xyzNet: Towards Machine Learning Camera Relocalization by Using a Scene Coordinate Prediction Network

Nam-Duong Duong, Amine Kacete, Catherine Sodalie, Pierre-Yves Richard, Jérôme Royan

xyzNet: Towards Machine Learning Camera Relocalization by Using a Scene Coordinate Prediction Network

Nam-Duong Duong* IRT b-com
Amine Kacete† IRT b-com
Catherine Sodalie‡ IETR/CentraleSupélec
Pierre-Yves Richard§ IETR/CentraleSupélec
Jérôme Royan¶ IRT b-com

ABSTRACT

Camera relocalization is a common problem in several applications such as augmented reality or robot navigation. Especially, augmented reality requires fast, accurate and robust camera localization. However, it is still challenging to have a both real-time and accurate method. In this paper, we present our hybrid method combing machine learning approach and geometric approach for real-time camera relocalization from a single RGB image. We propose a light Convolutional Neural Network (CNN) called *xyzNet* to efficiently and robustly regress 3D world coordinates of key-points in an image. Then, the geometric information about 2D-3D correspondences allows the removal of ambiguous predictions and the calculation of more accurate camera pose. Moreover, we show favorable results compared to previous machine learning based approaches about the accuracy and the performance of our method on different datasets as well as the capacity to address challenges concerning dynamic scene.

Index Terms: Real-time RGB Camera Relocalization; Deep Learning Regression;

1 INTRODUCTION

In recent years, Augmented Reality (AR), robotics, autonomous (self-driving) vehicles have become increasingly trendy. In Augmented Reality applications, the real environment is annotated or extended with computer-generated contents. These augmentations must be exactly registered with the real space, which requires an accurate and real-time camera localization. Most of the existing solutions (e.g. Tango, Hololens) to camera pose estimation problem employ multiple sensors such as camera, GPS, LIDAR or IMU [19]. Among these sensors, RGB sensor is used in the main solution of camera pose estimation for commercial systems: Simultaneously Localization And Mapping (SLAM): direct SLAM [7, 26]; indirect SLAM [6, 18, 20]. SLAM methods process an ordered sequence of images acquired from a set-up of one or more cameras potentially associated with an inertial measurement unit. SLAM methods first match features amongst the current and previous image, then they can both reconstruct a 3D scene model and estimate camera pose. Unfortunately, in the case of fast camera motion or sudden change of viewpoint such as in a hand-held camera, tracking failure interrupted camera pose estimation. Camera relocalization is then needed. SLAM methods solutions to camera relocalization use a large set of key-points [22, 23] or key-frames [10, 20]. Consequently, memory usage as well as processing time increase linearly with respect to the size of the tracking area. In this paper, we focus on an augmented reality solution based on camera relocalization only.

*e-mail: nam-duong.duong@b-com.com
†e-mail: amine.kacete@b-com.com
‡e-mail: catherine.soladie@centralesupelec.fr
§e-mail: pierre-yves.richard@centralesupelec.fr
¶e-mail: jerome.royan@b-com.com

In recent years, machine learning approaches have become widespread in computer vision and provide excellent results for many applications. Last year, [9, 25] succeeded in estimating 6-DoF (degrees of freedom) object pose in real-time by using a random forest and deep learning on depth images. Camera relocalization also aims to estimate 6-DoF camera pose except that it addresses a large-scale problem. For machine learning approaches, handling camera relocalization is as a regression problem solved by a supervised learning based on the informations known in advance about each scene. The training phase uses labeled images (images and their corresponding camera poses), then the testing phase uses that trained model to relocalize the camera from each unseen image. Although machine learning-based methods are only capable of performing on known scenes, they do not require any initial estimation pose and can handle individual frames independently contrary to SLAM methods. In particular, these methods [5, 15, 17, 28] can estimate camera pose in real-time from each whole RGB image. However, limitations of these methods lie in their accuracy, their jitter, and the lack of confidence score for each pose estimation.

Besides, [2–4, 11, 24, 27] presented hybrid approaches which combine both geometric approach and machine learning approach for camera relocalization with high accuracy. Machine learning approaches are applied to learn and predict 3D position of each pixel in world coordinates. From these correspondences, geometric-based methods infer camera pose. The hybrid methods are known as voting methods, which have been successfully used in [8] for object detection. Though, in order to obtain a final 6-DoF camera pose, each pixel does not vote directly for a global quantized 6-DoF. Indeed, the high dimensionality of the research space is likely to result in a poor estimation. Instead, each pixel makes a 3D continuous prediction about its own 3D position in the world coordinates system. The camera pose is then calculated by minimizing a re-projection error function based on these correspondences. The first hybrid methods [11, 24, 27] are limited by the use of RGB-D sensors in the testing phase. As an extension of SCoRe Forest [24], [3] uses an auto-context regression forest from only RGB image patches. [2] instead uses a VGG style architecture to predict scene coordinates and proposes a differentiable RANSAC, so that a matching function that optimizes pose quality can be learned. Although these methods achieve high accuracy, they need thousands of predictions about scene coordinates, so that time increases a lot to infer optimal camera pose by RANSAC.

In this paper, to help overcome the previous limitations, we propose a hybrid method based on both deep learning approach and geometric approach for real-time camera relocalization. Our method can balance between accuracy and run-time. We present a light convolutional neural network regression for scene coordinate prediction from local patches. The originality of our method is to only use patches extracted based on key-point detection. The patch extraction based on key-point provides appearance more discriminant. Therefore, our network predicts efficiently scene coordinates even for blurry images or texture-less scenes. This allows balance between run-time and accuracy of our method. Furthermore, thanks to the use of patches instead of a whole image, our method is robust to occlusion. From 2D-3D point correspondences are established by our network, we implement Perspective-n-Point (PnP) and Random

sample consensus (RANSAC) to estimate camera pose. The final camera pose is attached with a confidence score which is presented as the number of inliers.

In the following sections, we present our method including *xyzNet*'s architecture, means of training and predicting 3D points in world coordinates system, as well as camera pose calculation in the section 2. Section 3 shows and discusses our results on different datasets. Finally, section 4 provides some conclusions and perspectives.

2 PROPOSED METHOD

2.1 Overview

In this section, we present our hybrid method merging both deep learning approach and geometric approach for robust camera relocalization. Figure 1 illustrates our pipeline. That can be summarized in two principal steps: 3D points location of pixels in the world coordinates system based on local patches to define 2D-3D point correspondences; camera pose calculation from these correspondences.

2.2 Deep learning for 3D points localization based on local patches

The deep learning based methods in [5, 15, 17, 28] estimate directly camera pose from whole images. Inspired by success of deep learning with local patches for object detection and segmentation [14], we propose a light convolutional neural network for inferring 3D positions of pixels into the world coordinates system based on appearance of RGB patches around them. We extract local patches of RGB image which contain significant information. Homogeneous patches such as on the sky or on road do not provide distinctive information about camera pose in the scene. The use of these patches is not recommended since it produces some noise. A problem with deep learning regression for camera relocalization lies in the lack of confidence score. Therefore, predictions are uncertain. To solve this problem, [5, 15] create a probabilistic model of results by using dropout layer after every convolutional layer as a means of sampling the model weights. In our method instead, we use a set of patches to generate a set of probabilistic results from data.

Patches extraction and labeling

From each RGB image, we extract a set of patches for camera relocalization. Instead of randomly choosing them, we deliberately get patches around key-points to target only geometrically relevant regions. Thus each chosen patch is an image region around a key-point with a fixed size. We use SURF (Speed Up Robust Features) detector [1] to detect some sparse points which are scale and rotation invariant points, as repetitive representatives in a scene. This enhances the ability to locate 3D positions from patches. The Hessian minimum threshold is is appropriately chosen to ensure detection of hundreds of key-points even on blurry or texture-less images. So from every image, we have a set of patches $\mathscr{P} = \{\mathscr{P}_i\}$ centered on the SURF key-points $p = \{p_i = (u_i, v_i)\}$, with p_i defines an image coordinates.

For the training phase of *xyzNet*, we need to label each training data with the correspondent 3D world coordinates, $P_i^w = (X_i^w, Y_i^w, Z_i^w)$. A possible solution for labeling would be the execution of SfM once on all the training dataset to carry out a mapping between key-points and point cloud. However, our experiments in the section 3 is performed on RGB-D datasets. So we can use RGB-D images from the calibrated camera with their corresponding camera poses to define labels for the training phase. Note that in the testing phase, we only use RGB images and we do not need any depth information. From the position $p_i = (u_i, v_i)$ of each key-point detected in RGB image and the corresponding depth value D_i in the depth image, a 3D position $P_i^c = (X_i^c, Y_i^c, Z_i^c)$ of the key-point in camera coordinates system is calculated by using the standard pinhole camera model as follows:

$$P_i^c = D_i K^{-1} \begin{bmatrix} p_i \\ 1 \end{bmatrix}$$

Where K is a matrix of camera intrinsic parameters. The camera pose $T = [R|t]$ including rotation matrix R and translation vector t for each frame is supposed to be known in advance. The world coordinates $P_i^w = (X_i^w, Y_i^w, Z_i^w)$ corresponding pixel p_i are defined based on the transformation equation in the homogeneous coordinates system:

$$\begin{bmatrix} P_i^w \\ 1 \end{bmatrix} = \left[\begin{array}{c|c} R & t \\ 0 & 1 \end{array}\right] \begin{bmatrix} P_i^c \\ 1 \end{bmatrix}$$

xyzNet

We designed a novel regression network to directly predict from 2D pixels the 3D correspondences in the world coordinates system. It takes RGB-image patches with a fixed size of 49×49 pixels as inputs. The most common networks used in deep learning are not suitable for processing multiple patches in terms of calculation time. We make a light ConvNet dedicated to efficient and robust camera relocalization. *xyzNet* consists of five layers that perform convolution of the input with a set of filters of 3×3 kernels, max-pooling and subsampling over a 3×3 area and a rectified linear (ReLU) activation function. In the first stage, a local response normalization (LRN) is used to normalize patches with different light condition. These five layers are followed by two fully connected layers to regress 3D world coordinates. A dropout layer is added after every fully connected layer to deal with over-fitting. *xyzNet* is illustrated in Figure 2. The use of patches based on key-points instead of random-points or grid-points reduces noise of training data as well as search space. This does not require a complex network and our light network, *xyzNet*, can perform efficiently and robustly.

In the training phase, the weights of *xyzNet* are learned by minimizing an Euclidean objective loss function with an optimization algorithm that is a stochastic gradient descent. The loss function is defined as follows:

$$l(p) = \sum_{p_i \in p} \|P_i^w - \hat{P_i^w}\|_2^2$$

Where P_i^w and $\hat{P_i^w}$ are ground truth and prediction respectively about 3D coordinates of pixel p_i in the world coordinate system.

Each patch predicts a 3D continuous prediction about its own position in world coordinates instead of camera pose (6-DOF) as in [17]. This reduces considerably the complexity of the loss function producing more efficient optimization.

2.3 Camera pose calculation

In this section, we will show how to estimate camera pose from predictions of *xyzNet*. From each RGB image, we extract a set of patches centered on keypoints. Each patch passes through *xyzNet* to generate a prediction about the 3D position in the world coordinates system. When all patches have passed through *xyzNet*, we obtain a set of 2D-3D correspondences. Just three pixel exact predictions are theoretically required to infer the camera pose. Nevertheless, a well-known computer vision method can solve this problem, namely Perspective-n-Points (PnP) algorithm. But as the *xyzNet* can make some noisy predictions, we do not consider directly all 2D-3D points correspondences to calculate the camera pose. Instead, we first use PnP and Ransac to remove noise (outliers) and keep exact predictions (inliers). Ransac generates a set of hypothetical poses $\mathscr{T} = \{T_i\}$ by performing PnP on random subsets of 2D-3D point correspondences. The best inliers are defined by maximizing the number of inliers corresponding to each hypothesis based on re-projection error.

$$\max_{\forall T_i \in \mathscr{T}} \sum_{p_j \in p} \rho(\alpha_{ij})$$

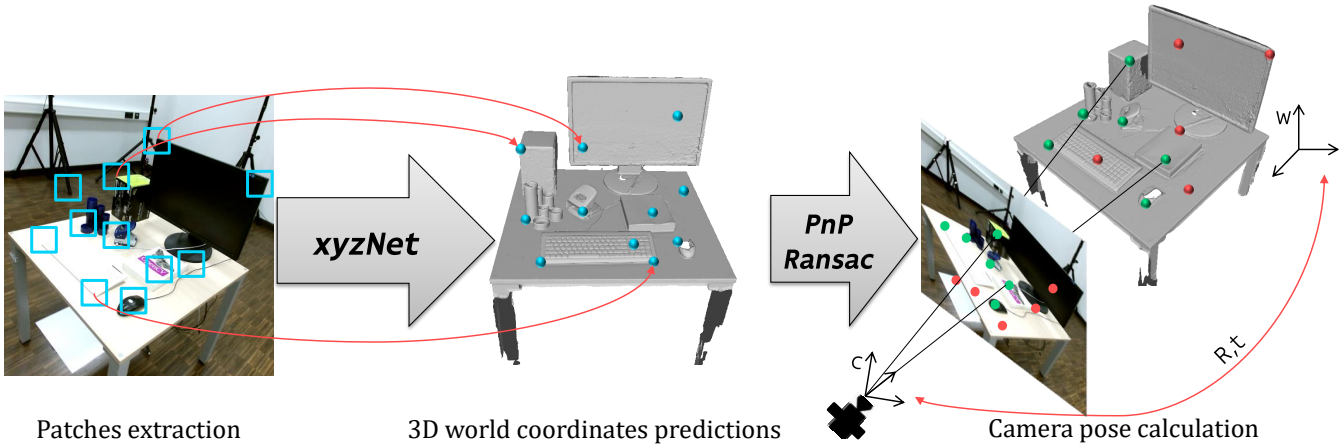


Figure 1: ***xyzNet* Camera Relocalization Pipeline**: From a set of patches (blue squares) extracted on each RGB image, we put them through *xyzNet* to predict a set of 3D positions (blue points) in the world coordinate system. Next PnP and Ransac algorithms are used to filter inliers (green points) and eliminate outliers (red points). Finally, camera pose is computed by re-running PnP once on all inliers.

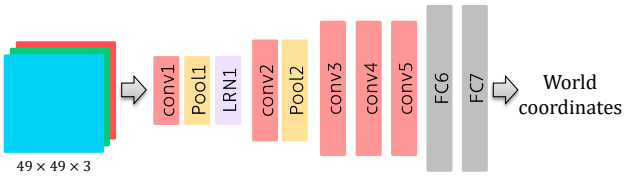


Figure 2: ***xyzNet***: A CNN regression for predicting world coordinates from RGB patches

$$\rho(\alpha_{ij}) = \begin{cases} 1, & \text{if } \alpha_{ij} < \tau \\ 0, & \text{otherwise} \end{cases}$$

Where $\alpha_{ij} = \|p_j - KT_i^{-1}\hat{P}_j^w\|^2$ and τ is the maximum threshold of re-projection error that defines inliers. Then pixel j is considered as an inlier of hypothesis T_i if $\rho(\alpha_{ij}) = 1$. Let $\mathscr{I}$ be the set of indices of the inliers associated with the best solution.

The final camera pose is carried out by running PnP once on all inliers to minimize the re-projection error function:

$$E(T) = \sum_{i \in \mathscr{I}} \|p_i - KT^{-1}\hat{P}_i^w\|^2$$

Inferring camera pose from multiple patches with filtering method based on PnP and Ransac algorithms allows to eliminate outliers on moving objects in the scene, in order to address the challenge of scenes with partial occlusion. Moreover, the number of inliers can be used as a confidence score of the final estimation which is not provided by the previous deep learning based methods. With this confidence score, we determine which frames can be used for the augmented reality. Frames with high confidence are considered as key-frames to infer camera pose for next frames when camera motion is not too great. Our results will be shown in the next section.

3 Experiments

All experiments are implemented on a NVIDIA GTX 1080 GPU using Caffe framework [12]. For the training phase, we extract up to 500 patches with fixed size from every image. We experiment many different configurations on a scene of [24] to optimize parameters that are used for other scenes. Our configuration includes: 500 epochs with a batch size of 2048; training begins at a learning rate of 10^{-2}; then the learning rate is dropped by multiplying it with a factor gamma of 0.8 after every 50 epochs; dropout probability is 0.5 for all fully connected layers; *xyzNet* is trained by using Stochastic Gradient Descent with a momentum of 0.9 and a weight decay of 10^{-5}.

3.1 Datasets

We evaluate our method on 7 scenes dataset [24] and CoRBS dataset [29], presented in Figure 3 and 4 respectively. All datasets are indoor scenes. Each one provides thousands of RGB-D images at 640 480 resolution, intrinsic matrix of camera and annotations (camera pose for every frame).

7 scenes dataset is introduced by [24]. This dataset contains seven scenes in room-scale. Each scene includes some sequences which are captured around a single room and annotated by using Kinect Fusion [21]. The data is extremely challenging with pure rotation or fast movement of camera that provides many blurry images.

CoRBS dataset [29] is also an indoor dataset. However, it contains scenes that are more texture-less. This dataset is more accurate than 7 scenes thanks to the use of multiple sensors. Visual data is achieved by using a Kinect v2. The ground truth are obtained by an external motion capture system. Each scene contains a 3D dense scene model which is created via an external 3D scanner.

3.2 Evaluation of patch extraction based on key-points

To evaluate benefits of patch extraction based on key-points, we compare it with patch extraction based on a grid-points 40×40 as proposed in [2], shown in Figure 5.

Firstly, in training phase, uses of patch extraction based on a grid of points generates a lot of training patches that includes much noisy data from homogeneous patches. Therefore, *xyzNet* training takes more time and converge with more difficulty. Figure 6 shows time of training convergence. Key-point based method is two times faster than grid-point based method.

Secondly, in testing phase, grid-point based method takes a lot of time to predict thousands patches even homogeneous patches that does not provide any information about translation and rotation of camera. So, processing these patches is redundant. As shown in Figure 5, almost patches extracted on wall or floor are outliers, since these patches did not contain enough individual feature to discriminate each other. High accurate predictions (inliers) belong to textured patches. In addition, if having too many predictions,

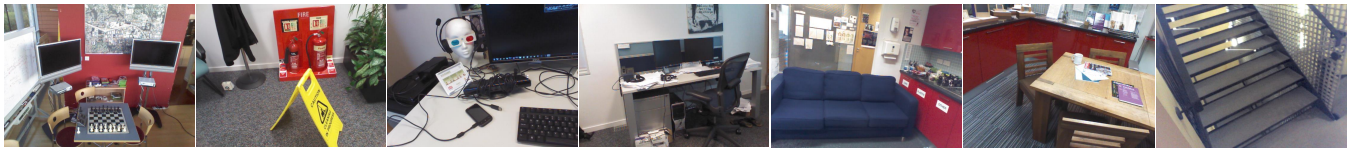


Figure 3: 7 scenes dataset: from left to right, this dataset consists of chess, fire, heads, office, pumpkin, red kitchen, stairs.

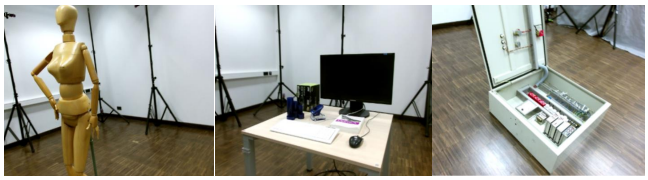


Figure 4: CoRBS dataset: three scenes of human, desk, electrical cabinet with flat surface.

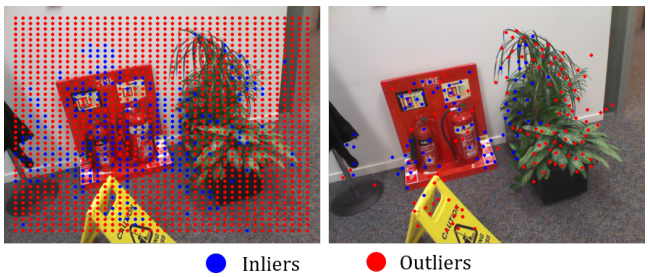


Figure 5: Detected inliers from patches extraction based grid-points (left) and key-points (right) on an image of the fire scene.

PnP Ransac need more and more time to define the best solution of camera pose. Figure 7 shows that for same computational time per image, key-point based is more accurate than grid-point based. So our method using patches extraction from key-points can reduce run-time as well as achieve high accuracy.

3.3 *xyzNet* accuracy

xyzNet is the core of our pipeline and has a powerful influence on the precision of camera pose. We consider the *xyzNet*'s accuracy through the distance error between predictions and ground truth. We give some results to clarify efficiency and robustness of *xyzNet*.

In table 1, we show an average of location error on the testing data of 7 scenes dataset. We calculate two distance errors on all predictions and all inliers which are $0.33m$ and $0.13m$ respectively. For the scenes *office, red kitchen, stairs*, we did not achieve good results, what could be explained by the repetitiveness of the scenes. That makes patches extraction on similar object ambiguous. However, filtering with Ransac and PnP algorithms greatly improve the accuracy of the estimation by decreasing the ambiguous predictions.

Scene	Chess	Fire	Heads	Office	Pumpkin	RedKitchen	Stairs
Err_P	0.25m	0.19m	0.14m	0.65m	0.27m	0.44m	0.34m
Err_I	0.13m	0.11m	0.06m	0.26m	0.11m	0.14m	0.13m

Table 1: ***xyzNet*'s error**: The mean of distance error between predictions and ground truths on the set of all predictions (Err_P) and the set of inliers (Err_I).

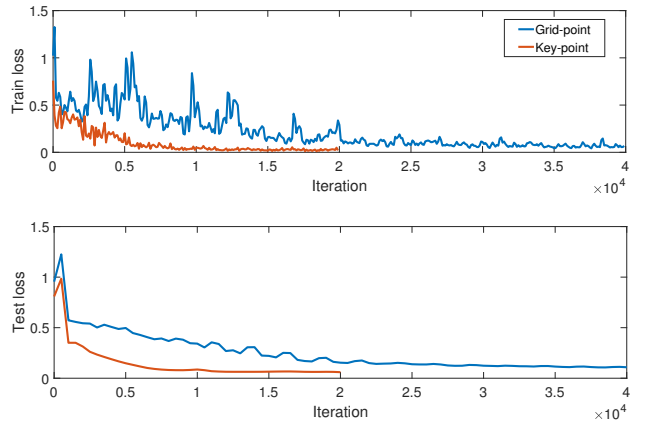


Figure 6: Training performance from patches extracted from key-points and grid-points on the chess scene.

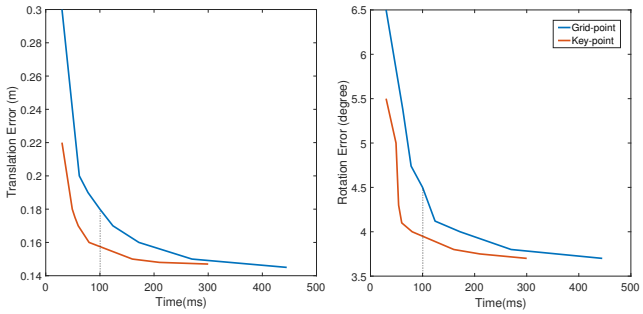


Figure 7: Relation of computational time per image and mean of accuracy by changing number of Ransac iteration for key-point based and grid-point based methods on the chess scene.

3.4 In terms of run-time and accuracy of camera relocalization

In this section, we evaluate our method on the 7 scene dataset to compare it with the state-of-the-art methods. We consider only RGB image based methods.

Baselines

We compare our method to three methods [2, 16, 23] which respectively belong to three different approaches: sparse feature based, machine learning based and hybrid based. In [23], a 3D point cloud with attached descriptors is built in the off-line phase using the training images. To accelerate feature matching, [23] uses a visual vocabulary. PoseNet 2 [16] takes a whole RGB image as an input to directly predict a camera pose. [2] presented an improvement of [3] using a CNN instead of a random forest and proposed a differentiable RANSAC so that a matching function that optimizes pose quality can be learned.

Computational time

7 scenes	Active Search [23]	PoseNet2 [16]	DSAC [2]	Ours
Chess	0.04m,2.0°	0.13m,4.5°	**0.02m,1.2°**	0.06m,2.4°
Fire	**0.03m,1.5°**	0.27m,11.3°	0.04m,1.5°	0.06m,2.2°
Heads	**0.02m,1.5°**	0.17m,13.0°	0.03m,2.7°	0.08m,4.8°
Office	0.09m,3.6°	0.19m,5.6°	**0.04m,1.6°**	0.13m,5.5°
Pumpkin	0.08m,3.1°	0.26m,4.8°	**0.05m,2.0°**	0.06m,2.0°
Kitchen	0.07m,3.4°	0.23m,5.4°	0.05m,2.0°	**0.05m,1.8°**
Stairs	**0.03m,2.2°**	0.35m,12.4°	1.17m,33.1°	0.21m,6.0°
Average	**0.05m,2.5°**	0.23m,8.1°	0.20m,6.3°	0.09m,3.5°
Time	100ms	**5ms**	1500ms	60ms

Table 2: Comparison of our methods with the state-of-the-art methods on 7 scenes dataset by measuring the median pose errors and computational time per frame on comparable configurations.

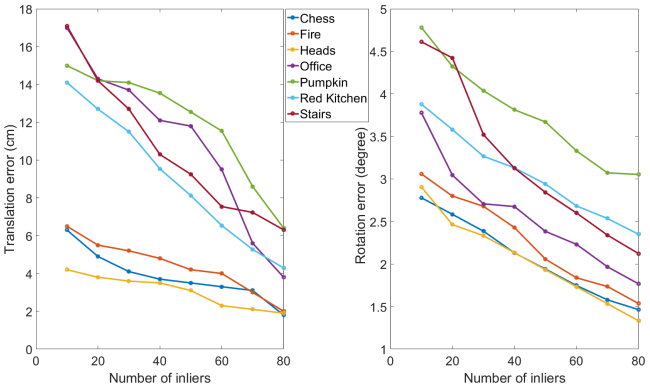


Figure 8: The camera pose error according to number of inliers: For each scene, we calculate the median of translation error (in the left) and rotation error (in the right) on the frames which have at least x inliers corresponding values x on the vertical axis.

We measure the time processing of our experiment. This takes about 60*ms* for each frame with 10*ms* for SURF feature detection, 25*ms* for time prediction of 500 patches on GPU and 25*ms* for 500 iterations of Ransac and PnP. Run-time depends on the number of iterations of Ransac to calculate camera pose. However, we fix the number of iteration at 500 being enough to balance between computational time and accuracy.

Accuracy

In Table 2, we compare our results on 7 scenes dataset to the baseline methods. To compare with PoseNet 2 [16], our method clearly outperforms all scenes in both translation and rotation error. Although PoseNet 2 [16] is very fast to relocalize camera with 5*ms* in the testing, its results are still moderately accurate. Our method is 2 times as accurate as theirs. Regarding the testing time, our method still performs in real-time, even if it is more time-consuming than PoseNet 2.

Our method is slightly higher than other methods on the *kitchen* scene. For the other scenes, our method is not as good as either Active Search [23] or DSAC [2]. However, our method is able to relocalize camera for each frame in 60*ms*. Whereas, the sparse feature based method [23] takes over 100*ms* per frame and the computational time scales with scene size. DSAC [2] requires more than a second for each frame, making difficult to use them for augmented reality systems requiring real-time processing. We obtain a worse result on *office* and *stairs* scenes. As our evaluation of *xyzNet*'s accuracy, Ransac can eliminate ambiguities on repetitive scenes such as *office, red kitchen* and *stairs*. Unfortunately, too many predictions are considered as outliers on the *office, stairs* scene, and resulting in too few 2D-3D inliers to achieve good results by the PnP.

3.5 Confidence score

The confidence score of camera pose is an important issue in camera relocalization as well as in deep learning regression, which is not provided in the state-of-the-art methods. In our solution, no confidence score is given from *xyzNet*. However, we leverage the number of inliers to quantify the accuracy of our method. The number of inliers is not an absolute confidence score, but the accuracy is correlated to it. Figure 8 shows the increasing accuracy of our method according to the number of inliers. This allows us to determine which frames can be used for augmented reality applications.

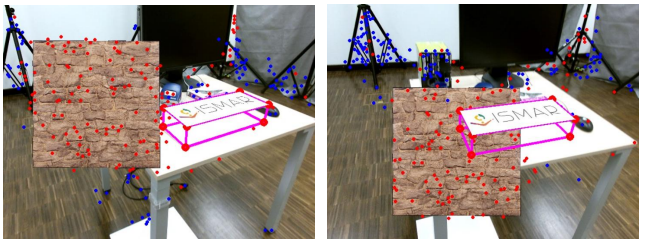
3.6 Performance on texture-less datasets and robustness to occlusion

Method	Translation Error (cm)	Rotation Error (°)
PoseNet	12.0	4.72
Kacete et al.	4.7	2.46
Ours	**3.5**	**0.97**

Table 3: Mean of median poses errors on three scenes of CoRBS dataset.

Scene	Translation Error (cm)		Rotation Error (°)	
	n/o	w/o	n/o	w/o
Human	**6.4**	6.7	**1.18**	1.44
Desk	**1.2**	1.8	**0.73**	1.18
Cabinet	**2.9**	3.5	**0.99**	1.37
Mean	**3.5**	4.0	**0.97**	1.33

Table 4: Robustness to occlusion. Evaluation our method on CoRBS dataset with synthetic occlusion. The accuracy is measured by the mean poses error. n/o: no occlusion, w/o: with occlusion.



b) Frame #005 d) Frame #100

Figure 9: Capacity of processing partial occlusion and application in the augmented reality of our method.

In this section, we evaluate our method on CoRBS dataset. With respect to the scale, the scenes in this data are simpler than those in 7 scenes dataset. CoRBS dataset contains scenes in desktop-scale, each of which focuses on a small environment such as around a desk or a cabinet. However, this dataset is challenging due to the presence of texture-less and many flat surfaces, as illustrated in the Figure 4. We choose three sequences (each scene contains over 2000 images) corresponding to three scenes of *human, desk, electrical cabinet* for our experiment as performed in [13].

We evaluate our method with this dataset in comparison with PoseNet [17] and the method proposed in [13] that uses a random forest to directly predict camera pose from each image patch, final pose being defined then by running mean-shift on all pose predictions. Table 3 shows that our method outperforms [13, 17] both on translation and rotation error.

Finally, we present an interesting advantage of our method that uses patches instead of a whole image. For camera relocalization,

the handling of non-rigid scenes, what mainly occurs in real life, is more difficult. When objects move through a scene, it makes partial occlusion on the scene. We randomly synthesize partial occlusion about 25% surface of each image (Figure 9) on CoRBS dataset. We extract a set of patches to generate a set of predictions about 3D location, in which we take even patches of occlusion.However, RANSAC and PnP algorithms eliminate outliers on the moving object and retains correspondences on rigid objects. Table 4 shows the average of our results for each scene. We still obtain good result with partial occlusion.

4 Conclusion

In this paper, we proposed a novel hybrid method combining deep learning and geometric approach for camera relocalization. We presented our light convolutional neural network to efficiently and robustly define correspondences of 2D pixels in the world coordinates system. A set of probabilistic results is generated to address uncertainty of deep learning regression in camera relocalization. Simultaneously, we exploit geometric information about 2D-3D correspondences to resolve the challenge of partial occlusion and calculate camera pose by using Ransac and PnP algorithms. Our method can perform in real-time. Besides, we also consider the number of inliers as a confidence score for each frame.

Although, we obtain better results on most of the test scenes compared to the state-of-the-art methods that can process in real-time, our method faces difficulties on repetitive scenes where we do not obtain enough inliers by removing almost ambiguity of unimodal prediction from our network. For future work, we wish to improve the prediction by using network of multi-output results.

References

[1] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[2] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother. Dsac - differentiable ransac for camera localization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[3] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold, et al. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3364–3372, 2016.

[4] E. Brachmann and C. Rother. Learning less is more-6d camera localization via 3d surface regression. *arXiv preprint arXiv:1711.10228*, 2017.

[5] R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen. Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[6] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1052–1067, 2007.

[7] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision*, pp. 834–849. Springer, 2014.

[8] J. Gall, A. Yao, N. Razavi, L. Van Gool, and V. Lempitsky. Hough forests for object detection, tracking, and action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 33(11):2188–2202, 2011.

[9] M. Garon and J.-F. Lalonde. Deep 6-dof tracking. *IEEE transactions on visualization and computer graphics*, 23(11):2410–2418, 2017.

[10] B. Glocker, J. Shotton, A. Criminisi, and S. Izadi. Real-time rgb-d camera relocalization via randomized ferns for keyframe encoding. *IEEE transactions on visualization and computer graphics*, 21(5):571–583, 2015.

[11] A. Guzman-Rivera, P. Kohli, B. Glocker, J. Shotton, T. Sharp, A. Fitzgibbon, and S. Izadi. Multi-output learning for camera relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1114–1121, 2014.

[12] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[13] A. Kacete, T. Wentz, and J. Royan. [poster] decision forest for efficient and robust camera relocalization. In *Mixed and Augmented Reality (ISMAR-Adjunct), 2017 IEEE International Symposium on*, pp. 20–24. IEEE, 2017.

[14] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab. Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation. In *European Conference on Computer Vision*, pp. 205–220. Springer, 2016.

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

[16] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[17] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2938–2946, 2015.

[18] 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*, pp. 225–234. IEEE, 2007.

[19] V. Malyavej, P. Torteeka, S. Wongkharn, and T. Wiangtong. Pose estimation of unmanned ground vehicle based on dead-reckoning/gps sensor fusion by unscented kalman filter. In *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2009. ECTI-CON 2009. 6th International Conference on*, vol. 1, pp. 395–398. IEEE, 2009.

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

[21] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pp. 127–136. IEEE, 2011.

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

[23] 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*, 39(9):1744–1756, 2017.

[24] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2930–2937, 2013.

[25] D. J. Tan, N. Navab, and F. Tombari. 6d object pose estimation with depth images: A seamless approach for robotic interaction and augmented reality. *arXiv preprint arXiv:1709.01459*, 2017.

[26] K. Tateno, F. Tombari, I. Laina, and N. Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[27] J. Valentin, M. Nießner, J. Shotton, A. Fitzgibbon, S. Izadi, and P. H. Torr. Exploiting uncertainty in regression forests for accurate camera relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4400–4408, 2015.

[28] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers. Image-based localization using lstms for structured feature correlation. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[29] O. Wasenmüller, M. Meyer, and D. Stricker. Corbs: Comprehensive rgb-d benchmark for slam using kinect v2. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pp. 1–7. IEEE, 2016.