re-Localization

Shuai Chen[1,2] Tommaso Cavallari[1] Victor Adrian Prisacariu[1,2] Eric Brachmann[1]
[1]Niantic [2]University of Oxford

2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) | 979-8-3503-5300-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/CVPR52733.2024.01953

Abstract

*Pose regression networks predict the camera pose of a query image relative to a known environment. Within this family of methods, absolute pose regression (APR) has recently shown promising accuracy in the range of a few centimeters in position error. APR networks encode the scene geometry implicitly in their weights. To achieve high accuracy, they require vast amounts of training data that, realistically, can only be created using novel view synthesis in a days-long process. This process has to be repeated for each new scene again and again. We present a new approach to pose regression, map-relative pose regression (**marepo**), that satisfies the data hunger of the pose regression network in a scene-agnostic fashion. We condition the pose regressor on a scene-specific map representation such that its pose predictions are relative to the scene map. This allows us to train the pose regressor across hundreds of scenes to learn the generic relation between a scene-specific map representation and the camera pose. Our map-relative pose regressor can be applied to new map representations immediately or after mere minutes of fine-tuning for the highest accuracy. Our approach outperforms previous pose regression methods by far on two public datasets, indoor and outdoor. Code is available:* https://nianticlabs.github.io/marepo.

1. Introduction

Today, neural networks have conquered virtually all sectors of computer vision, but there is still at least one task that they struggle with: visual relocalization. What is visual relocalization? Given a set of mapping images and their poses, expressed in a common coordinate system, build a scene representation. Later, given a query image, estimate its pose, i.e. position and orientation, relative to the scene.

Successful approaches to visual relocalization rely on predicting image-to-scene correspondences, either via matching [8, 21, 38–40, 42, 57] or direct regression [4–6, 14, 56], then solving for the pose using traditional and robust algorithms like PnP [18] and RANSAC [17].

Adopting a different perspective, approaches based on pose regression [12, 25, 32, 46] attempt to perform visual relocalization without resorting to traditional pose solving, by using a single feed-forward neural network to infer poses from single images. The mapping data is treated as a training set where the camera extrinsics serve as supervision. Generally, pose regression approaches come in two flavors, but they both struggle with accuracy compared to correspondence-based methods.

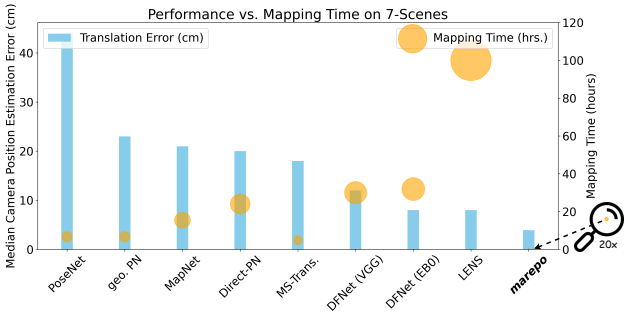

Figure 1. **Camera pose estimation performance vs. mapping time.** The figure shows the median translation error of several pose regression relocalization methods on the 7-Scenes dataset and the time required (proportional to the bubble size) to train each relocalizer on the target scenes. Our proposed approach, ***marepo***, achieves superior performance – by far – on both metrics, thanks to its integration of scene-specific geometric map priors within an accurate, map-relative, pose regression framework.

Absolute pose regression (APR) methods [7, 24, 25] involve training a dedicated pose regressor for each individual scene, enabling the prediction of camera poses to that particular scene. Though the scene coordinate space can be implicitly encoded in the weights of the neural networks, absolute pose regressors exhibit low pose estimation accuracy, primarily due to the often limited training data available for each scene, and struggle to generalize to unseen views [43].

Relative pose regression is a second flavor of pose regression methods [10, 16, 26, 51, 54]. The regressor is trained to predict the relative pose between two images. In a typical inference scenario, the regressor is applied to a pair formed by an unseen query and an image from the mapping set (typically selected via a nearest neighbor-type match-

2575-7075/24/$31.00 ©2024 IEEE
DOI 10.1109/CVPR52733.2024.01953

ing); then, the predicted relative pose can be combined with the known pose of the mapping image to yield the absolute query pose. These methods can be trained on a lot of scene-agnostic data, but their accuracy is still limited: a metric pose between two images can only be predicted approximately [2].

Motivated by those limitations, we propose a new flavor of absolute pose regression: map-relative pose regression (***marepo***). We couple a scene-specific representation – encoding the scale-metric reference space of each target scene – with a general, scene-agnostic, absolute pose regression network. In particular, we utilize a fast-training scene coordinate regression model as our scene representation and train, once and ahead of time, a pose regression network that learns the relationship between a scene coordinate prediction and the corresponding camera pose. This generic relationship allows us to train the pose regressor on hundreds of different scenes, effectively solving the issue of the limited availability of training data afflicting absolute pose regression models. On the other hand, since at localization time our pose regressor is conditioned on a scene-specific map representation, it is able to predict accurate scale-metric poses, unlike relative pose regressors.

Our experiments show that ***marepo*** is a pose regression network with an accuracy on par with structure-based relocalization methods (e.g. [6]), exceeding the accuracy of all other single-frame absolute pose regression methods by far (see Fig. 1). Our scene-agnostic pose regressor can be applied to each new scene representation right away, or (optionally) fine-tuned in just a few minutes for best accuracy. We summarize our main contributions as follows:

1. We propose ***marepo***, a novel Absolute Pose Regression approach that combines a generic and scene-agnostic Map-Relative Pose regression method with a scene-specific metric representation. We show that the network can perform end-to-end inference on previously unseen images and, thanks to the strong and explicit 3D geometric knowledge encoded by the scene-specific component, it can directly estimate accurate, absolute, metric poses.
2. We introduce a transformer-based network architecture that can process a dense set of correspondences between 2D locations in a query image and their corresponding 3D coordinates within the reference system of a previously mapped scene, and estimate the pose of the camera that captured the query image. We further show how a dynamic position encoding applied to the 2D locations in the query image can significantly improve the performance of the method by encoding the intrinsic camera parameters within the transformer input.

2. Related Works

Over the years many efforts in the literature have tackled the problem of visual relocalization, and we have seen a rough demarcation of the types of approach into two main fields: the more traditional approaches, relying on geometric concepts and the estimation of correspondences between images and maps; and the more recent "direct" approaches, relying on neural networks to predict the absolute position and orientation of the image without an intermediate, explicit, matching step linking the 2D image realm with a 3D map of the scene. In the remainder of this section, we briefly explore the main approaches in each of these categories.

2.1. Geometry-based Visual Relocalization

Geometry-based approaches rely on estimating correspondences between pixels in the query images and points in the scene's map. These correspondences effectively establish a 2D-to-3D matching that can be exploited by pose-solving methods such as PnP/RANSAC [17, 18] to compute the pose of the camera at the moment it captures the image.

There are several ways to estimate those correspondences: from classic computer vision approaches using off-the-shelf feature detectors and descriptors to compute matches between image pixels and a database of previously observed 3D points [8, 21, 41, 42]; to more advanced, neural-based approaches that rely on learned descriptors, improved matchers, and different map representations in order to estimate better correspondences from more challenging images or viewpoints [29, 35, 38, 40]. These approaches leverage the underlying geometric principles governing image formation and capture, yielding accurate pose estimations with low errors, often in the order of a few centimeters. However, they are not without a drawback: they generally require the creation of a map (e.g., in the form of a 3D point cloud created via Structure-from-Motion) of the scene ahead of time in order to associate the descriptors to 3D coordinates, and that is typically time-consuming.

In recent years, a new approach to geometry-based relocalization started to become prominent: scene coordinate regression (SCR). In this scenario, the map of the scene is directly encoded in a fixed-size set of weights of a neural network. At localization time, the query image is passed through the network, yielding per-pixel scene coordinates that can be directly used by a pose solver to estimate the camera pose [3–6, 14, 27, 56]. While effective, these methods have typically required training a new network for every new target scene, potentially taking several hours [4], thus hindering their large scale application. Recently, an approach to scene coordinate regression that can take mere minutes to be trained for every scene was presented in [6], making practical deployment of SCR networks a possibility. As the correspondence-based methods mentioned above, coordinate regression approaches are also very accurate by relying on geometric information on the structure of the scene. Nevertheless, scene coordinate regression methods still require an explicit stage where a pose solver has to

process each correspondence generated by the method to estimate the camera pose. Conversely, Absolute Pose Regression methods do not have this requirement since the regressor network can go directly from image to pose in an end-to-end fashion.

2.2. Absolute Pose Regression

Absolute Pose Regression (APR) approaches have also garnered notable attention recently, primarily due to their simplicity and efficiency. These methods directly predict camera poses via end-to-end neural networks. Kendall et al. introduced the first APR approach, named PoseNet [23–25], where a feed-forward neural network directly regresses a 7-dimensional pose vector for every query image. Successive works explore diverse architectural designs such as hourglass networks [30], bifurcated translation and rotation regression [34, 55], attention layers [45–47, 53], and LSTM layers [52]. Other research efforts attempt to improve APR performance with different supervisions, such as a geometric loss [24], relative pose constraints [7], uncertainty awareness [24, 33], or a sequential formulation like temporal filtering [13] and multitasking [37]. Despite these advancements, the accuracy of single-frame-based pose regression remains limited when compared to alternative approaches, such as those based on geometric principles.

Among recent advancements within APR, a promising direction is incorporating novel view synthesis techniques, either by synthesizing large amounts of training data to solve overfitting issues [12, 32, 36] or by integrating them into a fine-tuning process before test time [7, 11, 12]. One drawback of the former is that generating high-quality synthetic data can be a time-intensive process; as for the latter, in addition to time requirements, those approaches typically require extra data from the scene of interest. These limitations pose significant constraints in environments subject to rapid changes, such as those with frequent alterations in furnishings or appearances.

This paper introduces a new category of approach to the pose regression domain, one that tops the need for extensive mapping time, reducing it to mere minutes per scene. It demonstrates enhanced accuracy over previous single-frame APR methods and exhibits rapid scalability to new environments, making it flexible to deploy in a fast-changing world as we live in today.

3. Method

Prevalent pose regression methods are built on top of end-to-end neural network-based approaches. They can be formulated as $\hat{P} = \mathcal{F}(I)$: the camera poses $\hat{P}$ are directly predicted by providing an input image I to the network $\mathcal{F}$. A benefit of this type of approach is its conceptual simplicity and highly efficient inference speed. However, the forward process of typical APR networks – which rely on 2D operations over images and features – does not exploit any 3D geometric reasoning, resulting in insufficient performance compared to state-of-the-art geometry-based methods. In this paper, we propose a first map-relative pose regression approach empowered with explicit 3D geometric reasoning within its formulation, allowing us to regress accurate camera poses while maintaining real-time efficiency and end-to-end simplicity like any other pose regression method.

In the remainder of this section we first give an overview of the transformer-based network architecture we deploy to perform pose regression (Sec. 3.1); then we describe the main components and ideas behind the proposed approach (Sec. 3.2); the loss function optimized during training (Sec. 3.3); and, finally, we show how the scene-agnostic pose-regression transformer can be *optionally* fine-tuned for a specific testing scene in a matter of minutes, thus improving the performance of the method even further compared to a non-fine-tuned regressor (Sec. 3.4).

3.1. Architecture Overview

The main architecture of our method is formed of two components: (1) a CNN-based scene geometry prediction network $\mathcal{G}$ that maps pixels from the input image to 3D scene coordinates; and (2) a transformer-based map-relative pose regressor $\mathcal{M}$ that, given the scene coordinates, estimates the camera poses. Ideally, the network $\mathcal{G}$ is designed to associate each input image to scene-specific 3D information, thus requiring some training process for every new scene processed by the method. Conversely, the map-relative pose regressor $\mathcal{M}$ is a scene-agnostic module trained with large amounts of data and can generalize to unseen maps.

We illustrate our proposed network architecture in Fig. 2. Given an image I from scene S, we pass it to our model which outputs a pose $\hat{P}$. The process is formulated as:

$$\hat{P} = \mathcal{M}(\hat{H}, K) = \mathcal{M}(\mathcal{G}_S(I), K), \tag{1}$$

where $\hat{H} = \mathcal{G}_S(I)$ indicates the image-to-scene coordinates predicted by $\mathcal{G}$ (which was trained ad-hoc for scene S), and $K \in \mathbb{R}^{3\times3}$ is the camera intrinsic matrix associated with the input image. This formulation makes the approach similar to both standard Absolute Pose Regression, in that it generates poses via a feed-forward pass through a neural network, as well as Scene Coordinate Regression since the scene geometry prediction network regresses 3D coordinates directly from each input image. Unlike standard APR, our method has full geometric reasoning on the link between the image and the scene, and, unlike SCR approaches, it does not require a traditional, non-deterministic RANSAC stage to infer the pose. Theoretically, any algorithm capable of predicting 3D scene coordinates from an input image could be a viable candidate as $\mathcal{G}$, since the following transformer we deploy to perform pose regression ($\mathcal{M}$) does not depend on the prior component.

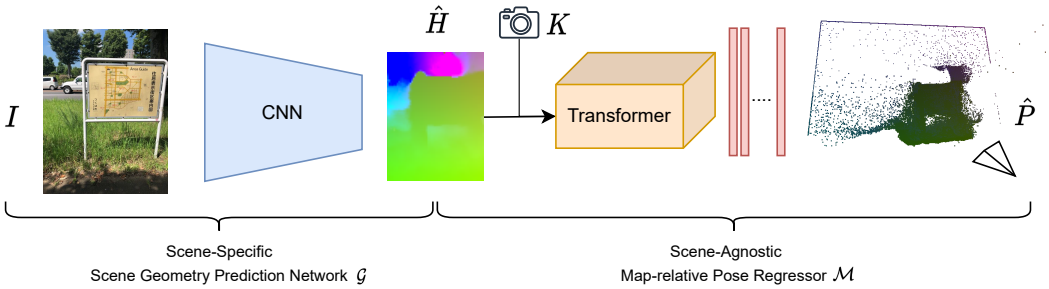

Figure 2. Illustration of the ***marepo*** network. A scene-specific geometry prediction module $\mathcal{G}_S$ processes a query image to predict a scene coordinate map $\hat{H}$. Then, a scene-agnostic map-relative pose regressor $\mathcal{M}$ is used to directly regress the camera pose. Our network's training and inference rely solely on RGB images I and camera intrinsics K without requiring depth information or pre-built point clouds.

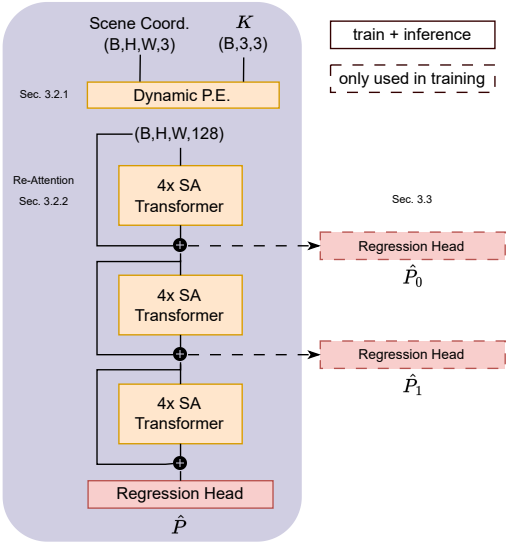

Figure 3. The map-relative pose regressor $\mathcal{M}$ takes as input a tensor of predicted scene coordinate maps and the corresponding camera intrinsics, embeds the information with dynamic positional encoding into higher dimensional features, and finally estimates the camera poses $\hat{P}$. During training, we also predict $\hat{P}_0$ and $\hat{P}_1$ for intermediate supervision.

In the next section, we will focus on detailing the map-relative pose regression network $\mathcal{M}$.

3.2. Map-Relative Pose Regression Architecture

To achieve a robust and scene-agnostic map-relative pose regression, we carefully design the simple yet effective architecture depicted in Fig. 3. The main components of this module are: (a) a novel dynamic positional encoding used to increase the dimensionality of the input scene coordinates – as well as embed their spatial location within the input image – taking into account the intrinsic properties of the camera that captured the frame; (b) several multi-layer self-attention transformer blocks; and finally (c), an MLP-based pose regression head. Given scene coordinate maps $\hat{H}$ (predicted by $\mathcal{G}_S$) and the corresponding camera intrinsic matrices K, the network is able to directly estimate 6-DoF (six degrees of freedom) metric camera poses. We detail the designs of each component in the following sub-sections.

3.2.1 Dynamic Positional Encoding

Unlike many vision transformers (ViTs) for high-level tasks [9, 15, 50], where the transformer is conditioned to operate directly upon input RGB images (or higher-dimensional features), our transformer is designed to interpret accurate 3D geometric information strongly connected to real-world physics. The content a camera captures in its frame is strictly associated with its intrinsic parameters; thus, we propose to use a positional encoding that is conditioned on each individual sensor, allowing us to train the main transformer blocks in a fashion that is generic, i.e., independent of the camera calibration parameters.

Our positional encoding scheme entails the fusion of two different components: (1) a camera-aware 2D positional embedding, associating each predicted scene coordinate to its corresponding pixel location; and (2) a 3D positional embedding that embeds the actual 3D scene coordinate values into a high-frequency domain.

Camera-Aware 2D Positional Embedding We draw inspiration from LoFTR's [50] positional embedding, but integrate information from the camera's intrinsics to generate the high-frequency components that are fed to the network.

Specifically, for each pixel coordinate (u, v) in the input image, we first compute the (x, y) components of the 3D ray originating in the camera center and passing through the pixel (ignoring the z component); then apply the positional embedding from [50] on the (now camera-invariant) ray's directional components. This generates a high-frequency/high-dimensionality embedding, allowing the transformer to correlate the input 3D coordinates (predicted by $\mathcal{G}_S$ and defined in a scene-specific coordinate system) with 3D rays originating from the current camera position, helping with the task of regressing the current camera

pose w.r.t. the origin of the scene coordinate system. Formally, we define the Camera-Aware 2D Positional Embedding as follows:

$$\mathcal{PE}^{i}_{2D(u,v)} := \begin{cases} \sin\left(\omega_k \cdot X_{\text{ray}}(u)\right), & i = 4k \\ \cos\left(\omega_k \cdot X_{\text{ray}}(u)\right), & i = 4k+1 \\ \sin\left(\omega_k \cdot Y_{\text{ray}}(v)\right), & i = 4k+2 \\ \cos\left(\omega_k \cdot Y_{\text{ray}}(v)\right), & i = 4k+3 \end{cases}, \quad (2)$$

where $\omega_k = \frac{1}{10000^{2k/d}}$ is the frequency band defined for d-dimensional features in which positional encoding is applied on, i is the current feature index, and X_{ray} and Y_{ray} are the X and Y components of the rays passing through (u, v):

$$\begin{aligned} X_{\text{ray}}(u) &= \lambda \frac{u - c_x - \varepsilon}{f_x}, \\ Y_{\text{ray}}(v) &= \lambda \frac{v - c_y - \varepsilon}{f_y}, \end{aligned} \quad (3)$$

with $f_{x/y}$ and $c_{x/y}$ corresponding to the intrinsics of the input frame, $\varepsilon = 0.5$ to achieve zero-mean in the center of the image, and $\lambda = 400$ chosen as a heuristic constant to keep a reasonable numerical magnitude for the final embedding.

3D Positional Embedding We use a 3D positional embedding to map the scene coordinates $p \in \mathbb{R}^3$ predicted by $\mathcal{G}_S$ to high frequency/dimensionality, inspired by [31]:

$$\begin{aligned} \mathcal{PE}_{3D}(p) = \text{Conv}^{d}_{3(2m+1)}[p, \sin\left(2^0\pi p\right), \cos\left(2^0\pi p\right), \\ \ldots, \sin\left(2^{m-1}\pi p\right), \cos\left(2^{m-1}\pi p\right)]. \end{aligned} \quad (4)$$

Here, in addition to the sinusoidal embedding mapping the 3D coordinates to a $3(2m+1)$-dimensional space, we also apply a further 1×1 convolution Conv^{d}_{6m+3} to ensure both $\mathcal{PE}_{2D}$ and $\mathcal{PE}_{3D}$ have the same number of channels.

Fused Positional Embedding Finally, we fuse the 2D and 3D embeddings before passing them to the transformer:

$$\mathcal{PE}_f = \mathcal{PE}_{3D} + \mathcal{PE}_{2D}. \quad (5)$$

3.2.2 Re-Attention for Deep Transformer

As illustrated in the left part of Fig. 3, the core of our map-relative pose regression architecture is formed by twelve self-attention transformers arranged over three blocks of four transformers each. In our implementation, we use Linear Transformers [22] as they reduce the computation complexity of each layer from quadratic to linear in the length of the input (i.e., the resolution of the scene coordinate map).

Since the Dynamic Positional Encoding is fed to the network only at the beginning, we found that the information flow became weaker as the depth of the network increases. To solve this problem, we add what we call a "Re-Attention" mechanism, introducing residual connections every four blocks. Experimentally, we find that this practice is quite effective, allowing the network to converge more quickly and leading to a better generalization.

3.2.3 Pose Regression Head

The last component of the ***marepo*** architecture is a pose regression head. Its structure is simple: first, a residual block formed of three 1×1 convolution layers followed by global average pooling generates a single embedding that represents the whole input scene-coordinate map. Such embedding is then passed to a small MLP (3 layers) that directly outputs the camera pose as a 10-dimensional representation. The pose representation can then be unpacked into translation and rotation: the translation is represented by four homogeneous coordinates (inspired by [6]); the rotation is encoded as a 6D vector representing two un-normalized axes of the coordinate system that are later used to form a full rotation matrix by normalization and cross-product, as in [58].

3.3. Loss Function

The map-relative pose regressor architecture described above is able to directly output a metric pose $\hat{\mathcal{P}}$ (formed of a 3×3 rotation matrix $\hat{R}$, and a translation vector $\hat{\mathbf{t}}$) for each image. In order to train such system we use a standard L1 pose regression loss proposed in [2], defined as follows:

$$\mathcal{L}_{\hat{P}} = \|\hat{R} - R\|_1 + \|\hat{\mathbf{t}} - \mathbf{t}\|_1. \quad (6)$$

Experimentally, we found that adding supervision at intermediate layers of the regressor is beneficial to the overall performance. Therefore, at training time, we additionally apply the pose regression head after each block of four self-attention transformers (see Fig. 3, right) and compute auxiliary losses $\mathcal{L}_{P_0}$ and $\mathcal{L}_{P_1}$ as described above. Thus, the overall loss we optimize during training is as follows:

$$\mathcal{L} = \mathcal{L}_{\hat{P}_0} + \mathcal{L}_{\hat{P}_1} + \mathcal{L}_{\hat{P}}, \quad (7)$$

During inference we only use the last output pose, $\hat{P}$.

3.4. (Optionally) Fine-Tuning the Pose Regressor

As described earlier, the proposed map-relative pose regressor is formed by two main components: an initial scene-specific network $\mathcal{G}_S$ that is able to predict metric scene coordinates for each pixel (in our implementation, we use an off-the-shelf scene coordinate regression architecture that can be trained in a few minutes for each scene [6]); and a scene-agnostic regressor $\mathcal{M}$ that exploits the geometric information encoded by the scene coordinates to predict the camera pose. The latter is trained once – ahead of time – over a large corpus of data, but it is reusable *as-is* for every new target scene. We find that this hybrid approach works

exceptionally well compared to other APR methods that are trained with the traditional end-to-end image-to-pose protocol, over the course of hours or days.

Still, we also explore whether applying a scene-specific adaptation stage to the transformer-based regressor can be beneficial to the performance of the method. In this scheme, for each new scene being evaluated, after training the scene-specific coordinate regressor $\mathcal{G}_\mathcal{S}$, we fine-tune the pose-regressor $\mathcal{M}$ on the same mapping images, using the same loss as in Sec. 3.3. Fine-tuning the transformer is very efficient in terms of resources required: in the next section we show that, with only two passes over the training dataset for each new scene (taking typically between 1-10 minutes, depending on the number of frames), our method can further improve its performance from what was already state-of-the-art compared to pose regression-based methods.

Notably, the final fine-tuning step is completely *optional* in our approach: the pre-trained $\mathcal{M}$ is already capable of predicting accurate camera poses, given 3D scene coordinates predicted by the geometric network module $\mathcal{G}_\mathcal{S}$.

4. Experiments

4.1. Implementation Details

In this section, we provide details of the process used to train the ***marepo*** network. We first detail the generation of training data, then describe the architectural configuration.

Training Data Generation In our implementation, we employ the Accelerated Coordinate Encoding architecture and training protocol proposed in [6] as $\mathcal{G}$ to train the scene-specific geometry prediction network $\mathcal{G}_\mathcal{S}$ for each scene S in the training dataset, since that allows the training of scene-specific coordinate-regression networks in a speedy fashion ($\sim$5 minutes for each new scene). To train $\mathcal{M}$, we use 450 scenes (indexing from 0 to 459, excluding 200-209) from the Map-Free Dataset [2]. Each image in the dataset has an associated ground truth camera pose computed by the authors of the dataset via SfM [44]. The data includes around 500K frames, with each scene containing images scanning a small outdoor location. Frames from each scan have been split into mapping and query, with $\approx$ 500 mapping frames and $\approx$ 500 queries. In practice, we train 900 scene-specific coordinate regressors $\mathcal{G}_\mathcal{S}$ using frames from each scene and its two splits. Given the efficient scaling capability of the method, we are able to generate a vast amount of 2D-3D correspondences between image pixels and scene coordinates by applying each $\mathcal{G}_\mathcal{S}$ to the frames of the unseen split of its corresponding scan. We use data augmentation during the generation of the correspondences, processing 16 variants of each frame. Specifically, we apply random image rotations of up to 15°; rescale each frame to $0.67 \sim 1.5$ times its original resolution; and, finally, extract random crops. We save the scene coordinate maps output of the preprocessing, together with their corresponding masks indicating which pixels are valid after rotation, the augmented intrinsics, and camera poses. This forms the fixed dataset we use to train the map-relative pose regressor $\mathcal{M}$.

Additionally, we perform online data augmentation by randomly jittering the scene coordinates by $\pm 1m$ and rotating them by up to 180° to further increase data diversity. Note that the random jittering is applied image-wise, i.e., all scene coordinates of an input frame are perturbed by the same transform. We do this to avoid overfitting, ensuring that the network $\mathcal{M}$ does not learn an absolute pose for each frame but rather a pose relative to the scene coordinates.

Network Configuration The scene-specific networks $\mathcal{G}_\mathcal{S}$ process images having the shortest side 480 pixels long and output dense scene coordinate maps with 8x smaller resolution. The map-relative pose regressor $\mathcal{M}$ is built upon a cascade of linear-attention transformer blocks [50] with $d_{model} = 256$ and $h = 8$ parallel attention layers. For the 3D position embedding, we prudently choose $m = 5$ frequency bands due to presence of potentially noisy input.

Training and Hardware Details We train $\mathcal{M}$ using 8 NVIDIA V100 GPUs with a batch size of 64. We use the AdamW [28] optimizer with a learning rate between $3e^{-4}$ to $2e^{-3}$ with a 1-cycle scheduler [49]. The model is trained for $\approx$ 10 days, iterating through the dataset for 150 epochs.

During inference our entire model, including the scene-specific network $\mathcal{G}_\mathcal{S}$ and the map-relative pose regressor $\mathcal{M}$, requires only one GPU, and can estimate camera poses with a real-time throughput, as later shown in Tab. 2.

4.2. Quantitative Evaluation

In the following paragraphs we show the performance of ***marepo*** on two public datasets: one depicting indoor scenes, and one outdoor. We show that the proposed map-relative pose regressor module $\mathcal{M}$ can generalize its predictions to previously unseen scenes, thanks to the scene-specific geometry prediction network $\mathcal{G}_\mathcal{S}$ providing it with 2D-3D correspondences in each scene's metric space.

7-Scenes Dataset We first evaluate our method on the Microsoft 7-Scenes dataset [19, 48], an indoor relocalization dataset that provides up to 7000 mapping images per scene. Each scene covers a limited area (between $1m^3$ and $18m^3$); despite that, previous APR methods require tens of hours or even several days [32] to train a model to relocalize in them. This is nonideal in a practical scenario as the appearance of the scene might have changed within that time frame, thus rendering the trained APR out of date. Conversely, ***marepo*** requires only minutes of training time ($\approx$ 5) for each new scene to generate a geometry-prediction network

Table 1. **Re-localization results on the indoor 7-Scenes dataset.** Pose errors are shown as median translation (cm) and rotation (°) errors. Numbers in **bold** represent the best performance among the APR-based approaches. ***marepo*** denotes our model with generic transformer-based pose regressor $\mathcal{M}$. ***marepo***$_S$ reports the performance of the model after $\mathcal{M}$ has been fine-tuned for each scene.

	Methods	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs	Average	Mapping Time
SCR	DSAC* [4]	1.9/1.11	1.9/1.24	1.1/1.82	2.6/1.18	4.2/1.41	3.0/1.70	4.2/1.42	2.7/1.41	Hours
	ACE [6]	1.9/0.7	1.9/0.9	0.9/0.6	2.7/0.8	4.2/1.1	4.2/1.3	3.9/1.1	2.8/ 0.93	**5 Minutes**
APR	PoseNet(PN)[25]	32/8.12	47/14.4	29/12.0	48/7.68	47/8.42	59/8.64	47/13.8	44/10.4	Hours
	PN Learn σ^2[24]	14/4.50	27/11.8	18/12.1	20/5.77	25/4.82	24/5.52	37/10.6	24/7.87	Hours
	geo. PN[24]	13/4.48	27/11.3	17/13.0	19/5.55	26/4.75	23/5.35	35/12.4	23/8.12	Hours
	LSTM PN[52]	24/5.77	34/11.9	21/13.7	30/8.08	33/7.00	37/8.83	40/13.7	31/9.85	Hours
	Hourglass PN[30]	15/6.17	27/10.8	19/11.6	21/8.48	25/7.01	27/10.2	29/12.5	23/9.53	Hours
	BranchNet[55]	18/5.17	34/8.99	20/14.2	30/7.05	27/5.10	33/7.40	38/10.3	29/8.30	Hours
	MapNet[7]	8/3.25	27/11.7	18/13.3	17/5.15	22/4.02	23/4.93	30/12.1	21/7.77	Hours
	Direct-PN[11]	10/3.52	27/8.66	17/13.1	16/5.96	19/3.85	22/5.13	32/10.6	20/7.26	Days
	MS-Transformer[47]	11/4.66	24/9.60	14/12.2	17/5.66	18/4.44	17/5.94	17/5.94	18/7.28	Hours
	DFNet (VGG) [12]	5/1.88	17/6.45	6/3.63	8/2.48	10/2.78	22/5.45	16/3.29	12/3.71	Days
	DFNet (EB0) [12]	3/1.15	9/3.71	8/6.08	7/2.14	10/2.76	9/2.87	11/5.58	8/3.47	Days
	LENS [32]	3/1.3	10/3.7	7/5.8	7/1.9	8/2.2	9/2.2	14/3.6	8/3.00	Days
	marepo (Ours)	2.6/1.35	2.5/1.42	2.3/2.21	3.6/1.44	4.2/1.55	5.1/1.99	6.7/1.83	3.9/1.68	**5 Minutes**
	marepo$_S$ (Ours)	**2.1/1.24**	**2.3/1.39**	**1.8/2.03**	**2.8/1.26**	**3.5/1.48**	**4.2/1.71**	**5.6/1.67**	**3.2/1.54**	$\leq$ 15 Minutes

Table 2. **Pose accuracy comparison on the outdoor Wayspots dataset.** Results are reported as the percentage of frames below $10cm/5°$ and $0.5m/5°$ pose error. The map-relative pose regressors $\mathcal{M}$ of our ***marepo***$_S$ experiment are fine-tuned in $\approx$ 1 minute for each scene.

Scene	DSAC* [4]	ACE [6]	PN [23–25]	MST [47]	***marepo*** Ours	***marepo***$_S$ Ours
Throughput (fps)	17.9	17.9	**166.7**	28.4	55.6	55.6
Bears	82.6%/91.6%	80.7%/92.6%	12.9%/35.7%	0.5%/12.8%	**80.7%**/99.3%	**80.7%/99.5%**
Cubes	83.8%/98.1%	97.0%/98.1%	0.0%/0.4%	0.00%/9.9%	**72.4%/96.9%**	71.8%/**96.9%**
Inscription	54.1%/69.7%	49.0%/69.6%	1.1%/6.3%	1.3%/9.7%	**37.8%/74.2%**	37.1%/74.1%
Lawn	34.7%/38.0%	35.8%/38.5%	0.0%/0.2%	0.0%/0.0%	32.6%/**41.6%**	**34.2%**/41.1%
Map	56.7%/87.1%	56.5%/84.7%	14.9%/49.1%	5.6%/25.7%	53.9%/87.7%	**55.1%/87.9%**
Square Bench	69.5%/97.9%	66.7%/97.8%	0.0%/3.0%	0.0%/0.0%	68.6%/**100%**	**70.7%/100%**
Statue	0.0%/0.0%	0.0%/0.0%	0.0%/0.0%	0.0%/0.0%	0.0%/0.0%	0.0%/0.0%
Tendrils	25.1%/26.5%	34.9%/36.8%	0.0%/0.0%	0.9%/23.6%	27.9%/33.4%	**29.3%/34.8%**
The Rock	100%/100%	100%/100%	24.2%/77.5%	10.7%/52.6%	98.1%/**100%**	**99.8%/100%**
Winter Sign	0.2%/5.7%	1.0%/7.6%	0.0%/0.0%	0.0%/0.0%	0.0%/**0.7%**	0.0%/0.3%
Average	50.7%/61.5%	52.2%/62.6%	5.3%/17.2%	1.9%/13.4%	47.2%/63.4%	**47.9%/63.5%**

$\mathcal{G}_S$ specifically tuned for the target environment. We compare our method with prior Pose Regression approaches in Tab. 1, showing that ***marepo*** is not only a partly scene-agnostic approach that enjoys the fastest mapping time of all APR-based methods, but also obtains $\approx 50\%$ better average performance (in terms of median error). We also show the performance of a fine-tuned variant of our method, ***marepo***$_S$, where, in addition to training the scene-specific $\mathcal{G}_S$ scene coordinate predictor, we also use the mapping frames to run two epochs of fine-tuning on the $\mathcal{M}$ regressor (see Sec. 3.4). The fine-tuned model ***marepo***$_S$ achieves further improvements in average performance, requiring only between $1.5 \sim 10$ extra minutes of training time, resulting in the only single-frame pose regression-based method able to achieve a similar level of accuracy as one of the current best 3D geometry-based methods, while being more efficient in terms of computational resources required.

Wayspots Dataset We further evaluate our method on the Wayspots dataset [2, 6], which depicts challenging outdoor scenes that even current geometry-based methods struggle with. The dataset contains scans of 10 different areas with associated ground truth poses provided by a visual-inertial odometry system [1, 20]. In Tab. 2 we show a comparison of the performance of the proposed ***marepo*** (as well as the ***marepo***$_S$ models, fine-tuned on the mapping frames of each scene) with two APR-based approaches we reproduced; we also include a comparison with two scene coordinate regression approaches: DSAC* [4] and the current state-of-the-art on Wayspots, ACE [6]. ***marepo*** significantly outperforms previous APR-based methods – such as PoseNet [25] and MS-Transformers [47] – that require on average several hours of training time, and compares favorably with geometry-based methods. We show, for the first time, that an end-to-end image-to-pose regression method relying on

Model	Accuracy	
	$5cm/5°$	$10cm/5°$
Full Architecture (***marepo***)	**16.6%**	**39.6%**
- Re-Attention	10.9%	28.3%
- Dynamic P.E.	3.9%	18.6%

Table 3. We gradually remove *Re-Attention* and *Dynamic Positional Encoding* and report the % of frames relocalized within $5cm/5°$ and $10cm/5°$.

# T Blocks	d_{model}	Accuracy	
		$5cm/5°$	$10cm/5°$
4	128	5.8%	22.2%
8	128	14.7%	39.1%
12	128	16.6%	39.6%
12	256	**19.0%**	**43.5%**

Table 4. Effect on performance of different dimensionality choices in the pose regressor's model. *# T Blocks* denotes the number of transformer blocks used in the model. d_{model} denotes the width of the transformer layers.

geometric priors can achieve a similar level of performance as methods that require the deployment of a (slower) robust solver to estimate the camera pose from a set of potentially noisy 2D-3D correspondences. More specifically, ***marepo*** requires only five minutes to train a network encoding the location of interest within the weights of the $\mathcal{G}_S$ scene-specific coordinate regressor and (*optionally*) approximately one minute to fine-tune the map-relative regressor $\mathcal{M}$ (as the Wayspot scans have significantly less frames then the 7-Scenes scenes above). At inference time ***marepo*** (or its fine-tuned variant) can perform inference at ≈ 56 frames per second, making it not only accurate, but also extremely efficient in comparison to other methods.

4.3. Ablation Experiments

In the following, we provide additional insights into the design choices adopted whilst designing our method.

Architecture Ablation We run several controlled experiments to justify our architectural design. Note that, for the experiments in this subsection, we trained the transformer-based pose regressor for 50 epochs instead of 150 as in the main experiments. This allows us to complete each experiment in approximately two days without affecting the relative ranking of results in the ablations. For the first experiment, see Tab. 3, we train a smaller transformer $\mathcal{M}$ (with $d_{model} = 128$) and gradually remove the *Re-Attention* and *Dynamic Positional Encoding* components to evaluate their impact on the performance of the Wayspots dataset. The pose accuracy is shown as the percentage of frames relocalized within $5cm/5°$ and $10cm/5°$ error. The table shows consistent degradation without the proposed components.

Next, we show the impact of diverse model configurations by deploying different numbers of transformer blocks and d_{model} dimensions in Tab. 4. We experimented with training even larger models with $d_{model} = 512$ and 16/20 transformer blocks, but found they necessitated substantially more GPU resources and time. We therefore cannot recommend them given the performance-time trade-offs.

Model	Accuracy	
	$10cm/5°$	$50cm/5°$
Per-scene ***marepo***	0.7%	6.0%
Per-scene $\mathcal{M}$	2.9%	18.7%
marepo (Ours)	**47.2%**	**63.4%**

Table 5. Effect of different training strategies. Per-scene ***marepo*** trains the entire network from scratch for every new mapping scene. In per-scene $\mathcal{M}$, we train only the regressor from scratch, on top of a pre-trained $\mathcal{G}_S$. Finally, ***marepo*** is trained over the entire training dataset and can generalize well to unseen scenes.

Per-Scene Training The proposed pose regressor component $\mathcal{M}$ is designed to be scene-agnostic and has been trained on a large corpus of data. Still, we are interested in evaluating its performance when trained ad-hoc for individual scenes, similar to existing APR-based methods. We conduct two experiments where, instead of using the full training set, we train scene-specific models using mapping sequences from the Wayspots dataset, as shown in Tab. 5. First, the models are trained from scratch, i.e., both the scene geometry regressor $\mathcal{G}_S$ and the map-relative pose regressor $\mathcal{M}$ are trained as a single entity, similar to PoseNet [25]. The results show extremely poor performance, likely due to the model's inability to learn explicit 3D geometry relations of the scene. For the second variant, we assume a pre-trained $\mathcal{G}_S$ is provided, then train $\mathcal{M}$ from scratch for each scene. We see results in a similar order of magnitude as other APR methods, such as PoseNet [25] or MST [47] (cf. Tab. 2); still, this training approach performed quite poorly compared to the full ***marepo*** model, where $\mathcal{M}$ is trained on a large-scale dataset to predict truly generic and scene-independent map-relative poses.

5. Conclusion

In conclusion, our paper introduces ***marepo***, a novel approach in Pose Regression that combines the strengths of a scene-agnostic pose regression network with a strong geometric prior provided by a fast-training scene-specific metric representation. The method addresses the limitations of previous APR techniques, offering both scalability and precision in predicting accurate scale-metric poses across diverse scenes. We demonstrate ***marepo***'s superior accuracy and its capability for rapid adaptation to new scenes compared to existing APR methods on two datasets. Additionally, we show how integrating the transformer-based network architecture with dynamic positional encoding ensures robustness to varying camera parameters, establishing ***marepo*** as a versatile and efficient solution for regression-based visual relocalization.

References

[1] Apple. ARKit. Accessed: 26 March 2024. 7

[2] Eduardo Arnold, Jamie Wynn, Sara Vicente, Guillermo Garcia-Hernando, Áron Monszpart, Victor Adrian Prisacariu, Daniyar Turmukhambetov, and Eric Brachmann. Map-free visual relocalization: Metric pose relative to a single image. In *ECCV*, 2022. 2, 5, 6, 7

[3] Eric Brachmann and Carsten Rother. Neural- Guided RANSAC: Learning where to sample model hypotheses. In *ICCV*, 2019. 2

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

[5] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC - Differentiable RANSAC for Camera Localization. In *CVPR*, 2017.

[6] Eric Brachmann, Tommaso Cavallari, and Victor Adrian Prisacariu. Accelerated coordinate encoding: Learning to relocalize in minutes using rgb and poses. In *CVPR*, 2023. 1, 2, 5, 6, 7

[7] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz. Geometry-Aware Learning of Maps for Camera Localization. In *CVPR*, 2018. 1, 3, 7

[8] Federico Camposeco, Andrea Cohen, Marc Pollefeys, and Torsten Sattler. Hybrid scene compression for visual localization. In *CVPR*, 2019. 1, 2

[9] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 4

[10] Kefan Chen, Noah Snavely, and Ameesh Makadia. Wide-baseline relative camera pose estimation with directional learning. In *CVPR*, 2021. 1

[11] Shuai Chen, Zirui Wang, and Victor Prisacariu. Direct-PoseNet: Absolute pose regression with photometric consistency. In *3DV*, 2021. 3, 7

[12] Shuai Chen, Xinghui Li, Zirui Wang, and Victor Prisacariu. DFNet: Enhance absolute pose regression with direct feature matching. In *ECCV*, 2022. 1, 3, 7

[13] Ronald Clark, Sen Wang, Andrew Markham, Niki Trigoni, and Hongkai Wen. Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. In *CVPR*, 2017. 3

[14] Siyan Dong, Shuzhe Wang, Yixin Zhuang, Juho Kannala, Marc Pollefeys, and Baoquan Chen. Visual localization via few-shot scene region classification. In *3DV*, 2022. 1, 2

[15] 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. *arXiv preprint arXiv:2010.11929*, 2020. 4

[16] Sovann En, Alexis Lechervy, and Frédéric Jurie. Rpnet: An end-to-end network for relative camera pose estimation. In *ECCVW*, 2018. 1

[17] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In *CACM*, 1981. 1, 2

[18] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE TPAMI*, 2003. 1, 2

[19] Ben Glocker, Shahram Izadi, Jamie Shotton, and Antonio Criminisi. Real-time rgb-d camera relocalization. In *ISMAR*, 2013. 6

[20] Google. ARCore. Accessed: 26 March 2024. 7

[21] 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, 2020. 1, 2

[22] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*, 2020. 5

[23] A. Kendall and R. Cipolla. Modelling uncertainty in deep learning for camera relocalization. In *ICRA*, 2016. 3, 7

[24] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. In *CVPR*, 2017. 1, 3, 7

[25] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, 2015. 1, 3, 7, 8

[26] Jake Levinson, Carlos Esteves, Kefan Chen, Noah Snavely, Angjoo Kanazawa, Afshin Rostamizadeh, and Ameesh Makadia. An analysis of svd for deep rotation estimation. *NeurIPS*, 2020. 1

[27] Xiaotian Li, Shuzhe Wang, Yi Zhao, Jakob Verbeek, and Juho Kannala. Hierarchical scene coordinate classification and regression for visual localization. In *CVPR*, pages 11983–11992, 2020. 2

[28] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *ICLR*, 2017. 6

[29] Simon Lynen, Bernhard Zeisl, Dror Aiger, Michael Bosse, Joel Hesch, Marc Pollefeys, Roland Siegwart, and Torsten Sattler. Large-scale, real-time visual-inertial localization revisited. *International Journal of Robotics Research*, 39(9), 2019. 2

[30] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu. Image-based localization using hourglass networks. In *ICCVW*, 2017. 3, 7

[31] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 5

[32] Arthur Moreau, Nathan Piasco, Dzmitry Tsishkou, Bogdan Stanciulescu, and Arnaud de La Fortelle. LENS: Localization enhanced by nerf synthesis. In *CoRL*, 2021. 1, 3, 6, 7

[33] Arthur Moreau, Nathan Piasco, Dzmitry Tsishkou, Bogdan Stanciulescu, and Arnaud de La Fortelle. Coordinet: uncertainty-aware pose regressor for reliable vehicle localization. In *WACV*, 2022. 3

[34] T. Naseer and W. Burgard. Deep regression for monocular camera-based 6-dof global localization in outdoor environments. In *IROS*, 2017. 3

[35] Vojtech Panek, Zuzana Kukelova, and Torsten Sattler. Meshloc: Mesh-based visual localization. In *ECCV*, pages 589–609. Springer, 2022. 2

[36] Pulak Purkait, Cheng Zhao, and Christopher Zach. Synthetic view generation for absolute pose regression and image synthesis. In *BMVC*, 2018. 3

[37] N. Radwan, A. Valada, and W. Burgard. Vlocnet++: Deep multitask learning for semantic visual localization and odometry. In *IEEE Robotics and Automation Letters*, 2018. 3

[38] 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

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

[40] Paul-Edouard Sarlin, Ajaykumar Unagar, Måns Larsson, Hugo Germain, Carl Toft, Viktor Larsson, Marc Pollefeys, Vincent Lepetit, Lars Hammarstrand, Fredrik Kahl, and Torsten Sattler. Back to the Feature: Learning robust camera localization from pixels to pose. In *CVPR*, 2021. 1, 2

[41] T. Sattler, B. Leibe, and L. Kobbelt. Improving image-based localization by active correspondence search. In *ECCV*, 2012. 2

[42] T. Sattler, B. Leibe, and L. Kobbelt. Efficient & Effective Prioritized Matching for Large-Scale Image-Based Localization. In *IEEE TPAMI*, 2017. 1, 2

[43] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *CVPR*, 2019. 1

[44] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 6

[45] Yoli Shavit and Yosi Keller. Camera pose auto-encoders for improving pose regression. In *ECCV*, 2022. 3

[46] Yoli Shavit, Ron Ferens, and Yosi Keller. Paying attention to activation maps in camera pose regression. In *arXiv preprint arXiv:2103.11477*, 2021. 1

[47] Yoli Shavit, Ron Ferens, and Yosi Keller. Learning multi-scene absolute pose regression with transformers. In *ICCV*, 2021. 3, 7, 8

[48] 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

[49] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, 2019. 6

[50] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. *CVPR*, 2021. 4, 6

[51] Mehmet Özgür Türkoğlu, Eric Brachmann, Konrad Schindler, Gabriel Brostow, and Áron Monszpart. Visual Camera Re-Localization Using Graph Neural Networks and Relative Pose Supervision. In *3DV*, 2021. 1

[52] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers. Image-based localization using lstms for structured feature correlation. In *ICCV*, 2017. 3, 7

[53] Bing Wang, Changhao Chen, Chris Xiaoxuan Lu, Peijun Zhao, Niki Trigoni, and Andrew Markham. Atloc: Attention guided camera localization. In *AAAI*, 2020. 3

[54] Dominik Winkelbauer, Maximilian Denninger, and Rudolph Triebel. Learning to localize in new environments from synthetic training data. In *ICRA*, 2021. 1

[55] J. Wu, L. Ma, and X. Hu. Delving Deeper into Convolutional Neural Networks for Camera Relocalization. In *ICRA*, 2017. 3, 7

[56] 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

[57] Qunjie Zhou, Sérgio Agostinho, Aljoša Ošep, and Laura Leal-Taixé. Is geometry enough for matching in visual localization? In *ECCV*, 2022. 1

[58] Yi Zhou, Connelly Barnes, Lu Jingwan, Yang Jimei, and Li Hao. On the continuity of rotation representations in neural networks. In *CVPR*, 2019. 5