IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

Implicit Learning of Scene Geometry from Poses for Global Localization

Mohammad Altillawi[1,2], Shile Li[1], Sai Manoj Prakhya[1], Ziyuan Liu[1], and Joan Serrat[2]

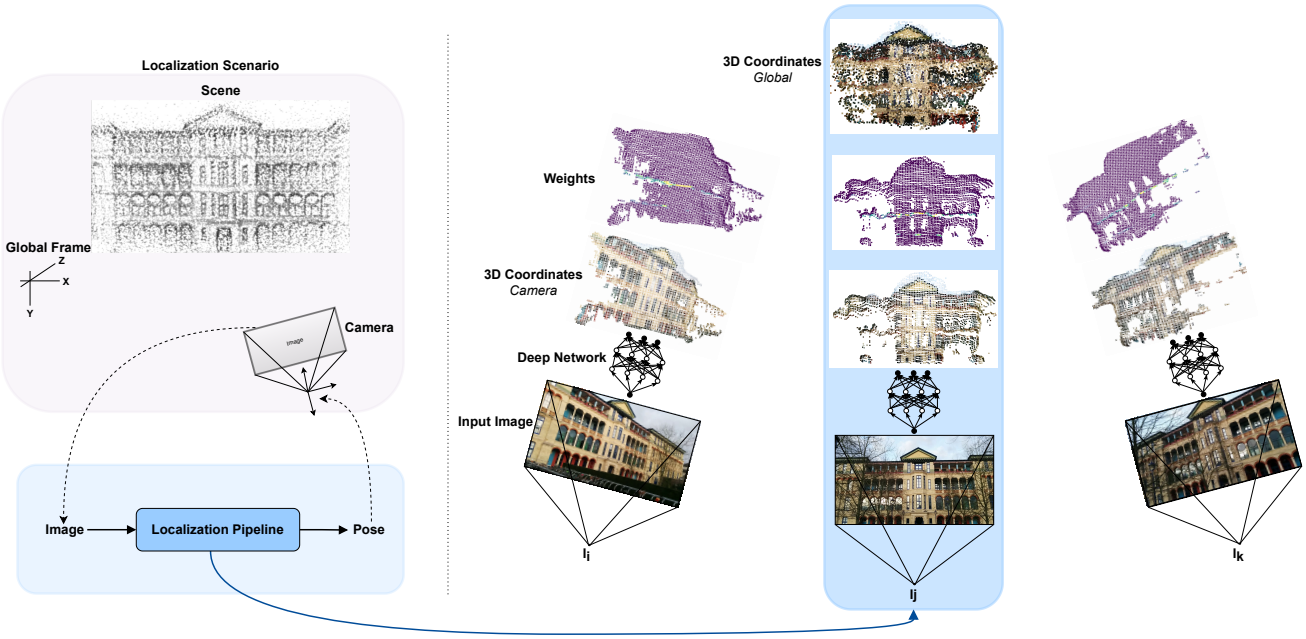

Fig. 1: Illustration of our visual localization proposal on samples from Cambridge Landmarks (Hospital scene). Our method requires a set of images and the corresponding poses as the only labels for training. Left side: Given a single image, our method estimates the global pose of the camera in a given scene. Right side: we display the intermediate outputs of our proposal which are used to estimate the pose. For an input image, the proposed pipeline estimates two point clouds and a set of weights. The first point cloud represents the scene geometry (*X, Y, Z coordinates*) in camera coordinate frame, while the second point cloud represents the scene geometry in global coordinate frame. These two point clouds, along with the predicted weights are used to estimate the camera's global pose. In the right side of Fig. 1, we visualize three sample input images, their corresponding indirectly estimated 3D scene representations (point clouds) and the weights. At the top, in the right side of Fig. 1, we can see only one 3D point cloud, which corresponds to three overlaid point clouds in the global coordinate frame, also estimated by our algorithm for the considered sample images. Though our method implicitly estimates 3D point clouds of the scene in local and global reference frames, it is not a mapping or 3D reconstruction algorithm, but a localization algorithm that implicitly learns and uses 3D scene geometry.

***Abstract*—Global visual localization estimates the absolute pose of a camera using a single image, in a previously mapped area. Obtaining the pose from a single image enables many robotics and augmented/virtual reality applications. Inspired by latest advances in deep learning, many existing approaches directly learn and regress 6 DoF pose from an input image. However, these methods do not fully utilize the underlying scene geometry for pose regression. The challenge in monocular relocalization is the minimal availability of supervised training data, which is just the corresponding 6 DoF poses of the images. In this paper, we propose to utilize these minimal available labels (.i.e, poses) to learn the underlying 3D geometry of the scene and use the geometry to estimate the 6 DoF camera pose. We present a learning method that uses these pose labels and rigid alignment to learn two 3D geometric representations (*X, Y, Z coordinates*) of the scene, one in camera coordinate frame and the other in global coordinate frame. Given a single image, it estimates these two 3D scene representations, which are then aligned to estimate a pose that matches the pose label. This formulation allows for the active inclusion of additional learning constraints to minimize 3D alignment errors between the two 3D scene representations, and 2D re-projection errors between the 3D global scene representation and 2D image pixels, resulting in improved localization accuracy. During inference, our model estimates the 3D scene geometry in camera and global frames and aligns them rigidly to obtain pose in real-time. We evaluate our work on three common visual localization datasets, conduct ablation studies, and show that our method exceeds state-of-the-art regression methods' pose accuracy on all datasets.**

***Index Terms*—Localization, Localization and Mapping, Deep Learning for Visual Perception, Visual Learning**

[1]M. Altillawi, S. Li, S. Prakhya, and Z. Liu are with the AI Robotics & Simulation team, Huawei Munich Research Center, Germany.

[2]M. Altillawi and J. Serrat are with the Computer Vision Center (CVC) and Department of Computer Science, Universitat Autònoma de Barcelona, 08193 Bellaterra (Cerdanyola del Vallès), Spain.

The work of Joan Serrat was supported by the Generalitat de Catalunya CERCA Program to CVC's general activities

I. INTRODUCTION

Global camera re-localization has driven many computer vision applications in augmented/virtual reality and robotics. The problem definition, which is to obtain position and orientation (in metric units) of a camera from a single image in a previously mapped area, requires the availability of ground-truth poses for training. With training data that is composed of images and the corresponding poses, these methods follow a common learning process that maps the input image to 6 DoF directly in an end-to-end manner. This learning process goes as follows: the input image is passed to a network that encodes it into a latent vector. Through one or more regression layers, the latent vector is then mapped into a pose as two quantities, one for the translation and the other for the rotation. A pose loss is deployed to update the weights of the network during training. With a new input image at localization time, the network utilizes its memory formed by the learned weights to estimate the pose. Recent works [1]–[16] build upon this scheme with different variations to constrain the weights to obtain a better pose estimate. This approach is attractive for several reasons. Firstly, it provides a pose directly in a single regression step. Secondly, the mapping from image to pose is fast; no classic feature matching is required, making it suitable for real-time applications. Thirdly, the whole pose estimation pipeline is saved in a compact form as network weights.

However, one limitation of this formulation is that it treats pose estimation as a regression problem and consequently, it strips out the geometry of the scene from pose estimation. This conditions the design of the deep network's last layer to be fully connected layers. This precludes incorporating geometric constraints such as depth or 3D scene coordinates. These can only be included in the loss terms in the training phase. Otherwise, they are explicitly required during inference (for example, 3D models from structure from motion).

In this work, we regard these limitations and propose a novel approach for global camera re-localization. Similar to other pose regression methods, our method is subject to the constraint of using minimalistic training data: a set of images with their intrinsics and the corresponding poses in a global reference frame. However, our pipeline does not estimate a pose by regression. Instead, it obtains geometric information of the scene which can be used, in turn, to estimate the pose geometrically; from pose labels only. The proposed pipeline learns 3D geometric representations (*X, Y, Z* coordinates) of the scene as seen by the image, given guidance from ground-truth poses alone as shown in Fig. 1. It takes a single image and obtains 3D point cloud of the scene in two coordinate systems: camera frame and a global reference frame. To accomplish this, we utilize rigid alignment as a means to adjust the network weights in order to obtain two geometric maps. The rigid alignment module aligns the two clouds to obtain a pose. This pose is adjusted, through gradient descent while training, so as to match the ground-truth pose, thus, implicitly adjusting the two geometric representations (3D clouds) as well.

In our end-to-end pipeline, deep learning is used to learn scene-specific geometric representations while we perform pose estimation in a geometric manner by parameter-free rigid alignment. During inference, it obtains a pose by rigidly aligning the two predicted 3D geometric representations. In contrast to pose regression, the proposed formulation allows the explicit use of additional constraints during training. We minimize re-projection errors in order to limit the deviation of the 3D map representation in the global frame from the corresponding 2D pixels. We complement that by constraining the deviation of the two 3D clouds according to ground-truth pose, which we refer to as consistency loss. This formulation results in higher localization accuracy than state-of-the-art regression methods. Additionally, we observe that our method can improve both position and orientation localization when finetuned with absolute position labels only. This is useful in applications where initially only a small set of training poses are available, but during operation, more data as partial geometric labels (position information from GPS) is available.

In summary, our contributions are:

- We propose to utilize poses to train a network to learn geometric representations of a given scene implicitly. These are 3D coordinates in camera and a scene/global reference frame. For this goal, we utilize parameter-free and differentiable rigid alignment supervised by pose loss to guide the learning of scene geometry.
- We also propose to employ additional loss terms, specifically, re-projection loss to constrain the learning of the 3D scene coordinates and introduce a consistency loss term that harmonizes the implicit geometric representations, according to the pose.
- Apart from extensive evaluation on public datasets, we conduct ablation studies to evaluate the influence of the three different losses on our method's performance. The ablation studies show that our method can be finetuned using partial pose labels, i.e., with position information alone and still offers fairly good performance by improving both position and orientation accuracy.

II. RELATED WORK

The problem at hand is monocular global re-localization, which is to obtain metric poses (in meters and degrees) in a previously mapped area. Existing works utilize different sets of labels for training a localization pipeline. While some works utilize 3D geometric information such as 3D scene coordinates and depth, others utilize only pose labels. Similar to pose regression methods, our proposed method learns from pose labels only).

Initial works [1]–[16] followed the regression approach and implemented different network architectures, learning strategies, and constraints on the learning process to reduce localization errors. In the following, we briefly review these methods and point out the coincidences and differences with respect to ours. PoseNetLSTM [4] utilizes LSTMs to reduce the dimensionality of the latent vector (that encodes the image) as a way to mitigate overfitting and obtain a more accurate pose estimate. PoseNetLearned [2] improves localization accuracy of their initial work PoseNet [1] by addressing the issue of loss imbalance between the orientation and translation losses by learning weighting factors for these terms using homoscedatic uncertainty. PoseNetGeo [2] further improves the accuracy

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

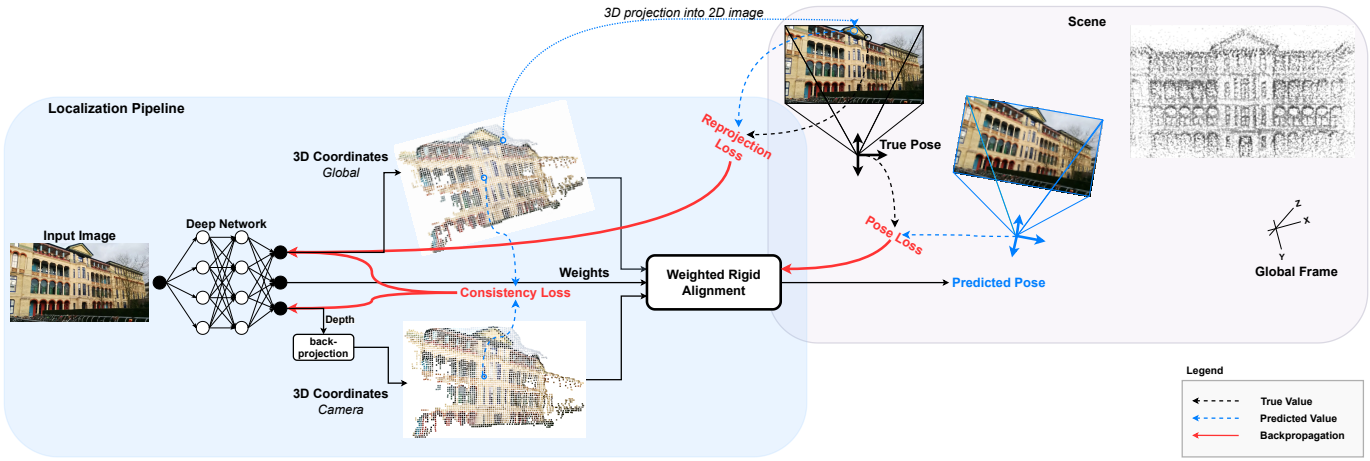

Fig. 2: A diagram of the proposed training method. Given images and the corresponding global pose labels, the network learns to indirectly (i.e., without explicit labels) estimate two 3D point clouds (one in global and the other in camera coordinate systems) and a set of weights. For the 3D coordinates in the camera coordinate system, the network predicts the depth, which is then back-projected to 3D space using Equation. (6). A rigid alignment module aligns the two point clouds according to the weights to estimate pose. The loss terms employed for training are visualized in red.

for some scenes by utilizing explicit 3D coordinate labels through re-projection loss in addition to pose loss. Though explicit 3D coordinate labels are used, the re-projection loss plays a limited role as it is used to constrain the image latent representation for regression instead of learning 3D scene geometry for pose estimation. Our proposal does not use 3D coordinate labels for training but still learns 3D geometry and uses it for pose estimation. Hourglass PN [6] replaces the initial architecture of PoseNet [1] by a Resnet34 [17] and complements it with up-convolutions to preserve the fine-grained information of the input image forming an Hourglass-like architecture. BranchNet [7] also adapts the architecture of PoseNet [1] to account for the complex coupling between position and orientation. The split in network branches for position and orientation is performed at an earlier stage of the series of convolutions. Other works [8], [10]–[12] utilize sequences of images to gain additional source of training signal by imposing relative pose constraints that are obtained between consecutive camera frames.

AtLoc [9] proposes a method based on attention to guide the network to output latent image representation that encodes robust objects and features for pose regression. It further utilizes a sequence of images to learn temporally consistent and informative features. CoordiNet [16] embeds pixel coordinates into the convolution operation to encode geometric feature locations into pose regression. It appends two additional channels that contain 2D pixel locations, to the input tensor before applying the convolution. MsTransformer [15] proposes a transformer-based approach for localization. It utilizes image latent representations that are obtained from CNN, for processing by separate Transformers to regress position and orientation. Transformer encoders are used to aggregate activation maps with self-attention and decoders convert latent features and scene encoding into pose predictions. With the advances in graph neural networks (GNN), PoGo-Net [13] and GNNPose [14] formulates the pose regression based on GNNs, naturally propagating information between different views for the benefit of pose regression.

Similar to previous regression works, we train from the available image-pose labels for the given datasets. In contrast, we propose using deep learning to implicitly learn representations of the 3D geometry of the scene instead of directly learning the pose regression. Accordingly, we solve for the pose by a single-step closed form solution through the rigid alignment of 3D scene representations. There are other works that use additional sensing modality such as LiDAR or labeled depth data or structure from motion results for training and/or inference [18]–[21]. However, in this work, we solely focus on training with posed images only.

III. METHOD

A. Overview

Figure 2 shows an overview of the proposed method. We propose to use the global camera pose $\mathbf{T}$ of a given input image I as a label to guide the training of a deep neural network to obtain representations of the scene.

For that purpose, we define our localization pipeline to take a given image as input and yield two sets of 3D points, each in a different coordinate system. The first one is a set of 3D coordinates $G = \{\hat{\mathbf{g}}_i, ..., \hat{\mathbf{g}}_M\}$ in the global reference frame of the scene. These are predicted directly by the network. The second one is a set of 3D coordinates $C = \{\hat{\mathbf{c}}_i, ..., \hat{\mathbf{c}}_M\}$ in the camera frame. For the latter, the network predicts depth, which is then back-projected using intrinsics parameters to get 3D coordinates in camera frame. Inherently, the two 3D points clouds are matched via the image pixel coordinates.

Using rigid alignment, a pose $\hat{\mathbf{T}}$ can be estimated by aligning the two point clouds. We utilize Kabsch algorithm [22] for this goal. It is differentiable, parameter-free, and obtains a closed-form solution in a single step. This makes the pipeline end-to-end trainable.

To account for imperfections in predictions, the network predicts a set of weights $W = \{w_i, ..., w_M\}$, which evaluates how much each 3D correspondence between point clouds

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

from camera and global coordinate frame contributes to the rigid alignment. Given such correspondences, the weighted Kabsch algorithm [22] is then applied to estimate the relative pose from camera coordinate system to the global coordinate system. Given M 3D coordinates, this weighted minimization goal is defined as:

$$\arg\min_{\hat{\mathbf{R}},\hat{\mathbf{t}}} \sum_{i}^{M} w_i ||\hat{\mathbf{g}}_i - \hat{\mathbf{R}}\hat{\mathbf{c}}_i - \hat{\mathbf{t}}||_2, \tag{1}$$

which can be described as follows: the translation $\hat{\mathbf{t}}$ of the pose is removed by centering both point clouds:

$$\boldsymbol{\mu}_{\mathbf{g}} = \frac{\sum_i w_i \hat{\mathbf{g}}_i}{\sum_i w_i}, \quad \bar{\mathbf{G}} = \mathbf{G} - \boldsymbol{\mu}_{\mathbf{g}}$$
$$\boldsymbol{\mu}_{\mathbf{c}} = \frac{\sum_i w_i \hat{\mathbf{c}}_i}{\sum_i w_i}, \quad \bar{\mathbf{C}} = \mathbf{C} - \boldsymbol{\mu}_{\mathbf{c}}.$$

Rotation $\hat{\mathbf{R}}$ and translation $\hat{\mathbf{t}}$ are then recovered with SVD as follows:

$$\mathbf{U}\mathbf{S}\mathbf{V}^T = \mathrm{svd}(\bar{\mathbf{C}}^T W \bar{\mathbf{G}})$$
$$s = \det(\mathbf{V}\mathbf{U}^T)$$
$$\hat{\mathbf{R}} = \mathbf{V}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & s \end{pmatrix}\mathbf{U}^T$$
$$\hat{\mathbf{t}} = -\hat{\mathbf{R}}\boldsymbol{\mu}_{\mathbf{c}} + \boldsymbol{\mu}_{\mathbf{g}}.$$

We apply a pose loss to guide the rigid alignment so that the network learns the 3D geometric representations. Given a ground-truth pose $\mathbf{T}$ with rotation $\mathbf{R}$ and translation $\mathbf{t}$ components, a cost function can be defined to minimize the difference between the estimated and ground-truth components. We define the loss as the summation of position loss and the rotation loss:

$$L_{pose} = L_{position} + L_{rotation}, \tag{2}$$

where

$$L_{position} = \| \mathbf{t} - \hat{\mathbf{t}} \|_2, \tag{3}$$

defines the position error between the computed translation $\hat{\mathbf{t}}$ and the actual translation $\mathbf{t}$ and

$$L_{rotation} = \cos^{-1}(\frac{1}{2}(\mathrm{trace}(\hat{\mathbf{R}}\mathbf{R}^{-1}) - 1)) \tag{4}$$

measures the angular error between computed rotation $\hat{\mathbf{R}}$ and the ground-truth rotation $\mathbf{R}$.

The predicted pose is adjusted by gradient descent that is guided, during training, by the pose loss Equation. (2), to match the ground-truth pose. This process indirectly adjusts the two geometric representations by passing gradients through the differentiable rigid alignment. The proposed formulation allows for the inclusion of additional constraints that actively guide the optimization of the implicit 3D geometric representations from the poses. Consequently, we introduce a consistency loss to constrain the geometric predictions to be aligned according to the ground-truth pose. We first transform the 3D points C from camera coordinate frame to global coordinate frame using the ground-truth pose. The consistency loss measures the error between the 3D points G in global coordinate frame and the 3D points C transformed from camera coordinate frame, using the ground-truth poses. We define the consistency loss as:

$$L_{consistency} = \frac{1}{M}\sum_{i}^{M} \| \hat{\mathbf{g}}_i - \mathbf{T}\hat{\mathbf{c}}_i \|_2, \tag{5}$$

Rather than predicting the 3D coordinates directly, we can adjust the network to predict one quantity which is depth. Given depth, which forms the *Z* coordinate in the camera perspective, the *X* and *Y* are obtained directly from the image pixels and depth, given camera intrinsics parameters. Accordingly, the 3D points C in camera coordinate frame are obtained by back-projecting the depth according to:

$$\hat{\mathbf{c}}_i = \hat{d}_i \mathbf{K}^{-1}\mathbf{u}_i, \tag{6}$$

where $\mathbf{u}_i$, $\mathbf{K}$, $\hat{d}_i$, and $\hat{\mathbf{c}}_i$ denote the homogeneous pixel coordinates, the camera intrinsic matrix, the depth, and the corresponding point in the camera frame, respectively.

In addition, the 3D global coordinates are further constrained by utilizing a re-projection loss to minimize the error between the re-projection of the 3D global coordinates into the image frame and the 2D image pixels. It is defined as:

$$L_{reprojection} = \frac{1}{M}\sum_{i}^{M} \| \mathbf{u}_i - \pi(\mathbf{T}\hat{\mathbf{g}}_i) \|_2, \tag{7}$$

where π projects points from the 3D global frame into the image frame.

With pose labels and the defined formulation, our method implicitly learns geometric representations of the scene. Given an image at inference, the proposed method estimates the scene's geometry and utilizes it for pose computation.

The overall loss is then the weighted combination of the pose loss, the re-projection loss, and the consistency loss:

$$L_{total} = \lambda_p L_{pose} + \lambda_c L_{consistency} + \lambda_r l_{reprojection}, \tag{8}$$

where λ_p, λ_c, and λ_r are the losses weighting factors.

IV. RESULTS

A. Datasets

We conduct our experiments on three common visual localization datasets. These are the outdoor Cambridge Landmarks [1], the indoor 7Scenes [23], and the indoor 12scenes [24] datasets. They exhibit different characteristics:

Cambridge landmarks is an outdoor re-localization dataset that covers different landmarks of several hundred or thousand square meters in Cambridge, UK. The provided reference poses are reconstructed from structure from motion. The challenges in this dataset arise from variety of illumination changes due to weather and dynamic objects such as cars and pedestrians. In addition, the training set size is relatively small (few hundred of images).

7Scenes contains 7 scenes that depict difficult scenarios, such as motion blur, reflective surfaces, repeating structures, and texture-less surfaces. Several thousand frames with corresponding ground-truth poses are provided for each scene's train and test splits.

12Scenes consists of 12 sequences with challenges similar to 7Scenes dataset. However, compared to 7scenes, it covers larger indoor environments with smaller number of training images, about several hundred frames for each scene.

B. Setup

We resize the input images to a standard 480 px height and normalize them by mean and standard deviation. During

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

TABLE I: Comparison against state-of-the-art localization methods on Cambridge Landmarks [1] and 7Scenes [23] datasets. First and second best results are marked in **bold** and underline respectively. '_' denotes unavailable results.

		Cambridge					7Scenes							
Method	Backbone	College	Hospital	Shop	Church	Average	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs	Average
PoseNetGeo [2]	GoogLeNet	0.88/1.04	3.20/3.29	0.88/3.78	1.57/3.32	1.63/2.86	0.13/4.48	0.27/11.3	0.17/13.0	0.19/5.55	0.26/4.75	0.23/5.35	0.35/12.4	0.23/8.12
PoseNetLSTM [4]	GoogLeNet	0.99/3.65	1.51/4.29	1.18/7.44	1.52/6.68	1.30/5.51	0.24/5.77	0.34/11.9	0.21/13.7	0.30/8.08	0.33/7.00	0.37/8.83	0.40/13.7	0.31/9.85
GPoseNet [3]	GoogLeNet	1.61/2.29	2.62/3.89	1.14/5.73	2.93/6.46	2.08/4.59	0.20/7.11	0.38/12.3	0.21/13.8	0.28/8.83	0.37/6.94	0.35/8.15	0.37/12.5	0.31/9.95
BranchNet [7]	GoogLeNet	_	_	_	_	-	0.18/5.17	0.34/8.99	0.20/14.2	0.30/7.05	0.27/5.10	0.33/7.40	0.38/10.3	0.29/8.32
SVS-Pose [5]	VGGNet	1.06/2.81	1.50/4.03	0.63/5.73	2.11/8.11	1.32/5.17	_	_	_	_	_	_	_	-
Hourglass PN [6]	ResNet34	_	_	_	_	-	0.15/6.17	0.27/10.8	0.19/11.6	0.21/8.48	0.25/7.01	0.27/10.2	0.29/12.5	0.23/9.54
MapNet [8]	ResNet34	0.94/1.99	2.03/3.60	0.80/6.34	1.66/4.01	1.36/3.99	0.08/3.25	0.27/11.7	0.18/13.3	0.17/5.15	0.22/4.02	0.23/4.93	0.30/12.1	0.21/7.78
AtLoc [9]	ResNet34	_	_	_	_	-	0.10/4.07	0.25/11.4	0.16/11.8	0.17/5.34	0.21/4.37	0.23/5.42	0.26/10.5	0.20/7.56
CoordiNet [16]	ResNet34	0.80/1.22	1.43/2.86	0.73/4.69	1.32/4.10	1.07/3.22	_	_	_	_	_	_	_	-
CoordiNet [16]	EffNet b3	0.70/0.92	0.97/2.08	0.69/3.74	1.32/3.56	0.92/2.58	0.14/6.7	0.27/11.6	0.13/13.6	0.21/8.6	0.25/7.2	0.26/7.5	0.28/12.9	0.22/9.72
PoGO-Net [13]	Not Applicable	-/0.94	-/1.69	-/2.40	-/**2.12**	-/1.78	-/1.72	-/6.23	-/7.34	-/3.93	-/3.56	-/3.85	-/7.88	-/4.93
MsTransformer [15]	EfficientNetB0	0.83/1.47	1.81/2.39	0.86/3.07	1.62/3.99	1.28/2.73	0.11/6.38	0.23/11.5	0.13/13.0	0.18/8.14	0.17/8.42	0.16/8.92	0.29/10.3	0.18/9.51
GNNPose [14]	ResNet34	0.59/**0.65**	1.88/2.78	0.50/2.87	1.90/3.29	1.22/2.40	0.08/2.82	0.26/8.94	0.17/11.41	0.18/5.08	0.15/2.77	0.25/4.48	0.23/8.78	0.19/6.32
Ours	ResNet34	0.48/0.84	0.67/1.14	0.47/1.91	0.90/3.05	0.63/1.74	**0.05**/1.86	**0.14/4.15**	**0.09/5.29**	0.14/3.61	**0.11**/2.78	**0.10**/2.66	0.19/4.14	**0.12**/3.50
	MobileNetV3	**0.46**/0.81	**0.61/1.09**	**0.44/1.71**	**0.87**/2.88	**0.60/1.62**	**0.05/1.42**	0.16/4.57	**0.09**/6.14	**0.13/3.46**	0.12/**2.48**	**0.10/2.51**	**0.17/3.48**	**0.12/3.44**

training, we apply, on the fly, color jittering and random in-plane rotations in the range [-30°, 30°].

We adjust every backbone network that we use in our experiments to obtain three outputs. The first is a 3 channels output that corresponds to the *X*, *Y*, *Z* coordinates in the global frame. The second is a one-channeled depth prediction. Depth is obtained from a Sigmoid function and is then scaled to a range of [0.1 10] for indoor scenes and [0.1 600] for outdoor scenes. These hyperparameters are generalizable and adjustable to specific scenes. The third is also of one channel, followed by a Sigmoid, which stores the weights that weigh the contribution of the correspondences to the rigid alignment.

For updating the network weights, we use Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and a weight decay of 5×10^{-4}. We train the whole architecture from scratch for 400 epochs with learning rate 10^{-4}. We set the weighting factors λ_p, λ_c, and λ_r of Equation. (8) to 1, 1, and 0.001 respectively to provide a balanced contribution to gradient updates. The low factor for the reprojection error aims to stabilize the training.

Following previous methods, we report localization errors as median translation (in meters) and median orientation (in degrees) errors for all the experiments below. For some experiments, we list as well the average localization errors that is computed on all scenes of the corresponding datasets.

C. Results: comparison to previous methods

In this section, we compare our proposed method against state-of-the-art. We consider the methods that utilize pose labels to train a deep network for global localization from a single image without including additional supervision signals or more sensory data. To marginalize improvements that may come out of using a recent backbone, we implement our method using ResNet34 backbone [17], which is adopted by previous methods. We report the median localization errors in Table I. As listed, our method, with both backbones, obtains the lowest localization errors on all of the scenes, except for the rotation measurements on the college and church scenes. Even though our method does not rely on labeled 3D ground-truth coordinates, it surpasses PoseNetGeo [2], which uses explicit 3D coordinates. PoseNetGeo [2] obtains pose by regression which doesn't directly incorporate these available geometric information. In contrast, our method uses poses to infer the geometry of the scene, which is directly embodied for pose estimation by rigid alignment. AtLoc [9] implements attention to utilize informative regions of a given image for pose regression. On the contrary, our method obtains weighting factors that are directly used to minimize contributions from outliers, thus, improving localization accuracy. We show the benefit of these weights in section IV-E. Previous methods also implement other strategies such as graph-neural-networks [13], [14], transformers [15] relative poses supervision [8], and LSTMs [4] to better encode the image representation for the task of pose estimation. However, their performances are leveled by regression incapability to utilize geometric quantities directly for localization. Our method uses pose labels to guide the network to learn certain geometric features, which, in return, are used to compute a pose. It lets the network, through pose targets, to freely choose the suitable geometric features that are best for localization. Besides using ResNet34 [17] as a backbone, we implement our method using MobileNetV3 [25] due to its efficiency. As shown in Tab. I, employing MobileNetV3 [25] as the backbone, gives the best results across datasets. In the rest experiments of following sections, we adopt MobileNetV3 [25] as our backbone while also providing a comparison with other backbones.

TABLE II: Ablation results of section IV-D on Cambridge Landmarks [1], 7Scenes [23], and 12Scenes [24] datasets.

	L_{pose}	L_{reproj}	$L_{consist}$	Cambridge	7Scenes	12Scenes
1	✓			0.72/3.91	0.138/3.88	0.067/2.48
2	✓	✓		0.71/2.89	0.119/3.49	0.063/2.41
3	✓		✓	0.67/1.75	0.118/3.49	0.066/2.34
4	✓	✓	✓	0.60/1.62	0.116/3.44	0.061/2.33
5	Input Resolution/4			0.59/1.62	0.115/3.13	0.060/2.31
6	Input Resolution/8			0.60/1.62	0.116/3.44	0.061/2.33
7	Input Resolution/16			0.68/1.77	0.134/4.15	0.068/2.65
8	Ours + 3D Coordinates			0.64/1.85	0.131/4.03	0.064/2.57
9	Ours + Depth			0.60/1.62	0.116/3.44	0.061/2.33
	Backbone		Run-time			
10	HRNetV2 [26]		256 ms	0.64/2.01	0.131/3.78	0.068/2.61
11	ResNet34 [17]		45 ms	0.63/1.74	0.124/3.50	0.064/2.49
12	DenseNet121 [27]		90 ms	0.62/1.69	0.121/3.52	0.062/2.43
13	MobileNetV3 [25]		14 ms	0.60/1.62	0.116/3.44	0.061/2.33

D. Results: ablation study

Here, we delve into our proposal by examining the effect of the employed three different losses, and the performance of our method with different output resolutions and backbones. We list the results in Tab. II.

Losses: we evaluate the contribution of every loss in our proposed method. Our method utilizes pose labels to guide the network through a pose loss Equation. (2). In addition, we apply consistency loss to align the two geometric representations (local and global) according to the input pose, and lastly a re-projection loss to align 3D global coordinates to 2D image pixels. The Tab. II (rows 1 to 4) presents the averaged results over all sequences on considered datasets, with different combination of loss terms.

The results from Tab. II (rows 1 to 4) show that adding the re-projection loss (row 2) or the consistency loss (row 3) to the pose loss (row 1) results in lower errors than when training the network with the pose loss only (row 1, rigid alignment module). The consistency loss is more effective than the re-projection loss in supporting the pose loss to reduce localization errors on outdoor scenes. On indoor scenes, they exhibit almost similar behavior. Combining all the losses (row 4) obtains the lowest errors on all datasets.

Resolutions: we evaluate the performance of our proposed method with different output resolutions. In our experiments, the resolution of the output geometric representations is $1/8$ of the input resolution. Here, we change the output resolution by changing the stride parameter. We report the results of two additional down-sampling factors: 4 and 16 in Tab. II (rows 5 to 7). For a down-sampling factor of 16, we observe a slight increase in localization errors compared to a down-sampling factor of 8. On average, down-sampling by a factor of 4 results in slight improvement in localization.

Depth versus 3D Camera coordinates: we can adjust the network to either obtain 3D coordinates in the camera frame directly or obtain the depth. For the latter, 3D camera coordinates can be computed by Equation. (6). Rows 8 and 9 in Tab. II suggest that learning just the depth results in a better localization performance. This constrains the 3D coordinates predictions and eases the learning process, so that the network learns one quantity rather than 3 quantities.

Backbones: while many backbones could be used to implement our method, we look into them from the perspective of localization accuracy, run-time and compactness. The rows 10 to 13 in Tab. II show the results using different backbones together with the run-time. Being the most compact, that is, with the smallest number of parameters (3.7 million), MobileNetV3 [25] obtains the best localization results with the fastest run-time (for a down-sampling factor of 8).

TABLE III: Effect of different filtering methods (section IV-E). Best results are marked in bold.

Method	College	Hospital	Shop	Church
Rigid + No Filtering	0.55/0.87	0.79/1.63	0.59/2.32	1.13/3.55
Rigid + Filter Dynamic	0.53/0.86	0.79/1.62	0.56/2.27	1.06/3.49
Rigid + Filter Dynamic + Others	0.52/0.93	0.76/1.51	0.50/2.19	1.01/3.46
PnP + RANSAC	0.49/1.04	**0.58**/1.64	**0.41**/2.72	1.45/4.72
Rigid + RNASAC	**0.46**/1.13	0.60/1.73	0.46/3.30	1.12/3.58
Rigid + Weights [ours]	**0.46/0.81**	0.61/**1.09**	0.44/**1.71**	**0.87/2.88**

E. Results: outliers filtering

Our method estimates weighting factors for each 3D-3D correspondence. These account for imperfections in 3D coordinates predictions and aim to down-weigh 3D correspondences that lead to inferior localization results (i.e., outliers). We conduct experiments to assess the impact of these weighting factors. Since our method relies on a single image for localization, we consider the following methods:

Rigid + No Filtering: we compute pose using rigid alignment without incorporating the obtained weights. That is, all predicted 3D correspondences are of equal importance.

Rigid + Filter Dynamic: we utilize semantics to filter out contributions from sources of outliers, mainly dynamic objects. Specifically, we utilize the recently released InternImage [28] to segment the input image into semantic classes. We filter out 3D points that correspond to pixels of dynamic classes. These are pedestrians, cars, bicycles, and trucks.

Rigid + Filter Dynamic + Others: we complement the removal of dynamic points by pruning 3D points that correspond to semantic classes that are presumably inferior to localization such as sky and trees.

Rigid + RANSAC: we utilize a robust outlier filter by pairing the rigid alignment algorithm with a RANSAC scheme [29]. We apply RANSAC with a maximum of 2000 iterations and an inlier threshold of 10 cm between corresponding 3D points. We use 10 correspondences for pose estimation.

PnP + RANSAC: our method uses poses to obtain 3D coordinates in the global coordinate system of the scene. Besides utilizing rigid-alignment, our method offers the flexibility to adopt perspective-n-point (PnP) algorithm for pose estimation using the 3D global coordinates and the corresponding 2D pixels. We compute pose using PnP [30] from 4 correspondences using RANSAC [29]. We set 2000 as maximum number of RANSAC iterations, with an inlier threshold of 10 pixels.

Rigid + Weights [ours]: we compute pose using weighted rigid alignment where the weights are the ones obtained by our method. These are obtained directly with a forward pass of the network without further processing of the obtained 3D correspondences or any off-the-shelf outlier filter.

Tab. III lists the results on Cambridge Landmarks [1]. Pruning off 3D points that correspond to dynamic objects by semantic segmentation shows improvements in localization compared to utilizing all the 3D points. In addition to removing dynamic points, filtering out correspondences from non-informative regions like sky and trees obtains additional improvements. The hospital scene does not contain dynamic objects which justifies the same result that is obtained without filtering any 3D-3D correspondence. While semantics can be used to filter complete dynamic objects, it might not filter all sources of outliers. Besides, it is subject to segmentation errors and may filter many useful features. Pairing a robust outlier filter such as RANSAC [29] with both PnP [30] and rigid alignment shows further improvements in estimated translation. However, it leads to increased errors on the Church scene and elevated orientation errors overall. We trace this back to the reason that many 3D points may be inferior and that few of them are subject to inlier checks. The Church scene poses a difficult localization scenario where the camera moves 360° around the scene. To overcome this and improve further, it would require increasing the number of considered points, relaxing the inlier threshold and/or increasing the maximum number of iterations. In summary, utilizing RANSAC [29] for

filtering outliers requires adjusting hyperparameters for each scene besides imposing a run-time burden.

In contrast to the mentioned methods, using the set of weights that are obtained by our method delivers a consistent reduction in position and orientation localization errors. These weights form a prior about the useful 3D coordinates for localization. It considers all correspondences for localization, nevertheless, with different weights. We complement these quantitative observations by showing visualizations of the network outputs in Fig. 3. Inspecting the visualization, we observe that the network guesses the architecture of the scene from pose labels only. Furthermore, the heatmaps of the weights show that it focuses on a small subset of features mainly edges and corners, implying that the correct geometric predictions lay on these features. It obtains an estimation of the depth and 3D scene coordinates without supervised labels for these quantities.

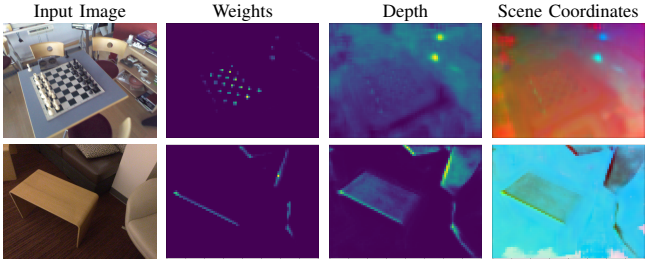

Fig. 3: Visual samples of the predictions obtained by the network on samples from 7Scenes. For visualizing the 3D coordinates, we map the X, Y, Z coordinates to RGB values.

F. Results: finetuning with position labels only

In this section, we test our method with limited training samples and also, explore the possibility of using only partial labels (only position information and not orientation). This partial position only labels can be easily available from GPS or can be available on the fly, using another sensor. Testing with limited data is challenging for deep learning-based methods as they require more data for generalization. Accordingly, we sample a third of the dataset for training by taking every third training sample. The testing samples are kept in the original size as provided by the dataset. After training, we finetune the network for a few epochs (10-15) by feeding it with the other two-thirds of the training samples. However, we assume that these additional training samples have only partial geometric labels (translation only instead of 6 DoF poses). Thus, we apply only the translation loss for finetuning from Equation. (3). The consistency (Equation. (5)) and re-projection (Equation. (7)) losses are excluded as they require full 6 DoF poses. In addition, we apply the same training scheme (training with absolute position labels) on one of state-of-the-art pose regression methods: MapNet [8]. The results are listed in Table IV. As expected, the localization accuracy dropped when training with fewer samples. MapNet [8] yields a larger drop in accuracy than our method. Our method reports a little reduction in accuracy on indoor scenes. Finetuning the model given only position labels has reduced both the translation and rotation errors on some of the scenes. In contrast, MapNet [8] has significant increase in orientation errors when finetuned with position labels. The reason behind the better performance of our algorithm is driven by two correlated reasons. The first is the separation between image representations and pose estimation that is inherent to our algorithm, while the second being, the computation of the pose using non-learned and parameter-free rigid-alignment which updates both 3D point clouds from position supervision. The rigid alignment module passes gradients to all the network branches that predict the geometric quantities. Finetuning using only positional labels works well. While the accuracy did not reach that of training on the complete dataset, the improvements for both location and orientation given only translation labels open doors for further research in this direction.

G. Results: run-time

In Table V, we report the run-time of our method for different down-sampling factors (resolutions) on an input with a standard resolution 480×640. We run our python implementation on a machine equipped with GTX Titan X and Intel Core i7-5960X CPU @ 3.00GHz. The results imply that our method localizes in real run-time. We also report the run-time of a minimal pose regression pipeline (PoseNet [1]) using MobileNetV3 backbone [25]. Some state-of-the-art regression methods further process the output of the network before pose regression by applying attention [9], graph neural networks [14] and transformers [15], demanding additional run-time.

V. CONCLUSION

We presented a novel method for global 6 DoF pose estimation from a single RGB image. The main novelty of our method is the use of pose targets only to guide a deep neural network, through differentiable rigid alignment, to estimate the scene geometry without explicit ground-truth of this geometry at training time. The proposed method takes a single image and implicitly obtains geometric representations of the scene using only pose labels. These implicitly learned geometric representations are the 3D scene geometry (X, Y, Z coordinates) in two reference frames: global and camera coordinate systems. We utilize a parameter-free and differentiable rigid alignment to pass gradients through a deep neural network to adjust its weights and continually learn these representations without explicit ground-truth labels. Besides pose loss, another novelty is that our method allows for the inclusion of additional learning losses as opposed to learning a localization pipeline by pose regression. We introduce a consistency loss to make the two geometric representations consistent with the geometric pose and a re-projection loss to constrain the 3D global coordinates to the 2D image pixels. Through extensive experiments, we show that the proposed method exceeds the localization accuracy of state-of-the-art regression methods and runs in real-time. As a final contribution, we show that the proposed formulation can utilize partial labels (instantaneous position labels only) to finetune a pre-trained model leading to improvements in both position and orientation localization. In future, we would like to leverage foundational models such as SAM [31] and CLIP [32] to generate embeddings and integrate them into our learned 3D representations to perform more accurate pose estimation using scene semantics.

TABLE IV: Results of experiment of section IV-F. Median errors (meters/degrees) are reported. Improvements as a result of finetuning by $L_{position}$ over training on 1/3 of samples are marked by underlines.

Method	Training Scheme	College	Hospital	Shop	Church	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs
Ours	trained on all samples	0.46/0.81	0.61/1.09	0.44/1.71	0.87/2.88	0.05/1.42	0.16/4.57	0.09/6.14	0.13/3.46	0.12/2.48	0.10/2.51	0.17/3.48
	trained on 1/3 of samples	0.52/0.85	0.69/1.49	0.59/2.53	0.95/3.04	0.06/1.60	0.18/4.73	0.13/7.7	0.14/3.48	0.11/2.73	0.12/2.81	0.19/3.50
	finetuned with $L_{position}$	0.50/0.88	0.68/1.32	0.59/2.44	0.93/2.95	0.05/1.55	0.18/5.28	0.13/7.7	0.12/3.41	0.12/3.13	0.11/2.63	0.18/3.74
MapNet [8]	trained on all samples	0.94/1.99	2.03/3.60	0.80/6.34	1.66/4.01	0.08/3.25	0.27/11.7	0.18/13.3	0.17/5.15	0.22/4.02	0.23/4.93	0.30/12.1
	trained on 1/3 of samples	1.12/6.10	3.14/7.81	1.29/8.76	2.65/6.54	0.12/4.75	0.30/11.51	0.18/14.02	0.20/6.20	0.21/5.33	0.27/5.76	0.37/11.53
	finetunned with $L_{position}$	1.25/93.02	3.03/139.79	1.34/95.01	2.63/53.50	0.12/38.98	0.31/24.68	0.18/20.28	0.19/41.52	0.22/35.62	0.26/32.00	0.37/24.23

TABLE V: Run-time analysis (section IV-G). Data Processing: time needed to run the network and obtain the 3D clouds. Pose Computation: time needed to obtain pose through rigid alignment. FPS: frames per second.

	Down-sampling Factor	Output Resolution	Data Processing	Pose Computation	Total (ms) -FPS
ours	4	120×160	9.0 ms	30.0 ms	39.0 ms - 26 Hz
	8	60×80	9.2 ms	4.8 ms	14.0 ms - 71 Hz
	16	30×40	9.6 ms	1.5 ms	11.1 ms - 90 Hz
				Regression	12.5 ms - 80 Hz

ACKNOWLEDGMENT

Joan Serrat acknowledges the support of the Generalitat de Catalunya CERCA Program to CVC's general activities.

REFERENCES

[1] A. Kendall, M. Grimes, and R. Cipolla, "PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization," in *IEEE International Conference on Computer Vision*, 2015, pp. 2938–2946.

[2] A. Kendall and R. Cipolla, "Geometric Loss Functions for Camera Pose Regression with Deep Learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5974–5983.

[3] M. Cai, C. Shen, and I. D. Reid, "A Hybrid Probabilistic Model for Camera Relocalization," in *British Machine Vision Conference*, 2018.

[4] F. Walch, C. Hazirbas, L. Leal-Taixé, T. Sattler, S. Hilsenbeck, and D. Cremers, "Image-Based Localization Using LSTMs for Structured Feature Correlation," in *IEEE International Conference on Computer Vision*, 2017, pp. 627–637.

[5] T. Naseer and W. Burgard, "Deep Regression for Monocular Camera-Based 6-DOF Global Localization in Outdoor Environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 1525–1530.

[6] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu, "Image-Based Localization Using Hourglass Networks," in *IEEE International Conference on Computer Vision Workshop*, 2017, pp. 870–877.

[7] J. Wu, L. Ma, and X. Hu, "Delving Deeper into Convolutional Neural Networks for Camera Relocalization," in *IEEE International Conference on Robotics and Automation*, 2017, pp. 5644–5651.

[8] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz, "Geometry-Aware Learning of Maps for Camera Localization," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2616–2625.

[9] B. Wang, C. Chen, C. X. Lu, P. Zhao, N. Trigoni, and A. Markham, "Atloc: Attention Guided Camera Localization," in *AAAI Conference on Artificial Intelligence*, vol. 34, no. 06, 2020, pp. 10 393–10 401.

[10] F. Ott, T. Feigl, C. Loffler, and C. Mutschler, "ViPR: Visual-Odometry-Aided Pose Regression for 6DOF Camera Localization," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 42–43.

[11] R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen, "Vidloc: A Deep Spatio-Temporal Model for 6-DOF Video-Clip Relocalization," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6856–6864.

[12] A. Valada, N. Radwan, and W. Burgard, "Deep Auxiliary Learning for Visual Localization and Odometry," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 6939–6946.

[13] X. Li and H. Ling, "PoGO-Net: Pose Graph Optimization with Graph Neural Networks," in *IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5875–5885.

[14] F. Xue, X. Wu, S. Cai, and J. Wang, "Learning Multi-View Camera Relocalization With Graph Neural Networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 372–11 381.

[15] Y. Shavit, R. Ferens, and Y. Keller, "Learning Multi-Scene Absolute Pose Regression with Transformers," in *IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2713–2722.

[16] A. Moreau, N. Piasco, D. Tsishkou, B. Stanciulescu, and A. de La Fortelle, "CoordiNet: Uncertainty-Aware Pose Regressor for Reliable Vehicle Localization," in *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 1848–1857.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[18] E. Brachmann and C. Rother, "Visual Camera Re-Localization From RGB and RGB-D Images Using DSAC," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5847–5865, 2022.

[19] M. Altillawi, "PixSelect: Less but Reliable Pixels for Accurate and Efficient Localization," in *IEEE International Conference on Robotics and Automation*, 2022, pp. 4156–4162.

[20] H. Blanton, S. Workman, and N. Jacobs, "A Structure-Aware Method for Direct Pose Estimation," in *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 205–214.

[21] M. Altillawi, Z. Pataki, S. Li, , and Z. Liu, "Global Localization: Utilizing Relative Spatio-Temporal Geometric Constraints from Adjacent and Distant Cameras," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023.

[22] W. Kabsch, "A Solution for the Best Rotation to Relate Two Sets of Vectors," *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, vol. 32, pp. 922–923, 1976.

[23] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2930–2937.

[24] J. Valentin, A. Dai, M. Nießner, P. Kohli, P. Torr, S. Izadi, and C. Keskin, "Learning to Navigate the Energy Landscape," in *IEEE International Conference on 3D Vision*, 2016, pp. 323–332.

[25] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, *et al.*, "Searching for Mobilenetv3," in *IEEE/CVF International Conference on Computer Vision*, 2019.

[26] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, *et al.*, "Deep High-Resolution Representation Learning for Visual Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 10, pp. 3349–3364, 2020.

[27] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[28] W. Wang, J. Dai, Z. Chen, Z. Huang, Z. Li, X. Zhu, X. Hu, T. Lu, L. Lu, H. Li, *et al.*, "Internimage: Exploring Large-Scale Vision Foundation Models with Deformable Convolutions," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 14 408–14 419.

[29] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[30] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, "Complete Solution Classification for the Perspective-Three-Point Problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 8, pp. 930–943, 2003.

[31] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, "Segment Anything," *arXiv preprint arXiv:2304.02643*, 2023.

[32] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning Transferable Visual Models from Natural Language Supervision," in *International Conference on Machine Learning*, 2021, pp. 8748–8763.