2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)
June 28 - July 1, 2017 at Maison Glad Jeju, Jeju, Korea

Towards Accurate Kidnap Resolution Through Deep Learning

Kent Sommer[1], Keonhee Kim[2], Youngji Kim[3], Sungho Jo[4]
[1,2,4]School of Computing, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, 305-701, Republic of Korea
(Tel : +82-42-350-7740; E-mail: kaistisnl@gmail.com)
[3]Department of Civil & Environmental Engineering, KAIST, Daejeon, 305-701, Republic of Korea
(Tel : +82-42-350-3632; E-mail: youngjikim@kaist.ac.kr)

***Abstract*—This paper presents a six degree of freedom position regression CNN (convolutional neural network) based on Google's Inception-V4 CNN. This network is then evaluated quantitatively and compared to previous state-of-the-art position regression CNNs. Our model achieves a 22% and 51% relative improvement compared to previous state-of-the-art methods for position and orientation accuracy respectively. A modular system for integrating our model into probabilistic localization algorithms for accurate kidnap resolution and global metric initialization in real-time is also introduced and evaluated. This modular system is able to globally initialize 85% of the time in under 70ms. If the robot is allowed to rotate in place and capture multiple views, this rises to 95%.**

***Keywords*—Simultaneous localization and mapping, Machine learning, Cognitive robotics, Computer vision**

1. Introduction

The ability to globally self-localize in a known area, also known as the kidnap problem, is an important requirement for mobile robotics as well as many other areas such as Structure-from-Motion (SfM) and Virtual or Augmented Reality (VR/AR). Solving the kidnap problem is required in many scenarios for mobile robotics such as a cleaning person moving a robot to a different location overnight, or for the task of long-term mapping. In both of these scenarios, it is vital for the robot to be able to recover its current location and continue to perform its job. While there are many approaches to the kidnap problem that utilize optical range finders [1–4], these methods suffer under highly repetitive structure even with large visual differences and are often so computationally intensive that they are impractical for kidnap resolution in large scale environments. Additionally, due to the high cost of optical range finders, we choose to focus our work on global self-localization using vision and deep learning.

Current state-of-the-art approaches [5–9] to image based localization still largely rely on hand-crafted image features such as those generated using scale-invariant feature transform (SIFT) [10], or oriented FAST and rotated BRIEF (ORB) [11]. In order to perform localization using these methods, a database containing position labeled images is often utilized. Correspondences between a given input image and every image in the database are calculated and the closest match is returned. If a 3D model of the scene is available along with mappings from every 3D point to its corresponding image features, it is possible to obtain finer location estimates. This is usually accomplished by using an n-point solver along with RANSAC [12] as demonstrated by Kukelova et al. in [13]. The inherent weakness of these approaches is that if few correct matches are found, localization is not possible.

Recently, some attempts have been made to apply deep learning to the task of camera localization. PoseNet [14, 15] formulates the localization task as a six degree-of-freedom (DoF) regression problem. Alternatively, PlaNet [16] formulates the localization task as a classification problem by dividing up the earth into distinct cells. So, during inference, every image is labeled, with some certainty, as belonging to one of the cells. While PlaNet allows for massive scale localization, it is largely unusable for mobile robotics as much finer localization proposals are required. Additionally, PlaNet is unable to recover orientation. To this end, we decided to formulate our approach as a regression task similar to PoseNet rather than a classification task like PlaNet.

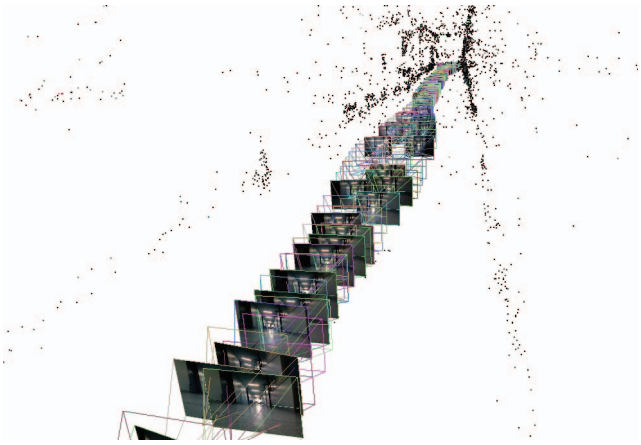

Fig. 1. six degree of freedom global localization using our model on our hallway dataset. Point cloud is included to better visualize localization.

PoseNet, although it is able to recover both orientation and position, still largely underperformed compared to the state-of-the-art hand-crafted feature methods such as [5–9]. To this end, we developed a new position regression network that provides a 25% relative improvement in position accuracy and 51% relative improvement in orientation accuracy over

978-1-5090-3056-9/17/$31.00 ©2017 IEEE

PoseNet. To achieve this, we make use of a state-of-the-art image recognition network, Inception-V4 [17], and modify it to the task of six degree of freedom position regression. We also introduce a framework for integrating our model into probabilistic localization frameworks allowing a robot to perform global metric initialization accurately and in real-time.

2. Approach

2.1. Architecture

In order to facilitate higher position regression accuracy, we decided to explore convolutional neural networks with high image recognition performance and utilize them for transfer learning to our task of kidnap resolution. The model that we use as a base for our network is Google's Inception-V4 [17] which is able to achieve a 5% and 20% error rate for the top-5 and top-1 performance measures on ImageNet [18] respectively. PoseNet was based on GoogLeNet which achieved an error rate of 6% for the top-5 performance measure [19]. The reason for utilizing a model with higher image classification performance for transfer learning to our task is that the early layers in the network are already tuned very precisely to output highly usable features. In a similar way that performance increases from having a high amount of SIFT features for pose estimation, high-quality CNN features increase performance for position regression.

Our model seeks to determine the position and orientation from a single RGB image in both indoor and outdoor settings. To accomplish this, we modify the Inception-V4 architecture as follows:

- Remove the final softmax classification layer.
- Add two regression layers after the final average pooling layer in order to output position (x,y,z) and orientation as a quaternion (w,p,q,r) respectively.

The overall schema for our network is then as shown in Fig. 2. Note that the stem portion of the network is identical to the stem described in [17].

2.2. Importance of Transfer Learning

It should be noted that training for the task of position regression directly with such a large model is extremely impractical. Due to a large number of parameters, an infeasible amount of labeled training data would have to be collected. Additionally, the network would likely be prone to over-fitting. Transfer learning provides a simple way to avoid these issues since it initializes the network weights with known useful starting points. This allows the network to more quickly learn to accomplish the final task and lowers the required amount of training data significantly compared to training from scratch. As the PoseNet authors discussed, the choice of the dataset to train the original network with can make a significant difference in how quickly the model approaches a good solution.

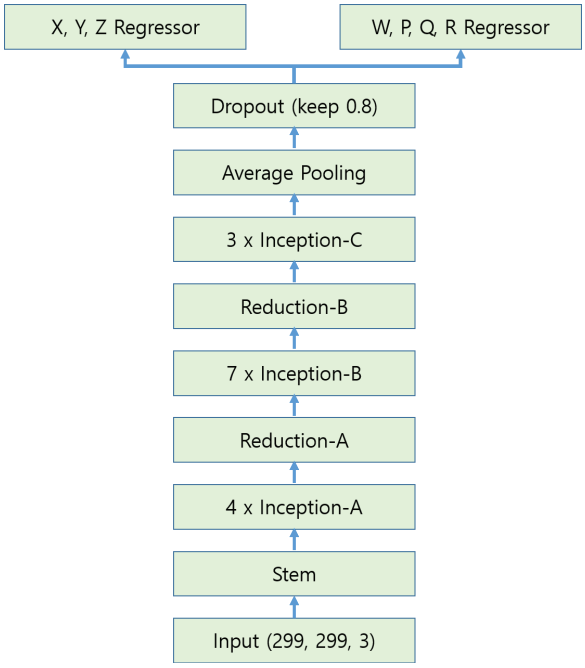

Fig. 2. Overview of Position Regression Network Schema.

2.3. Training

The problem of regressing a position from a single RGB image can be framed as in equation 1 where I_n is an input image to the mapping function $f(I_n)$. This mapping function is learned and approximated by a neural network whose architecture is as described in Fig. 2, and maps from the input RGB image to a position vector $P_n = [x, y, z, w, p, q, r]^T$.

$$f(I_n) \Rightarrow P_n \tag{1}$$

In order to let the network learn the mapping, it is necessary to formulate a loss function. To this end, we use the same loss function as [14] which is described in Equation 2.

$$L_i = \|\hat{x} - x\|_2 + \beta \left\| \hat{q} - \frac{q}{\|q\|} \right\|_2 \tag{2}$$

This is simply the sum of the L2-norm of the position and orientation along with a scaling parameter β. Here, $\hat{x}$ is the predicted position component of the output vector P_n, while x is the ground truth position label. Similarly, $\hat{q}$ is the predicted quaternion component of the output vector P_n, and q is the ground truth label. As in [14], β is hand picked to push the loss values for position and orientation to be roughly equal.

Adam [20] was chosen as the optimization function, which varies from [14], as we found Adam to be substantially more stable and require less hyperparameter tuning than Stochastic Gradient Descent. For Adam, we used an initial learning rate of 10^{-4} while leaving the rest of the hyperparameters as suggested in [20].

3. Application to the Kidnap Problem

In order for our localization network to be useful for application to the kidnap problem, we developed a simple integration architecture (Fig. 3) that can be used with most existing probabilistic localization algorithms.

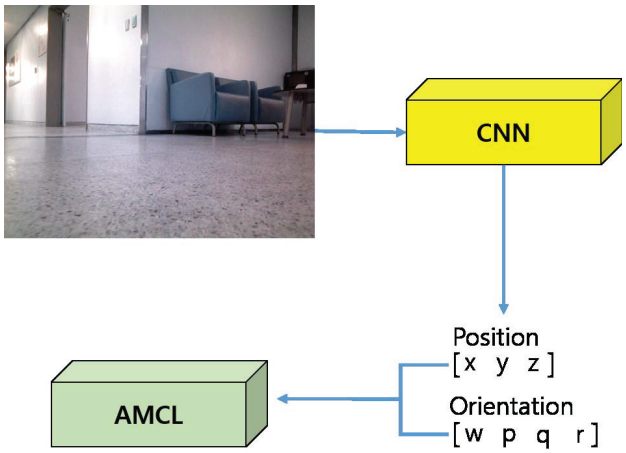

Fig. 3. Overview of integration architecture.

The neural network runs separately from the rest of the localization or SLAM architecture which allows for enhanced flexibility and modularity. For our tests we used adaptive Monte Carlo Localization (AMCL) [21], however, any probabilistic localization method could be used. AMCL uses a particle filter to track the pose of a robot against a known map. Normally, a user must enter an initial estimate of the position of a given robot in the map to initialize the particle filter. However, using the architecture shown in Fig. 3, it is possible to feed AMCL the output of our neural network and initialize the particle filter automatically. Through qualitative analysis, we found that this was able to correctly initialize the robot consistently. Since probabilistic localization algorithms such as [21] do not require precise initialize estimates, they make the perfect pairing with our neural network model for real world usage.

4. Experiments and Results

In this section, we first describe our dataset for real-world kidnap resolution evaluation. Next, we present an evaluation of our fully integrated system over our dataset. Last, we show a direct comparison to the current state-of-the-art over several publicly available datasets and discuss run-time of our model.

4.1. Data Collection

We used the TurtleBot robotics development platform to collect the data used to train our model. The device consisted of a Kobuki base, a laptop computer, and an RGB-D camera. The platform was controlled with via ROS Kinetic [22] and several Python based ROS nodes to facilitate image grabbing and labeling.

For the location of our dataset, we chose a hallway where there was limited sunlight as it interferes with the IR sensor on all RGB-D cameras. The TurtleBot was driven along the hallway manually several times. A Python based ROS node was used to save the position, orientation, and the view of the RGB-D camera every 0.5 seconds. A total of 1535 images were gathered, with 1152 used for training, and 383 used to evaluate the efficacy of the model.

Fig. 4. Example pictures from our dataset. Despite the similarity of many of our images, our model was able to recover position and orientation data accurately.

The six degree of freedom positional data was gathered through the use of the Cartographer [23] SLAM system. We chose to use SLAM as our ground truth label source as it shows the applicability to real world mobile robotics setups. Mapping of the dataset location was done in parallel with data collection.

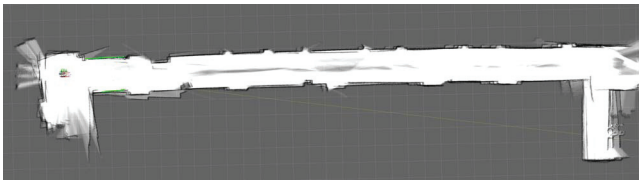

Fig. 5. Map of the localization area for our dataset

However, there were some limitations to the ground truth accuracy of the dataset. The RGB-D camera information is noisier and less accurate than a laser scanner. The wheel odometry sensors in the TurtleBot, while accurate, suffered from some loss of precision due to wheel slip caused by the smooth hallway surface. This problem was most commonly observed when making large turns. These inaccuracies caused slight drift during self-localization but were automatically corrected by loop closure algorithms in the mapping software. Using a real laser scanner would allow for more accurate SLAM and thus better ground truth labels for the images. Even with these challenges, however, we were still able to accurately estimate the robot's position given novel images from the test set.

4.2. Evaluation for Kidnap Resolution

In order to evaluate the efficacy of our model for real world kidnap resolution, we evaluated the accuracy of the position predictions over the test set of our dataset described in section 4.1 (Table 1).

Since probabilistic localization approaches such as AMCL can take noisy initial estimate and quickly converge to the correct location, we found that our model was able to produce initial position estimates that were consistently close enough to the true position for AMCL to converge. Out of 20 tests, the

robot was successfully able to initialize the particle filter for AMCL and localize itself globally on its first try 16 times (80% success rate). If the robot is allowed to rotate and average its position estimates from the network, we found the number of successful attempts rises to 95%. Alternatively, we tried feeding multiple random crops of a single view into the network. While this approach did improve performance (85% success rate), it did not improve performance nearly as much as averaging the outputs over multiple views. These results are summarized in Table 2.

TABLE 1
KIDNAP RESOLUTION ACCURACY.

–	Position	Orientation
Median Error	0.59m	0.77°

TABLE 2
KIDNAP RESOLUTION PERFORMANCE.

Prediction Method	Successful	Unsuccessful	Success Rate
Single View	16	4	80%
Multiple Crops	17	3	85%
Average Over Views	19	1	95%

4.3. Comparison to State-of-the-Art

Since PoseNet [14] is a novel approach to localization, it is the only direct comparison that can be made to our model. In order to show improvement over PoseNet, we compared the position and orientation accuracy of their model to that of our own model on the Kings College [24] and 7-Scenes [25] (Chess, Office, and Stairs) datasets. The Kings College dataset consists of images taken in an area of $5,600$ meters^2 which were then labeled with 6-DoF positions using structure from motion techniques. The 7-Scenes dataset is a collection of tracked RGB-D camera frames in seven small areas, however, we utilize only three of these scenes (Fig. 6).

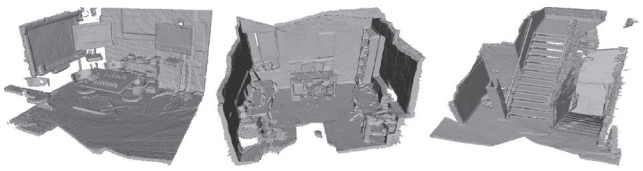

Fig. 6. 3D model of 7-Scenes selected datasets. Left to right: Chess, Office, Stairs.

Testing showed our model had, in general, a 22% relative improvement in position accuracy, and a 51% relative improvement in orientation accuracy over [14]. The results from our test are summarized in Table 3 (the relative improvement for position and orientation are shown under our models results). The substantial orientation improvement is due largely to more informative lower level features that are extracted from the Inception-V4 base of our model.

TABLE 3
COMPARISON TO STATE-OF-THE-ART ON SELECTED DATASETS

Model	Kings College	Chess	Office	Stairs
Conv. Nearest Neighbor	3.34m,5.92°	0.41m,11.2°	0.49m,12.0°	0.56m,15.4°
PoseNet	1.92m,5.40°	0.32m,8.12°	0.48m,7.68°	0.47m,13.8°
Ours	1.46m,2.67°	0.25m,4.02°	0.38m,3.69°	0.38m,10.2°
	+24%,+51%	+21%,+50%	+21%,+52%	+19%,+26%

4.4. Runtime

Unlike alternative approaches to the kidnap problem, such as scan matching, which become slower as the map size increases, our neural network has a constant time runtime of $\approx$ 11ms on a CPU (Quad Core Intel Core i7-6700 @3.4GHz) regardless of the map size. The reason for this constant runtime is that a single forward computational pass through the network per input image is all that is required. Since the input data size does not change, and the network parameters and model do not change after training, the runtime remains constant. This ensures our approach is real-time and can run at over 60 frames per second on only a CPU.

TABLE 4
RUNTIME COMPARISONS BY METHOD

Method	Runtime
Single Image	$\approx$ 11ms
Multiple Crops	$\approx$ 66ms
Multiple Views	$\approx$ 66ms

5. CONCLUSION AND FUTURE WORK

In this paper, we have presented an implementation of visual-based 6-DoF self-localization using deep learning, as well as an integration for use with probabilistic localization algorithms that provides accurate kidnap resolution as well as global metric initialization on mobile robots. We demonstrated a quantitative 22% increase in position accuracy and a 51% increase in orientation accuracy compared to [14]. We have also evaluated global metric initialization using images from our dataset, reaching 80% accuracy with a single image input, 85% with multiple crops of a single image, and 95% accuracy with multiple views.

For further development, we may also apply our position regression network to the task of visual loop closure. If graph based SLAM is used such as [26], our position regression network can be used to provide loop closure proposals by comparing the output of the network for every camera frame against known visited positions. If the euclidean distance between two nodes joined by a non-sequential edge is below some threshold, it can be considered as revisiting a known location and be used for loop closure. We leave this implementation for future work.

REFERENCES

[1] A. Burguera, G. Oliver, and J. D. Tardos, "Robust scan matching localization using ultrasonic range finders," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*. IEEE, 2005, pp. 1367–1372.

[2] X. Zezhong, L. Jilin, and X. Zhiyu, "Scan matching based on cls relationships," in *Robotics, Intelligent Systems and Signal Processing, 2003. Proceedings. 2003 IEEE International Conference on*, vol. 1. IEEE, 2003, pp. 99–104.

[3] A. Mallios, P. Ridao, D. Ribas, and E. Hernández, "Scan matching slam in underwater environments," *Autonomous Robots*, vol. 36, no. 3, pp. 181–198, 2014.

[4] P. Biber and W. Straßer, "The normal distributions transform: A new approach to laser scan matching," in *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 3. IEEE, 2003, pp. 2743–2748.

[5] Y. Li, N. Snavely, D. P. Huttenlocher, and P. Fua, "Worldwide pose estimation using 3d point clouds," in *Large-Scale Visual Geo-Localization*. Springer, 2016, pp. 147–163.

[6] B. Zeisl, T. Sattler, and M. Pollefeys, "Camera pose voting for large-scale image-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2704–2712.

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

[8] B. Zhang, Q. Zhao, W. Feng, M. Sun, and W. Jia, "Sift-based indoor localization for older adults using wearable camera," in *Biomedical Engineering Conference (NEBEC), 2015 41st Annual Northeast*. IEEE, 2015, pp. 1–2.

[9] J. Wang, H. Zha, and R. Cipolla, "Coarse-to-fine vision-based localization by indexing scale-invariant features," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 36, no. 2, pp. 413–422, 2006.

[10] T. Lindeberg, "Scale invariant feature transform," *Scholarpedia*, vol. 7, no. 5, p. 10491, 2012.

[11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2564–2571.

[12] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[13] Z. Kukelova, M. Bujnak, and T. Pajdla, "Real-time solution to the absolute pose problem with unknown radial distortion and focal length," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2816–2823.

[14] 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*, 2015, pp. 2938–2946.

[15] A. Kendall and R. Cipolla, "Modelling uncertainty in deep learning for camera relocalization," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4762–4769.

[16] T. Weyand, I. Kostrikov, and J. Philbin, "Planet-photo geolocation with convolutional neural networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 37–55.

[17] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," *arXiv preprint arXiv:1602.07261*, 2016.

[18] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

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

[21] P. Pfaff, W. Burgard, and D. Fox, "Robust monte-carlo localization using adaptive likelihood models," in *European robotics symposium 2006*. Springer, 2006, pp. 181–194.

[22] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, 2009, p. 5.

[23] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1271–1278.

[24] A. Kendall, M. Grimes, and R. Cipolla, "Research data supporting posenet: A convolutional network for real-time 6-dof camera relocalization: Kings college," 2015.

[25] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[26] S. Thrun and M. Montemerlo, "The graph slam algorithm with applications to large-scale mapping of urban structures," *The International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 403–429, 2006.