Understanding the Limitations of CNN-based Absolute Camera Pose Regression

Torsten Sattler[1] Qunjie Zhou[2] Marc Pollefeys[3,4] Laura Leal-Taixé[2]
[1]Chalmers University of Technology [2]TU Munich [3]ETH Zürich [4]Microsoft

Abstract

Visual localization is the task of accurate camera pose estimation in a known scene. It is a key problem in computer vision and robotics, with applications including self-driving cars, Structure-from-Motion, SLAM, and Mixed Reality. Traditionally, the localization problem has been tackled using 3D geometry. Recently, end-to-end approaches based on convolutional neural networks have become popular. These methods learn to directly regress the camera pose from an input image. However, they do not achieve the same level of pose accuracy as 3D structure-based methods. To understand this behavior, we develop a theoretical model for camera pose regression. We use our model to predict failure cases for pose regression techniques and verify our predictions through experiments. We furthermore use our model to show that pose regression is more closely related to pose approximation via image retrieval than to accurate pose estimation via 3D structure. A key result is that current approaches do not consistently outperform a handcrafted image retrieval baseline. This clearly shows that additional research is needed before pose regression algorithms are ready to compete with structure-based methods.

1. Introduction

Visual localization algorithms enable a camera to determine its absolute pose, *i.e.*, its position and orientation, in a scene and are a core component of intelligent systems [24, 38] and Augmented Reality applications [15, 48].

State-of-the-art algorithms for localization follow a 3D structure-based approach [8, 10, 16, 46, 59, 63, 68, 69]. They first establish correspondences between pixels in a test image and 3D points in the scene. These 2D-3D matches are then used to estimate the camera pose by applying an n-point-pose (PnP) solver [2, 31–34] inside a RANSAC [18, 22, 36, 54] loop. Traditionally, the first stage is based on matching descriptors extracted in the test image against descriptors associated with the 3D points. Alternatively, machine learning techniques can be used to directly regress 3D point positions from image patches [8, 10, 16, 43, 46, 47, 65].

In recent years, absolute pose regression (APR) approaches to visual localization have become popular [11, 12, 28–30, 44, 50, 53, 74, 76, 78]. Rather than using machine learning only for parts of the localization pipeline, *e.g.*, local features [63, 79], outlier filtering [49, 70], or scene coordinate regression [10, 46], these approaches aim to learn the full localization pipeline. Given a set of training images and their corresponding poses, APR techniques train Convolutional Neural Networks (CNNs) to directly regress the camera pose from an image. APR techniques are computationally efficient, given a powerful enough GPU, as only a single forward pass through a CNN is required. Yet, they are also significantly less accurate than structure-based methods [10, 63, 76]. In addition, updating the map, *e.g.*, when adding new data, requires expensive retraining of the CNN.

Rather than proposing a new APR variant, this paper focuses on understanding APR techniques and their performance. To this end, we make the following contributions: **i**) We develop a theoretical model for absolute pose regression (Sec. 3). To the best of our knowledge, ours is the first work that aims at looking at the inner workings of APR techniques. Based on this model, we show that APR approaches are more closely related to approximate pose estimation via image retrieval (Sec. 5) than to accurate pose estimation via 3D geometry (Sec. 4). **ii**) Using our theory, we show both theoretically and through experiments that there is no guarantee that APR methods, unlike structure-based approaches, generalize beyond their training data (Sec. 4). **iii**) Given the close relation between APR and image retrieval, we show that current APR approaches are much closer in performance to a handcrafted retrieval baseline [71] than to structure-based methods. We show that no published single image pose regression approach is able to consistently outperform this baseline. This paper thus introduces a highly necessary sanity check for judging the performance of pose regression techniques.

In summary, this work closes an important gap in the understanding of absolute pose regression methods to visual localization: It clearly demonstrates their short-comings and more clearly positions them against other ways to approach the visual localization problem. Overall, we show that a significant amount of research is still necessary before absolute pose regression techniques can be applied in practical applications that require accurate pose estimates.

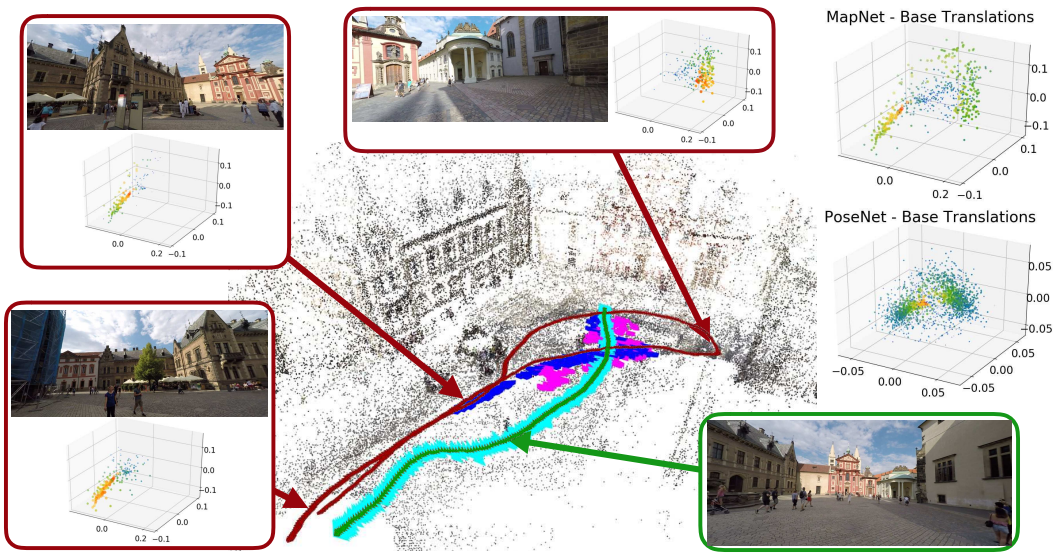

Figure 1. Visualization of the base translations $\{\mathbf{c}_j\}$ learned by PoseNet [29, 30] and MapNet [11]. Each point corresponds to one base translation. The scale of the base translations is in meters. We show the combinations of base translations for some training images for MapNet. The weight a translation received in Eq. 3 for a single image, respectively all images (on the right of the figure), is indicated by colors and point sizes, with warm colors and large points for translations with a large coefficient. The training and test trajectory are shown in red and green. The test predictions by PoseNet and MapNet and Active Search [59] are shown in blue, purple, and cyan, respectively.

2. Related Work

Structure-based localization approaches rely on 2D-3D matches between 2D pixel positions and 3D scene coordinates for pose estimation. These matches are established by descriptor matching [21, 37, 39, 59, 63, 68, 69, 81] or by regressing 3D coordinates from pixel patches [8–10, 16, 23, 43, 46, 47, 65]. Descriptor-based methods handle city-scale scenes [37, 39, 68, 81] and run in real-time on mobile devices [6, 38, 41, 48]. 3D coordinate regression methods currently achieve a higher pose accuracy at small scale, but have not yet been shown to scale to larger scenes [10, 69].

Image retrieval is typically used for place recognition [3, 5, 17, 58, 71, 72, 77, 80], *i.e.*, for determining which part of a scene is visible in a given image. State-of-the-art approaches use compact image-level descriptors to enable efficient and scalable retrieval [3, 52, 71]. Image retrieval can be used for visual localization by approximating the pose of a test image by the pose of the most similar retrieved image. More precise estimates can be obtained by using feature matches between the test image and the retrieved images for relative pose estimation [13, 82, 83]. Image retrieval has also been used as part of structure-based approaches [14, 26, 57].

Absolute camera pose regression (APR) approaches train CNNs to regress the camera pose of an input image [11, 28–30, 44, 50, 53, 74, 76, 78], thus representing the scene implicitly by the weights of the networks. They all follow the same pipeline: Features are extracted using a base network, *e.g.*, VGG [66] or ResNet [25], which are then embedded into a high-dimensional space. This embedding is then used to regress the camera pose in the scene. Existing approaches mainly differ in the underlying base architecture and the loss function used for training, *e.g.*, using a weighted combination of position and orientation errors [11, 30, 76], geometric reprojection errors [29], or adding visual odometry constraints [11, 53, 74]. [50, 78] extend the set of training images with synthetic data. [12, 28] also reason about the uncertainty of the estimated poses. Rather than using a single image, [11, 19, 53, 74] propose methods based on localizing sequences of images.

Recent results show that APR methods are significantly less accurate than structure-based methods [10, 46, 76]. This paper aims to understand these results by developing a theoretical model for APR. Based on this model, we show that, in contrast to structure-based methods, APR approaches struggle to generalize beyond their training data or might not generalize at all. Furthermore, we show that APR techniques are inherently closer related to image retrieval than to structure-based methods and that current APR algorithms do not consistently outperform a retrieval baseline.

Relative camera pose regression (RPR) approaches predict the pose of a test image relative to one or more training images rather than in absolute scene coordinates [7, 35, 45, 56]. The prediction is again handled by a CNN trained for regression. Relevant training images can be found using an explicit image retrieval step [7, 35] or by implicitly representing the images in the CNN [56]. APR is an instance-level problem, *i.e.*, APR techniques need to be trained for a specific scene. In contrast, RPR is a more general problem and RPR methods can be trained on multiple scenes [7, 35].

In this paper, we use our theory of APR to show that there is an inherent connection to RPR. We also show that, while being among the best-performing end-to-end localization approaches, current RPR techniques also do not consistently outperform an image retrieval baseline.

3. A Theory of Absolute Pose Regression

The purpose of this section is to develop a theoretical model for absolute camera pose estimation methods such as PoseNet [28–30] and its variants [11,50,76,78]. Our theory is not tied to a specific network architecture but covers the family of architectures used for pose regression. Based on this theory, Sec. 4 compares absolute pose regression and structure-based methods, using experiments to support our model. Sec. 5 then uses the theory to show the inherent similarities between pose regression and image retrieval.

Notation Let $\mathcal{I}$ be an image taken from a camera pose $\mathbf{p}_\mathcal{I} = (\mathbf{c}_\mathcal{I}, \mathbf{r}_\mathcal{I})$. Here, $\mathbf{c}_\mathcal{I} \in \mathbb{R}^3$ is the camera position and $\mathbf{r}_\mathcal{I}$ is the camera orientation. There are multiple ways to represent the orientation, *e.g.*, as a 4D unit quaternion [30,76] or its logarithm [11], or as a 3D vector representing an angle and an axis [7,73]. The exact choice of representation is not important for our following analysis. Without loss of generality, we thus simply represent the orientation as a r-dimensional vector $\mathbf{r}_\mathcal{I} \in \mathbb{R}^r$. Absolute camera poses are thus represented as points in $\mathbb{R}^{3+r}$.

Absolute pose regression. Given a test image $\mathcal{I}$, the task of absolute camera pose regression is to predict the pose from which the image was taken. This pose is defined with respect to a given scene coordinate frame. To solve this tasks, algorithms for absolute camera pose regression learn a visual localization function $L(\mathcal{I}) = \hat{\mathbf{p}}_\mathcal{I}$, where $\hat{\mathbf{p}}_\mathcal{I} = (\hat{\mathbf{c}}_\mathcal{I}, \hat{\mathbf{r}}_\mathcal{I})$ is the camera pose predicted for image $\mathcal{I}$. In the following, we will focus on methods that represent the function L via a convolutional neural network (CNN) [11,28–30,76].

Absolute camera pose regression is an instance level problem. Thus, CNN-based methods for absolute pose regression use a set of images of the scene, labeled with their associated camera poses, as training data. Additional image sequences without pose labels might also be used to provide additional constraints [11,74]. The training objective is to minimize a loss $\mathcal{L}(\hat{\mathbf{p}}_\mathcal{I}, \mathbf{p}_\mathcal{I})$ enforcing that the predicted pose $\hat{\mathbf{p}}_\mathcal{I}$ is similar to the ground truth pose $\mathbf{p}_\mathcal{I}$. The precise formulation of the loss is not important for our analysis.

A theory of absolute pose regression. We divide absolute pose regression via a CNN into three stages: The first stage, representing a function $F(\mathcal{I})$, extracts a set of features from the image. This stage is typically implemented using the fully convolutional part of a CNN such as VGG [66] or ResNet [25]. The second stage computes a (non-linear) embedding $E(F(\mathcal{I}))$ of the features into a vector $\alpha^\mathcal{I} = (\alpha_1^\mathcal{I}, \dots, \alpha_n^\mathcal{I})^T \in \mathbb{R}^n$ in a high-dimensional space. This embedding typically corresponds to the output of the second-to-last layer in a pose regression method. The last stage performs a linear projection from the embedding space into the space of camera poses. This third stage corresponds to the last (fully-connected) layer in the network. This three stage model covers all PoseNet-like approaches that have been published so far.

Treating the first two stages as a single network, we can write the trained visual localization function L as

$$\begin{aligned} L(\mathcal{I}) &= \mathbf{b} + \mathtt{P} \cdot E(F(\mathcal{I})) \\ &= \mathbf{b} + \mathtt{P} \cdot \left(\alpha_1^\mathcal{I}, \dots, \alpha_n^\mathcal{I}\right)^T \quad , \end{aligned} \tag{1}$$

where $\mathtt{P} \in \mathbb{R}^{(3+r)\times n}$ is a projection matrix and $\mathbf{b} \in \mathbb{R}^{3+r}$ is a bias term. The output of $L(\mathcal{I})$ is an estimate $\hat{\mathbf{p}}_\mathcal{I} = (\hat{\mathbf{c}}_\mathcal{I}, \hat{\mathbf{r}}_\mathcal{I})$ of the image's camera pose. Let $\mathbf{P}_j \in \mathbb{R}^{3+r}$ be the j^{th} column of $\mathtt{P}$. We can express the predicted camera pose as a linear combination of the columns of $\mathtt{P}$ via

$$L(\mathcal{I}) = \mathbf{b} + \sum_{j=1}^{n} \alpha_j^\mathcal{I} \mathbf{P}_j = \begin{pmatrix} \hat{\mathbf{c}}_\mathcal{I} \\ \hat{\mathbf{r}}_\mathcal{I} \end{pmatrix} \quad . \tag{2}$$

We further decompose the j^{th} column $\mathbf{P}_j$ of the projection matrix $\mathtt{P}$ into a translational part $\mathbf{c}_j \in \mathbb{R}^3$ and an orientation part $\mathbf{r}_j \in \mathbb{R}^r$, such that $\mathbf{P}_j = (\mathbf{c}_j^T, \mathbf{r}_j^T)^T$. Similarly, we can decompose the bias term $\mathbf{b}$ as $\mathbf{b} = (\mathbf{c}_b^T, \mathbf{r}_b^T)^T$, resulting in

$$\begin{pmatrix} \hat{\mathbf{c}}_\mathcal{I} \\ \hat{\mathbf{r}}_\mathcal{I} \end{pmatrix} = \begin{pmatrix} \mathbf{c}_b + \sum_{j=1}^{n} \alpha_j^\mathcal{I} \mathbf{c}_j \\ \mathbf{r}_b + \sum_{j=1}^{n} \alpha_j^\mathcal{I} \mathbf{r}_j \end{pmatrix} \quad . \tag{3}$$

Note that Eq. 3 also covers separate embeddings and projections for the position and orientation of the camera, *e.g.*, as in [78]. In this case, the projection matrix has the form

$$\mathtt{P} = \begin{pmatrix} \mathbf{c}_1 & \dots & \mathbf{c}_k & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \dots & \mathbf{0} & \mathbf{r}_{k+1} & \dots & \mathbf{r}_n \end{pmatrix} \quad . \tag{4}$$

Intuitive interpretation. Eq. 3 leads to the following interpretation: Absolute pose regression algorithms such as PoseNet and its variants learn a set $\mathcal{B} = \{(\mathbf{c}_j, \mathbf{r}_j)\}$ of *base poses* such that the poses of all training images can be expressed as a *linear combination* of these base poses[1]. How much a base pose contributes to a predicted pose depends on the appearance of the input image: The first stage $F(\mathcal{I})$ provides a set of feature response maps. The second stage $E(F(\mathcal{I}))$ then generates a high-dimensional vector $\alpha^\mathcal{I} = (\alpha_1^\mathcal{I}, \dots, \alpha_n^\mathcal{I})^T$. Each entry $\alpha_j^\mathcal{I}$ is computed by correlating feature activations from the first stage [76] and corresponds to a base pose $(\mathbf{c}_j, \mathbf{r}_j)$. The $\alpha_j^\mathcal{I}$ provide the importance of each base pose for a given input image.

Fig. 1 visualizes the translational part $\{\mathbf{c}_j\}$ of the base poses learned by PoseNet [29, 30] and MapNet [11], together with the combinations used for individual training

[1] In practice, most methods usually compute a *conical* combination as they use a ReLU activation before the linear projection.

images. As can be seen from the scale of the plots (in meters), $\{\mathbf{c}_j\}$ corresponds to a set of translations with small magnitude. Essentially, the network learns to sum these translations up to an absolute pose by scaling them appropriately via the coefficients in the embedding (*c.f.* Eq. 3). For this reason, we refer to the $\{\mathbf{c}_j\}$ as *base translations* rather than *base positions*. Notice that the base translations in Fig. 1 approximately lie in a plane since all training poses lie in a plane. The supplementary video shows how the base translations change with changing image content.

4. Comparison with Structure-based Methods

Visual localization algorithms represent a mapping from image content to the camera pose from which the image was taken. The current gold standard for localization are structure-based approaches [8, 10, 16, 46, 59, 63, 68, 69]. These methods establish correspondences between 2D pixel positions in an image and 3D point coordinates in the scene. The camera pose is then computed by solving the PnP problem, *i.e.*, by finding a pose that maximizes the number of 3D points projecting close to their corresponding 2D positions. As long as there are sufficiently many correct matches, structure-based methods will be able to estimate a pose.

In contrast to structure-based methods, pose regression algorithms do not explicitly use knowledge about projective geometry. Rather, they learn the mapping from image content to camera pose from data. Based on our theory, absolute pose regression methods are expressive enough to be able to learn this mapping given enough training data: Changes in image content lead to different features maps $F(\mathcal{I})$, which lead to a change in the embedding $E(F(\mathcal{I}))$, and thus a different pose (*c.f.* Eq. 3). Assuming the right network architecture, loss function, and sufficient training data, it should thus be possible to train an APR approach that accurate estimates camera poses for novel viewpoints.

In practice, collecting a vast amount of images, computing the training poses (*e.g.*, via SfM), and training a CNN on large amounts of data are all highly time-consuming tasks. Thus, methods that are able to accurately predict poses using as little training data as possible are preferable. In the following, we use our theoretical model to predict failure cases for pose regression techniques in scenarios with limited training data. We validate our predictions experimentally. In addition, we show that structure-based methods, as can be expected, are able to handle these situations.

Experimental setup. For the practical experiments used in this section, we recorded new datasets[2] . We deliberately limited the amount of training data to one or a few trajectories per scene and captured test images from differing viewpoints. Ground truth poses for training and testing data were obtained using SfM [62]. We scaled the resulting 3D models to meters by manually measuring distances. For evaluation, we use both PoseNet [29, 30] and MapNet [11]. We use the PoseNet variant that learns the coefficient weighting position and orientation errors during training [29]. Both methods are state-of-the-art absolute pose regression algorithms. We use Active Search [59] to obtain baseline results for structure-based methods. Active Search uses RootSIFT [4,40] features to establish 2D-3D matches. It is based on prioritized matching, terminating correspondence search once 200 matches have been found. The pose is estimated via a P3P solver [31] inside a RANSAC [18] loop, followed by non-linear refinement of the pose [1]. The 3D model required by Active Search is build by matching each training image against nearby training images and triangulating the resulting matches using the provided training poses.

Training data captured on a line or parallel lines. Let $\mathcal{T} = \{(\mathcal{I}, \mathbf{p}_\mathcal{I} = (\mathbf{c}_\mathcal{I}, \mathbf{r}_\mathcal{I}))\}$ be a set of training images with their corresponding camera poses. As shown in Sec. 3, camera pose regression techniques express the camera pose of an image as a linear combination of a set of learned base poses. Consider a scenario where all training camera positions lie on a line. This represents the most basic data capture scenario, *e.g.*, for data captured from a car such as the large-scale San Francisco [17] and RobotCar [42] datasets.

In this scenario, each camera position $\mathbf{c}_\mathcal{I}$ corresponds to a point on a line $\mathbf{o} + \delta\mathbf{d}$. Here, $\mathbf{o} \in \mathbb{R}^3$ is a point on the line, $\mathbf{d} \in \mathbb{R}^3$ is the direction of the line, and $\delta \in \mathbb{R}$ is a scaling factor. One admissible solution to the training problem, although not the only one, is thus to place all base translations $\mathbf{c}_j$ on the line $\mathbf{o} + \delta\mathbf{d}$. As any linear combinations of points on a line lies on the line, this solution *will never generalize*.

Fig. 2 shows two examples for this scenario: In the first one, training data was captured while riding an escalator upwards. Testing data was acquired while riding the escalator down (looking again upwards) in another lane. In the second example, training data was acquired while walking parallel to building facades while test data was acquired from a bit farther away. In both cases, MapNet clearly places most base translations along a line. While there are some translations not on the line, these are mostly used to handle camera shake (*c.f.* the supp. video). As a result, MapNet places its estimates of the test poses on or close to the resulting line and does not generalize to diverging viewpoints. This clearly shows that solutions to the training problem that are guaranteed to not generalize are not only of theoretical interest but can be observed in practice.

The base translations estimated by PoseNet are significantly more noisy and do not all lie on a line. Interestingly, PoseNet still places all test poses on a line through the training images. While the base poses thus span a space larger than positions on the line, PoseNet is still not able to generalize. This is due to a failure of mapping the image appear-

[2]The datasets are available at https://github.com/tsattler/understanding_apr.

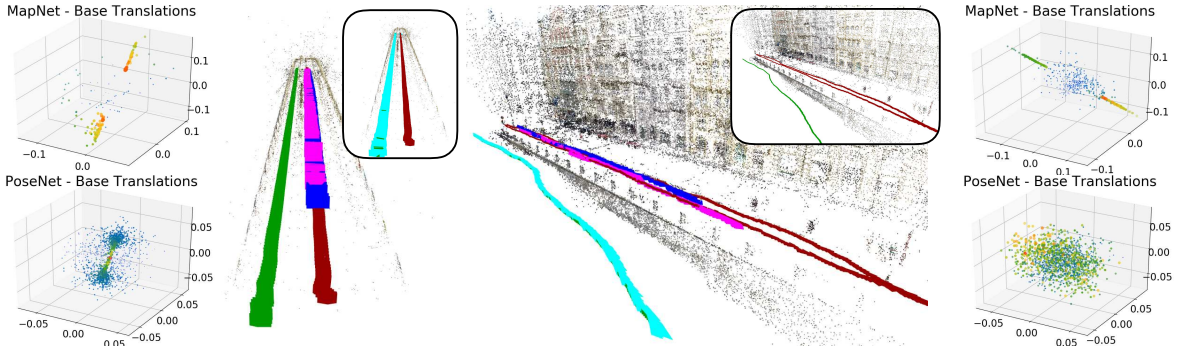

Figure 2. Example scenarios in which two absolute pose regression techniques, PoseNet [29, 30] and MapNet [11], fail to generalize. For both scenes, the networks learn to (roughly) interpolate along a line. Consequently, the poses of the test images are also placed along this line. Please see the caption of Fig. 1 for details on the color coding and the base translations shown for the two scenes.

ance to suitable weights for the base poses, showing that multiple solutions exists that do not generalize. In contrast, Active Search handles both scenarios well (*c.f*. Fig. 2).

More general trajectories. The argument above exploits that it is not necessary that the base translations span the space of all possible translations to explain a set of images taken on a line. If the training trajectory is more general, *e.g*., covering all directions in a plane in the case of planar motion, this argument is not applicable anymore.

For more general training trajectories, it is usually possible to express each viable test pose as a linear combination of the base poses. However, this is only a necessary but not a sufficient condition for generalization. As evident from Eq. 3, absolute pose regression techniques couple base poses to image appearance via the coefficients $\alpha_j^{\mathcal{I}}$.

Consider a part $\mathcal{P}'$ of the scene defined by a subset $\mathcal{T}' = \{\mathcal{I}\}$ of the training images. The corresponding relevant subset $\mathcal{B}'(\mathcal{P}')$ of the base poses $\mathcal{B} = \{(\mathbf{c}_j, \mathbf{r}_j)\}$ is

$$\mathcal{B}'(\mathcal{P}') = \{(\mathbf{c}_j, \mathbf{r}_j) | \text{ exists } \mathcal{I} \in \mathcal{T}' \text{ with } |\alpha_j^{\mathcal{I}}| > 0\} \quad . \quad (5)$$

A stronger necessary condition for generalization is that the linear span of each such $\mathcal{B}'(\mathcal{P}')$ contains the poses of all test images in $\mathcal{P}'$[3]. In the following, we show that this is not necessarily guaranteed in practice.

Figs. 1 and 3 show scenes with more general motion. For each scene, we show the training and ground truth testing trajectories, as well as the test trajectories estimated by PoseNet, MapNet, and Active Search. In addition, we show the base translations used by the two networks. Since the training images are taken in a plane, the base translations also lie in a plane (up to some noise). As can be seen, the networks are able to generalize in some parts of the scene, *e.g*., when the test trajectory crosses the training trajectory in Fig. 1. In other parts, they however seem to resort to some form of nearest neighbor strategy: Test poses are placed close to parts of the training trajectory with similar image appearance. In these parts, the relevant base translations are not sufficient to model the test positions more accurately. This shows that more training data is required in these regions. It also shows that networks do not automatically benefit from recording more data in unrelated parts of the scene.

As can be expected, Active Search fails or produces inaccurate pose estimates when there is little visual overlap between the test and training images (*c.f*. the example test image in Fig. 3(left), where the wall visible in the image is not seen during training). Still, Active Search overall handles viewpoint changes significantly better.

Fig. 4 shows a more complex example, where the training data is captured on multiple parallel lines and should be sufficient to explain the test poses. In this case, both networks are able to estimate poses close to these lines, but are not able to properly interpolate between them and do not generalize beyond them. Active Search mostly handles the large viewpoint changes between training and testing images. If the change is too large however, it fails to find enough matches and thus to estimate a pose. Local features that are more robust to large viewpoint changes are an active field of research [51, 55] and structure-based methods will automatically benefit from progress in this field.

Using densely sampled training data. Training using more data in a part of the scene should intuitively improve the prediction accuracy of pose regression techniques. To verify this assumption, we use synthetic data: We created a 3D model of the Shop Facade scenes from the Cambridge Landmarks dataset [30] using multi-view stereo [64]. We then rendered [75] the scene from the poses of the original training and testing images, as well as from a set of additional poses. These poses are placed on a regular grid in the plane containing the original poses, with a spacing of 25cm between poses. We only created poses up to 3 meters away from the original training poses. The orientation of each additional pose is set to that of the nearest training pose. Varying the maximum distance to the original poses and the grid spacing thus creates varying amounts of training data.

[3]This condition is not sufficient as a network might not learn the "right" embedding for expressing all test poses as linear combinations of $\mathcal{B}'(\mathcal{P}')$.

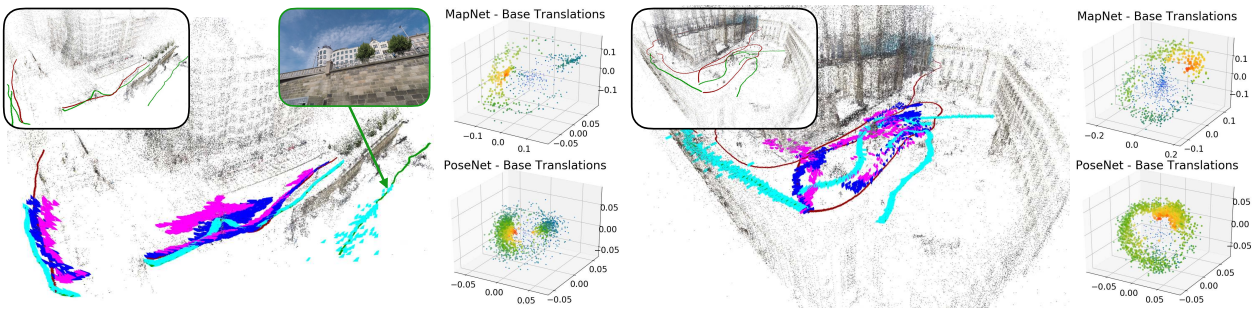

Figure 3. Example scenarios with more general training trajectories in which two absolute pose regression techniques, PoseNet [29, 30] and MapNet [11], fail to generalize. Please see the caption of Fig. 1 for details on the color coding.

max. distance	-	1m			2m			3m		
spacing	-	1m	0.5m	0.25m	1m	0.5m	0.25m	1m	0.5m	0.25m
# training images	203	501	1,315	4,576	683	2,035	7,425	806	2,531	9,412
PoseNet [29]	1.19 / 6.88	1.02 / 6.48	0.74 / 7.07	0.79 / 5.84	1.15 / 8.10	0.86 / 6.88	0.54 / 5.84	0.66 / 6.88	0.66 / 6.06	0.68 / 5.38
MapNet [11]	1.07 / 4.70	0.61 / 3.31	0.64 / 2.85	0.41 / 2.18	0.72 / 3.41	0.42 / 2.06	0.38 / 2.31	0.69 / 3.18	0.44 / 2.39	0.33 / 1.46
Active Search [59]	**0.01 / 0.04**									
DenseVLAD [71]	0.98 / 7.90	0.79 / 8.01	0.74 / 7.81	0.63 / 7.68	0.72 / 7.81	0.61 / 7.38	0.57 / 6.94	0.66 / 7.81	0.60 / 7.27	0.51 / 6.87
DenseVLAD+Inter.	0.89 / 5.71	0.75 / 5.62	0.52 / 6.65	0.45 / 6.93	0.57 / 5.96	0.48 / 6.13	0.41 / 6.41	0.49 / 6.07	0.46 / 6.26	0.38 / 6.41

Table 1. Median position / orientation errors in meters / degree on the **synthetic Shop Facade** dataset obtained by rendering a multi-view stereo reconstruction. We enhance the training set by additional images captured on a regular grid, varying the spacing between images. We only consider additional images within a certain maximum distance to the positions of the original training poses.

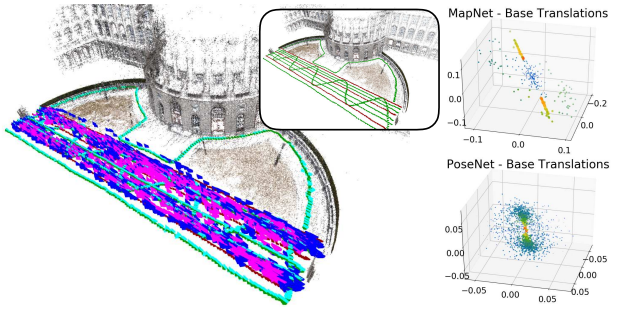

Figure 4. See caption of Fig. 1 for details.

Tab. 1 compares PoseNet and MapNet trained on varying amounts of data with Active Search using only renderings from the original training poses[4]. As expected, using more training data improves pose accuracy. However, PoseNet and MapNet do not perform even close to Active Search, even with one order of magnitude more data.

Discussion. Pose regression techniques are unlikely to work well when only little training data is available and significant viewpoint changes need to be handled. This clearly limits their relevance for practical applications. Even with large amounts of training data, pose regression does not reach the same performance as structure-based methods. This clearly shows a fundamental conceptual difference between the two approaches to visual localization. We attribute this divide to the fact that the latter are based on the laws of projective geometry and the underlying 3D geometry of the scene. This in turn enables them to better handle viewpoint changes.

[4]All images used in this experiment are renderings of the 3D model. We use a resolution of 455×256 pixels as input to all methods.

5. Comparison with Image Retrieval

As can be seen in Fig. 1 and Fig. 3(right), absolute pose regression (APR) techniques tend to predict test poses close to the training poses in regions where little training data is available. This behavior is similar to that of image retrieval approaches. Below, we show that this behavioral similarity is not a coincident. Rather, there is a strong connection between APR and image retrieval. We also show that APR methods do not consistently outperform a retrieval baseline.

Relation to image retrieval. Let $\mathcal{I}$ be a test image and $\mathcal{J}$ a training image observing the same part of the scene. We can write the embedding $\alpha^{\mathcal{I}}$ as $\alpha^{\mathcal{I}} = \alpha^{\mathcal{J}} + \Delta^{\mathcal{I}}$, for some offset $\Delta^{\mathcal{I}}$. Using Eq. 3, we can thus relate the pose $(\hat{\mathbf{c}}_{\mathcal{I}}, \hat{\mathbf{r}}_{\mathcal{I}})$ estimated for $\mathcal{I}$ to the pose $(\hat{\mathbf{c}}_{\mathcal{J}}, \hat{\mathbf{r}}_{\mathcal{J}})$ estimated for $\mathcal{J}$ via

$$\begin{pmatrix}\hat{\mathbf{c}}_{\mathcal{I}}\\ \hat{\mathbf{r}}_{\mathcal{I}}\end{pmatrix} = \begin{pmatrix}\hat{\mathbf{c}}_{\mathcal{J}}\\ \hat{\mathbf{r}}_{\mathcal{J}}\end{pmatrix} + \begin{pmatrix}\sum_{j=1}^{n} \Delta_j^{\mathcal{I}} \mathbf{c}_j\\ \sum_{j=1}^{n} \Delta_j^{\mathcal{I}} \mathbf{r}_j\end{pmatrix} = \begin{pmatrix}\hat{\mathbf{c}}_{\mathcal{J}}\\ \hat{\mathbf{r}}_{\mathcal{J}}\end{pmatrix} + \begin{pmatrix}\hat{\mathbf{c}}_{\mathcal{I},\mathcal{J}}\\ \hat{\mathbf{r}}_{\mathcal{I},\mathcal{J}}\end{pmatrix}. \quad (6)$$

Here, $(\hat{\mathbf{c}}_{\mathcal{I},\mathcal{J}}, \hat{\mathbf{r}}_{\mathcal{I},\mathcal{J}})$ is a pose offset, *i.e.*, the pose of $\mathcal{I}$ is predicted relative to the pose predicted for $\mathcal{J}$.

Eq. 6 highlights the conceptual similarity between absolute pose regression and image retrieval. Standard image retrieval approaches first find the training image $\mathcal{J}$ most similar to a given test image $\mathcal{I}$, where similarity is defined in some feature space such as Bag-of-Words (BoW) [67] or VLAD [3,27,71]. The pose of the test image is then approximated via the pose of the retrieved image, *i.e.*, $(\hat{\mathbf{c}}_{\mathcal{I}}, \hat{\mathbf{r}}_{\mathcal{I}}) = (\mathbf{c}_{\mathcal{J}}, \mathbf{r}_{\mathcal{J}})$, without adding an offset. However, retrieval methods can also estimate such an offset as an affine combination $\sum_{i=1}^{k} a_i(\mathbf{c}_{\mathcal{J}_i}, \mathbf{r}_{\mathcal{J}_i})$, $\sum a_i = 1$, of the poses of the top-k retrieved training images $\mathcal{J}_1, \ldots, \mathcal{J}_k$. Let $\mathbf{d}(\mathcal{I})$ be the descriptor for image $\mathcal{I}$ used during retrieval. The weights a_i

can be obtained by finding the affine combination of training image descriptors that is closest to the test descriptor $\mathbf{d}(\mathcal{I})$, *i.e.*, by minimizing $||\mathbf{d}(\mathcal{I}) - \sum_{i=1}^{k} a_i \mathbf{d}(\mathcal{J}_i)||_2$ subject to $\sum a_i = 1$. This approach has been shown to work well for linearly interpolating between two BoW representations [72]. Note the conceptual similarity between this interpolation and Eq. 3, where the trained base poses are used instead of the poses of the retrieved images.

Eq. 6 also establishes a relation between APR approaches and relative pose regression (RPR) algorithms. RPR methods first identify a set of training images relevant to a given test image, *e.g.*, using image retrieval [7, 35] or by encoding the training images in a CNN [56]. They then compute a pose offset from the training images to the test image via regression. RPR approaches naturally benefit from computing offsets to multiple training images [7].

5.1. Experimental Comparison

Baselines. We use *DenseVLAD* [71] as an image retrieval baseline. DenseVLAD densely extracts RootSIFT [4, 40] descriptors from an image and pools them into a VLAD [27] descriptor. Dimensionality reduction via PCA, trained on an unrelated outdoor dataset [71], is then used to reduce the dimensionality of the descriptor to 4096. The Euclidean distance is used to measure similarity between two DenseVLAD descriptors. We use the implementation provided by [71], but only extract RootSIFT descriptors at a single scale[5]. We chose DenseVLAD as it uses a handcrafted feature representation. At the same time, DenseVLAD has been shown to perform well even on challenging localization tasks [60, 61, 71].

DenseVLAD approximates the pose of the test image via the pose of the most similar training image. In addition, we also use a variant, denoted as *DenseVLAD + Inter.*, that uses the interpolation approach described above. We use all top-k ranked images for interpolation. As there might be some outliers among the top retrieved images, interpolation can potentially decrease pose accuracy. However, we decided to keep this baseline as simple as possible and thus did not implement an outlier filtering mechanism.

Cambridge Landmarks [30] and 7 Scenes [65]. In a first experiment, we compare state-of-the-art pose regression techniques to the two image retrieval baselines on the Cambridge Landmarks [30] and 7 Scenes [65] datasets. These two relatively small-scale datasets are commonly used to evaluate pose regression approaches. We only compare methods that predict a camera pose from a single image.

Tab. 2 shows the median position and orientation errors obtained by the various methods. As can be seen by the results marked in red, *none of the absolute and relative pose regression approaches is able to consistently outperform the retrieval baselines*. In addition, *pose regression techniques are often closer in performance to image retrieval than to structure-based methods*. In particular, these results verify our theoretical analysis that APR is much closer related to image retrieval than to structure-based methods.

Out of the four best-performing pose regression approaches (MapNet [11], RelocNet [7], Relative PN [35], AnchorNet [56]), three are RPR approaches (RelocNet, Relative PN, AnchorNet). AnchorNet comes closest to structure-based methods. It uses a brute-force approach that essentially estimates a pose offset between the input image and every 10^{th} training image. Considering the relative improvement[6], AnchorNet typically performs closer to other APR or RPR methods than to the best performing structure-based approach in each scene. It also fails to outperform the simple DenseVLAD baseline on the Street scene, which is the largest and most complex scene in the Cambridge Landmarks dataset [10, 50].

AnchorNet encodes the training images in the regression CNN and thus needs to be trained specifically per scene. In contrast, Relative PN and RelocNet both perform an explicit image retrieval step. They can thus also be trained on unrelated scenes. Besides RelocNet and Relative PN trained on 7 Scenes (7S), we thus also compare against variants trained on other datasets (ScanNet (SN) [20], University (U) [35]). As shown in Tab. 2, both approaches currently do not generalize well using this data, as they are less accurate than DenseVLAD (which requires no training).

One challenge of the Cambridge Landmarks and 7 Scenes datasets is that there are significant differences in pose between the training and test images. As shown in Sec. 4, this is a severe challenge for current regression techniques. In the following, we focus on scenes with less deviation between training and test poses, which should be much easier for pose regression techniques. We show results on two such datasets. A further experiment (on the DeepLoc dataset [53]) can be found in the supp. material..

TUM LSI [76]. The scenes in the Cambridge Landmarks and 7 Scenes datasets are typically rather well-textured. Thus, we can expect that the SIFT descriptors used by DenseVLAD and Active Search [59] work rather well. In contrast, the TUM LSI indoor dataset [76] contains large textureless walls and repeating structures. In general, we would expect learned approaches to perform significantly better than methods based on low-level SIFT features as the former can learn to use higher-level structures. Yet, as shown in Tab. 3, DenseVLAD still outperforms pose regression techniques on this more challenging dataset.

RobotCar dataset [42]. The training images of the LOOP and FULL scenes [11] correspond to trajectories of 1.1km

[5] Scale invariance is not desirable when searching for the training image taken from the most similar pose.

[6] Defined as the ratio of the position / orientation errors of two methods.

		Cambridge Landmarks					7 Scenes						
		Kings	Old	Shop	St. Mary's	Street	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs
APR	PoseNet (PN) [30]	1.92/5.40	2.31/5.38	1.46/8.08	2.65/8.48		0.32/8.12	0.47/14.4	0.29/12.0	0.48/7.68	0.47/8.42	0.59/8.64	0.47/13.8
	PN learned weights [29]	0.99/1.06	2.17/2.94	1.05/3.97	1.49/3.43	20.7/25.7	0.14/4.50	0.27/11.8	0.18/12.1	0.20/5.77	0.25/4.82	0.24/5.52	0.37/10.6
	Bay. PN [28]	1.74/4.06	2.57/5.14	1.25/7.54	2.11/8.38		0.37/7.24	0.43/13.7	0.31/12.0	0.48/8.04	0.61/7.08	0.58/7.54	0.48/13.1
	geo. PN [29]	0.88/1.04	3.20/3.29	0.88/3.78	1.57/3.32	20.3/25.5	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
	LSTM PN [76]	0.99/3.65	1.51/4.29	1.18/7.44	1.52/6.68		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
	GPoseNet [12]	1.61/2.29	2.62/3.89	1.14/5.73	2.93/6.46		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
	SVS-Pose [50]	1.06/2.81	1.50/4.03	0.63/5.73	2.11/8.11								
	Hourglass PN [44]						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
	BranchNet [78]						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
	MapNet [11]	1.07/1.89	1.94/3.91	1.49/4.22	2.00/4.53		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
	MapNet+ [11]						0.10/3.17	0.20/9.04	0.13/11.1	0.18/5.38	0.19/3.92	0.20/5.01	0.30/13.4
	MapNet+PGO [11]						0.09/3.24	0.20/9.29	0.12/8.45	0.19/5.42	0.19/3.96	0.20/4.94	0.27/10.6
RPR	Relative PN [35] (U)						0.31/15.0	0.40/19.0	0.24/22.2	0.38/14.1	0.44/18.2	0.41/16.5	0.35/23.6
	Relative PN [35] (7S)						0.13/6.46	0.26/12.7	0.14/12.3	0.21/7.35	0.24/6.35	0.24/8.03	0.27/11.8
	RelocNet [7] (SN)						0.21/ 10.9	0.32/11.8	0.15/13.4	0.31/ 10.3	0.40/ 10.9	0.33/10.3	0.33/11.4
	RelocNet [7] (7S)						0.12/4.14	0.26/10.4	0.14/10.5	0.18/5.32	0.26/4.17	0.23/5.08	0.28/7.53
	AnchorNet [56]	0.57/0.88	1.21/2.55	0.52/2.27	1.04/2.69	7.86/24.2	0.06/3.89	0.15/10.3	0.08/10.9	0.09/5.15	0.10/2.97	0.08/4.68	0.10/9.26
IR	DenseVLAD [71]	2.80/5.72	4.01/7.13	1.11/7.61	2.31/8.00	5.16/23.5	0.21/12.5	0.33/13.8	0.15/14.9	0.28/11.2	0.31/11.3	0.30/12.3	0.25/15.8
	DenseVLAD + Inter.	1.48/4.45	2.68/4.63	0.90/4.32	1.62/6.06	15.4/25.7	0.18/10.0	0.33/12.4	0.14/14.3	0.25/10.1	0.26/9.42	0.27/11.1	0.24/14.7
3D	Active Search [59]	0.42/0.55	0.44/1.01	0.12/0.40	0.19/0.54	**0.85/0.8**	0.04/1.96	0.03/1.53	0.02/1.45	0.09/3.61	0.08/3.10	0.07/3.37	**0.03/2.22**
	BTBRF [46]	0.39/0.36	0.30/0.41	0.15/0.31	0.20/**0.40**								
	DSAC++ [10]	**0.18/0.3**	**0.20/0.3**	**0.06/0.3**	**0.13/0.4**		**0.02/0.5**	**0.02/0.9**	**0.01/0.8**	**0.03/0.7**	**0.04/1.1**	**0.04/1.1**	0.09/2.6
	InLoc [69]						0.03/1.05	0.03/1.07	0.02/1.16	**0.03**/1.05	0.05/1.55	**0.04**/1.31	0.09/2.47

Table 2. Results on the **Cambridge Landmarks** [30] and **7 Scenes** [65] datasets. We compare absolute (APR) and relative (RPR) pose regression methods, image retrieval (IR) techniques, and structure-based (3D) approaches. We report the median position / orientation error in meters / degree. *DenseVLAD + Inter.* uses the top-20 (Cambridge Landmarks) respectively top-25 (7 Scenes) retrieved images. Red numbers show when a method fails to outperform the image retrieval (IR) baselines.

PoseNet [30]	MapNet [11]	LSTM PN [76]	Dense VLAD [71]	DenseVLAD +Inter.
1.87m, 6.14°	1.71m, 3.50°	1.31m, 2.79°	1.08m, 1.82°	0.49m, 2.01°

Table 3. Median position and orientation errors on the **TUM LSI** dataset [76]. The top-2 retrieved images are used for interpolation.

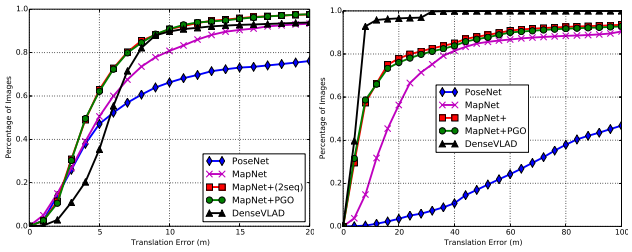

Figure 5. Cum. distribution of position errors for (left) **RobotCar LOOP** and (right) **RobotCar FULL**. On the larger dataset, DenseVLAD significantly outperforms pose regression techniques.

and 9.6km, respectively, driven by a car. The test images are obtained by driving the same trajectory. This dataset represents a scenario encountered during autonomous driving.

Fig. 5 shows the cumulative distributions of position errors for pose regression and image retrieval techniques. As expected, MapNet+ and MapNet+PGO outperform DenseVLAD on the smaller LOOP dataset. However, they perform significantly worse on the larger FULL scene[7]. This is despite MapNet+ using additional training sequences and MapNet+PGO using information from multiple images for its predictions. This scalability issue of pose regression is in line with similar observations in the literature [60, 63, 69].

[7]DenseVLAD is slightly more accurate on the FULL scene than on the LOOP dataset. We attribute this to the image quality, as the test set of the LOOP scene contains several overexposed images.

Using densely sampled data. As in Sec. 4, our final experiment compares image retrieval and APR techniques on a synthetic scene, where a vast amount of training data is available. As shown in Tab. 1, MapNet outperforms the image retrieval baselines when more training data is available. Still, it performs much closer to the retrieval baselines than to the structure-based method.

6. Conclusion

In this paper, we have derived a theoretic model for absolute pose regression (APR) algorithms. For the first time, this model allowed us to develop a better understanding of what APR method are and are not capable of. Based on our theory, we have predicted that APR techniques are not guaranteed to generalize from the training data in practical scenarios. We have also shown that APR is more closely related to image retrieval approaches than to methods that accurately estimate camera poses via 3D geometry. These predictions have been verified through extensive experiments.

The second main result of our paper is to show that pose regression techniques are currently competing with image retrieval approaches approximating the test pose rather than with methods that compute it accurately. More precisely, we have shown that no current pose regression approach consistently outperforms a handcrafted retrieval baseline. This paper has thus introduced an important sanity check for judging pose regression methods, showing that there is still a significant amount of research to be done before pose regression approaches become practically relevant.

Acknowledgements. This research was partially funded by the Humboldt Foundation through the Sofja Kovalevskaya Award.

References

[1] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. http://ceres-solver.org. 4

[2] Cenek Albl, Zuzana Kukelova, and Tomas Pajdla. R6P - Rolling Shutter Absolute Camera Pose. In *CVPR*, 2015. 1

[3] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016. 2, 6

[4] Relja Arandjelović and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012. 4, 7

[5] Relja Arandjelović and Andrew Zisserman. DisLocation: Scalable descriptor distinctiveness for location recognition . In *ACCV*, 2014. 2

[6] Clemens Arth, Daniel Wagner, Manfred Klopschitz, Arnold Irschara, and Dieter Schmalstieg. Wide area localization on mobile phones. In *ISMAR*, 2009. 2

[7] Vassileios Balntas, Shuda Li, and Victor Prisacariu. RelocNet: Continuous Metric Learning Relocalisation using Neural Nets. In *ECCV*, 2018. 2, 3, 7, 8

[8] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC - Differentiable RANSAC for Camera Localization. In *CVPR*, 2017. 1, 2, 4

[9] Eric Brachmann, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, and Carsten Rother. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *CVPR*, 2016. 2

[10] Eric Brachmann and Carsten Rother. Learning Less is More - 6D Camera Localization via 3D Surface Regression. In *CVPR*, 2018. 1, 2, 4, 7, 8

[11] Samarth Brahmbhatt, Jinwei Gu, Kihwan Kim, James Hays, and Jan Kautz. Geometry-Aware Learning of Maps for Camera Localization. In *CVPR*, 2018. 1, 2, 3, 4, 5, 6, 7, 8

[12] Ming Cai, Chunhua Shen, and Ian D. Reid. A Hybrid Probabilistic Model for Camera Relocalization. In *BMVC*, 2018. 1, 2, 8

[13] Federico Camposeco, Andrea Cohen, Marc Pollefeys, and Torsten Sattler. Hybrid Camera Pose Estimation. In *CVPR*, 2018. 2

[14] Song Cao and Noah Snavely. Graph-Based Discriminative Learning for Location Recognition. In *CVPR*, 2013. 2

[15] Robert O. Castle, Georg Klein, and David W. Murray. Video-rate Localization in Multiple Maps for Wearable Augmented Reality. In *ISWC*, 2008. 1

[16] Tommaso Cavallari, Stuart Golodetz, Nicholas A. Lord, Julien Valentin, Luigi Di Stefano, and Philip H. S. Torr. On-The-Fly Adaptation of Regression Forests for Online Camera Relocalisation. In *CVPR*, 2017. 1, 2, 4

[17] David M. Chen, Georges Baatz, Kevin Köser, Sam S. Tsai, Ramakrishna Vedantham, Timo Pylvänäinen, Kimmo Roimela, Xin Chen, Jeff Bach, Marc Pollefeys, Bernd Girod, and Radek Grzeszczuk. City-Scale Landmark Identification on Mobile Devices. In *CVPR*, 2011. 2, 4

[18] Ondrej Chum and Jiri Matas. Optimal Randomized RANSAC. *PAMI*, 30(8):1472 –1482, 2008. 1, 4

[19] 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. 2

[20] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *CVPR*, 2017. 7

[21] Michael Donoser and Dieter Schmalstieg. Discriminative Feature-to-Point Matching in Image-Based Locallization. In *CVPR*, 2014. 2

[22] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *CACM*, 24(6):381–395, 1981. 1

[23] Abner Guzman-Rivera, Pushmeet Kohli, Ben Glocker, Jamie Shotton, Toby Sharp, Andrew Fitzgibbon, and Shahram Izadi. Multi-Output Learning for Camera Relocalization. In *CVPR*, 2014. 2

[24] Christian Häne, Lionel Heng, Gim Hee Lee, Friedrich Fraundorfer, Paul Furgale, Torsten Sattler, and Marc Pollefeys. 3D visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection. *IMAVIS*, 68:14 – 27, 2017. Automotive Vision: Challenges, Trends, Technologies and Systems for Vision-Based Intelligent Vehicles. 1

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016. 2, 3

[26] Arnold Irschara, Christopher Zach, Jan-Michael Frahm, and Horst Bischof. From Structure-from-Motion Point Clouds to Fast Location Recognition. In *CVPR*, 2009. 2

[27] Herve Jégou, Florent Perronnin, Matthijs Douze, Jorge Sánchez, Patrick Pérez, and Cordelia Schmid. Aggregating local image descriptors into compact codes. *PAMI*, 34(9):1704–1716, 2012. 6, 7

[28] Alex Kendall and Roberto Cipolla. Modelling uncertainty in deep learning for camera relocalization. In *ICRA*, 2016. 1, 2, 3, 8

[29] Alex Kendall and Roberto Cipolla. Geometric Loss Functions for Camera Pose Regression With Deep Learning. In *CVPR*, 2017. 1, 2, 3, 4, 5, 6, 8

[30] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, 2015. 1, 2, 3, 4, 5, 6, 7, 8

[31] Laurent Kneip, Davide Scaramuzza, and Roland Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *CVPR*, 2011. 1, 4

[32] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Real-Time Solution to the Absolute Pose Problem with Unknown Radial Distortion and Focal Length. In *ICCV*, 2013. 1

[33] Viktor Larsson, Zuzana Kukelova, and Yinqiang Zheng. Making Minimal Solvers for Absolute Pose Estimation Compact and Robust. In *ICCV*, 2017. 1

[34] Viktor Larsson, Zuzana Kukelova, and Yinqiang Zheng. Camera Pose Estimation With Unknown Principal Point. In *CVPR*, 2018. 1

[35] Zakaria Laskar, Iaroslav Melekhov, Surya Kalia, and Juho Kannala. Camera Relocalization by Computing Pairwise Relative Poses Using Convolutional Neural Network. In *ICCV Workshops*, 2017. 2, 7, 8

[36] Karel Lebeda, Jirı Matas, and Ondrej Chum. Fixing the Locally Optimized RANSAC. In *BMVC*, 2012. 1

[37] Yunpeng Li, Noah Snavely, Dan Huttenlocher, and Pascal Fua. Worldwide pose estimation using 3d point clouds. In *ECCV*, 2012. 2

[38] Hyon Lim, Sudipta N. Sinha, Michael F. Cohen, Matt Uyttendaele, and H. Jin Kim. Real-time monocular image-based 6-DoF localization. *IJRR*, 34(4-5):476–492, 2015. 1, 2

[39] Liu Liu, Hongdong Li, and Yuchao Dai. Efficient Global 2D-3D Matching for Camera Localization in a Large-Scale 3D Map. In *ICCV*, 2017. 2

[40] David G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 4, 7

[41] Simon Lynen, Torsten Sattler, Mike Bosse, Joel Hesch, Marc Pollefeys, and Roland Siegwart. Get Out of My Lab: Large-scale, Real-Time Visual-Inertial Localization. In *RSS*, 2015. 2

[42] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The Oxford RobotCar dataset. *IJRR*, 36(1):3–15, 2017. 4, 7

[43] Daniela Massiceti, Alexander Krull, Eric Brachmann, Carsten Rother, and Philip H.S. Torr. Random Forests versus Neural Networks - What's Best for Camera Relocalization? In *ICRA*, 2017. 1, 2

[44] Iaroslav Melekhov, Juha Ylioinas, Juho Kannala, and Esa Rahtu. Image-based Localization using Hourglass Networks. In *ICCV Workshops*, 2017. 1, 2, 8

[45] Iaroslav Melekhov, Juha Ylioinas, Juho Kannala, and Esa Rahtu. Relative camera pose estimation using convolutional neural networks. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 675–687, 2017. 2

[46] Lili Meng, Jianhui Chen, Frederick Tung, James J. Little, Julien Valentin, and Clarence W. de Silva. Backtracking Regression Forests for Accurate Camera Relocalization. In *IROS*, 2017. 1, 2, 4, 8

[47] Lili Meng, Frederick Tung, James J. Little, Julien Valentin, and Clarence W. de Silva. Exploiting Points and Lines in Regression Forests for RGB-D Camera Relocalization. In *IROS*, 2018. 1, 2

[48] Sven Middelberg, Torsten Sattler, Ole Untzelmann, and Leif Kobbelt. Scalable 6-DOF Localization on Mobile Devices. In *ECCV*, 2014. 1, 2

[49] Kwang Moo Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. Learning to Find Good Correspondences. In *CVPR*, 2018. 1

[50] Tayyab Naseer and Wolfram Burgard. Deep regression for monocular camera-based 6-DoF global localization in outdoor environments. In *IROS*, 2017. 1, 2, 3, 7, 8

[51] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. LF-Net: Learning Local Features from Images. In *NIPS*, 2018. 5

[52] Filip Radenović, Giorgos Tolias, and Ondrej Chum. Fine-tuning CNN image retrieval with no human annotation. *TPAMI*, 2018. 2

[53] Noha Radwan, Abhinav Valada, and Wolfram Burgard. VLocNet++: Deep Multitask Learning For Semantic Visual Localization And Odometry. *RA-L*, 3(4):4407–4414, 2018. 1, 2, 7

[54] Rahul Raguram, Ondrej Chum, Marc Pollefeys, Jiri Matas, and Jan-Michael Frahm. USAC: A Universal Framework for Random Sample Consensus. *PAMI*, 35(8):2022–2038, 2013. 1

[55] IIgnacio Rocco, Mircea Cimpoi, Relja Arandjelović, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Neighbourhood Consensus Networks. In *NIPS*, 2018. 5

[56] Soham Saha, Girish Varma, and C. V. Jawahar. Improved Visual Relocalization by Discovering Anchor Points. In *BMVC*, 2018. 2, 7, 8

[57] Torsten Sattler, Michal Havlena, Filip Radenović, Konrad Schindler, and Marc Pollefeys. Hyperpoints and Fine Vocabularies for Large-Scale Location Recognition. In *ICCV*, 2015. 2

[58] Torsten Sattler, Michal Havlena, Konrad Schindler, and Marc Pollefeys. Large-Scale Location Recognition And The Geometric Burstiness Problem. In *CVPR*, 2016. 2

[59] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & Effective Prioritized Matching for Large-Scale Image-Based Localization. *PAMI*, 39(9):1744–1756, 2017. 1, 2, 4, 6, 7, 8

[60] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions. In *CVPR*, 2018. 7, 8

[61] Torsten Sattler, Akihiko Torii, Josef Sivic, Marc Pollefeys, Hajime Taira, Masatoshi Okutomi, and Tomas Pajdla. Are Large-Scale 3D Models Really Necessary for Accurate Visual Localization? In *CVPR*, 2017. 7

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

[63] Johannes Lutz Schönberger, Marc Pollefeys, Andreas Geiger, and Torsten Sattler. Semantic Visual Localization. In *CVPR*, 2018. 1, 2, 4, 8

[64] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *ECCV*, 2016. 5

[65] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. *CVPR*, 2013. 1, 2, 7, 8

[66] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015. 2, 3

[67] Josef Sivic and Andrew Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 6

[68] Linus Svärm, Olof Enqvist, Fredrik Kahl, and Magnus Oskarsson. City-Scale Localization for Cameras with Known Vertical Direction. *PAMI*, 39(7):1455–1461, 2017. 1, 2, 4

[69] Hajime Taira, Masatoshi Okutomi, Torsten Sattler, Mircea Cimpoi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, and Akihiko Torii. InLoc: Indoor Visual Localization with Dense Matching and View Synthesis. In *CVPR*, 2018. 1, 2, 4, 8

[70] Carl Toft, Erik Stenborg, Lars Hammarstrand, Lucas Brynte, Marc Pollefeys, Torsten Sattler, and Fredrik Kahl. Semantic Match Consistency for Long-Term Visual Localization. In *ECCV*, 2018. 1

[71] Akihiko Torii, Relja Arandjelović, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. 24/7 place recognition by view synthesis. In *CVPR*, 2015. 1, 2, 6, 7, 8

[72] A. Torii, J. Sivic, and T. Pajdla. Visual localization by linear combination of image descriptors. In *ICCVW*, 2011. 2, 7

[73] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. DeMoN: Depth and Motion Network for Learning Monocular Stereo. In *CVPR*, 2017. 3

[74] Abhinav Valada, Noha Radwan, and Wolfram Burgard. Deep Auxiliary Learning For Visual Localization And Odometry. In *ICRA*, 2018. 1, 2, 3

[75] Michael Waechter, Mate Beljan, Simon Fuhrmann, Nils Moehrle, Johannes Kopf, and Michael Goesele. Virtual Rephotography: Novel View Prediction Error for 3D Reconstruction. *ACM Transactions on Graphics (presented at SIGGRAPH 2017)*, 36(1):8:1–8:11, 2017. 5

[76] Florian Walch, Caner Hazirbas, Laura Leal-Taixe, Torsten Sattler, Sebastian Hilsenbeck, and Daniel Cremers. Image-based localization using lstms for structured feature correlation. In *ICCV*, 2017. 1, 2, 3, 7, 8

[77] Tobias Weyand, Ilya Kostrikov, and James Phiblin. PlaNet - Photo Geolocation with Convolutional Neural Networks. *ECCV*, 2016. 2

[78] Jian Wu, Liwei Ma, and Xiaolin Hu. Delving Deeper into Convolutional Neural Networks for Camera Relocalization. In *ICRA*, 2017. 1, 2, 3, 8

[79] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. LIFT: Learned Invariant Feature Transform. In *ECCV*, 2016. 1

[80] Amir Roshan Zamir and Mubarak Shah. Accurate image localization based on google maps street view. In *ECCV*, 2010. 2

[81] Bernhard Zeisl, Torsten Sattler, and Marc Pollefeys. Camera pose voting for large-scale image-based localization. In *ICCV*, 2015. 2

[82] Wei Zhang and Jana Kosecka. Image based Localization in Urban Environments. In *3DPVT*, 2006. 2

[83] Enliang Zheng and Changchang Wu. Structure from Motion Using Structure-Less Resection. In *ICCV*, 2015. 2