PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization

Alex Kendall, Matthew Grimes, and Roberto Cipolla - [ICCV 2015]

Presented by:
Kent Sommer
Outline:

- Motivation / Related work
- Problem Statement / Overview of approach
- Dataset
- Details and issues with approach
- Results
- Conclusion / Quiz
Review and Related Work
Review:

- Two approaches to localization
  - Metric
    - Estimate continuous position
  - Appearance/Topological
    - Classify scene to limited number of discrete locations
What does this have to do with search?

- Appearance/Topological localization can be presented as a search problem!
  - Database of known locations, given an input image, where are we?
    - Efficient retrieval is necessary, usually really large database
Related Work:

- Scene Coordinate Regression Forests
  - Use depth images to map each pixel from camera to global
  - Train a regression forest to regress these labels given an RGB-D image.
  - Limited to indoor use in practice (IR interference)
Related Work:

- Feature extraction and matching as in [1, 2, 3, 4]
  - (Generally) extract various types of image features
  - Match these features with those in the database with tagged known location to return position


Problem Statement and Overview of Approach
Problem Statement:

- Estimate the 3D position and orientation of the camera, given a single monocular image taken from a large previously explored area.

Colors:
- **Green** ○ Training
- **Blue** ○ Testing
- **Red** ○ System output
Overview of Approach:

- Perform end-to-end supervised learning with euclidean loss to regress 6-DOF pose.
  - Does not require large landmark database (instead it learns robust high level features to regress 6-DOF pose.)
Dataset
Figure 4: **Map of dataset** showing training frames (green), testing frames (blue) and their predicted camera pose (red). The testing sequences are distinct trajectories from the training sequences and each scene covers a very large spatial extent.

Figure 5: **7 Scenes dataset** example images from left to right; Chess, Fire, Heads, Office, Pumpkin, Red Kitchen and Stairs.
Details and Issues with Approach
Details of Approach (Neural network):

- PoseNet is a modified version of Googles 22 layer Inception Network (27 if counting pooling layers)
  - Includes 6 ‘inception modules’ and 2 additional intermediate classifiers which are discarded during testing
Details of Approach (Neural network):

- Modifications to LeNet
  - Replace all softmax classifiers with affine regressors
  - Insert another fully connected layer with size 2048 before the final regressor (used for generalization exploration)
  - At test time, normalize quaternion orientation vector to unit length
- Results in a 23 layer (28 layers including pooling) network
Details of Approach (Neural network):

- Euclidean Loss / Affine Regressor layers

```plaintext
layer {
  name: "loss3/loss3_xyz"
  type: "EuclideanLoss"
  bottom: "cls3_fc_xyz"
  bottom: "label_xyz"
  top: "loss3/loss3_xyz"
  loss_weight: 1
}

layer {
  name: "loss3/loss3_wpqr"
  type: "EuclideanLoss"
  bottom: "cls3_fc_wpqr"
  bottom: "label_wpqr"
  top: "loss3/loss3_wpqr"
  loss_weight: 500
}
```
Details of Approach (Neural network):

- Learning location and orientation
  - Train network on Eucliden loss
  - Found that training on just position or orientation performed poorly compared to training on both simultaneously

\[
\text{loss}(I) = \| \hat{x} - x \|_2 + \beta \left\| \hat{q} - \frac{q}{\|q\|} \right\|_2
\]
Details of Approach (Neural network):

- Learning location and orientation

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

- Balance \( \beta \) must be struck between orientation and translation penalties.
- Optimal \( \beta \) given by ratio between expected error of position and orientation at the end of training (not beginning)
Details of Approach (Neural network):

- PoseNet model was implemented in Caffe and trained using stochastic gradient descent
  - Base learning rate was $10^{-5}$
    - Reduced by 90% every 80 epochs
  - Momentum of 0.9
  - Batch size of 75
  - Subtract separate image mean for each scene
Issues with Approach:

- Starting network weights (LeNet pretrained on XX) are very important for PoseNet performance
**Issues with Approach:**

- No output uncertainty produced by network
- Relatively large error compared to SCoRe Forest (indoors - as SCoRe Forest cannot handle the large outdoor datasets)
- Even utilizing transfer learning yields semi-long training times (3-6 hours on Nvidia Titan X)
Results
## Results:

<table>
<thead>
<tr>
<th>Scene</th>
<th># Frames</th>
<th>Spatial Extent (m)</th>
<th>SCoRe Forest (Uses RGB-D)</th>
<th>Dist. to Conv. Nearest Neighbour</th>
<th>PoseNet</th>
<th>Dense PoseNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>King’s College</td>
<td>1220</td>
<td>140 x 40m</td>
<td>N/A</td>
<td>3.34m, 2.96°</td>
<td>1.92m, 2.70°</td>
<td>1.66m, 2.43°</td>
</tr>
<tr>
<td>Street</td>
<td>3015</td>
<td>500 x 100m</td>
<td>N/A</td>
<td>1.95m, 4.51°</td>
<td>3.67m, 3.25°</td>
<td>2.96m, 3.00°</td>
</tr>
<tr>
<td>Old Hospital</td>
<td>895</td>
<td>50 x 40m</td>
<td>N/A</td>
<td>5.38m, 4.51°</td>
<td>2.31m, 2.69°</td>
<td>2.62m, 2.45°</td>
</tr>
<tr>
<td>Shop Façade</td>
<td>231</td>
<td>35 x 25m</td>
<td>N/A</td>
<td>2.10m, 5.20°</td>
<td>1.46m, 4.04°</td>
<td>1.41m, 3.59°</td>
</tr>
<tr>
<td>St Mary’s Church</td>
<td>1487</td>
<td>80 x 60m</td>
<td>N/A</td>
<td>4.48m, 5.65°</td>
<td>2.65m, 4.24°</td>
<td>2.45m, 3.98°</td>
</tr>
<tr>
<td>Chess</td>
<td>4000</td>
<td>3 x 2 x 1m</td>
<td>0.03m, 0.66°</td>
<td>0.41m, 5.60°</td>
<td>0.32m, 4.06°</td>
<td>0.32m, 3.30°</td>
</tr>
<tr>
<td>Fire</td>
<td>2000</td>
<td>2.5 x 1 x 1m</td>
<td>0.05m, 1.50°</td>
<td>0.54m, 7.77°</td>
<td>0.47m, 7.33°</td>
<td>0.47m, 7.02°</td>
</tr>
<tr>
<td>Heads</td>
<td>1000</td>
<td>2 x 0.5 x 1m</td>
<td>0.06m, 5.50°</td>
<td>0.28m, 7.00°</td>
<td>0.29m, 6.00°</td>
<td>0.30m, 6.09°</td>
</tr>
<tr>
<td>Office</td>
<td>6000</td>
<td>2.5 x 2 x 1.5m</td>
<td>0.04m, 0.78°</td>
<td>0.49m, 6.02°</td>
<td>0.48m, 3.84°</td>
<td>0.48m, 3.62°</td>
</tr>
<tr>
<td>Pumpkin</td>
<td>4000</td>
<td>2.5 x 2 x 1m</td>
<td>0.04m, 0.68°</td>
<td>0.58m, 6.08°</td>
<td>0.47m, 4.21°</td>
<td>0.49m, 4.06°</td>
</tr>
<tr>
<td>Red Kitchen</td>
<td>7000</td>
<td>4 x 3 x 1.5m</td>
<td>0.04m, 0.76°</td>
<td>0.58m, 5.65°</td>
<td>0.59m, 4.32°</td>
<td>0.58m, 4.17°</td>
</tr>
<tr>
<td>Stairs</td>
<td>2000</td>
<td>2.5 x 2 x 1.5m</td>
<td>0.32m, 1.32°</td>
<td>0.56m, 7.71°</td>
<td>0.47m, 6.93°</td>
<td>0.48m, 6.54°</td>
</tr>
</tbody>
</table>
Figure 7: Localization performance. These figures show our localization accuracy for both position and orientation as a cumulative histogram of errors for the entire testing set. The regression convnet outperforms the nearest neighbour feature matching which demonstrates we regress finer resolution results than given by training. Comparing to the RGB-D SCoRe Forest approach shows that our method is competitive, but outperformed by a more expensive depth approach. Our method does perform better on the hardest few frames, above the 95th percentile, with our worst error lower than the worst error from the SCoRe approach.
Conclusion
Conclusion / Summary:

- PoseNet is an end-to-end 6DOF pose regression convnet
- 5ms run-time, 50MB total storage space
- Large Scale indoor and outdoor relocalization
- Release of public dataset consisting of over 10,000 pose annotated images
Thanks!
Questions?
Quiz
Quiz:

1. PoseNet is able to output uncertainty
   a. True
   b. False

2. PoseNet is based off which of the following models?
   a. VGG16
   b. AlexNet
   c. LeNet
   d. ResNet