6-DOF Monocular Relocalization with Convolutional Neural Networks

Avi Singh and Aayush Mudgal

Abstract—This project deals with the task of 6-DOF re-localization in large outdoor environments using a single monocular image. It is a modified re-implementation of PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization [8]. We used a labeled data that was generated using videos recorded around prominent landmarks around Cambridge. The labels were generated using computationally heavy Structure From Motion software [21], which served as supervision for training the CNN. We employ transfer learning to regress camera poses on a modified version of VGG network [15]. Our modified CNN network obtained a mean translation error of 4.44m and a mean rotational error of 6.06° on the King’s College sequence from the dataset.

I. INTRODUCTION

Localization is a widely studied problem in mobile robotics. It is the process of determining the \((x,y,z)\) and \((\alpha, \beta, \gamma)\) of a robot with respect to a fixed coordinate frame, and a map is often known in advance. A host of sensors such as LIDAR, Ultrasound sensors, and cameras can be used to solve this localization problem. In our project, we focus on the problem of localizing from a single monocular camera, and no other sensor. Such an estimation of camera location and its orientation can be quite helpful in indoor environments which are often GPS-denied, and would also provide 6-DOF information which most GPS sensors do not provide even in outdoor conditions. Traditionally, geometry-based approaches (Structure from Motion, SLAM) are used for solving these kinds of problems, but we make use of convolutional neural networks, and pose the localization problem as a supervised learning problem.

We provide a modified re-implementation of a Convolutional Neural Network based re-localization system recently proposed by [8]. It proposes a novel, real-time approach to solve the kidnapped robot problem

II. RELATED WORK

The Convolutional Neural Network approach by [8] attempts to combine the strength of the existing localization approaches. Metric and appearance-based approaches have been the two most common approaches to localization. Metric SLAM focuses on creating a sparse [9], [7] or dense [12], [4] map of the environment. It is dependent on the initial pose estimate for estimating the camera’s pose. Appearance-based localization classifies the scene among a limited number of discrete locations to provide a coarse estimate. [16] have used Convolutional Neural Networks to classify a scene into one of several location variables. The approach by [8] tries to combine the strengths of these methods by producing a continuous pose without the need of an initial pose estimate. The training of the neural network is much more efficient compared to the methods that require the need of map building. SIFT-like point based methods as used by [20], [11] require a large database of features and efficient retrieval methods. The Scene Coordinate Regression Forests for re-localization proposed by [14] creates scene coordinate labels using depth images by mapping pixel from camera coordinates to global scene coordinates. [8] work is closely related to the work of [14].

Neural networks have shown to advance the state of art in object detection [17] and human pose regression [19]. However such techniques limit the regression targets to necessarily lie in the 2-D image plane. Contrasting to this, [8], regresses the full 6-DOF camera pose including depth and out-of-plane rotation, and utilizes [6] to generate a dense visualization of re-localization results. Convolutional networks trained on classification problems have shown to generalize well on other problems [1], [13]. [8] attempts to apply these representations of classification to 6-DOF regression tasks.

III. PRELIMINARIES

Some of the building blocks of the model are briefly described below

- **Convolutional Neural Networks** Most modern visual recognition systems use Convolutional Neural Networks, and they are well known to outperform other classifiers on large-scale visual recognition tasks[10]. CNN architectures have two main components - convolutional layers and pooling layers, which often appear in an alternating fashion in the network. This combination is repeated several times over, and fully connected layers are added towards the end, with Dropout layers added in the mix for regularization.

- **Convolution Operation** Convolutional operation forms the basic building block of CNNs. Convolution operation is used to obtain a feature mapping of image by repeated application of a function across the sub-regions of the entire image.

IV. Labeled Dataset

A labeled data was generated using videos recorded around prominent landmarks around Cambridge. [8] were kind enough to release their outdoor urban localization dataset Cambridge Landmarks\(^2\) and it contains five se-

\(^1\)The kidnapped robot problem commonly refers to a situation where an autonomous robot in operation is carried to an arbitrary location [5], [3], and needs to know its new location

\(^2\)The data set is available for download at: http://mi.eng.cam.ac.uk/projects/relocalisation/
quences for different scenes, with a total of more than 6000 frames. Every frame is annotated with 6-DOF pose of the camera, providing data to train and test pose for regression algorithms. The labels were generated using Structure From Motion software [21], which served as supervision for training the CNN, reducing the manual human labor to just the video recording of each scene.

V. METHODOLOGY
The original paper [8] used a 23-layered modified version of the GoogleNet [18] architecture. Our methodology closely followed that of [8]. We trained a modified version of the VGG architecture [15] that takes as input a monocular image $I$, and regresses the pose vector $p$. The pose vector $p$ itself consists of a 3D co-ordinates $x$ and quaternion $q$:

$$p = [x, q]$$

The pose $p$ is measured with respect to an arbitrary global frame that we can define.

![Original GoogleNet Architecture](image1)

**Fig. 1. Original GoogleNet Architecture**

A. Modified VGG Architecture
Closely following [8], we modified the VGG Architecture for our regression task in the following ways:

- Replaced the softmax layers with affine regression layers. Each of the regression layers have a 7D output, three of which will correspond to the $x$, $y$, $z$, and four will correspond to the quaternions
- Inserted a fully connected layer of size 2048 before the final regression layer
- A normalization layer for the quaternions was added.

3 A quaternion is a 4-dimensional representation of euler angles $\alpha, \beta, \gamma$, and it better suited for regression than euler angles.

B. The Loss Function
Training any neural network requires a loss function, and the loss function that we used is as follows:

$$loss(I) = \|\hat{x} - x\|_2 + \beta \left\| \hat{q} - \frac{q}{|q|} \right\|_2$$

(1)

Where $\beta$ is a scale factor that balances the expected rotational and translational error. The loss function as given by Equation (1), ensures that the Euclidean distance between the ground truth and the prediction is minimized for both translation and rotation. This enables us to learn both the translation and rotation simultaneously in the same training process. Though the quaternions should ideally live on the unit sphere, but to avoid complex optimization constraints, the spherical restriction is neglected. [8] report that during training $q$ becomes closer to $\hat{q}$, somewhat validating the constraint relaxation.

VI. TRAINING DETAILS
To regress the camera pose we train the modified VGG architecture on the loss function as defined in Equation (1) using Stochastic Gradient Descent with a learning rate of $10^{-5}$, and Nesterov Momentum. The model was implemented using the Keras library [2]. Training was done for 400 epochs. We trained with about 1200 images, which took about 48 seconds per epoch on an NVidia Titan X GPU with a batch size of 64.

VII. RESULTS
We trained and tested our model on the King’s College Scene, the longer sequences of the Cambridge Landmarks Dataset. The sequence comprises of 1220 training frames and 343 testing frames, with a spatial extent of 140mx40m. Though our model could not outperform the PoseNet model [8], it was still far better than just a random guesser, which means that the model was able to learn meaningfully from the data. Our model reported a translation error of 4.44m and a rotational error of 6.06° on the test set, as opposed to 1.92m and 2.70° which were reported by the PoseNet model [8], for this sequence.

<table>
<thead>
<tr>
<th>Scene</th>
<th>#Frames Train</th>
<th>#Frames Test</th>
<th>Spatial Extent (m)</th>
<th>PoseNet</th>
<th>Our Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>King's College</td>
<td>1220</td>
<td>343</td>
<td>140x40m</td>
<td>1.92m</td>
<td>4.44m</td>
</tr>
</tbody>
</table>

VIII. CONCLUSION AND FUTURE WORK
We presented a re-implementation of the system developed in [8]. Our system is capable of relocalizing in large outdoor environments from just a single image, and is able do that in...
a real time fashion. We evaluated the accuracy of our system on a sequence from the Cambridge landmarks dataset, and they have been presented above.

Possible future directions for the work could include testing a wider variety of neural network architectures, varying hyper parameters and training settings.

ACKNOWLEDGMENT

We would like to thank Prof. Gaurav Pandey for his encouragement, guidance, and teaching. We also would like to thank Prof. Vinay P. Namboodiri for providing us with state-of-art resources. Without their support, this project would not have been possible.

REFERENCES