Geometric loss functions for camera pose regression with deep learning

Alex Kendall and Roberto Cipolla
University of Cambridge
{agk34, rc10001}@cam.ac.uk

Abstract

Deep learning has shown to be effective for robust and real-time monocular image relocalisation. In particular, PoseNet [22] is a deep convolutional neural network which learns to regress the 6-DOF camera pose from a single image. It learns to localize using high level features and is robust to difficult lighting, motion blur and unknown camera intrinsics, where point based SIFT registration fails. However, it is trained using a naive loss function, with hyper-parameters which require expensive tuning. In this paper, we give the problem a more fundamental theoretical treatment. We explore a number of novel loss functions for learning camera pose which are based on geometry and scene reprojection error. Additionally we show how to automatically learn an optimal weighting to simultaneously regress position and orientation. By leveraging geometry, we demonstrate that our technique significantly improves PoseNet's performance across datasets ranging from indoor rooms to a small city.

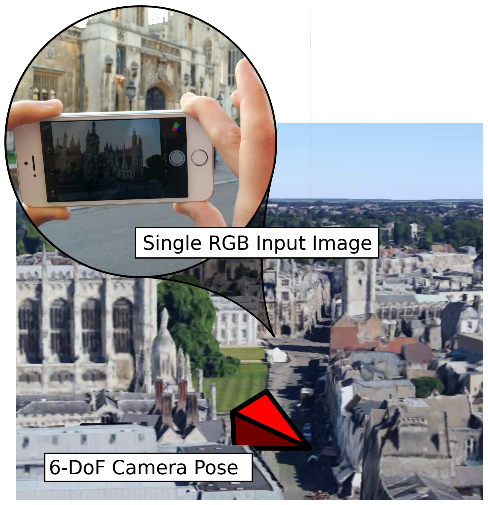


Figure 1: **PoseNet** [22] is trained end-to-end to estimate the camera's six degree of freedom pose from a single monocular image. In this paper we show how to apply a principled loss function based on the scene's geometry to learn camera pose without any hyper-parameters.

1. Introduction

Designing a system for reliable large scale localisation is a challenging problem. The discovery of the positioning system in mammalian brains, located in the hippocampus, was awarded the 2014 Nobel prize in Physiology or Medicine [36, 32]. It is an important problem for computer vision too, with localisation technology essential for many applications including autonomous vehicles, unmanned aerial vehicles and augmented reality. State of the art localisation systems perform very well within controlled environments [24, 34, 12, 33, 45]. However, we are yet to see their wide spread use in the wild because of their inability to cope with large viewpoint or appearance changes.

Many of the visual localisation systems use point landmarks such as SIFT [30] or ORB [40] to localise. These features perform well for incremental tracking and estimating ego-motion [33]. However, these point features are not able to create a representation which is sufficiently robust to challenging real-world scenarios. For example, point features are often not robust enough for localising across different weather, lighting or environmental conditions. Additionally, they lack the ability to capture global context, and require robust aggregation of hundreds of points to form a consensus to predict pose [58].

To address this problem, we introduced PoseNet [22, 19] which uses end-to-end deep learning to predict camera pose from a single input image. It was shown to be able localise more robustly using deep learning, compared with point features such as SIFT [30]. PoseNet learns a representation using the entire image context based on appearance and shape. These features generalise well and can localise across challenging lighting and appearances changes. It is also fast, being able to regress the camera's pose in only a few milliseconds. It is very scalable as it does not require a large database of landmarks. Rather, it learns a mapping from pixels to a high dimensional space linear with pose.

The main weakness of PoseNet [22] was that despite its scalability and robustness it did not produce metric accuracy which is comparable to other geometric methods

[45, 50]. In this paper we argue that a contributing factor to this was because PoseNet naively applied a deep learning model end-to-end to learn camera pose. In this work, we reconsider this problem with a grounding in geometry. We wish to build upon the decades of research into multi-view geometry [14] to improve our ability to use deep learning to regress camera pose.

The main contribution of this paper is improving the performance of PoseNet with geometrically formed loss functions. It is not trivial to simply regress position and rotation quantities using supervised learning. PoseNet required a weighting factor to balance these two properties, but was not tolerant to the selection of this hyperparameter. In Section 3.3 we explore loss functions which remove this hyperparameter, or optimise it directly from the data. In Section 3.3.4 we show how to train directly from the scene geometry using the reprojection error.

In Section 4 we demonstrate our system on an array of datasets, ranging from individual indoor rooms, to the Dubrovnik city dataset [26]. We show that our geometric approach can improve PoseNet's efficacy across many different datasets – narrowing the deficit to traditional SIFT feature-based algorithms. For outdoor scenes ranging from $50,000m^2$ to $2km^2$ we can achieve relocalisation accuracies of a few meters and a few degrees. In small rooms we are able to achieve accuracies of $0.2 - 0.4m$.

2. Related work

Large scale localisation research can be divided into two categories; place recognition and metric localisation. Place recognition discretises the world into a number of landmarks and attempts to identify which place is visible in a given image. Traditionally, this has been modelled as an image retrieval problem [6, 9, 54, 46] enabling the use of efficient and scalable retrieval approaches [35, 38] such as Bag-of-Words (BoW) [48], VLAD [17, 10], and Fisher vectors [18]. Deep learning models have also been shown to be effective for creating efficient descriptors. Many approaches leverage classification networks [39, 13, 3, 53], and fine tune them on localisation datasets [4]. Other work of note is PlaNet [56] which trained a classification network to localise images on a world scale. However, all these networks must discretise the world into places and are unable to produce a fine grained estimate of 6-DOF pose.

In contrast, metric localisation techniques estimate the metric position and orientation of the camera. Traditionally, this has been approached by computing the pose from 2D-3D correspondences between 2D features in the query image and 3D points in the model, which are determined through descriptor matching [7, 28, 27, 41, 43, 50]. This assumes that the scene is represented by a 3D structure-from-motion model. The full 6 degree-of-freedom pose of a query image can be estimated very precisely [45]. However these methods require a 3D model with a large database of features and efficient retrieval methods. They are expensive to compute, often do not scale well, and are often not robust to changing environmental conditions [55].

In this work, we address the more challenging problem of metric localisation with deep learning. PoseNet [22] introduced the technique of training a convolutional neural network to regress camera pose. It combines the strengths of place recognition and localisation approaches: it can globally relocalise without a good initial pose estimate, and produces a continuous metric pose. Rather than building a map (or database of landmark features), the neural network learns features whose size, unlike a map, does not require memory linearly proportional to the size of the scene.

Later work has extended PoseNet to use RGB-D input [25], learn relative ego-motion [31], improve the context of features [55], localise over video sequences [8] and interpret relocalisation uncertainty with Bayesian Neural Networks [19]. Additionally, [55] demonstrate PoseNet's efficacy on featureless indoor environments, where they demonstrate that SIFT based structure from motion techniques fail in the same environment.

Although PoseNet is scalable and robust [22], it does not produce sufficiently accurate estimates of Pose compared to traditional methods [45]. It was designed with a naive regression loss function which trains the network end-to-end without any consideration for geometry. This problem is the focus of this paper – we do not want to throw away the decades of research into multi view geometry [14]. We improve PoseNet's performance by learning camera pose with a fundamental treatment of scene geometry.

3. Model for camera pose regression

In this section we describe the details of the convolutional neural network model we train to estimate camera pose directly from a monocular image, I. Our network outputs an estimate, $\hat{\mathbf{p}}$, for pose, $\mathbf{p}$, given by a 3-D camera position $\hat{\mathbf{x}}$ and orientation $\hat{\mathbf{q}}$. We use a quaternion to represent orientation, for reasons discussed in Section 3.2. Pose $\mathbf{p}$ is defined relative to an arbitrary global reference frame. In practice we centre this global reference frame at the mean location of all camera poses. We train the model with supervised learning using pose labels, $\mathbf{p} = [\mathbf{x}, \mathbf{q}]$, obtained through structure from motion, or otherwise (Section 4.1).

3.1. Architecture

Our pose regression formulation is capable of being applied to any neural network trained through back propagation. For the experiments in this paper we adapt a state of the art deep neural network architecture for classification, GoogLeNet [52], as a basis for developing our pose regression network. This allows us to use pretrained weights, for example from a model trained to classify images in the

ImageNet dataset [11]. We observe that these pretrained features regularise and improve performance in PoseNet through transfer learning [37]. Although, to generalise PoseNet, we may apply it to any deep architecture designed for image classification as follows:

1. Remove the final linear regression and softmax layers used for classification
2. Append a linear regression layer. This fully connected layer is designed to output a seven dimensional pose vector representing position (3 dimensions) and orientation (4 dimensional quaternion)
3. Insert a normalisation layer to normalise the four dimensional quaternion orientation vector to unit length

3.2. Pose representation

An important consideration when designing a machine learning system is the representation space of the output. We can easily learn camera position in Euclidean space [22]. However, learning orientation is more complex. In this section we compare a number of different parametrisations used to express rotational quantities; Euler angles, axis-angle, $SO(3)$ rotation matrices and quaternions [2]. We evaluate their efficacy for deep learning.

Firstly, Euler angles are easily understandable as an interpretable parametrisation of 3-D rotation. However, they have two problems. Euler angles wrap around at 2π radians, having multiple values representing the same angle. Therefore they are not injective, which causes them to be challenging to learn as a uni-modal scalar regression task. Additionally, they do not provide a unique parametrisation for a given angle and suffer from the well-studied problem of gimbal lock [2]. The axis-angle representation is another three dimensional vector representation. However like Euler angles, it too suffers from a repetition around the 2π radians representation.

Rotation matrices are a over-parametrised representation of rotation. For 3-D problems, the set of rotation matrices are 3×3 dimensional members of the special orthogonal Lie group, $SO(3)$. These matrices have a number of interesting properties, including orthonormality. However, it is difficult to enforce the orthogonality constraint when learning a $SO(3)$ representation through back-propagation.

In this work, we chose quaternions as our orientation representation. Quaternions are favourable because arbitrary four dimensional values are easily mapped to legitimate rotations by normalizing them to unit length. This is a simpler process than the orthonormalization required of rotation matrices. Quaternions are a continuous and smooth representation of rotation. They lie on the unit manifold, which is a simple constraint to enforce through back-propagation. Their main downside is that they have two mappings for each rotation, one on each hemisphere. However, in Section 3.3.1 we show how to adjust the loss function to compensate for this.

3.3. Loss function

This far, we have described the structure of the pose representation we would like our network to learn. Next, we discuss how to design an effective loss function to learn to estimate the camera's 6 degree of freedom pose. This is a particularly challenging objective because it involves learning two distinct quantities - rotation and translation - with different units and scales.

This section defines a number of loss functions and explores their efficacy for camera pose regression. We begin in Section 3.3.2 by describing the original weighted loss function which was proposed by PoseNet [22]. We improve on this in Section 3.3.3 by introducing a novel loss function which can learn the weighting between rotation and translation automatically, using an estimate of the *homoscedastic* task uncertainty. Further, in Section 3.3.4 we describe a loss function which combines position and orientation as a single scalar using the reprojection error geometry. In Section 4.2 we compare the performance of these loss functions, and discusses their trade-offs.

3.3.1 Learning position and orientation

We can learn to estimate camera position by forming a smooth, continuous and injective regression loss in Euclidean space, $\mathcal{L}_x(I) = \|\mathbf{x} - \hat{\mathbf{x}}\|_\gamma$, with norm given by γ ([22] used the L_2 Euclidean norm).

However, learning camera orientation is not as simple. In Section 3.2 we described a number of options for representing orientation. Quaternions are an attractive choice for deep learning because they are easily formulated in a continuous and differentiable way. The set of rotations lives on the unit sphere in quaternion space. We can easily map any four dimensional vector to a valid quaternion rotation by normalising it to unit length. [22] demonstrates how to learn to regress quaternion values:

$$\mathcal{L}_q(I) = \left\| \mathbf{q} - \frac{\hat{\mathbf{q}}}{\|\hat{\mathbf{q}}\|} \right\|_\gamma \tag{1}$$

Using a distance norm, γ, in Euclidean space makes no effort to keep $\mathbf{q}$ on the unit sphere. We find, however, that during training, $\mathbf{q}$ becomes close enough to $\hat{\mathbf{q}}$ such that the distinction between spherical distance and Euclidean distance becomes insignificant. For simplicity, and to avoid hampering the optimization with unnecessary constraints, we chose to omit the spherical constraint. The main problem with Quaternions is that they are not injective because they have two unique values (from each hemisphere) which map to a single rotation. This is because quaternion, $\mathbf{q}$, is identical to $-\mathbf{q}$. To address this, we constrain all quaternions to one hemisphere such that there is a unique value for each rotation.

3.3.2 Simultaneously learning position and orientation

The challenging aspect of learning camera pose is designing a loss function which is able to learn both position and orientation. Initially, we proposed a method to combine position and orientation into a single loss function with a linear weighted sum [22], shown in (2):

$$\mathcal{L}_\beta(I) = \mathcal{L}_x(I) + \beta\mathcal{L}_q(I) \tag{2}$$

Because $\mathbf{x}$ and $\mathbf{q}$ are expressed in different units, a scaling factor, β, is used to balance the losses. This hyperparameter attempts to keep the expected value of position and orientation errors approximately equal.

Interestingly, we observe that a model which is jointly trained to regress the camera's position and orientation performs better than separate models trained on each task individually. Figure 2 shows that with just position, or just orientation information, the network was not able to determine the function representing camera pose with as great accuracy. The model learns a better representation for pose when supervised with both translation and orientation labels. We also experimented with branching the network lower down into two separate components to regress position and orientation. However, we found that it too was less effective, for similar reasons: separating into distinct position and orientation features denies each the information necessary to factor out orientation from position, or vice versa.

However the consequence of this was that the hyperparameter β required significant tuning to get reasonable results. In the loss function (2) a balance β must be struck between the orientation and translation penalties (Figure 2). They are highly coupled as they are regressed from the same model weights. We found β to be greater for outdoor scenes as position errors tended to be relatively greater. Following this intuition it is possible to fine tune β using grid search. For the indoor scenes it was between 120 to 750 and outdoor scenes between 250 to 2000. This is an expensive task in practice, as each experiment can take days to complete. It is desirable to find a loss function which removes this hyperparameter. Therefore, the remainder of this section explores different loss functions which aim to find an optimal weighting automatically.

3.3.3 Learning an optimal weighting

Ideally, we would like a loss function which is able to learn position and orientation optimally, without including any hyper parameters. For this reason, we propose a novel loss function which is able to learn a weighting between the position and orientation objective functions. We formulate it using *homoscedastic uncertainty* which we can learn using probabilistic deep learning [20]. Homoscedastic uncertainty is a measure of uncertainty which does not depend on the input data, as opposed to heteroscedastic uncertainty which is a function of the input data [20]. Rather, it captures the uncertainty of the task itself. In [21] we show how to use this insight to combine losses for different tasks in a probabilistic manner. Here we show how to apply this to learn camera position and orientation (with a Laplace likelihood):

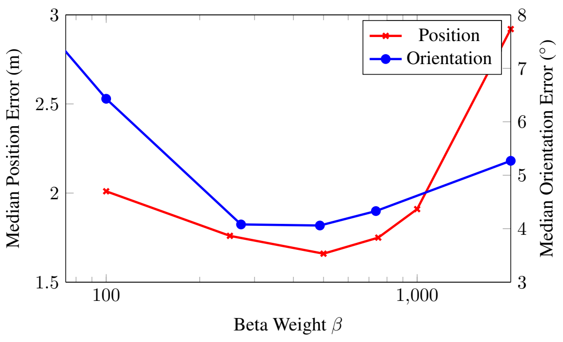


Figure 2: Relative performance of position and orientation regression on **a single model with a range of scale factors** for an indoor scene from the King's College scene in Cambridge Landmarks, using the loss function in (2). This demonstrates that learning with the optimum scale factor leads to the network uncovering a more accurate pose function.

$$\mathcal{L}_\sigma(I) = \mathcal{L}_x(I)\hat{\sigma}_x^{-2} + \log\hat{\sigma}_x^2 + \mathcal{L}_q(I)\hat{\sigma}_q^{-2} + \log\hat{\sigma}_q^2 \tag{3}$$

where we optimise the homoscedastic uncertainties, $\hat{\sigma}_x^2$, $\hat{\sigma}_q^2$, through backpropagation with respect to the loss function. These uncertainties are free scalar values, not model outputs. They represent the homoscedastic (task) noise.

This loss consists of two components; the residual regressions and the uncertainty regularization terms. We learn the variance, σ^2, implicitly from the loss function. As the variance is larger, it has a tempering effect on the residual regression term; larger variances (or uncertainty) results in a smaller residual loss. The second regularization term prevents the network from predicting infinite uncertainty (and therefore zero loss). As we expect quaternion values to have much smaller values (they are constrained to the unit manifold), their noise, σ_q^2 should be much smaller than the position noise, σ_x^2, which can be many meters in magnitude. As σ_q^2 should be much smaller than σ_x^2, orientation regression should be weighted much higher than position – with a similar effect to β in (2).

In practice, we learn $\hat{s} := \log\hat{\sigma}^2$ because it is more numerically stable [21]:

$$\mathcal{L}_\sigma(I) = \mathcal{L}_x(I)\exp(-\hat{s}_x) + \hat{s}_x + \mathcal{L}_q(I)\exp(-\hat{s}_q) + \hat{s}_q \tag{4}$$

This is more numerically stable than regressing the variance, σ^2, because the loss avoids a potential division by zero. The exponential mapping also allows us to regress unconstrained scalar values, where $\exp(-s_i)$ is resolved to the positive domain giving valid values for variance. We find that this loss is very robust to our initialisation choice for the homoscedastic task uncertainty values. Only an approximate initial guess is required, we arbitrarily use initial values of $\hat{s}_x = 0.0, \hat{s}_q = -3.0$, for all scenes.

3.3.4 Learning from geometric reprojection error

Perhaps a more desirable loss is one that does not require balancing of rotational and positional quantities at all. Reprojection error of scene geometry is a representation which combines rotation and translation naturally in a single scalar loss [14]. Reprojection error is given by the residual between 3-D points in the scene projected onto a 2-D image plane using the ground truth and predicted camera pose. It therefore converts rotation and translation quantities into image coordinates. This naturally weights translation and rotation quantities depending on the scene and camera geometry.

To formulate this loss, we first define a function, π, which maps a 3-D point, $\mathbf{g}$, to 2-D image coordinates, $(u, v)^T$:

$$\pi(\mathbf{x}, \mathbf{q}, \mathbf{g}) \mapsto \begin{pmatrix} u \\ v \end{pmatrix} \tag{5}$$

where $\mathbf{x}$ and $\mathbf{q}$ represent the camera position and orientation. This function, π, is defined as:

$$\begin{pmatrix} u' \\ v' \\ w' \end{pmatrix} = \mathsf{K}(\mathsf{R}\mathbf{g} + \mathbf{x}), \quad \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u'/w' \\ v'/w' \end{pmatrix} \tag{6}$$

where K is the intrinsic calibration matrix of the camera, and R is the mapping of $\mathbf{q}$ to its $SO(3)$ rotation matrix, $\mathbf{q}_{4\times1} \mapsto \mathsf{R}_{3\times3}$.

We formulate this loss by taking the norm of the reprojection error between the predicted and ground truth camera pose. We take the subset, $\mathcal{G}'$, of all 3-D points in the scene, $\mathcal{G}$, which are visible in the image I. The final loss (7) is given by the mean of all the residuals from points, $g_i \in \mathcal{G}'$:

$$\mathcal{L}_g(I) = \frac{1}{|\mathcal{G}'|} \sum_{g_i \in \mathcal{G}'} \|\pi(\mathbf{x}, \mathbf{q}, \mathbf{g_i}) - \pi(\hat{\mathbf{x}}, \hat{\mathbf{q}}, \mathbf{g_i})\|_\gamma \tag{7}$$

where $\hat{\mathbf{x}}$ and $\hat{\mathbf{q}}$ are the predicted camera poses from PoseNet, with $\mathbf{x}$ and $\mathbf{q}$ the ground truth label, with norm, γ, which is discussed in Section 3.3.5.

Note that because we are projecting 3-D points using both the ground truth and predicted camera pose we can apply any arbitrary camera model, as long as we use the same intrinsic parameters for both cameras. Therefore for simplicity, we set the camera intrinsics, K, to the identity matrix – camera calibration is not required.

This loss implicitly combines rotation and translational quantities into image coordinates. Minimising reprojection error is often the most desirable balance between these quantities for many applications, such as augmented reality. The key advantage of this loss is that it allows the model to vary the weighting between position and orientation, depending on the specific geometry in the training image. For example, training images with geometry which is far away would balance rotational and translational loss differently to images with geometry very close to the camera.

Interestingly, when experimenting with the original weighted loss in (2) we observed that the hyperparameter β was an approximate function of the scene geometry. We observed that it was a function of the landmark distance and size in the scene. Our intuition was that the optimal choice for β was approximating the reprojection error in the scene geometry. For example, if the scene is very far away, then rotation is more significant than translation and vice versa. This function is not trivial to model for complex scenes with a large number of landmarks. It will vary significantly with each training example in the dataset. By learning with reprojection error we can use our knowledge of the scene geometry more directly to automatically infer this weighting.

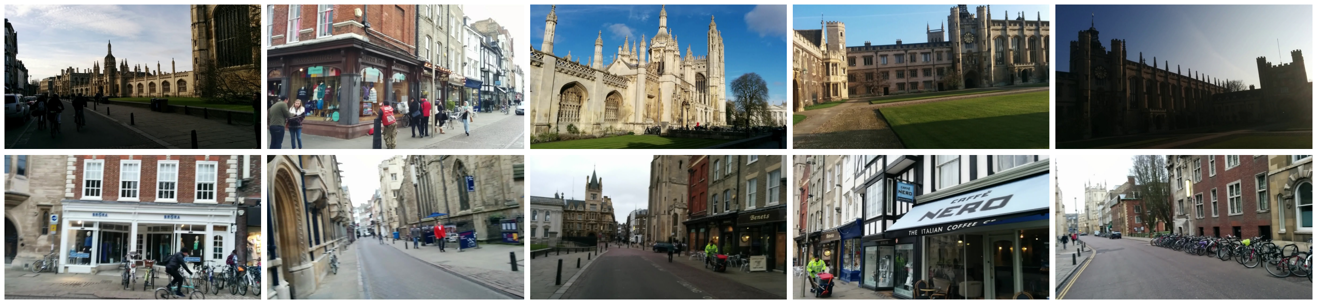
Projecting geometry through a projection model is a differentiable operation involving matrix multiplication. Therefore we can use this loss to train our model with stochastic gradient descent. It is important to note that we do not need to know the intrinsic camera parameters to project this 3-D geometry. This is because we apply the same projection to both the model prediction and ground truth measurement, so we can use arbitrary values.

It should be noted that we need to have some knowledge of the scene's geometry in order to have 3-D points to reproject. The geometry is often known; if our data is obtained through structure from motion, RGBD data or other sensory data (see Section 4.1). Only points from the scene which are visible in the image I are used to compute the loss. We also found it was important for numerical stability to ignore points which are projected outside the image bounds.

3.3.5 Regression norm

An important choice for these losses is the regression norm, $\| \|_\gamma$. Typically, deep learning models use an $L_1 = \| \|_1$ or $L_2 = \| \|_2$. We can also use robust norms such as Huber's loss [16] and Tukey's loss [15], which have been successfully applied to deep learning [5]. For camera pose regression, we found that they negatively impacted performance by over-attenuating difficult examples. We suspect that for more noisy datasets these robust regression functions might be beneficial. With the datasets used in this paper, we found the L_1 norm to perform best and therefore use use $\gamma = 1$. It does not increase quadratically with magnitude or over-attenuate large residuals.

(a) 7 Scenes Dataset - 43,000 images from seven scenes in small indoor environments [47].


(b) Cambridge Landmarks Dataset - over 10,000 images from six scenes around Cambridge, UK [22].

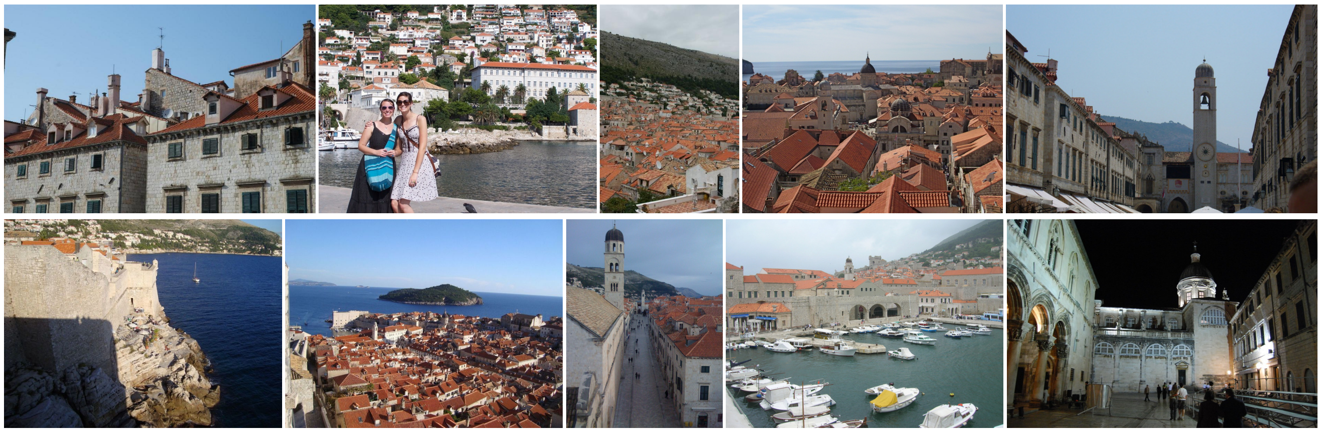

(c) Dubrovnik 6K Dataset - 6,000 images from a variety of camera types in Dubrovnik, Croatia [29].

Figure 3: **Example images randomly chosen from each dataset.** This illustrates the wide variety of settings and scales and the challenging array of environmental factors such as lighting, occlusion, dynamic objects and weather which are captured in each dataset.

Dataset	Type	Scale	Imagery	Scenes	Train Images	Test Images	3-D Points	Spatial Area
7 Scenes [47]	Indoor	Room	RGB-D sensor (Kinect)	7	26,000	17,000	-	4×3m
Cambridge Landmarks [22]	Outdoor	Street	Mobile phone camera	6	8,380	4,841	2,097,191	100×500m
Dubrovnik 6K [26]	Outdoor	Small town	Internet images (Flikr)	1	6,044	800	2,106,456	1.5×1.5km

Table 1: **Summary of the localisation datasets used in this paper's experiments.** These datasets are all publicly available. They demonstrate our method's performance over a range of scales for both indoor and outdoor applications.

4. Experiments

To train and benchmark our model on a number of datasets we rescale the input images such that the shortest side is of length 256. We normalise the images so that input pixel intensities range from -1 to 1. We train our PoseNet architecture using an implementation in TensorFlow [1]. All models are optimised end-to-end with ADAM [23] using the default parameters and a learning rate of 1×10^{-4}. We train each model until the training loss converges. We use a batch size of 64 on a NVIDIA Titan X (Pascal) GPU, training takes approximately 20k - 100k iterations, or 4 hours - 1 day.

	Cambridge Landmarks, King's College [22]			*Dubrovnik 6K* [29]		
	Median Error		Accuracy	Median Error		Accuracy
Loss function	x [m]	q [°]	$< 2m, 5°$ [%]	x [m]	q [°]	$< 10m, 10°$ [%]
Linear sum, $\beta = 500$ (2)						
Learn weighting with homoscedastic uncertainty (3)	1.11	1.99	72.5%	8.88	6.54	44.0%
Reprojection loss	does not converge					
Learn weighting pretraining $\mapsto$ Reprojection loss (7)	0.96	2.21	78.1%	7.90	4.40	48.6%

Table 2: **Comparison of different loss functions.** We use an L1 distance for the residuals in each loss. *Linear sum* combines position and orientation losses with a constant scaling parameter β [19] and is defined in (2). Learn weighting is the loss function in (3) which learns to combine position and orientation using homoscedastic uncertainty. Reprojection error implicitly combines rotation and translation by using the reprojection error of the scene geometry as the loss (7). We find that homoscedastic uncertainty is able to learn an effective weighting between position and orientation quantities. The reprojection loss was not able to converge from random initialisation. However, when used to fine-tune a network pretrained with (3) it yields the best results.

4.1. Datasets

Deep learning performs extremely well on large datasets. However annotating ground truth labels on these datasets is often expensive or very labour intensive. We can leverage structure from motion [49], or similar algorithms [47], to autonomously generate training labels (camera poses) from image data [22]. We use three datasets to benchmark our approach. These datasets are summarised in Table 1 and example imagery is shown in Figure 3. We use these datasets to demonstrate our method's performance across a range of settings and scales. We endeavour to demonstrate the general applicability of the approach.

Cambridge Landmarks [22] provides labelled video data to train and test pose regression algorithms in an outdoor urban setting. It was collected using a smart phone and structure from motion was used to generate the pose labels [57]. Significant urban clutter such as pedestrians and vehicles were present and data was collected from many different points in time representing different lighting and weather conditions. Train and test images are taken from distinct walking paths and not sampled from the same trajectory making the regression challenging.

7 Scenes [47] is an indoor dataset which was collected with a Kinect RGB-D sensor. Ground truth poses were computed using Kinect Fusion [47]. The dataset contains seven scenes which were captured around an office building. Each scene typically consists of a single room. The dataset was originally created for RGB-D relocalization. It is extremely challenging for purely visual relocalization using SIFT-like features, as it contains many ambiguous textureless features.

Dubrovnik 6K [26] is a dataset consisting of 6,044 train and 800 test images which were obtained from the internet. They are taken from Dubrovnik's old town in Croatia which is a UNESCO world heritage site. The images are predominantly captured by tourists with a wide variety of camera types. Ground truth poses for this dataset were computed using structure from motion.

4.2. Comparison of loss functions

In Table 2 we compare different combinations of losses and regression norms. We compare results for a scene in the Cambridge Landmarks dataset [22] and the Dubrovnik 6K dataset [26], which has imagery from a range of cameras.

We find that modelling homoscedastic uncertainty with the loss in (3) is able to effectively learn a weighting between position and orientation. It outperforms the constant weighting used in loss (2). The reprojection loss in (7) is unable to train the model from a random initialisation. We observe that the model gets stuck in a poor local minima, when using any of the regression norms. However, the reprojection loss is able to improve localisation performance when using weights pretrained with any of the other losses. For example, we can take the best performing model using the loss from (3) and fine tune with the reprojection loss (7). We observe that this loss is then able to converge effectively. This shows that the reprojection loss is not robust to large residuals. This is because reprojected points can be easily placed far from the image centre if the network makes a poor pose prediction.

Therefore, we recommend the following two-step training scheme:

1. Train the model using the loss in (3), learning the weighting between position and orientation.
2. If the scene geometry is known (for example from structure from motion or RGBD camera data) then fine-tune the model using the reprojection loss in (7).

4.3. Benchmarking localisation accuracy

In Table 3 we show that our geometry based loss outperforms the original PoseNet's naive loss function [22]. We observe a consistent and significant improvement across both indoor *7 Scenes* outdoor *Cambridge Landmarks* datasets. We conclude that we can simultaneously learn both position and orientation more effectively by considering scene geometry. The improvement is notably more pronounced for the 7Scenes dataset. We believe this is

Scene	Area or Volume	Active Search (SIFT) [44]	PoseNet (β weight) [22]	Bayesian PoseNet [19]	PoseNet Spatial LSTM [55]	PoseNet (this work) Learn σ^2 Weight	PoseNet (this work) Geometric Reprojection
King's College	$5600m^2$	0.42m, 0.55°	1.66m, 4.86°	1.74m, 4.06°	0.99m, 3.65°	1.11m, 1.99°	0.96m, 2.21°
Old Hospital	$2000m^2$	0.44m, 1.01°	2.62m, 4.90°	2.57m, 5.14°	1.51m, 4.29°	2.28m, 5.36°	3.02m, 5.22°
Shop Façade	$875m^2$	0.12m, 0.40°	1.41m, 7.18°	1.25m, 7.54°	1.18m, 7.44°	0.98m, 5.44°	1.02m, 4.69°
St Mary's Church	$4800m^2$	0.19m, 0.54°	2.45m, 7.96°	2.11m, 8.38°	1.52m, 6.68°	1.52m, 5.87°	1.93m, 5.04°
Great Court	$8000m^2$	–	–	–	–	4.70m, 6.49°	4.82m, 2.96°
Chess	$6m^3$	0.04m, 1.96°	0.32m, 6.60°	0.37m, 7.24°	0.24m, 5.77°	°	°
Fire	$2.5m^3$	0.03m, 1.53°	0.47m, 14.0°	0.43m, 13.7°	0.34m, 11.9°	0.27m, 11.2°	°
Heads	$1m^3$	0.02m, 1.45°	0.30m, 12.2°	0.31m, 12.0°	0.21m, 13.7°	0.18m, 12.1°	0.17m, 13.0°
Office	$7.5m^3$	0.09m, 3.61°	0.48m, 7.24°	0.48m, 8.04°	0.30m, 8.08°	0.20m, 5.77°	0.19m, 5.55°
Pumpkin	$5m^3$	0.08m, 3.10°	0.49m, 8.12°	0.61m, 7.08°	0.33m, 7.00°	0.25m, 4.82°	0.26m, 5.18°
Red Kitchen	$18m^3$	0.07m, 3.37°	0.58m, 8.34°	0.58m, 7.54°	0.37m, 8.83°	0.24m, 5.52°	0.23m, 5.35°
Stairs	$7.5m^3$	0.03m, 2.22°	0.48m, 13.1°	0.48m, 13.1°	0.40m, 13.7°	0.37m, 10.6°	0.36m, 12.4°

Table 3: **Median localization results for the *Cambridge Landmarks* [22] and *7 Scenes* [47] datasets.** We compare the performance of various RGB-only algorithms. Active search [44] is a state-of-the-art traditional SIFT keypoint based baseline. We demonstrate a notable improvement over PoseNet's [22] baseline performance using the learned σ^2 and reprojection error proposed in this paper, narrowing the margin to the state of the art SIFT technique.

due to the significantly larger amount of training data for each scene in this dataset, compared with Cambridge Landmarks. We also outperform the improved PoseNet architecture with spatial LSTMs [55]. However, this method is complimentary to the loss functions in this paper, and it would be interesting to explore the union of these ideas.

We observe a difference in relative performance between position and orientation when optimising with respect to reprojection error (7) or homoscedastic uncertainty (3). Overall, optimising reprojection loss improves rotation accuracy, sometimes at the expense of some positional precision.

4.4. Comparison to SIFT-feature approaches

Table 3 also compares to a state-of-the-art traditional SIFT feature based localisation algorithm, Active Search [44]. This method outperforms PoseNet, and is effective in feature-rich outdoor environments. However, in the 7Scenes dataset this deficit is less pronounced. The indoor scenes contain much fewer point features and there is significantly more training data. As an explanation for the deficit in these results, it is worth noting that PoseNet only uses 256×256 pixel images, while SIFT based methods require images of a few mega-pixels in size [44]. Additionally, PoseNet is able to localise an image in $5ms$, scaling constantly with scene area, while traditional SIFT feature approaches require over $100ms$, which scales with scene size [44].

In Table 4 we compare our approach on the Dubrovnik dataset to other geometric techniques which localise by registering SIFT features [30] to a large 3-D model [26]. Although our method improves significantly over the original PoseNet model, it is still yet to reach the fine grained accuracy of these methods [51, 58, 42, 29]. We hypothesise that this is due to a lack of training data, with only 6k images across the town. However, our algorithm is significantly faster than these approaches. Furthermore, it is worth noting that PoseNet only sees a 256×256 resolution image, while these methods register the full size images, often with a few million pixels.

Method	Position		Orientation	
	Mean [m]	Median [m]	Mean [°]	Median [°]
PoseNet (this work)	40.0	7.9	11.2	4.4
APE [51]	-	0.56	-	-
Voting [58]	-	1.69	-	-
Sattler, et al. [42]	14.9	1.3	-	-
P2F [29]	18.3	9.3	-	-

Table 4: **Localisation results on the Dubrovnik dataset** [26], comparing to a number of state-of-the-art point-feature techniques. Our method is the first deep learning approach to benchmark on this challenging dataset. We achieve comparable performance, while our method only requires a 256×256 pixel image and is much faster to compute.

5. Conclusions

We have investigated a number of loss functions for learning to regress position and orientation simultaneously with scene geometry. We present an algorithm for training PoseNet which does not require any hyper-parameter tuning. We demonstrate PoseNet's efficacy on three large scale datasets. We observe a large improvement of results compared to the original loss proposed by PoseNet and show that PoseNet can outperform traditional SIFT feature-based methods in indoor scenes with sufficient training data.

For many applications which require localization, such as mobile robotics, video data is readily available. Ultimately, we would like to extend the architecture to video input with further use of multi-view stereo [14].

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] S. L. Altmann. *Rotations, quaternions, and double groups*. Courier Corporation, 2005.

[3] A. Babenko and V. Lempitsky. Aggregating deep convolutional features for image retrieval. 2015.

[4] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. 2014.

[5] V. Belagiannis, C. Rupprecht, G. Carneiro, and N. Navab. Robust optimization for deep regression. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2830–2838. IEEE, 2015.

[6] D. Chen, G. Baatz, K. Köser, S. Tsai, R. Vedantham, T. Pylvänäinen, K. Roimela, X. Chen, J. Bach, M. Pollefeys, B. Girod, and R. Grzeszczuk. City-scale landmark identification on mobile devices. 2011.

[7] S. Choudhary and P. J. Narayanan. Visibility probability structure from sfm datasets and applications. 2012.

[8] R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen. Vidloc: 6-dof video-clip relocalization. *arXiv preprint arXiv:1702.06521*, 2017.

[9] M. Cummins and P. Newman. Highly Scalable Appearance-Only SLAM - FAB-MAP 2.0. In *RSS*, 2019.

[10] J. Delhumeau, P.-H. Gosselin, H. Jégou, and P. Pérez. Revisiting the VLAD image representation. In *ACM Multimedia*, Barcelona, Spain, Oct. 2013.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[12] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular slam. In *Computer Vision–ECCV 2014*, pages 834–849. Springer, 2014.

[13] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. 2014.

[14] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[15] D. C. Hoaglin, F. Mosteller, and J. W. Tukey. *Understanding robust and exploratory data analysis*, volume 3. Wiley New York, 1983.

[16] P. J. Huber. *Robust statistics*. Springer, 2011.

[17] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. pages 3304–3311, jun 2010.

[18] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local image descriptors into compact codes. 34(9):1704–1716, Sept. 2012.

[19] A. Kendall and R. Cipolla. Modelling uncertainty in deep learning for camera relocalization. *arXiv preprint arXiv:1509.05909*, 2015.

[20] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? *arXiv preprint arXiv:1703.04977*, 2017.

[21] A. Kendall, Y. Gal, and R. Cipolla. Multi-task deep learning using task-dependant homoscedastic uncertainty for depth regression, semantic and instance segmentation. *arXiv preprint arXiv*, 2017.

[22] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. *arXiv preprint arXiv:1505.07427*, 2015.

[23] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[24] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE, 2007.

[25] R. Li, Q. Liu, J. Gui, D. Gu, and H. Hu. Indoor relocalization in challenging environments with dual-stream convolutional neural networks. *IEEE Transactions on Automation Science and Engineering*, 2017.

[26] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide pose estimation using 3d point clouds. In *Computer Vision–ECCV 2012*, pages 15–29. Springer, 2012.

[27] Y. Li, N. Snavely, D. Huttenlocher, and P. Fua. Worldwide Pose Estimation Using 3D Point Clouds. 2012.

[28] Y. Li, N. Snavely, and D. P. Huttenlocher. Location Recognition using Prioritized Feature Matching. 2010.

[29] Y. Li, N. Snavely, and D. P. Huttenlocher. Location recognition using prioritized feature matching. In *European Conference on Computer Vision*, pages 791–804. Springer, 2010.

[30] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[31] I. Melekhov, J. Kannala, and E. Rahtu. Relative camera pose estimation using convolutional neural networks. *arXiv preprint arXiv:1702.01381*, 2017.

[32] E. I. Moser, E. Kropff, and M.-B. Moser. Place cells, grid cells, and the brain's spatial representation system. *Annu. Rev. Neurosci.*, 31:69–89, 2008.

[33] R. Mur-Artal, J. Montiel, and J. D. Tardós. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[34] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2320–2327. IEEE, 2011.

[35] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. 2006.

[36] J. O'keefe and L. Nadel. *The hippocampus as a cognitive map*, volume 3. Clarendon Press Oxford, 1978.

[37] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1717–1724. IEEE, 2014.

[38] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object Retrieval with Large Vocabularies and Fast Spatial Matching. 2007.

[39] A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson. A baseline for visual instance retrieval with deep convolutional networks. In *arXiv:1412.6574*, 2014.

[40] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. IEEE, 2011.

[41] T. Sattler, B. Leibe, and L. Kobbelt. Fast Image-Based Localization using Direct 2D-to-3D Matching. 2011.

[42] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *2011 International Conference on Computer Vision*, pages 667–674. IEEE, 2011.

[43] T. Sattler, B. Leibe, and L. Kobbelt. Improving Image-Based Localization by Active Correspondence Search. 2012.

[44] T. Sattler, B. Leibe, and L. Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.

[45] T. Sattler, C. Sweeney, and M. Pollefeys. On Sampling Focal Length Values to Solve the Absolute Pose Problem. 2014.

[46] G. Schindler, M. Brown, and R. Szeliski. City-Scale Location Recognition. 2007.

[47] 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 *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2930–2937. IEEE, 2013.

[48] J. Sivic and A. Zisserman. Video Google: A Text Retrieval Approach to Object Matching in Videos. 2003.

[49] N. Snavely, S. Seitz, and R. Szeliski. Modeling the World from Internet Photo Collections. 80(2):189–210, 2008.

[50] L. Svarm, O. Enqvist, M. Oskarsson, and F. Kahl. Accurate Localization and Pose Estimation for Large 3D Models. 2014.

[51] L. Svarm, O. Enqvist, M. Oskarsson, and F. Kahl. Accurate localization and pose estimation for large 3d models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 532–539, 2014.

[52] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

[53] G. Tolias, R. Sicre, and H. Jgou. Particular object retrieval with integral max-pooling of cnn activations. In *ICLR*, 2016.

[54] A. Torii, J. Sivic, T. Pajdla, and M. Okutomi. Visual Place Recognition with Repetitive Structures. 2013.

[55] F. Walch, C. Hazirbas, L. Leal-Taixé, T. Sattler, S. Hilsenbeck, and D. Cremers. Image-based localization with spatial lstms. *arXiv preprint arXiv:1611.07890*, 2016.

[56] T. Weyand, I. Kostrikov, and J. Philbin. Planet-photo geolocation with convolutional neural networks. *arXiv preprint arXiv:1602.05314*, 2016.

[57] C. Wu. Towards linear-time incremental structure from motion. In *3D Vision-3DV 2013, 2013 International Conference on*, pages 127–134. IEEE, 2013.

[58] B. Zeisl, T. Sattler, and M. Pollefeys. Camera pose voting for large-scale image-based localization. In *International Conference on Computer Vision (ICCV)*, 2015.