Pamukkale Univ Muh Bilim Derg, 28(2), 277-285, 2022

Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi

Pamukkale University Journal of Engineering Sciences

Direct pose estimation from RGB images using 3D objects

3 Boyutlu nesneleri kullanarak imgelerden poz kestirimi

Muhammed Ali DEDE[1*], *Yakup GENÇ*[1]

[1]Department of Computer Engineering, Faculty of Engineering, Gebze Technical University, Gebze, Turkey.
muhammetalidede@gmail.com, yakup.genc@gtu.edu.tr

Received/Geliş Tarihi: 29.01.2021 Revision/Düzeltme Tarihi: 05.05.2021 doi: 10.5505/pajes.2021.08566
Accepted/Kabul Tarihi: 18.05.2021 Research Article/Araştırma Makalesi

Abstract

We present a real-time monocular camera pose estimation algorithm for augmented reality applications. Proposed model is a small convolutional neural network that is trained to directly estimate 6 Degree of Freedom (6-DOF) camera pose from an RGB image. Our model is designed to run on real-time devices with low memory and computation power. Our model can estimate the camera pose in less than 1ms while keeping accuracy comparable to the state-of-the art. This was made possible by employing geometrically sound loss functions and algebraic constraints. Furthermore, we introduce a new synthetic dataset for demonstrating the proposed methods capabilities.

Keywords: Augmented reality, Pose estimation, Deep learning.

Öz

Artırılmış gerçeklik uygulamalarında kullanılmak üzere tek bir kameradan bir nesnenin yön ve konum kestirimini yapan bir algoritma sunulmaktadır. Küçük bir konvolüsyon ağından oluşan bu model 6 serbestlik dereceli konum ve yön bilgisini tek bir KYM (kırmızı-mavi-yeşil) imgeden elde etmektedir. Önerilen model yüksek başarım ve hafıza içermeyen mobil cihazlar için idealdir. Algoritma verilen bir imgeyi 1ms içinde işlemekte ve güncel algoritmaların performansına yakın performans sergilemektedir. Önerilen geometrik kayıp fonksiyonu ve kullanılan cebirsel kısıtlamaları modelin performansını sağlamaktadır. Aynı zamanda sentetik bir veri kümesi de bu tür modellerin performansını ölçmek için önerilmiştir.

Anahtar kelimeler: Arttırılmış Gerçeklik, Poz kestirimi, Derin öğrenme.

1 Introduction

6D accurate pose estimation is an essential part of the computer vision research since it has important applications in robotics, navigation an augmented reality. This problem is not trivial; many problems like occlusion, illumination and dynamic background can interfere with the estimation. Like many other fields in computer vision, pose estimation also received its fair share of deep learning-based attention. Recent deep learning methods SSD-6D [9] and BB8 [15] is built upon well-known object detector SSD [16][28] and YOLO [17]. These models focus on estimating the 6D pose of an object, depending on success of 2D counterparts on bounding box estimation of given objects. However, estimating 6DOF pose is a much harder problem in RGB images. Variations in appearance, ambiguities and the lack of geometric information and depth [12] may complicate robust estimation. This manuscript addresses recovering camera pose relative to a particular object from a monocular RGB image. This approach is analogous to object pose estimation since our research is focused on augmented reality (AR) perspective. In AR applications, estimating egocentric motion and pose relative to the scene is important as objects in the scene usually stay stationary. Camera pose estimation is an important part of Structure-from-Motion and image-based localization. This problem is traditionally well-studied [19],[20],[22]. Algebraically, given a set of correspondences between an image and its 3D model, the camera pose can be calculated by solving a six-degree polynomial. However, in practice, there are several problems with algebraic approaches. The first problem is finding reliable matching. SIFT [21] and other gradient-based variants are good at finding such informative regions between the images. However, the output of such methods is usually noisy and requires robust methods like RANSAC. The second problem is with these methods is that they try to understand very low level-features (corners and edges) without attending to global structures in the image. To address such problems Deep Learning Based camera pose estimators such as PoseNet [1], are usually applied onto large scale localization. Following these works, new models are used to learn relative ego-motion [12] and compute pair-wise camera pose [25], improve the context of features [26] and Bayesian Neural Networks [27],[29]. Main advantage of such probabilistic learning, they do not suffer from the curse of dimensionality since they can learn to represent many important and complicated features in the image context. Furthermore, given enough diverse samples, they cope with changing environmental factors well. However, deep learning-based approaches poses some drawbacks. First, they require immense amounts of labeled training data. Gathering and annotating such data requires lots of human effort and it is expensive. Moreover, this process is usually error prone and it results in contaminated training data that can impair the performance of the model. Secondly, neural networks require lots of computation power and memory. This is especially a drawback from AR perspective, since contemporary AR devices are mobile phones, which have limited computational and energy resources. Our work tries to address such problems with following contributions.

- We present a new dataset with a class of objects(cars) and a fast and efficient pipeline that can produce realistic synthetic images with varying illumination and natural occlusion, unlike real-time data augmentation in which any image manipulation causes distortions in the scene. We constructed a

*Corresponding author/Yazışılan Yazar

small CNN architecture, which is designed to run on with computationally limited capabilities that runs on Realtime,

- We introduced an algebraic constraint into loss function as a regularizer. Experimental study shows that our constraint reduces training time and slightly increases performance of our proposal on our dataset. Following the literature, we have also tested capabilities of our method on competitive benchmark datasets and achieved comparable results while estimating the camera pose in less than a few milliseconds.

2 Related work

Direct of pose estimation is to train network for estimating 3D translation and 3D rotation of an object of interest in a scene or to infer camera pose relative to a canonical frame. Current pose estimation methods can be split into two categories. Methods in the first category infer the object pose using local features extracted from the target image and matched to the 3D model. Using these 2D-3D correspondences 6DoF pose can be recovered [12],[14],[15],[32],[33]. These methods usually show high robustness against partial occlusion. However, they require robust visual features to operate and fail when the objects are texture-less. Additionally, these features unable to corporate global structure of the scene or the target object to the estimation process. Contemporary pose estimation approaches heavily employ deep learning methods either by directly estimating the pose or use it as part of a multi-stage pipeline [8],[15],[24]. In the seminal work of Xiang [30], PoseCNN uses a CNN architecture to directly regress the 6DoF pose. However, this method heavily suffers when the object is occluded. Obwerger et al [13] show that occluders have a corrupting effect on CNN activations far beyond the receptive field. Another approach is to turn this problem into a classification problem [15] by discretizing the poses. In response to those failures, many researchers adopt a multi-stage pipeline similar to the traditional methods. In the first stage, they predict the2D key-points and then compute the pose with PnP algorithm. These methods train CNNs to detect semantic key-points. Lepetit et al [24] uses segmentation and sparse correspondences for detecting the 3D bounding box. Tekin et al [9] use YOLO [17] architecture to estimate object key-points on a multi-resolution feature map. An alternative to sparse approaches where a few selected key-points are estimated, in a dense-approach every object pixel has a contribution to the pose estimation. Similar to a Hough voting scheme, all pixels cast a vote to a set of hypotheses. The hypothesis with the most votes is selected to be the proposal. Doumanoglou [34] and Kehl [35] use CNN to sample RGB-D image patches and extract features for the voting. Peng et al [15] uses the Farthest Point Sampling algorithm (FPS) to select key-points on the target object. Then they use a heavy-weight pre-trained CNN [36] to predict the key-points and a segmentation map. Furthermore, they notice that the quality of predicted key-points may vary. Thus, they introduced Uncertainty-driven PnP that minimizes the Mahalanobis distance between each predicted spatial key-point distribution and the 3D correspondence. The main advantage of multi-stage and key-point-based networks is the improved accuracy and their resilience to the weak occlusion. Especially with the help of RANSAC these methods, outperform direct estimation methods and single-stage methods. On the other hand, cost of robustness is large neural models and computational time spent on post-refinement and solving PnP-RANSAC. This cost makes modern approaches inapplicable to real-time pose estimation on cheaper industrial devices.

3 Direct estimation of camera or object pose

Given a monocular input image, we aim to estimate the location and orientation of the camera with respect to the single object (see Figure 1). This is the same as estimating the object pose with respect to the camera. Algebraically, the imaging process is captured by Equation (1) where K is a 3×4 matrix capturing the camera internal parameters, [x, y, z] is a point in object coordinate system and [u, v] is corresponding location on the image plane and ρ is the projective depth.

$$\rho \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1}$$

Camera pose and orientation can be represented as a 7dimensional vector θ= [**T, R**] where T is a three-dimensional vector representing the pose or the translation of the camera. And R is a four-dimensional vector representing camera rotation as quaternions. As in [1] we choose quaternions for representing rotations since they lie on a unit hypersphere, any arbitrary four-dimensional vector represents a rotation provided they are normalized. Another advantage of this property is that, we can use this term as a constraint to help to train the network Unlike Euler angles, which require a complex and expensive orthonormalization process and suffer from singularities, quaternion representations are easy to manipulate algebraically thus enabling a simple normalization process that allows us to calculate gradients easily. Direct estimation of camera pose from an image is done by a neural model (see Figure 1). A custom neural model optimized for speed and small memory footprint is designed in order to run the model on devices that have limited computational capabilities. Proposed architecture (see Table 1), down-scales the given image by three consecutive 5x5 convolutions with strides of 2 and applies a normalization. Then, we construct a convolution block that consists of 3 densely connected [4] traditional 3x3 convolutions with a bottleneck 1x1convolutions which is followed by an average pooling and a normalization layer. This is applied three times. The model is continued with two fully connected layers each 1024 neurons. Model output is a 7-dimensional vector in which the first three represent the location and the remaining four represents the rotation.

The model is trained through an optimization on a specially designed loss function. The proposed loss function is an amalgamation of different losses penalizing various estimations. The first part of the loss penalizes the difference between estimated position and the ground truth:

$$L_t = \|t_g - t_e\|_2 \tag{2}$$

This loss treats all three components of the position equally including the distance to the object. Our datasets have comparable values for lateral motion of the camera as well as the distance to the object. The second loss is the simple difference between estimated rotation and actual rotations:

$$L_r = \|r_g - r_e\|_2 \tag{3}$$

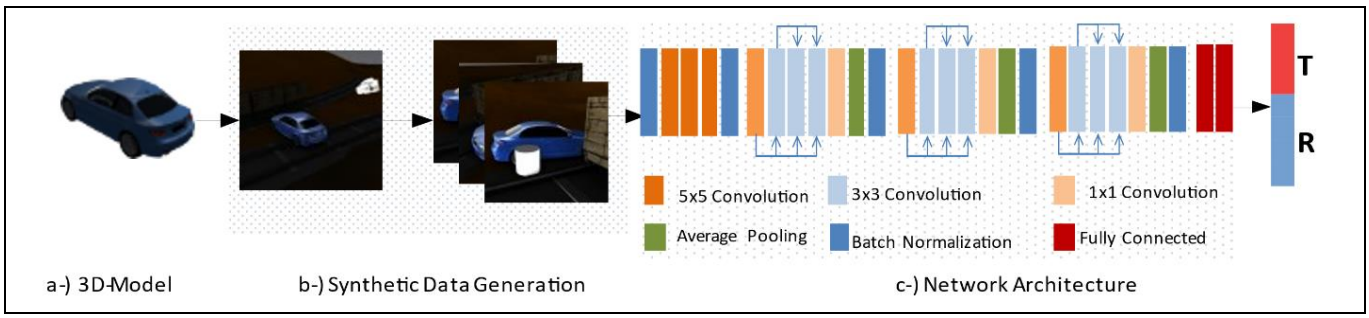


Figure 1. Direct estimation of camera poses with neural models using synthesized images. (a): Given a 3D model we generate random texture maps and distinct shapes and material properties. (b): We render images on dynamic and transparent backgrounds with varying illumination. Random 3D shapes and objects are placed between camera and the target object. (c): We train a custom densely connected custom neural architecture to directly estimate either camera or target object pose.

Table 1. Layer by layer details of the proposed architecture. We highlight the repeating dense block by drawing a border. We use highlighted block two more times before fully connected layers.

Layer No	Type	Kernel Size	Stride	Filter Count
1	Batch Norm	-	-	-
2	2DConvolution	5x5	2x2	48
3	2DConvolution	5x5	2x2	48
4	2DConvolution	5x5	1x1	48
5	Batch Norm	-	-	-
6	2DConvolution	3x3	1x1	24
7	2DConvolution	3x3	1x1	48
8	2DConvolution	3x3	1x1	96
9	2DConvolution	1x1	1x1	48
10	Average Pooling	2x2	2x2	-
11	Fully Connected	Global	-	1024
12	Fully Connected	Global	-	1024

Although not all the ranges of the rotation are covered in our training and test data due to limited access to a real-world object, our loss yields again comparable values for the four components of the rotation in quaternion representation. The final loss is the projection error. Projection error is calculated by the Euclidean distance between the actual image coordinates [u_g,v_g] of a random set of points in the world and their estimated image coordinates[u_e_v_e](as given in Equation 4). During training we choose four random points to calculate a mean of the projection as the final projection loss.

$$L_p = \frac{1}{n}\sum_{i=1}^{n} \left\| \begin{bmatrix} u_g \\ v_g \end{bmatrix} - \begin{bmatrix} u_e \\ v_e \end{bmatrix} \right\|_2 \quad (4)$$

Once again, we do not distinguish between the locations of the points on image or object. Randomly choosing these points gives us enough variety, as their placement on the object as well as their distribution in the image. Fixed set of points may create biases in this loss as they may lie on a singular setup (e.g., along a line)or they may map on the boundary of some images (effecting the range of values as well as being subjected to different camera distortions).We also introduced the unit hyper-sphere constraint of the quaternions as a regularizer on the network. Kendall et al. [6] mentioned about this fact and they decided not to use as part of optimization, believing that such constraints can impair model training. Furthermore, they observed that as the training advances, estimated pose comes very close to the ground truth and renders such additions unnecessary. However, our experiments show that such a constraint can be beneficiary to the overall training process. This will be discussed in detail in Section 4. We note that [6] argues that as the training progresses, the estimated values of the camera rotation come close to the unit sphere of a quaternion. Instead of leaving relaxing the rotation outputs, we supply the unit norm constraint directly with a continuous surrogate.

$$\hat{Q} = e^{(Q-1)^2} \quad (5)$$

Where Q represents the quaternion norm $\hat{Q}$ denotes the surrogate regularization term. We finally combine all four losses given in equations (2),(3),(4) and (5) into a single loss:

$$J = \alpha L_t + \beta L_r + \gamma L_p + \lambda \hat{Q} \quad (6)$$

Where α, β, γ and λ are hyper-parameters which are used to fine-tune and balance the overall training process. This combined loss function is differentiable and can easily be optimized by modern deep learning optimizers. It should be noted that some portions of the loss may seem redundant. For example, the re-projection loss can represent both the translation and the rotation losses together. However, stating these losses separately may allow the optimization process to remove biases due to imperfections in the projection models, noises, and other possible biases. And our experiments show that this improves the estimation accuracy significantly.

4 Datasets used in experimental evaluation

It is well known that deep learning performs well on large datasets. However, producing such datasets are usually expensive or requires hand-annotated labels. We solved this problem by using generated 3D models. Given a 3D model, we have rendered an image with random object colors under

different lighting conditions with random backgrounds. We used 6 different free 3D models from open source communities. For each model, we have generated 600 images for training, 300 images for validation and final 1000 images with transparent backgrounds for testing. Rendered images have size of 640x640 pixels. They are scaled down to 224x224 pixels during training. Rendering camera is randomly placed between two hemispheres with radius of $\mathbf{r_1}$ and $\mathbf{r_2}$ and oriented such that the target object is always visible within the image. For initial experiments these values are assigned 5and 15 respectively. Additionally, we placed random 3D models between camera and the rendered object to simulate occlusion. During dataset generation, camera hemisphere is divided into four quadrants along the x-axis and y-axis. 540 of the rendered training images come from three quadrants and 60 of them comes from the last one. We employed this strategy to test the interpolating capabilities of the neural network. Validation and test images are equally distributed among the quadrants. We have also rendered pure silhouette of the object from the same camera perspective so that we can use IoU (Intersection over Union) score. This score choice is further discussed in the experiments section. We have also demonstrated the performance of our method on Cambridge Landmarks dataset [1]. This dataset (Sample images is presented in the Figure 2) includes 5 large scale urban scenery with challenging conditions like pedestrians, vehicles and confusing environmental conditions like different lightning and weather conditions. For indoor comparison, we used 7Scenes Dataset [2]. This dataset contains 7 in-door scenes with various numbers of RGB-D images for each category. Although this dataset is designed for RGB-D re-localization, we use it to demonstrate the performance our proposed method. With the use of depth data, some of the difficulties can be alleviated. Our method will not make use of the depth data in our experiments. Additionally, this dataset is also challenging for methods using image level features like SIFT, as it contains many ambiguous texture-less areas.



Figure 2. Sample images from cambridge dataset.

5 Experimental evaluation

5.1 Setup

We have pre-trained our model using PlacesNet [5] aiming multi-label scene classification. This dataset contains 1,803,460 training images with 365 categories. Each category has varying number of images from 3,068 to 5,000. PlacesNet dataset focuses on indoor and outdoor scenes. Alternatively, ImageNet is used for pre-training networks in many applications. However, objects in ImageNet does not have the depth variations that we would like to see. The scene categories in PlacesNet has better depth variation which might be useful in our pose estimation problem. Our model was implemented in TensorFlow [6]. We used Adam [7] optimizer with a learning rate $5x10^{-4}$, which is halved for every 40 epochs. Overall training took 170 epochs and we employed early stopping to counter overfitting. Because of the limited GPU resources, batch size is selected to be 8. We also applied random geometric crops onto training images for data augmentation (See Figure1).

5.2 Hyper-parameter selection

In the experiments with our synthetic datasets, we selected α, β, γ, and λ as 3, 10, 20 and 8 respectively. α, β, and γ values are found using a grid search in the range [0,20] with 1 step increments. Finding a healthy λ value is essential since selected loss approaches its minimum very slowly and many different conFigureurations of quaternion may result in a similar regularization loss. In our experiments, we have hand tuned λ with less than 10 different tries. Obviously that the introduction of multiple hyper-parameters complicates the training process. Since the weight of each loss partition will introduce a bias that may drag the network to a point that is numerically favorable from an optimization perspective while the estimator showing relatively poor performance. On the other hand, Scene geometry and the priorities of the estimator can vary depending on the problem. In some cases, we could estimate the camera position in large-scale outdoor scene while in others, we could be estimating the precise pose of a hand-held object for augmented reality. Additionally, there could be large numerical variations between different parts of the loss which can also hinder the training process. Such partitioning allows us to adjust the contribution of each source and stabilizes the training process. This problem is visited in a follow up article by Kendall et al. [6] and they used a geometric loss like our re-projection error again based on the fundamentals of multi-view computer vision geometry. They managed to improve their original performance [1] while simplifying the training process. However, in our experiments with synthetic dataset, we observed that, merging different losses with coefficients have a slight improvement on validation results.

5.3 Training pipeline

During training, for each image batch we have selected random images from MS-COCO [3] dataset as back-ground for images. Each synthetic image is overlayed on top of the chosen image. Thus, neural model almost sees a unique background for every rendered image. We have also employed a different occlusion strategy in training batches. We have applied random shaped crops in 20 percent of the images. While crop size depends on the shape that is assigned, for rectangular shapes, we randomly choose width and height between 10 and 40 pixels. For spherical crops, radius was again between 10 and 40 pixels. The cropped area is either filled with random uniform colors or we have cropped the area with same coordinates from the background and applied on top of the cropped region.

5.4 Experiments and results

During inference we normalize the estimated rotation by its length so that quaternion constraint is applied using $Q/\|Q\|$. We conduct experiments and make comparisons with the normalized estimations. We show that proposed model can estimate camera pose effectively using only a fraction of the number of parameters used by PoseNet. In fact, the number of parameters is reduced from 22M to 2.4M as shown in Table 1. Following [8], we calculate median translation and rotation errors for all datasets and categories in Table 3. Additionally, we used IoU (see Figure 3 for an archetypal representation) score for demonstrating the accuracy of tested models.

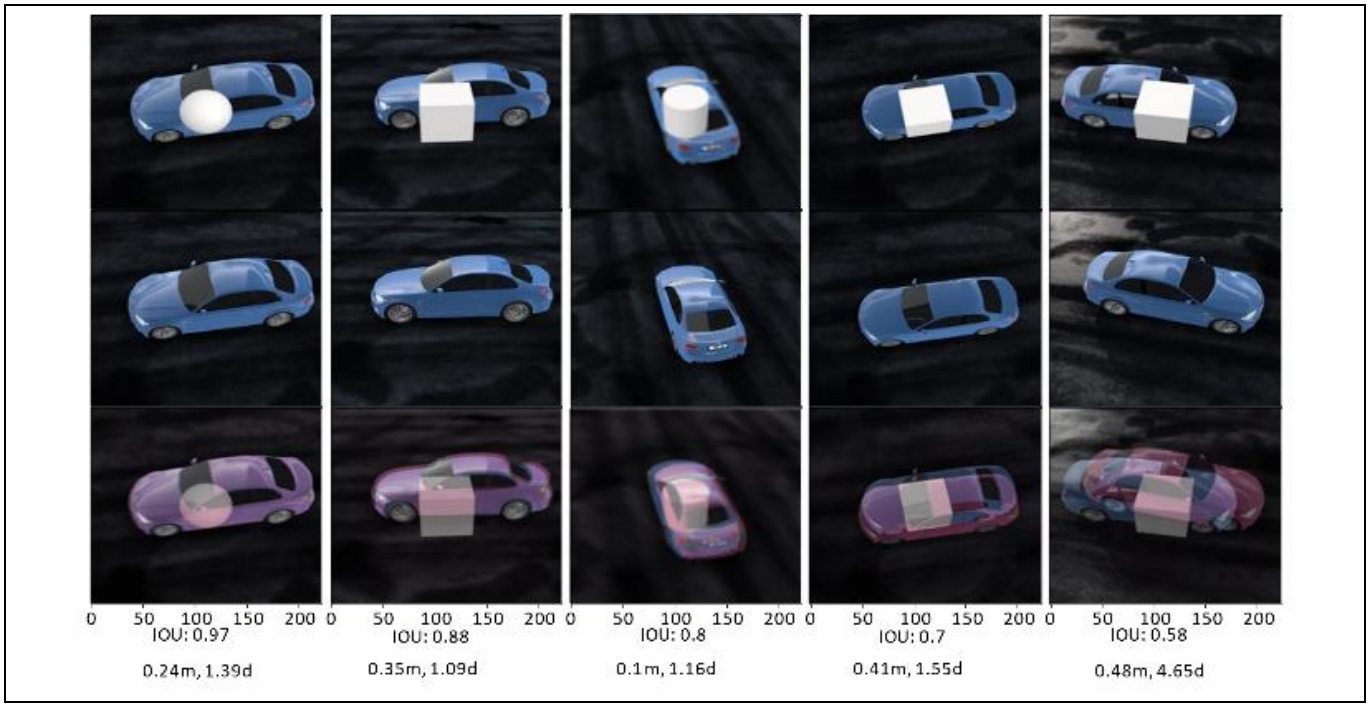


Figure 3. Archetypal representations for each IOU threshold. Top row represents the ground truth. Middle row is the view of the estimated camera location and orientation. Bottom row is the composite image of estimation and ground truth. Axis labels shows the IoU and the translation and the rotation error.

There are multiple ways of measuring estimator's performance. Although using such a score is non-traditional, we propose using a simple intersection over a union of silhouette images can give both quantitative and qualitative nature of the comparison. It can be argued that our method has an inherent weakness, that different pose estimation can result in similar IoU scores. However, during our evaluation of predictions, we observed that even if the prediction is very close to the truth value, IoU may vary noticeably, since our objects have complex surfaces and cover a large area in the image. To calculate IoU score we rendered silhouette images of from both prediction and original pose. Table 2 also shows that how well model performed on the task and the number of worst-case scenarios across different models.

A significant number is the estimated angles are much closer to the ground truth labels. We believe this result is due to the application of a quaternion constraint in Equation (4). We have also observed that application of such constraint on both models improves the average IoU by decreasing the number of worst-case predictions despite it has a negligible detrimental effect on high accuracy predictions. Following the tradition, we have also tested our model's performance on Cambridge Landmark dataset and 7 Scenes dataset. In Table 3, we demonstrate that our model performs comparably to the state-of-the-art camera re-localization using other deep neural networks while having one tenth of trainable parameters. We have also tested our model's generalization capabilities when the training is done with views obtained from a close by camera and test the performance on images obtained further away from the object.

For this, we have generated 1000 test images rather distant from the target object. Camera radii $\mathbf{r}_1$ and $\mathbf{r}_2$ are selected as 15 and 30 meters for this test dataset. Training images were obtained within 15 meters. Table 4 shows that our model performs well up to 20-meter radius and breaks down after that. This is expected as object details become indiscernible after certain distances. Another robustness test is conducted to calculate orientation variations. In the training data we let only portion of orientations present for each car type. We fix the tilt angle w.r.t. the object while varying the pan angle freely. We make sure that the entire range of orientation is covered by the two different cars. When the first car covers the angle range 0°–45°, the second car covers the next range 30°-60° and soon.

The trained model is then tested on a car that was imaged for the remaining range of pan angles (not used in the training data for that car). As the results show in Table 5, the model successfully learns the representation of the car for non-existing poses. This suggests that pose transfer from one car to the other is accomplished by the model. Of course, we cannot conclude that the pose of a completely new can be estimated by the model as good as the existing cars that is used in test. Figure 4 shows the effect of the choice of regularization term on the validation performance during training. As discussed earlier, we expect $\sim Q$ to behave better than the direct loss or the quadratic regularization$(Q-1)^2$. Even though exponential regularization starts slowly, it picks up and gives slightly better validation error. The positive effect of this regularization on other models can also be seen in Table 1. When we apply this regularization on PoseNet, although higher accuracy levels did not change, for the lower accuracy levels, performance has seen a dramatic increase (columns under <0.2 and <0.1). We also used this regularizer on another model. When applied in training PoseNet, the number of iterations to get to the same performance are decreased by about 50% (see Figure 5). Our model shows good experimental performance on synthetic as well as real data.

Table 2. Comparison of intersection over union (IoU) for number of correct model predictions for various levels of accuracy is shown. Abbreviations Const., Pro. and Geo. correspond to constrained, re-projection loss and geometric loss respectively.

Model	Param Size	Median IOU	Accuracy				
			>0.9	>0.8	>0.7	<0.2	<0.1
PoseNet [1]	22M	0.547	47	269	426	202	157
PoseNet [1] (Constrained)	22M	0.675	53	297	586	57	44
PoseNet [6] (Const.,Geo.)	22M	0.857	363	792	908	35	28
Our Model (Const.)	2.4M	0.739	139	508	722	36	30
Our Model (Const., Pro.)	2.4M	0.845	345	790	912	15	12

Table 3. Performance of the proposed model in comparison to the state-of-the art. Displayed numbers show median translation (in meters) and rotation (in degrees) estimation errors of the study in that scene.

Scene	Bayesian PoseNet	LSTM PoseNet	PoseNet Reprojection	Our Model
College	1.74 m,4.06°	0.99 m,3.65°	0.88 m,1.04°	0.93 m, 2.64°
Hospital	2.67 m,5.14°	1.41m,4.20°	3.20 m,3.29°	2.18 m, 3.67°
Church	2.11 m,8.38°	1.18 m,7.44°	1.57 m,3.32	2.11 m, 6.8°
Shop Facade	1.25 m,7.54°	1.52 m,6.84°	0.88 m,3.78°	1.16 m,5.43
Chess	0.37 m, 7.24°	0.24 m,5.77°	0.14 m,4.48°	0.26 m, 4.24°
Fire	0.43 m, 13.7°	0.34 m,11.9°	0.27 m,11.3°	0.43 m, 13.11
Heads	0.31 m, 12.0°	0.21 m,13.7°	0.17 m,13.0°	0.21 m, 13.24°
Office	0.48 m, 8.04°	0.33 m,8.08°	0.19 m,5.55°	0.21 m, 5.98°
Pumpkin	0.61 m, 7.08°	0.37 m,7.00°	0.26 m,4.75°	0.53 m, 5.11°
Kitchen	0.58 m, 7.54°	0.58 m,7.54°	0.24 m,5.52°	0.24 m, 5.35°
Stairs	0.48 m, 13.1°	0.48 m,13.1°	0.37 m,12.4°	0.38 m, 12.1°
Synthetic Cars(ours)	Failed	Failed	0.18,1.19°	0.21 m, 1.07°

Table 4. Model performance when tested on images with larger ranges of motion.

Motion Range	IoU Score	Translation Error (in meters)	Rotation Error (in degrees)
5-15m	0.845	0.21m	1.07°
15-20m	0.842	0.22m	1.07°
20-25m	0.46	0.54m	4.64°
25-30m	Failed	Failed	Failed

Table 5. The model performance when tested on images with diverse orientation ranges.

Motion Range	IoU Score	Translation Error (in meters)	Rotation Error (in degrees)
5-15m	0.845	0.21 m	1.07°
Sparse data	0.812	0.45 m	1.64°

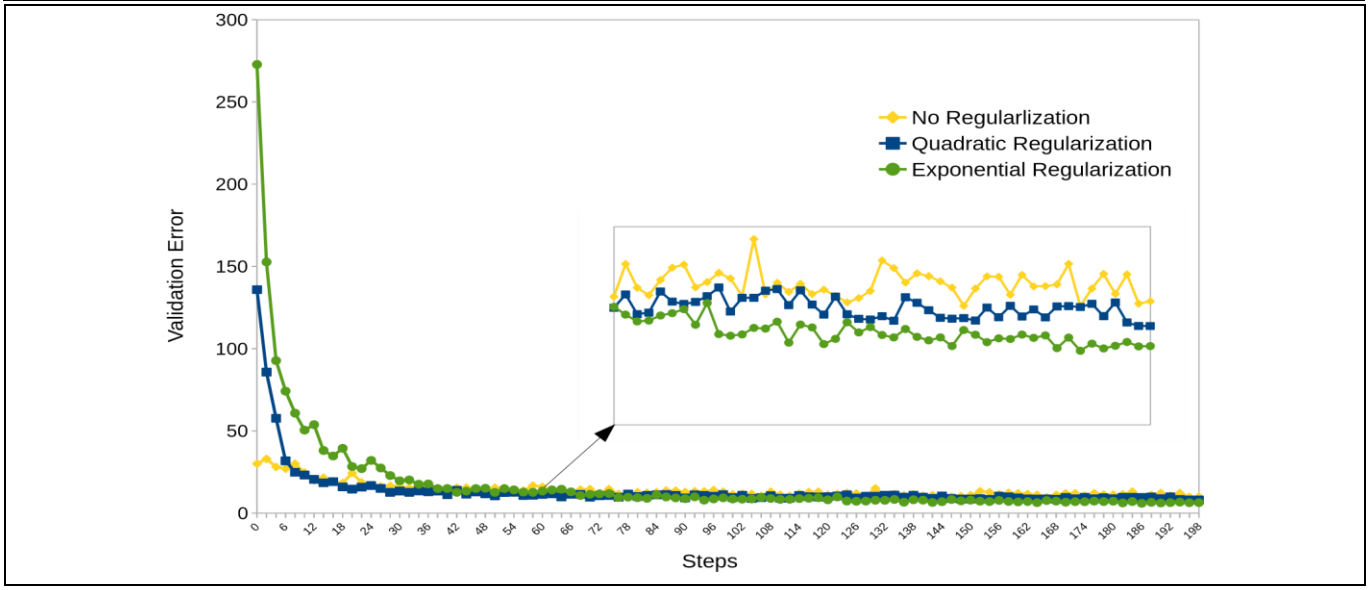


Figure 4. Effect of quaternion norm regularization on the validation loss. In the beginning of the training cumulative loss is significantly higher compared to the vanilla loss due to the chaotic estimations of the network. However, as the networks becomes more stable, constraint helps, optimization to find a slightly better minimum and faster convergence.

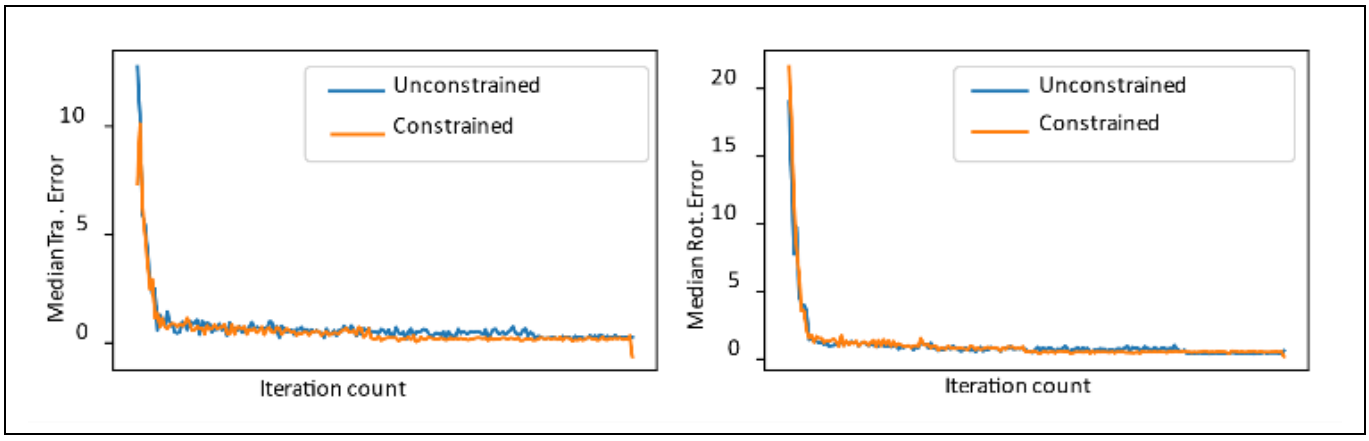


Figure 5. Median validation errors of PoseNet. Constrained quaternion curve shows the effect of exponential regularizer. Both translation and rotation error approaches local minima at faster rate, typically about 50%.

5.5 Ablation studies on LINEMOD dataset.

In this section, we provide a detailed analysis of our approach to the LINEMOD [22] dataset. LINEMOD dataset is one of the standard pose estimation benchmark datasets used in academic and industrial applications. The dataset contains 13 object classes with around 1200 images per class. This dataset is highly challenging due to background clutter, variations in illumination furthermore texture-less objects. Following the literature, we take 200 images from the original dataset for training, and the remaining are used in testing. Additionally, we synthetically generate 3000 images per class for training in which poses are sampled from a 3-meter radius Gaussian sphere. We adjust the hyper-parameters to 1, 1, 6, and 8 to accommodate the different scene compositions. We compare the proposed architecture in 4 different setups. The first setup is using a combination translation and rotation errors as the sole loss function. The second setup uses vanilla projection to estimate the object pose. The third setup incorporates reprojection error and translation and rotation errors. Finally, in the last setup, we introduce the constraint term on the rotation. In Table 6, we present our findings in the 2D Projection metric. This metric measures how close the 2D projected vertices are to the ground-truth, in pixel domain. Following the common practice in the recent papers, we consider the pose as correct if the mean 2D projection error is below 5 pixels. Our study suggests that neither reprojection error nor the translation and rotation error perform significantly better than each other on average. However, when we merge the losses as discussed, we observe a significant performance increase. We believe that this phenomenon is due to different priorities regarding the nature of the losses. We suspect that while the projection errors attend to the 2D projection of the scene and translation and rotation infers some implicit information from the latent 3D geometry.

Similar to the original experiments, the proposed constraint on the quaternion seems to alter regression performance quite a little. On the other hand, we observe the impact of the constrain on the training process. While the unconstrained network seems to settle around the 110th epoch. Constrained models settle around the 70th epochs.

6 Discussion

In experiments sections, we presented the effectiveness of our engineered model with a set of combined losses and constraints. Moreover, additional experiments on a different dataset clearly show the contribution of careful design choices, training regimes, and regularizing constraints. On the other hand, we see the shortcomings of the proposed method in the ablation studies. The parameter size of the network, one of the strongest aspects of our architecture that provides almost real-time pose estimation, does not have the expressive power to estimate the pose of the objects across categories. For each object class, we train the network from scratch. In LINEMOD experiments, we observe that the performance of the neural network is not competitive. Specifically, multi-stage pose estimation techniques perform much better due to their large number of parameters and combining traditional Computer Vision algorithms with neural networks. On the other hand, the proposed method is a fully convolutional architecture consequently we scale the parameter size effortlessly and observe that our network benefits from cardinality on the LINEMOD dataset. Another shortcoming of our approach is to adjust hyper-parameters again for the dataset to adjust the new scene composition. It may be burdensome and expensive for some applications to perform a hyper-parameter search for each application. Our final observation is the necessity of a quaternion constraint on larger models. While we see the significance of the constraint in earlier experiments with small-scale models, we observe that as the model gets larger, the accuracy contribution of the quaternion regularizer to the diminishes in contrast to our expectations. However, training duration still seems benefitting from the additional error signal since we re-observe the faster convergence in the latter experiments. This behavior needs further investigation across multiple datasets.

7 Conclusion

In this manuscript, we have presented a method for direct estimation of the camera pose from a given image of known objects. Starting with a CAD model of the object for which the pose to be estimated, we first render synthetic images of the object under varying pose, occlusion, and illumination conditions. Augmenting these synthetic images with natural background yields a training data. This data is then used to train a carefully designed neural network. Proposed model estimates the camera pose in our data better than any other direct method that we know. The same model performs comparable to the other methods on benchmark datasets such as Cambridge and 7 Scenes.

Table 6 .We show the effects of different loss terms on the regression process. L1 shows pure translation and rotation loss L2 loss shows the projection loss. Given numbers denote the accuracy in 2D projection metric. We consider an estimated pose correct if the distance between the ground truth pixels and the projected pixels lie between 5 pixels.

Object	L1	L2	L1 + L2	L1+L2+ Constraint
ape	22.9	22.7	31.3	32.2
benchwise	32.1	33.0	48.1	48.7
cam	38.4	39.0	36.3	39.4
can	40.2	39.8	66.2	66.3
cat	41.6	41.2	42.7	43.2
driller	34.8	34.8	63.5	64.1
duck	30.2	30.6	32.4	30.8
eggbox	36.7	36.9	61.4	62.0
glue	22.4	23.4	63.2	56.7
holepuncher	36.6	37.5	42.6	44.3
iron	58.4	61.3	74.0	74.9
lamp	41.0	44.1	68.4	68.5
phone	34.7	36.2	43.6	47.3

Proposed model is fine-tuned for this purpose with a specific architecture which is much smaller than many reported in the literature. We also introduced the random projective loss function that helps the performance of the estimator, especially in the orientation estimation up to 1 degree. The exponential regularizer we have used also helps in the performance of both our model as PoseNet [1]. Combined, these three improvements in our model leads to a fast and accurate direct pose estimation method. We are in the process of extending this method to include a refinement step [10],[23]. Our refinement process however will be incorporated into our existing model by making use of not only the 2D image structures as well as the silhouette of the object. Such method has the potential to outperform feature-based pose estimation methods, as they can have difficulty around occluding contours.

8 Author contribution statements

In the scope of this study, the Muhammed Ali DEDE in the synthesizing the dataset, design of the architecture, experiments and the writing of the manuscript; Yakup GENÇ in the literature review, assessment of obtained results, optimization strategies, examining the results, spelling and checking the article in terms of content were contributed.

9 Ethics committee approval and conflict of interest statement

There is no need to obtain permission from the ethics committee for the article prepared.

10 References

[1] Kendall A, Grimes M, Cipolla R. "PoseNet: A convolutional network for real-time 6-DOF camera relocalization". *In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV),* Las Condes, Chile, 11-18 December 2015.

[2] Shotton J, Glocker B, Zach C, Izadi S, Criminisi A, Fitzgibbon A. "Scene coordinate regression forests for camera relocalization in RGB-D images". *In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, Portland, USA, 23-28 June 2013.

[3] Lin C. "Microsoft COCO: Common objects in context". *In the European Conference on Computer Vision (ECCV),* Zurich, Switzerland, 14-21 August 2014.

[4] Gao H, Zhuang L, Kilian QW. "Densely connected convolutional networks". *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, 27-30 June 2016.

[5] Zhou B, Lapedriza A, Khosla A, Oliva A, Torralba T. "Places: A 10 million image database for scene recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6), 1452-1464, 2018.

[6] Martin A, Agarwal A, Barham A. "TensorFlow: Large-Scale machine learning on heterogeneous systems". *arXiv*, 2016. https://www.tensorflow.org/.

[7] Diederik P, Ba J. "Adam: A method for stochastic optimization" *3rd International Conference on Learning Representations, {ICLR} 2015,* San Diego, USA, 7-9 May 2015.

[8] Kehl W, Manhardt F, Tombari F, Ilic S, Navab N. "SSD-6D: making RGB-Based 3D detection and 6D pose estimation great again". *IEEE International Conference on Computer Vision* (ICCV), Venice, Italy, 22-29 October 2017.

[9] Tekin B, Sinha S, Fua P. "Real-Time seamless single shot 6D object pose prediction". In *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, USA, 18-22 June 2018.

[10] Wang H, Sridhar S, Huang J, Valentin J, Song S, Guibas L. "Normalized object coordinate space for category-level 6D object pose and size Estimation". *In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, USA, 16-20 June 2019.

[11] Sock J, In Kim K, Sahin C, Kyun Kim T." Multi-Task deep networks for depth-based 6D object pose and joint registration in crowd scenarios" *British Machine Vision Conference (BMVC),* New Castle, UK, 3-6 November 2018.

[12] Sundermeyer M, Marton ZC, Durner M, Brucker M, Triebel, R. "Implicit 3D orientation learning for 6D object detection from RGB images". *In the European Conference on Computer Vision (ECCV)*, Munich, Germany, 8-14 September 2018.

[13] Oberweger M, Rad M, Lepetit V. "Making deep heatmaps robust to partial occlusions for 3D object pose estimation". *In the European Conference on Computer Vision (ECCV),* Munich, Germany, 8-14 September 2018.

[14] Rad M, Lepetit V. "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using Depth". *In the IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 22-29 October 2017.

[15] Peng S, Liu, Y, Huang Q, Zhou X, Bao H. "PVNet: Pixel-Wise voting network for 6DoF pose estimation". *In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, USA, 16-20 June 2019.

[16] Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC. "SSD: Single shot multibox detector*". In the European Conference on Computer Vision (ECCV)*, Amsterdam, Netherlands, 8-16 October 2016.

[17] Redmon J, Divvala S, Girshick R, Farhadi A." You only look once: unified, real-time object detection". *In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* Las Vegas, USA, 27-30 June 2016.

[18] Walch F, Hazirbas C, Leal-Taixe L, Sattler T, Hilsenbeck S, Cremers D. "Image-Based localization Using LSTMs for structured feature correlation". *In the IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 22-29 October 2017.

[19] Durrant-Whyte H, Bailey T. "Simultaneous localization and mapping: part I". *IEEE Robotics Automation Magazine,* 13(2), 99-110, 2006.

[20] Lowe D. "Distinctive image features from Scale-Invariant key points". *International Journal of Computer Vision,* 60(2), 91–110, 2004.

[21] Forsyth D, Ponce J. *Computer Vision: A Modern Approach*. 1st ed. New York, USA, Pitman, 2002.

[22] Hinterstoisser S, Lepetit V, Ilic S, Holzer G. B, et al. "Model based training,detection and pose estimation of texture-less 3D objectsin heavily cluttered scenes". *11th Asian Conference on Computer Vision*, Daejeon, Korea, 5-9 November 2012.

[23] Zakharov S, Shugurov I, Ilic S. "DPOD: 6D pose object detector and refiner". *In the IEEE International Conference on Computer Vision (ICCV)*, Seoul, Korea, 27 October, 2 November 2019.

[24] Laskar Z, Melekhov I, Kalia S, Kannala J. "Camera relocalization by computing pairwise relative poses using convolutional neural network". In t*he IEEE International Conference on Computer Vision (ICCV) Workshops,* Venice, Italy, 22-29 October 2017.

[25] Melekhov E. "Relative camera pose estimation using convolutional neural networks". *In Advanced Concepts for Intelligent Vision Systems,* Springer International Publishing, 2017.

[26] Simonyan K, Vedaldi A, Zisserman A. "Deep inside convolutional networks: Visualising image classification models and saliency Maps*". International Conferance on Learning Representations*, Bandiff, Canada, 14-16 April 2014.

[27] Kendall A, Cipolla R. "Geometric loss functions for camera pose regression with deep learning*". In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, Hawaii, 21-26 July 2017.

[28] Liu W. et al. SSD: "Single Shot MultiBox Detector". *In the European Conference on Computer Vision (ECCV),* Amsterdam, Netherlands, 8-16 October 2016.

[29] Kendall A, Cipolla R. "Modelling uncertainty in deep learning for camera relocalization". *In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweeden, 16-21 May 2016.

[30] Xiang Y, Schmidt T, Narayanan V, Fox D. "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scene". *Robotics: Science and Systems*, Pittsburgh, USA, 26-30 June 2018

[31] Hinterstoisser S. et al., "Gradient response maps for real-Time detection of textureless objects". *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(5), 876-888, 2012.

[32] Martinez M, Collet A, and Srinivasa S S. "MOPED: A scalable and low latency object recognition and pose estimation system". *2010 IEEE International Conference on Robotics and Automation*, Anchorahe, USA, 4-8 May 2010.

[33] Wang C, Xu D, Zhu Y, Martin-Martin R, Lu C, et Al . "DenseFusion: 6D object pose estimation by iterative dense fusion". *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Salt Lake City, USA, 16-20 June 2019.

[34] Doumanoglou A, Kouskouridas R, Malassiotis S, Kim T. "Recovering 6d object pose and predicting next-best-view in the crowd". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, 27-30 June 2016.

[35] Kehl W, Milletari F, Tombari F, Ilic S, and Navab N. "Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation*". In the European Conference on Computer Vision (ECCV)*, Amsterdam, Netherlands, 8-16 October 2016.

[36] He K, Zhang X, Ren S, Sun J, "Deep residual learning for image recognition". *2016 IEEE Conferenceon Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, 27-30 June 2016.