Recurrent Neural Networks for Range-only SLAM

Hyungtae Lim[1] , Junseok Lee[1], Changgyu Park[1], Ye Eun Kim[1],

***Abstract*— Range-only SLAM is a method for localizing a mobile robot and beacons by mainly utilizing distance measurements. Unlike the traditional probability-based range-only SLAM method, we present a novel approach using a recurrent neural network architecture that directly learns the end-to-end mapping between distance data and robot position.**

I. INTRODUCTION

Trilateration is a conventional algorithm for locating a vehicle in the metropolitan area by range measurements between the vehicle and fixed beacon sensors. [1]. Due to the convenience of trilateration that estimates the position of a receiver of range sensors if one only knows range measurement, trilateration algorithm has been widely incorporated into robotics fields, especially utilized in the indoor environment to estimate the position of an object by distance measurements obtained from range sensors such as UWB, ultrasonic, laser-based beacon sensors [2]–[4]. Specifically, range-only Simultaneous Localization and Mapping(RO-SLAM) methods are utilized popularly, which not only estimate the position of the receiver of range sensors, but also localize the position of range sensors regarded as features on a map, and studies have been conducted continuously in terms of probability-based approach [5]–[8].

In the meantime, as deep learning age has come [9], various kinds of deep neural architectures have been proposed for many tasks related to robotics field, such as detection [10]–[12], navigation [13], [14], pose estimation [15], and so on. Especially, recurrent neural networks (RNNs), originated from Natural Language Process(NLP) area [16], have been shown to achieve better performance in case of dealing with time variant information, thereby RNNs are widely utilized such as not only speech recognition, but also pose estimation and localization [15], [17]–[20].

In this paper, we propose a deep learning-based localization method by stacked bidirectional Long Short-Term Memory(stacked Bi-LSTM) for more accurate localization of the robot. Using deep learning, our structure directly learns the end-to-end mapping between range measurements and robot position. This operation non-linearly maps the relationship not only considering the long-range dependence of sequential distance data by the LSTM, but also using the correlation of the backward information and the forward information of the sequence of each time step by virtue of its bidirectional architecture.

[1]Hyungtae Lim, [1]Junseok Lee, [1]Changgyu Park, and [1]Ye Eun Kim are with the Urban Robotics Laboratory, Korea Advanced Institute of Science and Technology (KAIST) Daejeon, 34141, South Korea. {shapelim, ljs630, cpark, yeeunk}@kaist.ac.kr

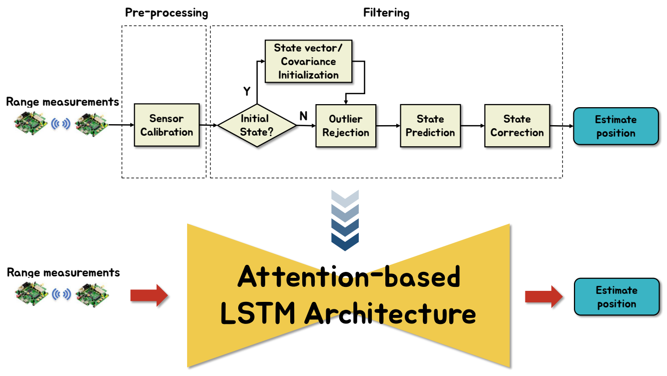


Fig. 1. System overview. A robot localizes its own pose through distance data and the derivative of distance data.

II. RELATED WORKS

In this section, we briefly survey previous researches closely focused on Long Short-Term Memory(LSTM) model and applications of LSTMs to solve domain problems.

1) LSTM: LSTM is a type of Recurrent Neural Networks(RNNs) that has loops so that infer output based on not only the input data, but also the internal state formed by previous information. In other words, while the RNN deals with sequential data, the network has remembered the previous state generated by past inputs and might be able to output the present time step via internal state and input, which is very similar to filtering algorithms.

However, RNNs often have a *vanishing gradient problem*,i.e., RNNs fail to propagate the previous matter into present tasks as time step gap grows by. In other words, RNNs are not able to learn to store appropriate internal states and operate on long-term trends. That is the reason why the Long Short-Term Memory (LSTM) architecture was introduced to solve this long-term dependency problem and make the networks possible to learn longer-term contextual understandings [21]. By virtue of the LSTM architecture that has memory gates and units that enable learning of long-term dependencies [22], LSTM are widely used in most of the deep learning research areas and numerous variations of LSTM architecutres have been studied.

2) Localization with Deep Learning: There have been many approaches combining Simultaneous Localization and Mapping (SLAM) with deep learning, aiming to overcome the limitations on SLAM only technique such as difficulty on tuning the proper parameters in different environments and recovering an exact scale. Actually, those researches are

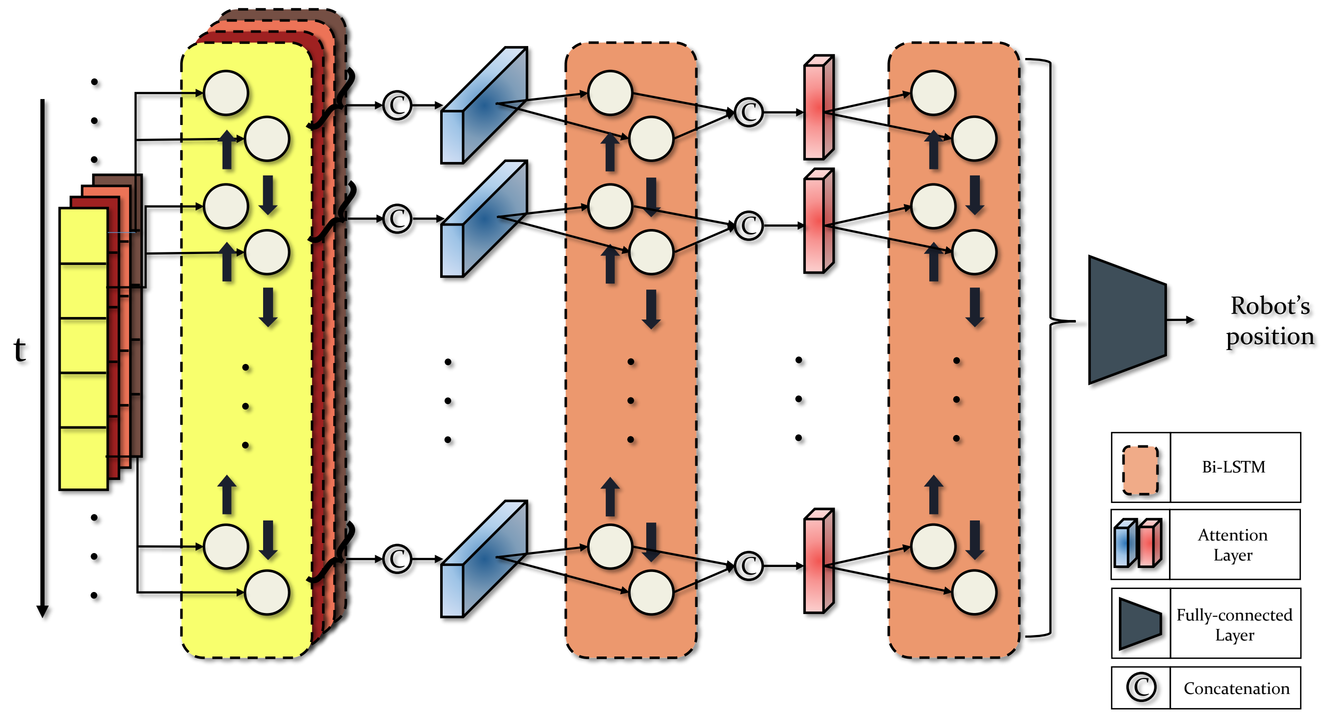


Fig. 2. Overall architecture of stacked Bi-LSTM.

showing the superior performance to the traditional SLAM approaches.

One of the popular SLAM techniques with deep learning is CNN-SLAM [23] which takes Convolutional Neural Networks (CNNs) to precisely predict the depth from a single image without any scene-based assumptions or geometric constraints, allowing them to recover the absolute scale of the reconstruction. Another approach using deep learning for localization is Deep VO [24] In this method, Recurrent Convolutional Neural Networks (RCNNs) is utilized. Specifically, feature representation is learned by Convolutional Neural Networks and Sequential information and motion dynamics are obtained by deep Recurrent Neural Networks without using any module in the classic VO pipeline.

3) Applications of LSTMs: There are many variations of LSTM architecture. As studies of deep learning are getting popular, various modified architectures of LSTM have been proposed for many tasks in a wide area of science and engineering. Because LSTM is powerful when dealing with sequential data and infering output by using previous inputs, LSTM is utilized to estimate pose by being attached to the end part of deep learning architecture [18]–[20] as a stacked form of LSTM. In addition, LSTM takes many various data as input; LSTM is exploited for sequential modeling using LiDAR scan data [17], images [15], [18], IMU [25], a fusion of IMU and images [24].

III. Our approaches

In this chapter, we introduce our proposed neural network model which is used for estimating the robot's pose and Landmarks' position when only the range sensor data is given from each distance sensor. Firstly, the overall network architecture will be provided. Then, the details of each part will be explained.

A. Network Architectures

As it is illustrated in Fig. 2, our proposed stacked Bi-LSTM can be decomposed into 3 parts: (1) Input part which accepts and preprocess the sequences of the sensor with multiple bidirectional LSTM layers. (2) Hidden layer part consisting of attention modules and bidirectional LSTM layer (3) Output part where a fully-connected output layer gives Robot's pose and position of landmarks as its results.

B. Multimodal LSTM

Since to effectively accept the input collected from the multiple sensors, instead of just using a single layer as an input layer, we use the number of LSTM layers, thinking that each single sensor represents a different modality. Each layer corresponds to the input of each distance sensor. In other words, if N sensors are used for measuring the distance of the robot, the number of input layer also would be N and the MTh layer accepts an input from the MTh sensor. By doing so, we can further expect that the input layers act as the sensor calibration process in traditional RO-SLAM, allowing the sensors to be tuned respectively with input layer's parameters.

C. Bidirectional LSTM

As traditional RO-SLAM [5], [6] takes an odometry which is an accumulated data from the beginning to the present point, our network takes the sensor data for the time period l. So, if the current time stamp is t, the input layers take the sensor data obtained from timestamp t-l to t. For dealing with such sequential information, we apply LSTM network with l cells which is one of the most appropriate network for the sequential data to our network. Furthermore, to take an advantage from the bidirectional time flow, normal time order and reverse time order, we design our network with 3 bidirectional LSTM layers that consist of 2 independent LSTMs corresponding to normal and reverse time order. Individual LSTM layers play a different role. LSTM layers of input part take and preprocess the sequence of sensor data. LSTM layer placed between input and output layer takes a spatial information from a previous spatial attention layer and send it to another temporal Attention layer. Lastly, LSTM layer of output layer outputs the results.

D. *Attention layer*

To precisely estimate the Robot's pose and landmarks' position, it is important for the network to distinguish which is more meaningful information and which is less for estimation and prevent to focus on less significant information. So, we add the two different types of attention modules [26] which extract something more important and related to the task information. The first attention modules placed between the input LSTM layer and the second LSTM layer are called "Spatial Attention layer" and is represented as blue blocks in Fig. 2 These attention modules judge which sensor is more informative. The second attention module corresponding to the red blocks in Fig. 2 is the temporal attention module. These temporal attention modules determine which time stamp which has more useful information allowing the network to attend that time stamp more.

E. *Stacked Architecture*

In deep learning, the number of layers stacked is getting large, intending to increase the non-linearity and correspondingly to improve the performance. Likewise, the multiple LSTM layers can be stacked as well [27], enabling more complex representation and higher performance. In stacked Bi-LSTM, total 3 LSTM layers are stacked in series.

F. *Multimodal Architecture*

G. *Training Loss*

We set the experiment on the virtual situation and generate distance data set which corresponds to the position with 10% noise error and let RNN be trained using these distance data. Train data are just zigzag paths and test data is an arbitrary path, so we also check if RNN can estimate the position despite the variation of distance data as input.

Let Θ be the parameters of our RNN model, then our final goal is to find optimal parameters Θ^* for localization by minimizing Mean Square Error (MSE) of Euclidean distance between ground truth position Y_k and estimated position $\hat{Y}_k$.

$$\Theta^* = \underset{\Theta}{\operatorname{argmin}} \sum_{k=1}^{N} \parallel Y_k - \hat{Y_k} \parallel^2 \tag{1}$$

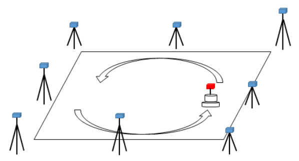
IV. Experiment

A. *Experiment settings*

Our experimental system consists of a UWB(ultra wideband) sensor tag and eight UWB sensor anchors that have a UWB transceiver, the motion capture system with 12 cameras, and a mobile robot and a small form-factor computer. The UWB tag and anchors are attached to a robot and landmarks respectively. The tag and anchor system operates like that an anchor transmits the Ultra wideband signal and a tag receives the signal and measures the range between two devices. Each UWB sensors have a UWB transceiver that is DW1000 UWB-chip made by Decawave and supports 6 RF bands from 3.5 GHz to 6.5 GHz and has centimeter-level accuracy. And The motion capture system is Eagle Digital Realtime system of motion analysis corporation that operates with the principle of stereo pattern recognition that is a kind of photogrammetry based on the epipolar geometry and the triangulation methodology. the system has $<$ 1mm accuracy and $>$ 500 frames/s frame rate. And a mobile robot is iClebo Kobuki from yujinrobot that has 70 cm/s maximum velocity. And the small form-factor computer is a gigabyte Ultra compact PC kit that CPU is intel dual core i7 / 2.7GHz and ram is DDR4SDRAM. Deep learning framework used for our network is pytorch 0.4.0 on python 3.6. The network is trained on the machine that OS is Ubuntu mate 16.04 LTS and GPUs are gtx 1080ti and gtx titan. The network inferences on the same machine that we used for training.

B. *Training/Test Dataset*

Fig. 3 shows the description of experimental environment. The UWB tag and a small computer are attached to mobile robot. The UWB anchors are attached to stands that have two different heights and positioned randomly. Inside of the square space, a mobile robot goes on various random paths. And the distance data is measured by the UWB tag and the global position data is measured by the motion capture system. In the computer two different thread receive these two kinds of data separately. So, to synchronize these data, we make an independent thread that concatenates and saves these data and the thread is running at 20Hz frequency shown in Fig. 3. After the experiment, we separate the entire data to two types of dataset, some are the training datasets and others are test datasets. Each type of datasets is independent of each other.


(a) Description.

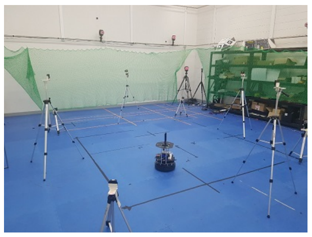

(b) Actual.

Fig. 3. Experimental system overview.

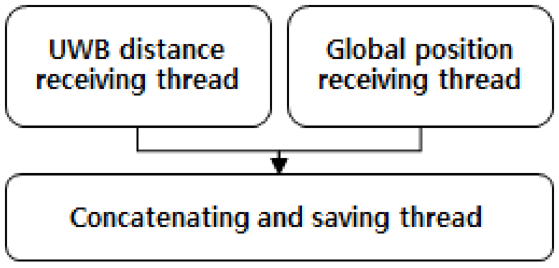
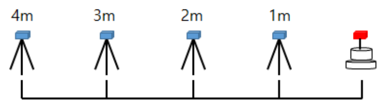

UWB distance receiving thread
Global position receiving thread
Concatenating and saving thread

Fig. 4. Data syncronizing method.

In addition, to use the distance data for traditional RO-slam we calibrate the distance from each anchors. As you can see in Fig. 4, we measure the data from a tag to each anchors at the points where the actual distance was measured by 1m, 2m, 3m, 4m. By using the linear regression, we compute the ratio between the measurement and the actual distance. And the ratios of each anchor are used to calibrate it.



(a) Description.

(b) Actual

Fig. 5. Sensor calibration overview.

V. Results

To verify our proposal that RNNs can estimate the robot's position through varying range data, we trained our RNN-based multimodal architecture. Plus, to compare to previous traditional SLAM algorithm, we also estimate robot's position by particle filter(PF) based algorithm. Train data are our own data gathered by UWB sensors and motion capture camera, so neural networks take range-only measurements as input and outputs robot's position. Ground truth data is robot's position measured by eagle eye motion capturer, whose error is in mm units. The results of trajectory prediction are shown in Fig. 6 and Root-Mean-Squared Error (RMSE) are shown in Table **??**.

A. Analysis of errors

We set two test trajectory cases. However, unexpectedly, Performance of PF based localization is better than performance of our architecture. In case of GRU, it has only two gates which is less complex structure than LSTM [28]. However, due to GRU's less complexity, GRU has less the number of neurons than LSTM so their non-linear mapping achieves less performance. Likewise, Bi-LSTM consists of two LSTMs to process sequence in two directions so that infer output using the correlation of the backward information and the forward information of the sequences of each time step with its two separate hidden layers. Thus, Bi-LSTM has better nonlinear mapping capability than LSTM. For similar reasons, stacked Bi-LSTM is the architecture that stacks two Bi-LSTMs, so inference performance is better than Bi-LSTM. As a result, the stacked Bi-LSTM showed the best performance among unit RNN architectures. Therefore, we can conclude that the performance improves as the non-linearity of the architecture increases.

Results of RMSE[cm]		
Model	Test1	Test2
Particle filter-based	**9.1827**	9.8803
Bidirectional Multimodal	11.3301	**9.7528**

VI. Conclusion

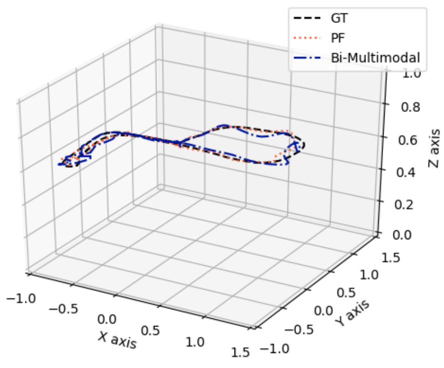
In this paper, we proposed a novel approach to range-only measurements localization using recurrent neural network models and tested various types of LSTM models for more accurate localization of the mobile robot.

Using deep learning, our structure directly learns the end-to-end mapping between distance data and robot position. The stacked bidirectional LSTM structure exhibits the best estimates of robot positions than other RNN structure units. Therefore, we conclude that the LSTM-based structure improves performance as non-linearity of structures increase and even if the robot position is not included in the ground truth dataset, our method is able to predict robot positions with small errors through sequential distance data.

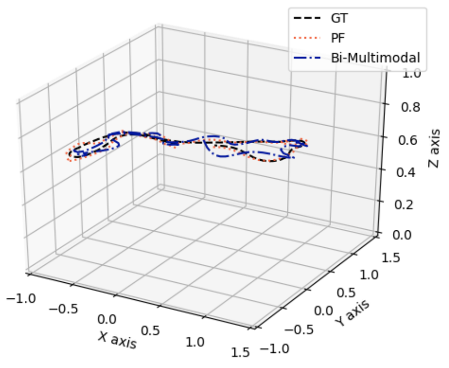
As a future work, because train/test dataset are generated on simulated environment, the proposed method needs to be tested in the real-world to check whether RNNs can deal with multipath problems and line of non-line of sight(NLOS) issues.

References

[1] H. Staras and S. Honickman, "The accuracy of vehicle location by trilateration in a dense urban environment," *IEEE Transactions on Vehicular Technology*, vol. 21, no. 1, pp. 38–43, 1972.
[2] F. Thomas and L. Ros, "Revisiting trilateration for robot localization," *IEEE Transactions on robotics*, vol. 21, no. 1, pp. 93–101, 2005.
[3] H. Cho and S. W. Kim, "Mobile robot localization using biased chirp-spread-spectrum ranging," *IEEE transactions on industrial electronics*, vol. 57, no. 8, pp. 2826–2835, 2010.
[4] A. N. Raghavan, H. Ananthapadmanaban, M. S. Sivamurugan, and B. Ravindran, "Accurate mobile robot localization in indoor environments using bluetooth," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 4391–4396.
[5] J.-L. Blanco, J. González, and J.-A. Fernández-Madrigal, "A pure probabilistic approach to range-only slam." in *ICRA*. Citeseer, 2008, pp. 1436–1441.
[6] J.-L. Blanco, J.-A. Fernández-Madrigal, and J. González, "Efficient probabilistic range-only slam," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, 2008, pp. 1017–1022.
[7] F. R. Fabresse, F. Caballero, I. Maza, and A. Ollero, "Undelayed 3d ro-slam based on gaussian-mixture and reduced spherical parametrization," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. Citeseer, 2013, pp. 1555–1561.
[8] N. S. Shetty, "Particle filter approach to overcome multipath propagation error in slam indoor applications," Ph.D. dissertation, The University of North Carolina at Charlotte, 2018.
[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

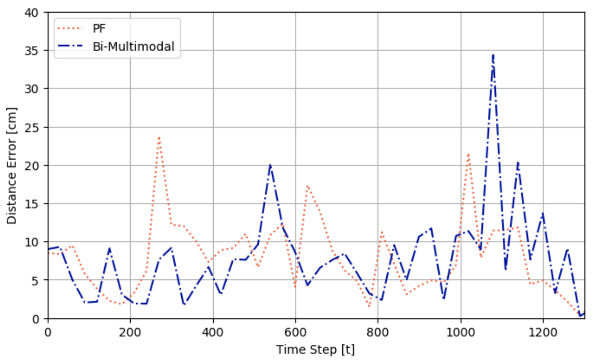


(a) Description.

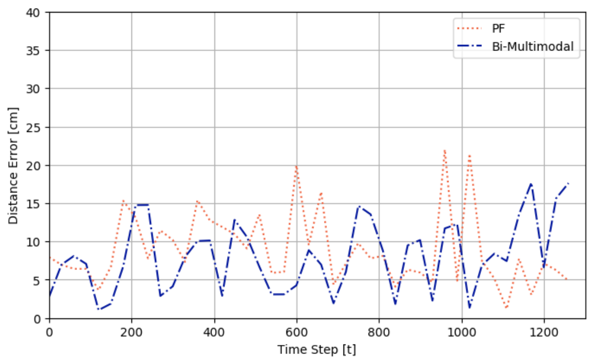


(b) Actual

Fig. 6. Sensor calibration overview.



(a) Description.



(b) Actual

Fig. 7. Sensor calibration overview.

[10] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 705–724, 2015.

[11] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in *European Conference on Computer Vision*. Springer, 2016, pp. 354–370.

[12] H. H. Smith, "Object detection and distance estimation using deep learning algorithms for autonomous robotic navigation," 2018.

[13] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 3357–3364.

[14] M. Hamandi, M. D'Arcy, and P. Fazli, "Deepmotion: Learning to navigate like humans," *arXiv preprint arXiv:1803.03719*, 2018.

[15] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers, "Image-based localization using lstms for structured feature correlation," in *Int. Conf. Comput. Vis.(ICCV)*, 2017, pp. 627–637.

[16] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[17] S. Gladh, M. Danelljan, F. S. Khan, and M. Felsberg, "Deep motion features for visual tracking," in *Pattern Recognition (ICPR), 2016 23rd International Conference on*. IEEE, 2016, pp. 1243–1248.

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

[19] 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.

[20] M. Turan, Y. Almalioglu, H. Araujo, E. Konukoglu, and M. Sitti, "Deep endovo: A recurrent convolutional neural network (rcnn) based visual odometry approach for endoscopic capsule robots," *Neurocomputing*, vol. 275, pp. 1861–1870, 2018.

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

[22] W. Zaremba and I. Sutskever, "Learning to execute," *arXiv preprint arXiv:1410.4615*, 2014.

[23] K. Tateno, F. Tombari, I. Laina, and N. Navab, "Cnn-slam: Real-time dense monocular slam with learned depth prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2017.

[24] R. Clark, S. Wang, H. Wen, A. Markham, and N. Trigoni, "Vinet: Visual-inertial odometry as a sequence-to-sequence learning problem." in *AAAI*, 2017, pp. 3995–4001.

[25] F. J. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016.

[26] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[27] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, and N. A. Smith, "Transition-based dependency parsing with stack long short-term memory," *arXiv preprint arXiv:1505.08075*, 2015.

[28] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation

of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.