ContextualNet: Exploiting Contextual Information using LSTMs to Improve Image-based Localization

Mitesh Patel[1], Brendan Emery[2], Yan-Ying Chen[1]

***Abstract*— Convolutional Neural Networks (CNN) have successfully been utilized for localization using a single monocular image [1]. Most of the work to date has either focused on reducing the dimensionality of data for better learning of parameters during training or on developing different variations of CNN models to improve pose estimation. Many of the best performing works solely consider the content in a single image, while the context from historical images is ignored. In this paper, we propose a combined CNN-LSTM which is capable of incorporating contextual information from historical images to better estimate the current pose. Experimental results achieved using a dataset collected in an indoor office space improved the overall system results to 0.8 m & $2.5°$ at the third quartile of the cumulative distribution as compared with 1.5 m & $3.0°$ achieved by PoseNet [1]. Furthermore, we demonstrate how the temporal information exploited by the CNN-LSTM model assists in localizing the robot in situations where image content does not have sufficient features.**

Keywords: Localization, Long Short Term Memory (LSTM), Computer Vision, Deep Neural Network, Pose estimation

I. Introduction

Estimating the position and orientation of a robot or a device using an image is a fundamental requirement for various computer vision and robotic applications such as motion estimation [2], pedestrian detection for autonomous cars [3], navigation of mobile robots [4], Simultaneous Localization and Mapping (SLAM) [4], and augmented reality [5]. Traditional approaches rely on identifying distinct image features and extracting local neighborhood descriptors for each feature. Scale Invariant Feature Transform (SIFT) [6] and Oriented FAST and Rotated BRIEF (ORB) [7] have been used to extract representable features from image content which are used for map building and/or localization within a map.

Lately, different flavors of Deep Neural Networks (DNN) have been explored to perform relocalization using camera images. Various Convolutional Neural Networks have been utilized to regress camera position and orientation [8], [1], [9], [10]. These models learn the 2D spatial features from raw camera images to perform localization in a given space. However, most of these models fail to exploit the temporal relationship between current and historical images, which may enable a model to perform better relocalization.

It is common to find very limited distinct patterns in images, particularly in indoor areas such as office buildings and convention centers. Images containing blank walls, repetitive textures and consistent space layouts may not offer sufficient information in a single image for an accurate prediction. Using the context provided by a sequence of previous images may introduce more distinguishable features and help these challenging cases.

[1]Mitesh Patel, and Yanying Chen are with FX Palo Alto Laboratory Inc., Palo Alto, CA - 94304, USA {mitesh,yan-ying}@fxpal.com

[2]Brendan Emery is with Faculty of Engineering and IT, University of Technology Sydney, Ultimo, NSW 2007, Australia brendan.emery@student.uts.edu.au

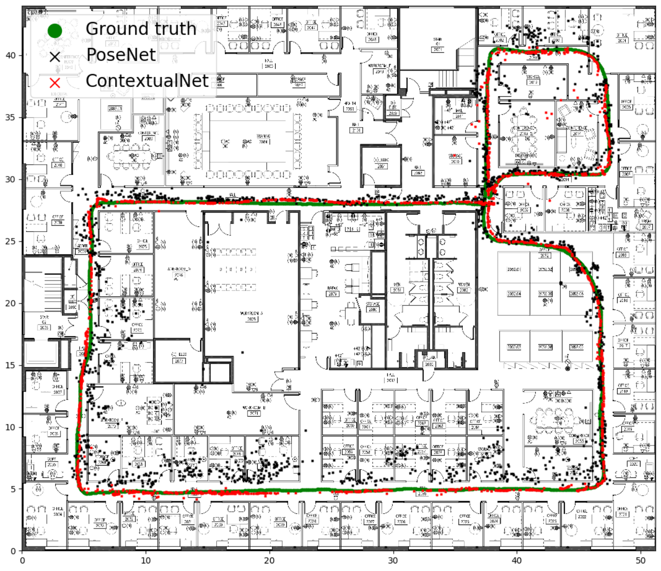


Fig. 1: Comparison of estimated position and orientation by our Contextual model and PoseNet with ground truth

In this paper, we propose a model which uses Long Short Term Memory (LSTM) layers to handle a sequence of previous images which provide contextual information for localization. A CNN architecture is used for extracting a vector of features from each single image which is passed through the LSTM layers. The results of the CNN-LSTM model against the ground truth is shown in Figure 1. The contribution of our proposed system are as follows:

i. We propose to use a CNN-LSTM framework (we call it ContextualNet) which exploits spatio-temporal information from a sequence of camera images to estimate the position and orientation of a robot. Unlike other approaches which utilize either traditional feature engineering [6] and/or other feature learning techniques [8], [9], the CNN-LSTM framework becomes effective in situations where there is a lack of features available from a single image.
ii. We investigate the effectiveness of the CNN-LSTM model by comparing it with other state of the art deep learning techniques [8], [10], [9] which rely on the

content of a single image. We evaluate the methods on both datasets logged in an office space as well as publicly available datasets. The overall median errors are reduced to 0.30 m, 1.49° as compared with the PoseNet model proposed in [8], which had median error of 0.94 m, 1.71° for the indoor dataset. We also perform a sensitivity analysis to better understand the relation between image sampling rate and sequence size used in the LSTM layers. Lastly, we test our model's ability to estimate position and orientation in real-time on a mobile robot navigating an office space.

II. Related Work

The idea of localizing from images is not new and has been extensively studied by researchers in the robotics and computer vision communities. Traditional computer vision techniques to perform localization can be divided into two main categories: keypoint based localization and keyframe based localization. Lately, with the success of CNNs, a wide variety of computer vision based applications have adopted this approach to solve various classification and detection problems. In this section, we review camera relocalization methods using both traditional computer vision techniques and Deep Neural Networks.

A. *Keypoint and Keyframe based localization*

Keypoint based localization approaches utilize feature points that are extracted from each image frame which are matched with a database of image features. Different methods such as SIFT [6] and ORB [7] are used to extract feature points. Since the database of feature points can be large, these methods require efficient retrieval methods. Shotton *el. al.* in [11] utilized random forests to regress scene coordinate labels to perform relocalization. The uncertainty of the results achieved from the random forest was further exploited using a probabilistic framework by Valentin *et. al.* [12]. Klein and Murphy used a keyframe based method where the position and orientation are determined by computing image similarity with a set of keyframe images with known poses [13]. The final pose is calculated using either the weighted average of key poses or the pose of a database image with the highest similarity score.

B. *Localization using Deep Neural Network Techniques*

Deep Neural Networks have been successfully utilized in various computer vision problems including image classification [14], image segmentation [15], object localization [16] and activity recognition [17]. The layered architecture of CNNs are used to learn features from the image content and is optimized towards a better representation that fulfills specific tasks or reduces reconstruction errors. PoseNet was one of the preliminary models proposed by Kendall *et. al.*, which regresses camera position and orientation from images [8]. An extension of PoseNet was proposed by Kendall *et. al.*, where they used a Bayesian model trained with dropout [1] to determine the model uncertainty. Wu *et. al.* proposed to use a multitask CNN model which learns the complex coupling between orientation and translation. Their model shared the lower layers of the CNN model and are branched at the higher layers to specifically optimize for orientation and translation [10]. While CNN layers can extract representative features, the aforementioned works only rely on the features in a single image and do not take the contextual information from a sequence of images into consideration.

Our work most closely follows the CNN-LSTM model proposed by Walch *et. al.*, where they utilized CNN layers to extract representative features of an image and further use LSTM layers to model the spatial relationships of the extracted CNN features [9]. The additional LSTM layer integrates structural correlations of features within an image and also reduces the feature dimension before being forwarded to a fully connected layer. Instead of using LSTM layers for modeling structural correlation in an image, we use LSTM layers to integrate the temporal correlations of a sequence of images ordered by time.

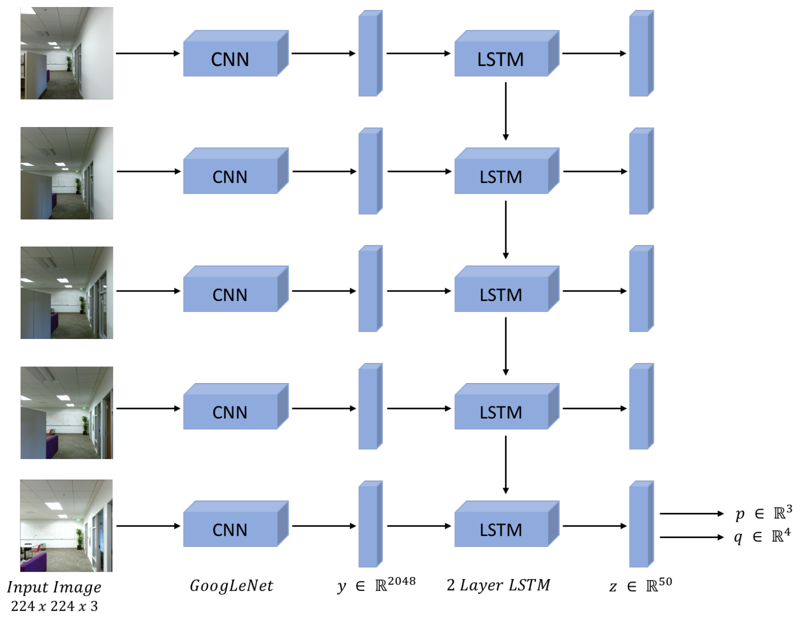
Using LSTM layers to aggregate sequential context has been addressed in photo geolocation. Weyand *et. al.* combines photo albums with an LSTM architecture that aims at exploiting temporal coherence to locate uncertain travel photos in the most likely geographic cells [18]. The integration of temporal coherence improves the performance by 50% over a single-image model, which supports the capability of LSTM in modeling temporal coherence. Instead of locating loosely coupled photos in specific geographic cells, our work is targeted at estimating real-valued positions and orientations of continuous image frames captured by a robot in an indoor/outdoor space and in a real-time setting. The temporal context is expected to offer additional cues in localizing individual image frames as well as reconstructing a route that the robot passed through.

In conclusion, our method differs from the prior methods in the following ways:

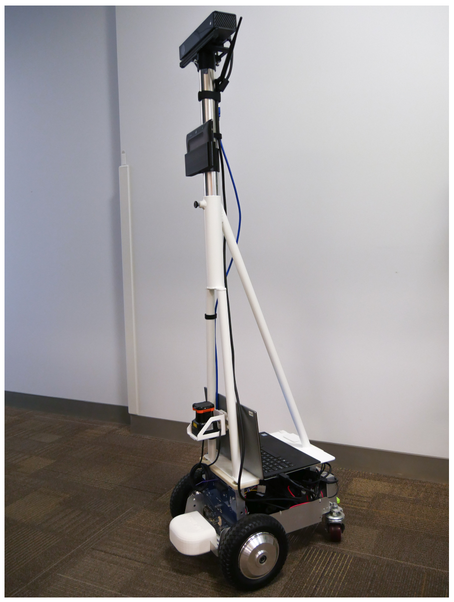
i. We develop a CNN-LSTM based model which learns the contextual relationship between images in a sequence. The model learns both the inter and intra image features to better estimate the position and orientation of a camera.
ii. Our proposed model is robust to estimating the position and orientation within images having minimal features. We demonstrate how the contextual information is exploited by the CNN-LSTM model in such a situation.

III. Proposed Methodology

Our system (ContextualNet) consists of a combined CNN-LSTM framework which learns the mapping between image content and the pose of the robot that captures the image. The CNN layers consist of a combination of convolution, max-pooling and dropout layers which are combined hierarchically to generate a DNN model. The CNN layers of ContextualNet extract representative features from each image in a sequence. CNN layers are spatial in nature; hence they are only capable of learning the features present in a single image. In order to learn the temporal relationship between images, we utilize LSTM Layers [19]. LSTMs are



(a) ContextualNet Model



(b) Magni Robot Platform

Fig. 2: (a) Architecture of the proposed ContextualNet Model which consists of both CNN and LSTM layers. The output of the CNN layers are flatten to a vector of size 2048 which are utilized as input by the LSTM layer. (b) Modified Robot platform used for data collection and testing in real-time.

a type of Recurrent Neural Network (RNN) designed to accumulate or forget relevant contextual information in hidden states. Further, LSTMs are a better fit than RNN layers as they are equipped to deal with the problem of vanishing and exploding gradients that can result from propagation of gradients through numerous layers [19]. In addition to the hidden units h_t, the LSTMs consist of an input gate i_t, forget gate f_t, output gate o_t, input modulation gate g_t, and memory cell c_t. The input gate and forget gate provides LSTMs with the ability to learn to selectively pass or forget parts of the current input and previous state, respectively. Similarly, the output gate learns how much of the memory cell to transfer to the hidden state. The additional cells of the LSTMs allow them to learn the complex and long-term temporal dynamics between images [17] (Further details of CNN and LSTMs can be found at [20], [19], [17]).

Various DNN architectures such as AlexNet [21], VGGNet [22] and GoogLeNet [20] can be used to extract representative features from an image. In this paper, we utilized the GoogLeNet architecture for the CNN layers primarily because we wanted to compare the performance of our proposed CNN-LSTM model with other models such as PoseNet [8] and Directional-PoseNet [9]. The feature vectors are then utilized by the LSTM layers to learn the inter image dependencies to estimate the robot pose ($\mathbf{P} = [\mathbf{p}, \mathbf{q}]$). The pose of the robot is represented by the 3D camera position, $\mathbf{p} \in \mathbb{R}^3$, and quaternion, $\mathbf{q} \in \mathbb{R}^4$. The position and quaternion are optimized using a Euclidean Loss function given below:

$$loss_i = \|\hat{\mathbf{p}}_i - \mathbf{p}_i\|_2 + \beta \cdot \left\| \hat{\mathbf{q}}_i - \frac{\mathbf{q}_i}{\|\mathbf{q}_i\|} \right\|_2 \tag{1}$$

where, $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{q}}_i$ are the estimated position and quaternion and $\mathbf{p}_i$ and $\mathbf{q}_i$ are the ground truth position and quaternion. The β term is used as a weighting factor in the loss function so that the position and orientation error values have a similar magnitude. This ensures that the weights of the model are influenced by both the position and orientation error during optimization. Once defined, the loss function can be optimized using different optimization techniques such as Stochastic Gradient Decent (SGD) [23] and Adaptive movement estimation (Adam) [24], which utilize the pose labels, $\mathbf{P}$,(i.e. 3D position, $\mathbf{p}$, and quaternion, $\mathbf{q}$) for every image I_i in the dataset.

A. *ContextualNet Model*

Our proposed ContextualNet model consists of 27 CNN layers and 2 LSTM layers. The 27 CNN layers are similar to that of GoogLeNet [20], where the input image size is $224 \times 224 \times 3$. The last fully connected layer is modified to generate a feature vector of size 2048 instead of 1024. This feature vector is used by the first LSTM layer as input. The first LSTM layer contains 512 hidden cells while the second layer layer contains 50. The second LSTM layer is connected to 2 separate fully connected layers which are used for estimating the robot pose, $\mathbf{P}$ The sequence size of the LSTM layers was set to 3. Furthermore, the LSTM layers are implemented in a *many-to-one* configuration, so that it takes a sequence of images to infer the pose, $\mathbf{P}$, of the robot at the current timestep. An overview of ContextualNet used in the experiments is shown in Figure 2a.

Furthermore, to better optimize the weights of both the CNN and LSTM layers, we used a multistage training process. In the first stage, the weights of the CNN layers were

TABLE I: Median position and orientation errors for different datasets

Scene	Area/Volume	PoseNet [8]	Directional PoseNet [9]	ContexualNet (PoseNet+LSTM)
King's College	5600 m^2	1.92m, 5.40°	**0.99m**, 3.65°	1.11m, **1.76°**
Old Hospital	2000 m^2	2.31m, 5.38°	**1.51m**, **4.29°**	4.17m, 4.39°
Shop Facade	875 m^2	1.46m, 8.08°	1.18m, 7.44°	**0.79m**, **6.70°**
St Mary's Church	4800 m^2	2.65m, 8.48°	**1.52m**, **6.68°**	1.85m, 7.20°
Average All	–	2.08m, 6.83°	**1.30m**, 5.52°	1.98m, **5.01°**
Chess	6 m^3	0.32m, 8.12°	0.24m, **5.77°**	**0.15m**, 6.12°
Fire	2.5 m^3	0.47m, 14.40°	0.34m, 11.90°	**0.16m**, **10.93°**
Heads	1 m^3	0.29m, 12.0°	**0.21m**, 13.70°	0.25m, **13.2°**
Office	7.5 m^3	0.48m, 7.68°	0.30m, 8.08°	**0.25m**, **7.45°**
Pumpkin	5 m^3	0.47m, 8.42°	0.33m, 7.00°	**0.26m**, **6.62°**
Red Kitchen	18 m^3	0.59m, 8.64°	0.37m, 8.83°	**0.20m**, **6.97°**
Stairs	7.5 m^3	0.47m, 13.80°	0.40m, 13.70°	**0.17m**, **10.83°**
Average All	–	0.44m, 10.43°	0.31m, 9.85°	**0.20m**, **8.87°**
Indoor Dataset	2000 m^2	0.94m, 1.71°	–	**0.30m**, **1.49°**

optimized and the weights of the LSTM layers were frozen. The weights of the CNN layers were initialized with the weights trained on the Places [25] dataset and were further fine-tuned using our dataset. In the second stage, we only optimized the weights of the LSTM layers and froze the weights of the CNN layers. The weights of the LSTM layers were initialized using a glorot uniform distribution. For both the training stages, we utilized the loss function defined in Equation 1, which was optimized using the Adam algorithm.

IV. Datasets

We evaluate and compare the performance of our proposed ContextualNet with PoseNet [8] on a dataset collected locally in an office space which spans over 2000 m^2. For completeness we also tested the performance of ContextualNet on several indoor and outdoor datasets which are publicly available. For the publicly available datasets, we compared the performance of ContextualNet with both PoseNet and Directional-PoseNet [9] as the results from these models were readily available. Details of each of the datasets are shown in Table I.

A. Indoor Dataset

We collected a large indoor dataset in our office space which consists of typical office furniture, corridors, offices, desks, and chairs (example images shown in Figure 2a). The RGB images were captured using a Microsoft Kinect V2 sensor[1] which was mounted on a magni robot frame[2]. The robot (shown in Figure 2b) was also equipped with a laser range finder which was used for ground truth pose estimation using the hector mapping [26] package of the Robot Operating System (ROS) [27]. The RGB images and the ground truth from hector mapping were logged at a sampling rate of 25 Hz. Two different datasets for training and testing the ContextualNet model were collected which consist of around 14000 images each. The resolution of each image was 960 × 540 pixels.

B. Public Datasets

We also used the Cambridge outdoor dataset [8] and 7Scenes [11] small-scale indoor dataset to compare the performance of ContextualNet with the accuracy achieved by PoseNet and Directional-PoseNet.

The images for the Cambridge outdoor dataset are captured using a Google LG Nexus 5 smartphone. Labels for the data set were generated using Structure from Motion (SfM) [28]. The dataset consists of urban clutter in the form of pedestrians and vehicles present while the data was collected. Further details of the dataset can be found in [8]. The sampling rate of the captured images were downsampled to 2Hz due to the processing time taken by SfM to generate the ground truth.

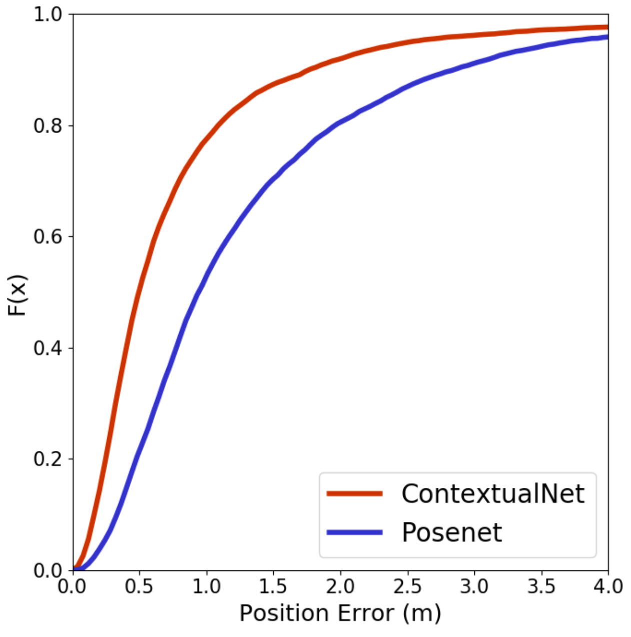
On the contrary, the 7scene dataset consists of RGB-D data captured in an indoor space. The image resolution was 640 × 480 and contained significant ambiguities in terms of motion blur, flat surfaces and varied lighting conditions. For our experiments, we only utilized RGB images. This dataset was collected in a smaller area as compared with the Cambridge outdoor dataset.

V. Experimental Results

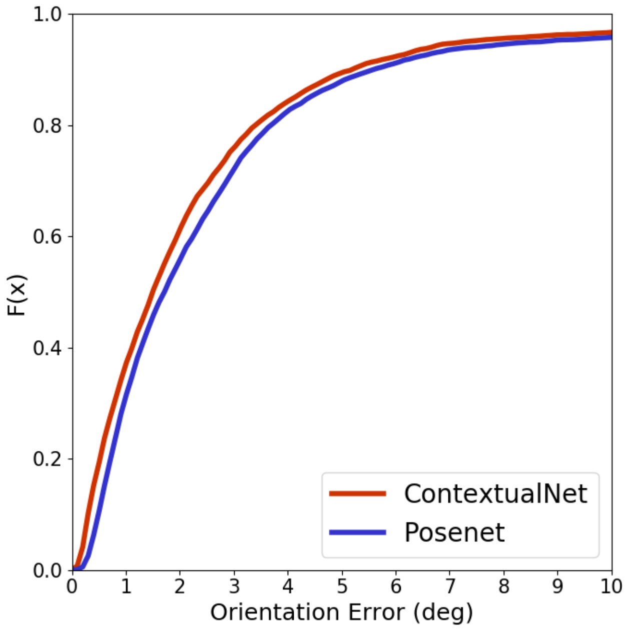
In all our experiments, the weights of both the CNN and LSTM layers were initialized as described in Section III-A. The images are downsampled and center-cropped to 224 × 224 × 3 pixels, which is the input to the model. A mean image is computed from the training data and is subtracted from all images used during training and testing. The models were trained using the Tensorflow libraries [29] on NIVIDIA titan X GPUs. The batches of image sequences were randomized for faster convergence. For the public dataset, we compared the performance of our model with

[1]https://www.microsoft.com/en-us/store/d/kinect-sensor-for-xbox-one/91hq5578vksc

[2]http://ubiquityrobotics.com/magni.html



(a) Cumulative Distribution for Position



(b) Cumulative Distribution for Orientation

Fig. 3: Comparison of the empirical cumulative distribution functions using ContextualNet and PoseNet models for both position and orientation on the Indoor dataset.

PoseNet, which is a CNN based model [8], and Directional-Posenet [9], where the LSTM layer is used to capture the spatial structure of features in an image. In order to find the best hyperparameters for all the models, we used a grid search based approach. The hyperparameters described in [8] [9] were used as baseline parameters for grid search to find the best hyperparameters. Further, the model was trained using multi-stage training process as described in Section III-A.

A. Indoor Dataset

The hyper parameters used for training the indoor dataset were batch_size $= 64$, regularization $\lambda = 10^{-4}$, learning rate $= 0.0001$, decay factor $= 10^{-7}$, and image sequence length $= 3$. The parameters used for the Adam [24] optimizer were kept as $\epsilon = 10^{-8}$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. The β value used for weighting the quaternion in the loss function was set to 750. The median error reported by the ContextualNet model was 0.30 m for position and $1.49°$ for orientation, whereas the PoseNet model had an error of 0.94 m and $1.71°$. Furthermore, the errors for both the position and orientation for the third quartile of the cumulative distribution function [30] were less then 0.8 m and $2.5°$ for ContextualNet compared with the PoseNet model, which was 1.5 m and $3°$ respectively. The comparison of the cumulative error for both position and orientation along the trajectory is shown in Figure 3.

B. Public Dataset

The hyperparameters used for training both the Cambridge and 7scene datasets were batch_size $= 64$, regularization $\lambda = 10^{-4}$, learning rate $= 0.0001$, decay factor $= 10^{-8}$ and the image sequence length was set to 3. Parameters for Adam were kept the same as those used for training the indoor dataset. The β value was set to 250 for both datasets.

For all the Cambridge datasets, the median error for both the position and orientation of our model were better then PoseNet. Our model did not perform as well compared with the Directional-PoseNet, especially for the position for three out of the four datasets, but gave minor improvement for orientation for two datasets (refer to Table I). The minor or no improvement in the position and orientation can be explained by two specific reasons. Firstly, the data was collected at 2 Hz, which potentially leads to less information between consecutive images which further dampens the results of ContextualNet. To further test our hypothesis, we performed an experiment on the *Old Hospital* dataset and trained the model using a sequence size of 7. The longer sequence size gave more contextual information to the model which resulted in a median error of 2.58 m for position and $4.47°$ for orientation. Lastly, by visualizing the training and testing images, we observed that the sequence of images used in the testing data does not have the same inter-image correlation as that of the training data.

For the indoor 7scene dataset, our model performed better than both Posenet and Directional-PoseNet. The average error for all the datasets with our model for position and orientation was 0.25 m and $8.87°$, respectively, whereas the same for Directional-PoseNet was 0.31 m and $9.85°$. A similar trend was observed when comparing the results of our model with PoseNet. Details on median accuracy using all three models for each dataset are listed in Table I.

TABLE II: Sensitivity Analysis for correlating Sampling Rate & Sequence Size

Freq(Hz.) / Seq. Size	3	5	8	12
25	0.304	0.303	0.300	0.269
8	0.295	0.272	0.328	0.465

C. Sensitivity Analysis

The image sequence size is an influential factor for sequence-based localization because it determines how many previous images are integrated in the prior context. However, the best recommended size may be affected by the image sampling rate. For a higher sampling rate, the image content will change more slowly, so a short sequence of images may not provide much additional information. For a lower sampling rate, the change between images will be faster, which leads to a different optimal sequence size.

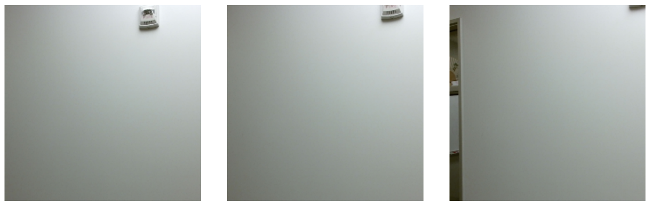
In order to better understand the correlation between the image sampling rate and the image sequence size used in our model, we performed a sensitivity analysis on the indoor dataset. We trained and tested ContextualNet with different image sequence sizes used in the LSTM layers (seq. size = $3, 5, 8, 12$), while the image sampling rate was kept at 25 and 8 Hz. The median error for estimated position for all the combinations is listed in Table II. The analysis suggests that with higher frequency, a larger sequence size should be selected for the LSTM layer. On the contrary, when the sampling rate is reduced from 25 Hz to 8 Hz, smaller sequences are sufficient for the LSTM to get a better estimation. The results provided by the sensitivity analysis are in line with our hypothesis that when the sampling rate is high, the difference between consecutive images will be low and hence a larger sequence size is required for the LSTM to exploit the contextual data and vice versa.

D. Real-time Testing

Our model consists of 29 layers in total. The first 27 consist of various Convolutional, Maxpooling and dropout layers, while the remaining 2 are LSTM layers. The total model size is approximately 48 MB, which contains weights for all the layers. Using this configuration, the maximum rate for estimating the pose using the CPU on the robot was 1.1 Hz. This leads to inferior performance of the robot, specifically for tracking applications due to the lower sampling rate for pose estimation. By streaming the downsampled images over WiFi and running the prediction on the TitanX GPU server, the maximum estimation speed that can be achieved was in the range of $50-55$ Hz. Hence, performing estimation on the GPU server allowed the estimation sampling rate to match the data capture sampling rate of the kinect sensor (25 Hz in our case) which in-turn lead to a smoother trajectory estimation of the robot.

VI. Discussion & Conclusion

The results presented in Section V illustrate how our model exploits features between consecutive images by utilizing a sequence of images as the model input to estimate the position and orientation. Exploring contextual information of earlier images in a sequence becomes especially important when there are not many distinct features in the current image that can be used by the model for estimation. To explain this better, we used a sequence containing 3 images (shown in Figure 4), in ContextualNet and only a single image (shown in Figure 4c), in PoseNet. The position error for ContextualNet was 2.78 m, whereas the same for PoseNet was 17.09 m. The large error in PoseNet can primarily be accounted for by the fact that there were minimal features available in the image for PoseNet's pose estimation. This is further visible through the normalized visual saliency features extracted at the deepest layer of the CNN model, as shown in Figure 4d-4f. The number of representative features in the first two images, 4d and 4e, are minor, containing only a small object on the wall, while the third image, 4f, has no features. The higher then expected error in ContextualNet can also be explained by the lack of informative features in the entire image sequence. This can be improved (as is evident in the sensitivity analysis) by utilizing a larger sequence size in the LSTM layers, as it will capture more contextual information from the longer historical image sequence.


(a) Image 1 $(t-2)$ (b) Image 2 $(t-1)$ (c) Image 3 (t)

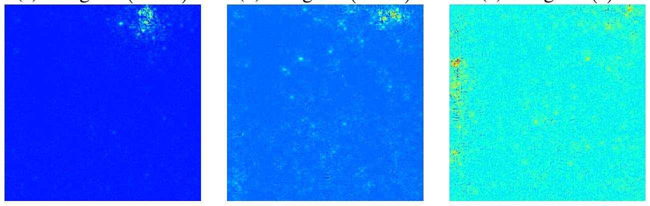

(d) Feature Image 1 (e) Feature Image 2 (f) Feature Image 3

Fig. 4: Image 1 - 3 are the sequence of images used by ContextualNet to take historical image features into consideration, whereas Image 3 was solely used by PoseNet to estimate the current position and orientation. Feature image 1 - 3 represents the normalized visual saliency features extracted by deepest CNN Layer of ContextualNet directly preceding the LSTM layers.

In this paper, we present a CNN-LSTM framework used for robot localization. Our model combines CNN-LSTM layers which exploits spatio-temporal features from image sequences to estimate the current position and orientation of the robot. Furthermore, we demonstrate how historical image features are utilized in situations where the current image does not have many representative features. We also performed a sensitivity analysis to understand the co-relation between the sampling rate and sequence size used in the LSTM layer. Our analysis suggests that for a higher sampling rate, a larger sequence size should be used in the LSTM layer so that the contextual relationship in the images can better be exploited by the model.

VII. Future Work

We plan to extend this work in three different areas. Firstly, we would experiment with depth data along with RGB images to estimate the pose of the robot. Depth data contains more information related to the features present in the surrounding environment which can be useful for further optimization of the model. Secondly, to quantitatively compare the performance of the ContextualNet model with the BranchNet model, proposed by Wu *et. al.* [10], and the Directional-PoseNet model, proposed by Walch *et. al.* [9]. We plan to replicate their proposed models and data sampling techniques and test them using the indoor dataset. Lastly, we plan to release the indoor dataset to the research community.

Acknowledgment

The authors would like to thank Tony Dunnigan and John Doherty from FX Palo Alto Laboratories Inc. for designing the robot platform and sensor placement rigs which was used for both data collection and testing in real-time.

References

[1] A. Kendall and R. Cipolla, "Modelling uncertainty in deep learning for camera relocalization," *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2016.

[2] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time," in *IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 963–968.

[3] R. Benenson, M. Omran, J. Hosang, and B. Schiele, *Ten Years of Pedestrian Detection, What Have We Learned?* Workshops on Computer Vision, 2015.

[4] T. Lupton and S. Sukkarieh, "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions," *Transaction on Robotics*, vol. 28, no. 1, pp. 61–76, 2012.

[5] I. M. Zendjebil, F. Ababsa, J.-Y. Didier, and M. Mallem, "On the hybrid aid-localization for outdoor augmented reality applications," in *2008 ACM Symposium on Virtual Reality Software and Technology*, ser. VRST '08. ACM, 2008, pp. 249–250.

[6] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

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

[8] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[9] F. Walch, C. Hazirbas, L. Leal-Taix, T. Sattler, S. Hilsenbeck, and D. Cremers, "Image-based localization using lstms for structured feature correlation," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[10] J. Wu, L. Ma, and X. Hu, "Delving deeper into convolutional neural networks for camera relocalization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5644–5651.

[11] 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 *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2930–2937.

[12] J. Valentin, M. Nießner, J. Shotton, A. Fitzgibbon, S. Izadi, and P. H. S. Torr, "Exploiting uncertainty in regression forests for accurate camera relocalization," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[13] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2007, pp. 225–234.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[15] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, *FuseNet: Incorporating Depth into Semantic Segmentation via Fusion-Based CNN Architecture*. Springer International Publishing, pp. 213–228.

[16] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[17] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[18] T. Weyand, I. Kostrikov, and J. Philbin, "Planet - photo geolocation with convolutional neural networks," in *European Conference on Computer Vision (ECCV)*, 2016.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computing*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[20] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Computer Vision and Pattern Recognition*, 2015.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[23] T. Zhang, "Solving large scale linear prediction problems using stochastic gradient descent algorithms," in *Proceedings of the Twenty-first International Conference on Machine Learning*, New York, NY, USA, 2004, pp. 116–.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference for Learning Representations*, 2014.

[25] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *International Conference on Neural Information Processing Systems*, 2014, pp. 487–495.

[26] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, 2011.

[27] 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, 2009, p. 5.

[28] C. Wu, "Towards linear-time incremental structure from motion," in *Proceedings of the International Conference on 3D Vision*, 2013, pp. 127–134.

[29] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[30] "Evaluating AAL systems through competitive benchmarking," http://evaal.aaloa.org/2016/competition-results, accessed: 2017-Sept-09.