Convolutional Neural Network based Sensors for Mobile Robot Relocalization

Jay Patrikar [*,⋆] **Eeshan Gunesh Dhekane** [**,⋆] **Harsh Sinha** [*,⋆]
Gaurav Pandey [***] **Mangal Kothari** [*]

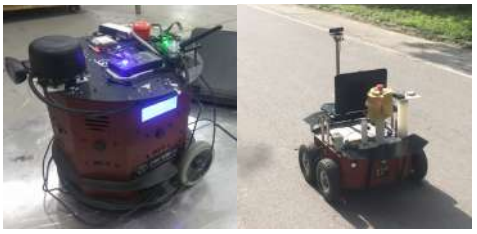
[*] *Department of Aerospace Engineering, Indian Institute of Technology, Kanpur. (e-mail: {jaypat, harshsin, mangal}@iitk.ac.in)*
[**] *Department of Electrical Engineering, Indian Institute of Technology, Kanpur. (e-mail: eeshangd@iitk.ac.in)*
[***] *Research and Innovation Center, Ford Motor Company, Palo Alto, USA. (e-mail: gpandey2@ford.com)*

Abstract: In this paper, we present a robust and real-time convolutional neural network (CNN) based localization algorithm that can be implemented on cheap and low power computation devices, thereby making it more practical. The proposed method first trains a CNN that takes RGB images from a monocular camera as input and performs regression for robot pose. It then incorporates the output of the trained CNN in an Extended Kalman Filter (EKF) for robot localization. Here we present a complete framework for localization of mobile robots in GPS–denied indoor and outdoor environments that do not require costly and computationally expensive sensors for real-time deployment. We demonstrate the performance of the proposed approach using a low power GPU, NVIDIA Jetson TX1, mounted on an indoor and an outdoor mobile robot platform. We show that the proposed approach gives promising results with localization error of 0.3 m in indoor environments and 1.3 m in outdoor environments. This makes CNN-sensors real-time, robust, low-cost and less computation-intensive substitutes for other sensors, with immense potential for a wider use in mobile robotics.

Keywords: Mobile Robotics, Convolutional Neural Networks, Robot Relocalization

1. INTRODUCTION

Robot localization, or inferring the location of a robot from the sensory information of its environment, is one of the most important tasks in mobile robotics. Traditionally, there have been many vision-based techniques that rely on point-landmark features. Some of the iconic examples of the feature representations used in these techniques are SIFT by Lowe (2004), SURF by Bay et al. (2006) and ORB by Rublee et al. (2011). These approaches, though successful and widely implemented, have significant limitations. They can not extract feature representations that can work competitively in newer and challenging environments. The extracted features do not capture the high-level and semantic abstractions of the scene. Consequently, they fail to work in scenarios with poor lighting conditions, significant lighting variations, motion blur and other artifacts that deteriorate image quality. Visual odometry for mobile robotics using feature tracking based on monocular and stereo-vision has been an active area of research in the past decade, with notable applications by Maimone et al. (2007) in Mars Exploration Rover (MER). But these approaches, although significantly accurate, are often computationally very expensive and cost-intensive.



(a) Indoor platform (b) Outdoor platform

Fig. 1. Robotic platforms used to generate datasets and consequently used to demostrate realtime relocalization

Within the last few years, convolutional neural networks (CNNs) have been successfully applied to an increasing and wide range of computer vision applications after their introduction by Krizhevsky et al. (2012). Earlier approaches to computer vision tasks have relied on training algorithms on low-level and hand-crafted features. On the other hand, the ability of CNNs to learn hierarchical feature representations from huge amounts of training data to specialize in the particular task at hand has been the main reason for this paradigm shift. Subsequently, CNNs have been incorporated in several robotics tasks like vision-based navigation. In the works of Giusti et al. (2016),

⋆ The authors have contributed equally.

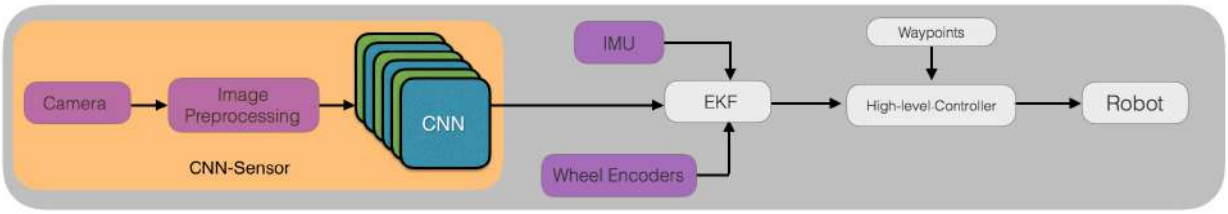


Fig. 2. The CNN-EKF architecture with inputs from CNN-sensor, IMU and wheel encoders

deep CNNs have been applied for image classification to predict the direction of navigation for trails in forests. Obstacle avoidance tasks have been addressed by Xie et al. (2017) using a combination of deep CNNs with reinforcement learning. Recent work by Wang et al. (2017), named DeepVO, performs end-to-end visual odometry on monocular RGB images. Additionally, the applications of CNNs in prey-predator settings have been demonstrated by Moeys et al. (2016). Deep CNNs have also been applied to relocalization tasks and one of the initial applications, named PoseNet, was conceived by Kendall et al. (2015). PoseNet trains deep CNN architectures for 6 Degree-of-Freedom pose regression from monocular RGB images. The extension of this work by Kendall and Cipolla (2016), named Bayesian PoseNet, models relocalization uncertainty for the same task. In addition to PoseNet, encoder-decoder style Hourglass Networks have been proposed by Melekhov et al. (2017) to perform indoor vision-based localization. These relocalization approaches are accurate as well as robust on large-scale indoor and outdoor datasets, with the only downside being their computationally intensive nature that hinders their real-time application. The CNN architectures incorporated in these approaches are very deep; for instance, the deep CNN architecture of PoseNet is a slight variant of the GoogLeNet architecture by Szegedy et al. (2015), which is a 23 layer deep network. The real-time application of these networks requires considerably large computing power. Benchmark tests by NVIDIA (Inference (2015)) show that on a Titan X, shallower nets like AlexNet are twice as energy efficient (2.5 img/sec/W vs 1.2 img/sec/W) than larger networks like GoogleNet. Hence, these architectures are not suited for real-time applications to fast-moving mobile robots working in relatively smaller environments. Thus, the deep architecture of CNNs is one of the major reasons because of which wide-spread use of CNNs in relocalization is yet to be achieved.

Mobile robots possess huge potential for a wide range of applications while offering a unique combination of constraints and challenges. Unlike the scenarios of applications of PoseNet, the environments of mobile robots are not large-scale. Mobile robots with wide-spread applications, like robots working in laboratories or workshops, need to model relatively smaller indoor or outdoor environments of the order of magnitude $\sim 10^1$m only. In addition, such robots are smaller in size, with dimensions of the order of magnitude $\sim 10^0$m, with modest weight-carrying capacities. They also have constraints on power capacity available while running. Hence, deploying heavier and large number of computing devices on the robot is not preferred. Mobile robots have speeds of the order of magnitude $\sim 10^0$–10^1m/s and hence, the sensors should have frequencies suitable for such speeds.

In view of these constraints and requirements, our work presents following major contributions—

- We propose to incorporate shallow CNN architectures, named `CNN-sensors`, for pose regression from a monocular RGB camera and use it to perform localization in an EKF framework. This is in contrast with deep CNN architectures implemented in PoseNet and other similar approaches. Despite the shallow nature of the architecture, the performance of CNN-sensors/CNN-EKF is at par with its deeper counterparts.
- We implement the complete CNN-EKF framework on FireBird VI Ltd. (2005 - 2017b) and 0x-Delta Ltd. (2005 - 2017a) robots, with a low cost and low power GPU–NVIDIA Jetson TX1, and a low-cost camera in GPS–denied environments to demonstrate the viability of the proposed CNN-EKF.

The rest of the paper is organized as follows– Section 2 provides the details of the CNN-sensor architecture implemented for mobile robots. Sections 3 and 4 provide details of indoor and outdoor experimentation respectively. This in-depth description includes dataset generation, training of the system, results, performance and cases of failure. Section 5 concludes the paper by summarizing our contributions, their demonstrations and future directions.

2. CNN-SENSORS

To incorporate the CNNs as vision-based sensors of mobile robots, several constraints and considerations are crucial. The neural networks should run in real-time and for mobile robots operating at the speeds of $\sim 10^0$–10^1m/s, we require the CNN to output predictions at sufficiently high rates ($\sim$ 10Hz–20Hz). Also, the available hardware should be able to load the CNN-sensor for online applications. TX1 has an available memory of 4GB, which needs to be shared for running ROS as well as CNNs. In view of these constraints, we fix the architecture of CNN-sensors to variants of AlexNet architecture by Krizhevsky et al.

Dataset	Dimensions	Training Frames : Testing Frames	CNN-sensor	PoseNet
Reading Room (Indoor)	10m×8m	1101 : 312	1.16m	0.50m
King's College (Outdoor)	140m×40m	1252 : 313	3.25m	1.92m

Table 1. Performance of CNN-sensor and PoseNet in terms of relocalization error

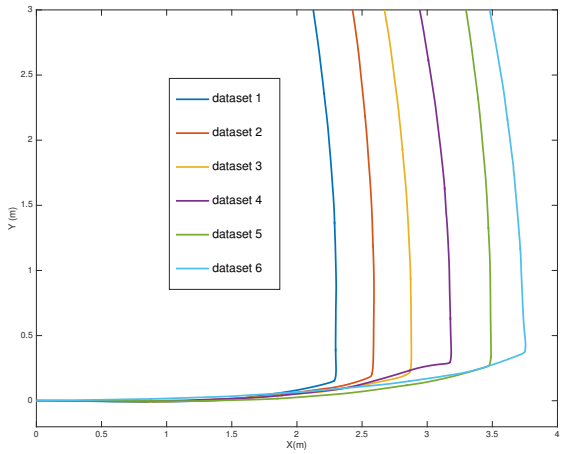


Fig. 3. Representation of the collected dataset

(2012). We have chosen Caffe by Jia et al. (2014) as the framework for CNN-sensors.

The architecture consists of eight layers, first five layers of which, namely `conv1` to `conv5`, are convolutional layers with the number of kernels being 96, 256, 384, 384 and 256 respectively. This number can be varied subject to the applications and environment wherein the applications are carried out. The output of convolutional layers is passed through non-linear ReLU activations. The pooling operation is performed after `conv1`, `conv2` and `conv5` layers. Fully connected layers `fc6` and `fc7` follow the convolutional layers and carry 4096 neurons each. This number can also be varied subject to the constraints on the size of the architecture. The output layer `fc8` consists of neurons for pose estimation $\mathbf{p}$. We initialize the weights of the network using `xavier` algorithm. The loss function $\mathcal{L}$ to train the two networks are chosen to be $\mathcal{L}_2$ norms and are defined as follows–

$$\mathcal{L}_{\mathbf{x}} = \|\hat{\mathbf{x}} - \mathbf{x}\|_2 ,$$

where x is the ground truth pose and $\hat{x}$ is the regressed pose. We use Adagrad algorithm by Duchi et al. (2011) and Adam algorithm by Kingma and Ba (2014) to train the networks.

We train these shallow networks on indoor and outdoor datasets to check their performance. Table 1 shows the performance, measured in terms of relocalization error in meter, of the shallow networks in comparison with that of PoseNet. Two datasets, one indoor and one outdoor, were used to train both PoseNet and CNN-Sensor architecture to benchmark their performances. The outdoor dataset is taken from the ones provided by Kendall et al. (2015) while the indoor dataset is generated inside a room. The ground truth for the indoor dataset is obtained using Structure-from-Motion (SfM) algorithm by Wu et al. (2011). We see that the CNN-sensors perform competitively with their deep counterparts. It should be noted, though, that the former are trained on scenarios suited for mobile robots and the later are trained for large-scale datasets. As the accuracies from CNN-sensor alone are not sufficient to estimate pose, an EKF architecture incorporating onboard sensors is used for pose estimation.

3. INDOOR EXPERIMENTATION

The following experiments were performed to establish the relocalization capabilities of the proposed CNN-EKF in indoor environments. The architecture was trained on compartmentalized, equally-spaced sections of an indoor arena and then implemented on the entire continuous domain. A laboratory is chosen as the indoor environment for demonstration purposes.

3.1 DATASET GENERATION

To acquire the indoor dataset, we used the Fire Bird VI robotic research platform from NexRobotics Ltd. (2005 - 2017b). The dataset constitutes images from a monocular RGB camera mounted on the robot. The robot was commanded to follow 'L'-shaped trajectories inside the laboratory complex. In order to cover the entire environment, the dataset was collected by turning at a number of equally spaced points in the trajectory. The pose $\mathbf{x}$ which is taken as the ground truth was estimated by using an array of sensors that included the following– (a) magnetic wheel encoders, (b) raw IMU data from the Pixhawk open-source hardware by Meier et al. (2011) and (c) a 2D LiDAR. A schematic of the sensor fusion algorithm is shown in Fig 5. The encoder and IMU were fused using an open-source EKF by Meeussen (2005 - 2017), the output of which is transformed using a LiDAR-based `gmapping` algorithm that uses improved Rao-Blackwellized particle filter by Grisetti et al. (2007) to solve the simultaneous localization and mapping (SLAM) problem. A less cost-intensive Logitech C270 camera was mounted on the front of the ground robot to capture the images with auto-controlled focus, exposure and white balance. Each image was tagged with the synchronized corresponding robot pose $\mathbf{x}$ and was stored onboard at 5 Hz. The synchronization was performed using an approximate time policy by Josh Faust (2005 - 2017) which matches messages that have different time stamps. The setup for indoor dataset collection can be seen in Fig 1a. In total, six different turn locations were chosen and the dataset was created by carrying out 2 runs per location choice. The lighting conditions varied during the duration of collection of the datasets. A representation of the dataset is shown in Fig 3 and a sample image from the dataset is shown in Fig 4a.

3.2 TRAINING

The dimensions of the chosen indoor environment are 10m×5m. Also, the challenging lighting conditions in the laboratory help in establishing the robustness of the proposed system. We generated 5717 images for the dataset,

(a) Indoor dataset (b) Outdoor dataset

Fig. 4. Sample images from dataset

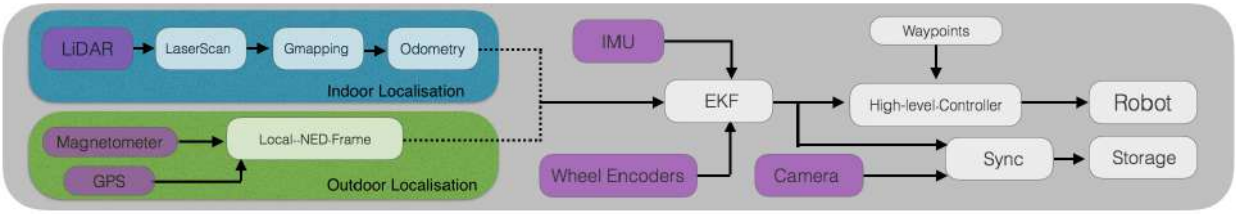


Fig. 5. Flow graph for pose estimation

out of which around 20% of the images were reserved for testing the performance of trained CNN-sensors. The rest of the images were split roughly in the ratio of 4:1 to generate training and validation datasets. We preprocessed, scaled and stored the images in LMDB format. We trained the CNN-sensors off-line on NVIDIA GeForce GTX TITAN X GPU system. The training time was around 6 hours with 50,000 iterations. After every 100 iterations, we evaluated the performance of the partially trained network on the validation dataset. Keeping track of this performance helped in avoiding over-fitting and in increasing generalization capability of the trained network. The trained CNN-sensor was also tested for its accuracy before its deployment in CNN-EKF. We observed that the indoor relocalization error using CNN-sensors was 0.38m±0.08m, which is very close to the performance obtained by very deep CNN architectures.

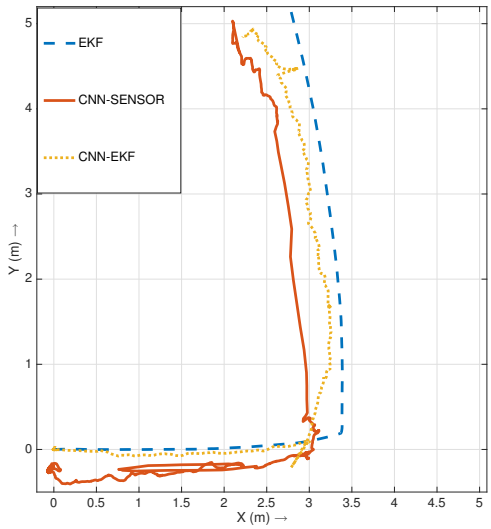


Fig. 6. A comparison of ground truth form the dataset, Raw CNN Sensor output and EKF on CNN Sensor Output on Indoor dataset

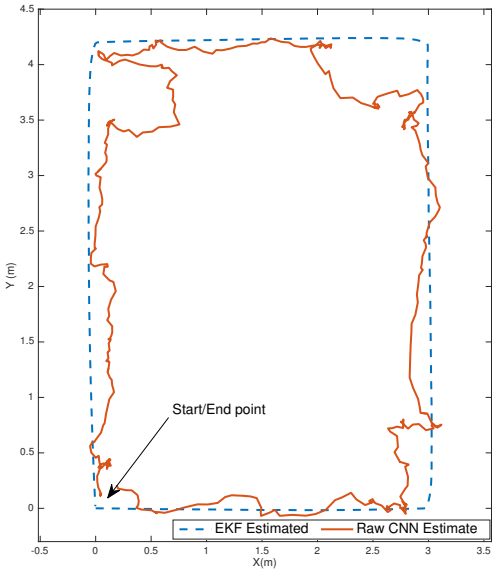


Fig. 7. Indoor failure case highlighting the inaccuracies in turns

3.3 TESTING AND PERFORMANCE

The trained network was run in forward mode, in real-time, on images from the onboard camera. The integration of Caffe with ROS was performed using a derivative of an earlier work by Tzutalin (2017). The CNN-sensor output was fused with wheel-odometry and IMU output to generate a CNN-EKF pose estimate, the schematic of which is shown in Fig 2. Both the CNN-EKF and LiDAR-assisted pose estimation pipelines were simultaneously and independently run onboard. The GPU used in this task was NVIDIA Jetson TX1 development board. The test runs were performed inside the same corridor, but the turns were made at locations different from those in the training dataset. The comparison of rates of CNN-EKF on TX1 and Titan X are shown in Table 2 and the resulting pose estimates are shown in Fig 6.

3.4 FAILURE CASES

One of the shortcomings of CNN-based pose estimation was observed at sharp turns in the robot trajectory, which was demonstrated in a specifically designed set of experiments. In this setting, the robot was made to follow a square-shaped trajectory of dimensions 3m × 4.2m. Training and testing were conducted on similar trajectories and the results are shown in Fig 7. It was observed that the estimation fails near the turns. Despite these deviations at the turns, CNN-EKF output quickly converges with the true trajectory when the robot proceeds on a straight path. This is a marked difference from other feature-based algorithms, like ORB-SLAM by Rublee et al. (2011), that rely on temporal correlation between images. The proposed CNN-EKF does not incorporate this aspect of modeling temporal correlation between images for pose

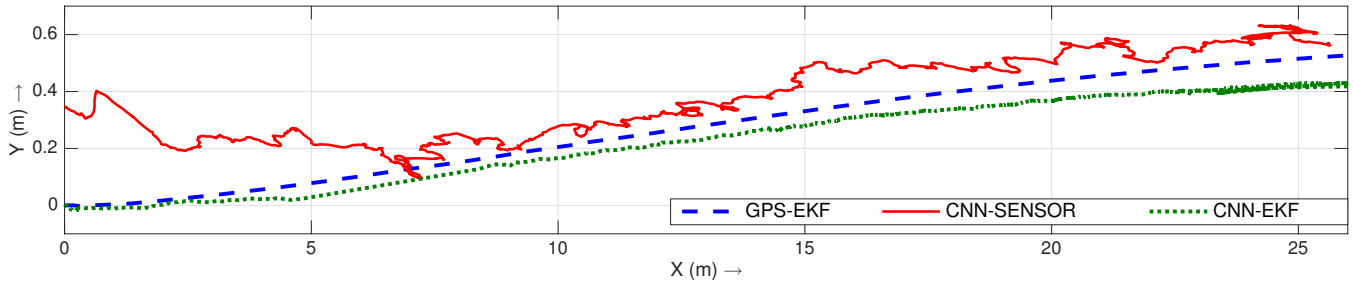


Fig. 8. A comparison of ground truth form the dataset, Raw CNN Sensor output and EKF on CNN Sensor Output on Outdoor dataset

estimation, which constitutes another direction for further investigations.

4. OUTDOOR EXPERIMENTATION

The following experiments were performed to establish the relocalization capabilities of the proposed CNN-EKF in outdoor environments and to demonstrate its robustness against changes in lighting conditions and noise.

4.1 DATASET GENERATION

For outdoor dataset generation, we used the 0x-Delta 4 Wheel Drive robot by NexRobotics Ltd. (2005 - 2017a). A road-section in the campus was chosen as the outdoor environment. Six runs were conducted to form a dataset that covered the entire environment. The ground truth was estimated using an array of sensors, which included the following– (a) A set of magnetic wheel encoders, (b) raw IMU data from the Pixhawk open-source hardware by Meier et al. (2011) and (c) a Ublox Neo-M8N GPS attached to the Pixhawk. GPS output was converted to the local frame with the initial position as its reference. The outputs of the encoder, IMU and GPS in the local frame were fused using an open-source EKF by Meeussen (2005 - 2017). A Genius 120-Degree Ultra Wide Angle fish-eye lens camera mounted at the front of the robot captured images. The synchronization of images with poses was performed using the same algorithm which was used in the indoor case. The lighting conditions varied for different runs; some of the runs were conducted in sunny conditions, while others were conducted in cloudy conditions. The schematic of the fusion algorithm is shown in Fig 5, the robot setup is shown in Fig 1b and a sample image from the dataset is shown in Fig 4b.

4.2 TRAINING

The dimensions of the chosen outdoor environment are 30m×7m. We generated 6402 images for the outdoor dataset, out of which 1317 images were reserved for testing purposes. The rest 5085 images were split roughly in the ratio of 4:1 to generate training and validation splits. Training is carried out off-line on NVIDIA GeForce GTX TITAN X GPU system. The training time is around 6 hours for 50,000 iterations and after every 100 iterations, the performance of the network was checked using the validation dataset. The trained CNN-sensor was tested for accuracy and then deployed in CNN-EKF. The outdoor relocalization error is 2.01m±0.90m, which is close to the performance of very deep CNN architectures.

4.3 TESTING AND PERFORMANCE

The network was tested in feed forward mode with the camera providing the images in real-time. The pipeline is the same as that described in Section 3.3. An onboard Jetson TX1 is used as the platform to simultaneously and independently run the GPS-EKF system, as depicted in Fig 5, and the CNN-EKF system, as depicted in Fig 2, to produce the respective pose estimates. Table 2 shows the details of the feed-forward rates required to run the network in real time. Fig 8 shows the results of the experimentation. As is shown here, the CNN-EKF output follows the ground-truth accurately.

Hardware	CNN-sensor Output Rate
Jetson TX1	18.5 Hz
TITAN X	200Hz

Table 2. Real-time performance of proposed architecture

4.4 FAILURE CASES

As the CNN-EKF architecture does not assume any temporal correlation between input images, it may regress similar looking images incorrectly. To highlight this issue, an outdoor dataset with repetitive visual content was generated. The network was trained as shown in Section 4.2 and the results are shown in Fig 9. The results indicate that the network fails to predict the pose correctly when the environment becomes repetitive. In such cases, the predictions start to repeat with the recurring visual content. It was also observed that as the visual content becomes distinctive, the network prediction accuracy increases again.

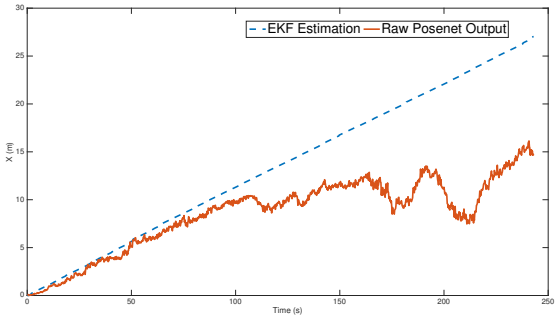


Fig. 9. Outdoor failure case highlighting the inaccuracies in a repetitive environment

5. CONCLUSIONS

In this paper, we demonstrate that the CNN-sensors, which have shallow architectures, are extremely accurate and robust in pose regression from RGB images. We also provide ROS-Caffe based implementation for incorporating the pose prediction of CNN-sensors into EKF algorithms. It should be noted that the CNN-sensors can also be incorporated in any other localization algorithm. The proposed CNN-EKF system is tested on indoor as well as outdoor environments; we are able to generate a competitively accurate, robust and real-time pose regression even with very basic mobile robot sensors. Hence, the proposed system is ideal for wide-spread mobile robot applications. Despite its merits, the proposed framework has some limitations. It fails to accurately regress the robot pose in environments with repetitive scenes or visual contents. Also, the pose regression output wavers at sharp turns. It should be noted that unlike ORB and other similar algorithms, this error in regression is not propagated to further predictions due to the absence of temporal correlation in predictions. One of the possible directions for future works can be incorporating temporal correlation between the input images for pose regression. Some other possible extensions might include incorporating visual inputs from two or more RGB cameras or multiple CNN-sensors and incorporating inputs from depth or other visual modalities, like point-clouds, using appropriate CNN-sensors.

REFERENCES

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up robust features. *Computer vision–ECCV 2006*, 404–417.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul), 2121–2159.

Giusti, A., Guzzi, J., Cireşan, D.C., He, F.L., Rodríguez, J.P., Fontana, F., Faessler, M., Forster, C., Schmidhuber, J., Di Caro, G., et al. (2016). A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, 1(2), 661–667.

Grisetti, G., Stachniss, C., and Burgard, W. (2007). Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE transactions on Robotics*, 23(1), 34–46.

Inference, G.B.D.L. (2015). A performance and power analysis. *Nvidia Whitepaper, November*.

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

Josh Faust, V.P. (2005 - 2017). Ros:message-filters::sync::approximatetime. `http://wiki.ros.org/message_filters/ApproximateTime`. Online.

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

Kendall, A., Grimes, M., and Cipolla, R. (2015). Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, 2938–2946.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A., Sutskever, I., and Hinton, G.E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

Lowe, D.G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2), 91–110.

Ltd., N.R.P. (2005 - 2017a). 0x-delta 4 wheel drive robot. `http://www.nex-robotics.com/products/0x-series-robots/0x-delta-4-wheel-drive-robot.html`. Online.

Ltd., N.R.P. (2005 - 2017b). Fire bird vi robotic research platform. `http://www.nex-robotics.com/products/fire-bird-vi-robot/fire-bird-vi-robotic-research-platform.html`. Online.

Maimone, M., Cheng, Y., and Matthies, L. (2007). Two years of visual odometry on the mars exploration rovers. *Journal of Field Robotics*, 24(3), 169–186.

Meeussen, W. (2005 - 2017). robot-pose-ekf. `https://github.com/ros-planning/navigation`. Github,Online.

Meier, L., Tanskanen, P., Fraundorfer, F., and Pollefeys, M. (2011). Pixhawk: A system for autonomous flight using onboard computer vision. In *Robotics and automation (ICRA), 2011 IEEE international conference on*, 2992–2997. IEEE.

Melekhov, I., Ylioinas, J., Kannala, J., and Rahtu, E. (2017). Image-based localization using hourglass networks. *arXiv preprint arXiv:1703.07971*.

Moeys, D.P., Corradi, F., Kerr, E., Vance, P., Das, G., Neil, D., Kerr, D., and Delbrück, T. (2016). Steering a predator robot using a mixed frame/event-driven convolutional neural network. In *Event-based Control, Communication, and Signal Processing (EBCCSP), 2016 Second International Conference on*, 1–8. IEEE.

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE international conference on*, 2564–2571. IEEE.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9.

Tzutalin, D. (2017). ros-caffe. `https://github.com/tzutalin/ros_caffe`. Online.

Wang, S., Clark, R., Wen, H., and Trigoni, N. (2017). Deepvo: towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, 2043–2050. IEEE.

Wu, C. et al. (2011). Visualsfm: A visual structure from motion system.

Xie, L., Wang, S., Markham, A., and Trigoni, N. (2017). Towards monocular vision based obstacle avoidance through deep reinforcement learning. *arXiv preprint arXiv:1706.09829*.