A Deep Learning Framework for Robust Semantic SLAM

Rana Azzam, Tarek Taha, Shoudong Huang, and Yahya Zweiri

***Abstract*—Semantic simultaneous localization and mapping (SLAM) is susceptible to several sources of noise that hinder the accuracy of its trajectory and map estimates. Such sources include inaccurate landmark pose estimation and sensor limitations. In this paper, a novel deep learning based approach is proposed to improve the accuracy of semantic SLAM by reducing the trajectory estimation error. A deep neural network consisting of various non-linear activation functions is structured and pre-trained by means of an unsupervised greedy layer-wise pre-training technique. The network is then fine-tuned using the adaptive moment estimation (ADAM) optimizer. The training datasets were collected using several simulated and real-time experiments and are composed of two parts, the estimated trajectory and the corresponding ground truth. The effectiveness of the proposed approach was shown through simulated experiments, real-time experiments, and a sequence from the TUM RGB-D dataset. The performance of the DNN was tested with different pre-training techniques and the proposed unsupervised greedy layer-wise pre-training technique proved to perform the best across training, validation, and testing datasets in terms of reducing the mean absolute trajectory error (ATE).**

***Index Terms*—Deep Neural Network, Semantic SLAM, Estimation Error**

I. Introduction

Deep neural networks (DNN) are trained to perform a particular task using massive amounts of data. The training process adjusts the weight-space of the network through several epochs to ensure that the network's predictions resemble the target data [1]. However, arriving at the optimal weight-space after the DNN training is extremely challenging. The performance of the gradient descent approaches during training depends on the initial weights assigned to the network [2]. Consequently, the research activity in the field of DNNs remained dormant until the proposal of the greedy layer-wise pre-training technique in 2006 [2], [3], where a restricted Boltzman machine was used to train a deep belief network. Afterwards, a stacked auto-encoder (SAE) architecture was proposed in [4] to replace the restricted Boltzman machine in an analogous pre-training approach.

Greedy layer-wise pre-training initializes the weights of a neural network to values in the neighborhood of a local minimum, and hence aids the optimization process and induces better model generalization [4]. The term *greedy* refers to pre-training every layer individually, irrespective of subsequent layers in the network [4]. Accordingly, each layer is fed with data at a different abstraction level, and hence, learns to represent the data in a distinct manner [3]. Ultimately, the entire model is fine-tuned altogether by means of a gradient descent method [4]. Ever since then, DNNs have been witnessed attaining state-of-the-art performance in a wide range of applications, such as computer vision [5] and robotics [6].

When a robot maneuvers in an environment while perceiving information through sensors mounted on it, simultaneous localization and mapping (SLAM) can be performed to estimate the robot's trajectory and the map of the environment. SLAM has been heavily studied in the past three decades and a wide range of solutions have been proposed, like [7], [8], [9]. Deep learning has been involved in the development of solutions to SLAM in several ways [10] as can be seen in [11], [12], [13], [14], [15]. Not only was it able to replace techniques in the classical SLAM pipeline, but it motivated the development of semantic SLAM [16], [17] which exploits understanding of the surrounding environment [18], [19], [20] to perform trajectory and map estimation. The quality of the obtained estimates is affected by several factors, such as the quality of landmark detection and localization [21], sensor noise [22], [23], and occlusions [24]. Developing an error model that takes into account all the factors contributing to measurement inaccuracies is considered very challenging and hence, estimating a highly accurate trajectory and map is difficult.

Enhancing state estimation using deep learning approaches can be found in very few research work. For instance, the work presented in [25] improves the accuracy of wheel odometry using a shallow neural network. In [6], sequential learning was utilized to enhance the robustness of visual odometry. On a different note, a deep neural network was proposed in [15] to better estimate the relative poses between consecutive images and hence improve visual localization.

In this paper, we propose a deep learning based approach that can be used to minimize the semantic SLAM estimation error by identifying different noise patterns and trying to reduce them. This is done through a deep neural network that is pre-trained using the unsupervised greedy layer-wise pre-training technique then fine-tuned using the adaptive moments (Adam) optimizer. The training datasets are comprised of semantic SLAM trajectory estimations and the corresponding ground truth which are obtained from simulated and real-time experiments. In summary, the contributions of this paper are:

- Developing a deep learning based approach for minimizing semantic SLAM estimation errors.
- Pre-training the developed DNN using unsupervised greedy layer-wise training and demonstrating its superiority over other pre-training approaches.

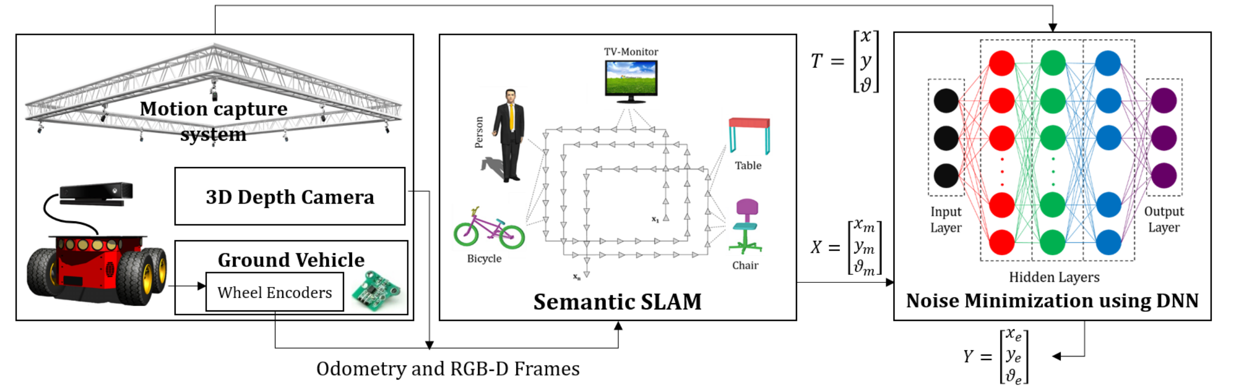

Motion capture system
3D Depth Camera
Ground Vehicle
Wheel Encoders
TV-Monitor
Person
Table
Bicycle
Chair
Semantic SLAM
$T = \begin{bmatrix} x \\ y \\ \vartheta \end{bmatrix}$
$X = \begin{bmatrix} x_m \\ y_m \\ \vartheta_m \end{bmatrix}$
Input Layer
Output Layer
Hidden Layers
Noise Minimization using DNN
$Y = \begin{bmatrix} x_e \\ y_e \\ \vartheta_e \end{bmatrix}$
Odometry and RGB-D Frames

Fig. 1: Semantic SLAM with deep learning based noise minimization

- Fine-tuning the DNN to generalize well to previously unseen data.
- Validation of the proposed filtration approach using datasets obtained from simulation, real-time experiments, and publicly available domains.

II. Proposed Approach

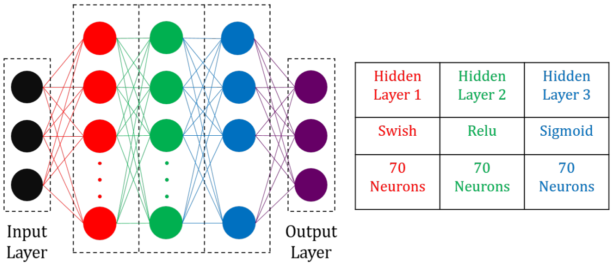
Figure 1 shows a general block diagram of the approach proposed in this paper. A wheeled vehicle navigates in an unknown GPS-denied environment and records its odometry and visual measurements, obtained through an RGB-D sensor. A motion capture system is used to obtain the ground truth of the vehicle's trajectory. Using odometry and visual measurements, a semantic SLAM system is run to estimate the trajectory of the vehicle. The estimations are then passed to the proposed DNN for further refinement and error minimization. A description of the semantic SLAM block and the DNN are provided in the next sections.

A. Semantic SLAM

Using the measurements recorded by the robotic vehicle in the environment, semantic SLAM estimates the trajectory and the map of the environment using factor graph optimization. Odometry is used to construct motion constraints which are relative poses of the robot in the environment. In parallel, RGB-D frames are processed to compute measurement constraints. RGB frames are passed to you only look once (YOLO) object detector [18] that detects the label and the bounding box of objects in the image. The corresponding depth frame is then converted to point cloud, which is clustered to extract the points that belong to the detected object. The geometric centroid of the object segment is used to compute the relative position between the object and the vehicle. Since multiple objects of the same category might be present in the environment, a distance threshold is employed to associate each observation to a landmark in the map.

B. DNN-based Estimation Error Minimization

In this section, the architecture of the proposed DNN, including its depth, the size of each layer, and its activation functions are presented. Then, the adopted pre-training and fine-tuning techniques are demonstrated.



Hidden Layer 1	Hidden Layer 2	Hidden Layer 3
Swish	Relu	Sigmoid
70 Neurons	70 Neurons	70 Neurons

Fig. 2: Proposed neural network architecture

1) Proposed DNN Architecture: The architecture of the adopted DNN, shown in Figure 2, exhibited the highest performance among many other tested architectures with different depths, activation functions, and layer sizes. It consists of an input layer, three hidden layers, and an output layer. The input layer takes in three inputs, the 2D position and associated orientation of the robotic vehicle represented as $X = [x_m, y_m, \vartheta_m]^T$. The output layer layer has three neurons, and its activation function is sigmoid. The ground truth trajectory of the robot, referred to as $T = [x, y, \vartheta]^T$ is provided to the network as a target and is recorded using a motion capture system in real-time experiments.

The output of the network is an improved estimate of the 2D position and orientation of the robot and is referred to as $Y = [x_e, y_e, \vartheta_e]^T$. The three hidden layers are of the same size, where 70 neurons reside in each layer. Employing hidden layers of the same size was recommended in [3] due to its convenience when using the pre-training technique that will be presented shortly. As for the activation functions, Swish [26] was chosen to activate the first hidden layer, the rectified linear unit (ReLU) to activate the second hidden layer, and sigmoid to activate the last hidden layer.

2) DNN Pre-training: Since the performance of a neural network heavily relies on the assigned weights [27], it is essential to guarantee that the initial weights are in the neighborhood of a good estimate so as to ensure that the

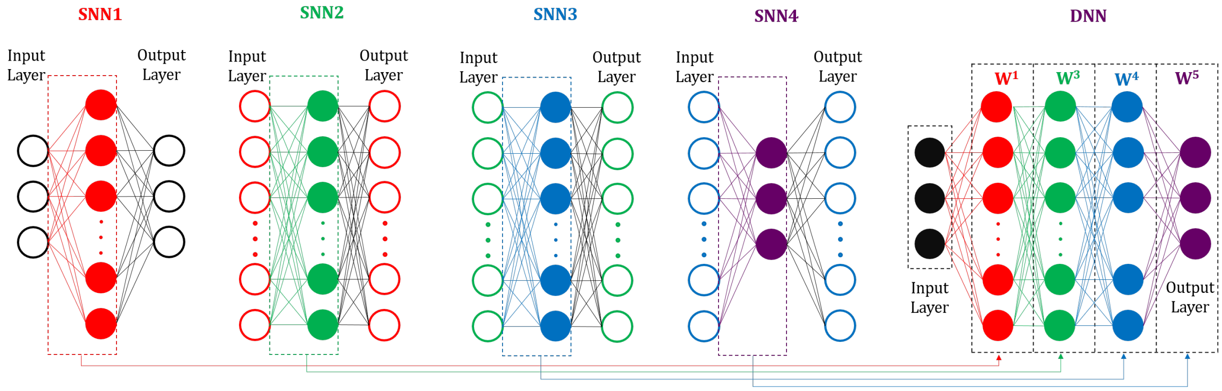


Fig. 3: Neural network unsupervised greedy layer-wise pre-training (or SAE) approach

employed gradient descent method works well during training [2], [4]. To that end, a pre-training technique is adopted in the proposed approach to initialize the network's weights.

The adopted unsupervised greedy layer-wise pre-training technique, which is also known as SAE, is depicted in Figure 3. The proposed DNN is pre-trained in four stages, where the DNN's non-input layers are trained sequentially, starting from the first hidden layer onward. Each layer is individually trained using an SNN, which is also referred to as an auto-encoder, of suitable input and output layers. More specifically, the input and output layers of the SNN used to train layer (p) should be equal in dimension to layer $(p-1)$ in the DNN. The size of its hidden layer is the size of layer (p) in the DNN. For example, to train the first hidden layer in the proposed DNN, the employed SNN has three inputs, 70 neurons in the hidden layer, and three outputs. The architectures of the remaining SNNs are illustrated in Figure 3. Each SNN takes the output of the preceding layer in the DNN as its input and output. The inputs and outputs of the first SNN, however, are the same as the inputs to the DNN. The weights and biases resulting from training each SNN are assigned as initial weights to the corresponding layers in the DNN in the fine-tuning stage as described in the next section. At this point, the network's weights are initialized.

3) DNN Fine-tuning: Although pre-training better initializes the weights of the DNN, it suffers from sub-optimality, especially for the weights in the first hidden layers [3]. Therefore, global network fine-tuning is carried out to replace stochastic weights, resulting from unsupervised learning in the pre-training process, with more deterministic ones by means of the backpropagation algorithm [2]. The optimization process in this case is much simpler compared to optimizing random initial weights and yields better generalization [4].

The DNN is trained by updating the weights for each entry in the dataset through several training epochs using backpropagation. Backpropoagation [28] is a widely used supervised learning technique for neural networks, where the differences between the DNN's predictions and the corresponding target outputs are employed to adjust the internal weights of the network.

III. Experimental Validation

In this section, the proposed approach is validated through a set of simulated and real-time experiments, including publicly available datasets. After preparing the datasets, an automated search for the optimal DNN structure was conducted by varying the number of layers, number of neurons per layer, the activation functions, and the initial random seeds. More than 2000 different DNN structures with three, four, and five hidden layers were trained and tested. The structure shown in Figure 2 demonstrated the highest accuracy among all other structures, where the mean absolute trajectory error (ATE) was the lowest. Comparisons between three alternatives for pre-training the adopted structure were then conducted, more specifically unsupervised greedy layer-wise pre-training, supervised greedy layer-wise pre-training, and backpropagation.

The rest of this section is organized as follows: The experimental set-up used to record the training dataset is presented in Section III-A. In Section III-B, the structure of the training datasets is described. Section III-C includes comparisons of different pre-training techniques used to initialize the network's weights, including unsupervised greedy layer-wise pre-training (or SAE), supervised greedy layer-wise pre-training, and backpropagation. Finally, the proposed approach is tested on a SLAM sequence from the TUM RGB-D dataset [29].

A. Experimental Set-up

An 80 m long trajectory of a ground vehicle was recorded in a simulated $10 \times 10\ m^2$ static environment that is populated with multiple instances of objects from three different categories; people, monitors, and bottles. The ground vehicle is equipped with wheel encoders and an RGB-D camera mounted on top.

For real-time experiments, a Pioneer 3AT robot maneuvered a trajectory of 60 m long in a $9 \times 6\ m^2$ previously unexplored environment, where a motion capture system was installed. Multiple instances of two different object categories, chairs and monitors, were placed in the environment.

B. Dataset Preparation

The dataset used for training the network is composed of data from both simulated and real-time experiments. It contained a total of 6751 samples, each consists of two parts; the estimated 2D position of the robotic vehicle and its orientation, which are computed by semantic SLAM, and the corresponding ground truth. The dataset was randomly split into three parts; 80% for training, 10% for validation to aid the model's regularization [27], and 10% for testing.

C. Network Pre-Training

In this section, a comparison between three network pre-training techniques is conducted. The architecture presented in Figure 2 was pre-trained using unsupervised greedy layer-wise pre-training, supervised greedy layer-wise pre-training, and backpropagation. In supervised greedy layer-wise pre-training, the network is pre-trained in four stages, followed by fine-tuning using the backpropagation algorithm. First, the first hidden layer is trained given the input and the target of the DNN. Then, the first hidden layer is frozen, the second hidden layer is added, and the new network is trained, given the DNN's input and target. The same is repeated to train the remaining layers, including the output layer. The DNN is finally fine-tuned using backpropagation. In the last approach, the weights of the network are randomly initialized and backpropagation is used to update them.

In all cases, the weights to any DNN or SNN were randomly initialized. To that end, and to fairly compare the pre-training approaches, each network was pre-trained and fine-tuned several times using different seeds, and the best results for each were used for comparison. The mean ATE was used to evaluate the trajectory filtered by any network. Figure 4 shows the distribution of error values for each run using the three different pre-training techniques.

Figure 5 depicts the trajectories estimated by the network when pre-trained using backpropagation, unsupervised and supervised greedy layer-wise pre-training approaches and Table I shows the corresponding minimum, maximum, and mean ATE. The input to the network in all cases is the trajectory estimated by semantic SLAM which is shown in Figure 5d. As can be noticed, the estimated trajectory by the network exhibits much higher accuracy than the path estimated by semantic SLAM in all three cases, where the mean ATE dropped to almost one third that of the input. However, the network that was pre-trained using the unsupervised greedy layer-wise pre-training showed superior performance compared to the other two. The resulting mean ATE was around 10 cm, compared to 11 cm for both backpropagation and supervised greedy layer-wise pre-training. Hence, for the targeted problem and the selected DNN architecture, unsupervised greedy layer-wise approach is the most suited for pre-training the DNN. This is attributed to initializing the weights of the DNN near a good solution and hence backpropagation will perform better compared to starting from a completely random solution.

The Pearson correlation coefficient was computed to evaluate the regression performance of the proposed DNN and it was close to 1 for all outputs. The model's convergence can be evidently shown in Figure 6, which shows the training accuracy of the DNN during the fine-tuning stage.

TABLE I: ATE comparison for tested approaches

Approach	**Max ATE (m)**	**Min ATE (m)**	**Mean ATE (m)**
Semantic SLAM	1.1179	0	0.3024
DNN - unsupervised pre-training	0.8288	8.44e-04	0.0998
DNN - supervised pre-training	0.6946	6.63e-04	0.1119
DNN - backpropagation	0.8334	0.0012	0.1149

D. Performance Analysis on Different Dataset

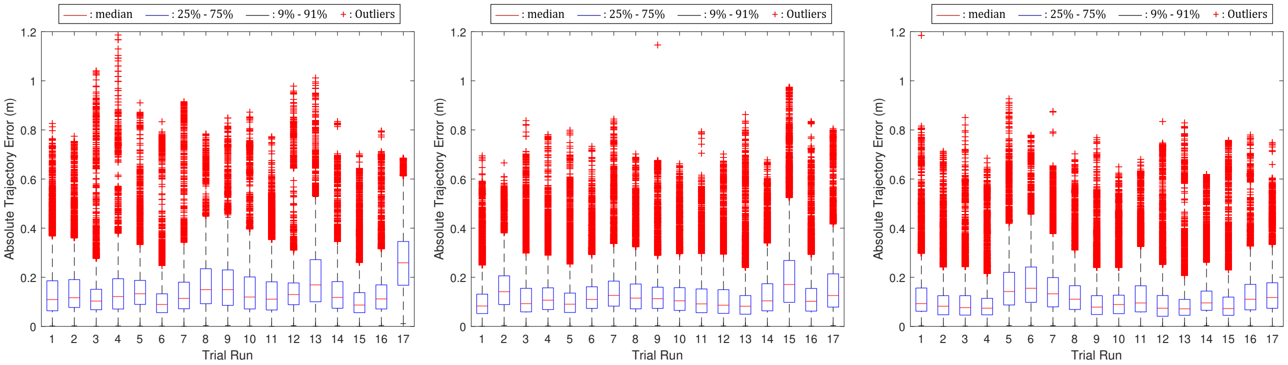
To confirm the generality of the model and its ability to enhance semantic SLAM estimations from previously unseen datasets, several simulated and real-time semantic SLAM experiments were conducted. In addition, the semantic SLAM algorithm was run on the fr2/pioneer_slam sequence from the TUM RGB-D dataset [29], where several objects appeared in the environment and were used as landmarks. The outcomes of such experiments were passed to the DNN for further refinement. For each dataset, the DNN was fine-tuned to achieve the sought performance. During fine-tuning, the first two hidden layers are frozen, while the last two are set to be trainable. Unlike trainable layers, the weights of a frozen layer do not change during fine-tuning. The data used for fine-tuning is a set of $10-20\%$ randomly selected samples from each dataset.

Figure 7 shows the results of filtering a set of trajectories computed by semantic SLAM during multiple simulated and real-time experiments. The first row includes the trajectories that were fed to the DNN while the filtered trajectories are depicted in the lower part of the same figure.

In all cases, the model was able to generalize well to new datasets where it was able to refine the estimations of different trajectories computed by semantic SLAM. The mean ATE decreased from 0.21 m to 0.03 m in experiment 1, from 0.11 m to 0.03 m in experiment 2, from 0.04 m to 0.01 m in experiment 3, and from 0.5 m to 0.07 m in experiment 4 which is a recorded sequence from the TUM RGB-D dataset.

IV. Conclusion

In this paper, a deep learning based approach to enhance semantic SLAM estimations was proposed. A DNN was structured using various nonlinear activation functions and pre-trained using the unsupervised greedy layer-wise pre-training approach. The adopted pre-training approach demonstrated higher performance than other approaches including the supervised greedy layer-wise pre-training approach and backpropagation. This is attributed to initializing the DNN's weights near a good solution, rather than a random one. The proposed DNN was trained using the ADAM optimizer, which is known to be resource-efficient, on a dataset that includes samples from simulated and real-time experiments. Combining data from both simulations and real-time experiments in the training dataset led to better model generalization when tested on simulated and real data. The proposed approach has proven work well in several simulated and real-time experiments including the publicly available fr2/pioneer_slam sequence from

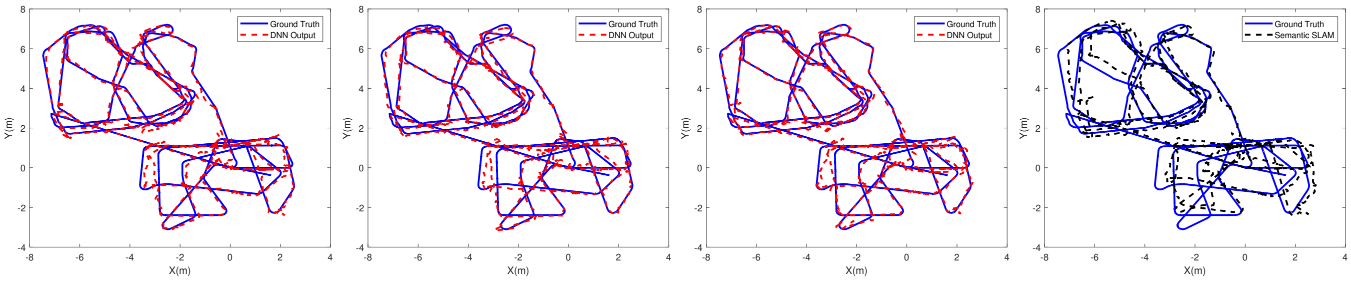


(a) Backpropagation pre-training

(b) Supervised pre-training

(c) Unsupervised pre-training

Fig. 4: ATE distribution of multiple training results. Blue boxes indicate values between the $25th$ and $75th$ percentiles, dotted black lines extend from the $9th$ to the $91st$ percentiles, and red whiskers represent the remaining values. The median of the error values in indicated as a horizontal line inside each box.



(a)

(b)

(c)

(d)

Fig. 5: Comparison between pre-training techniques, (a) Proposed DNN with backpropagatoin pre-training, (b) Proposed DNN with supervised pre-training, (c) Proposed DNN with unsupervised pre-training, (d) Semantic SLAM output

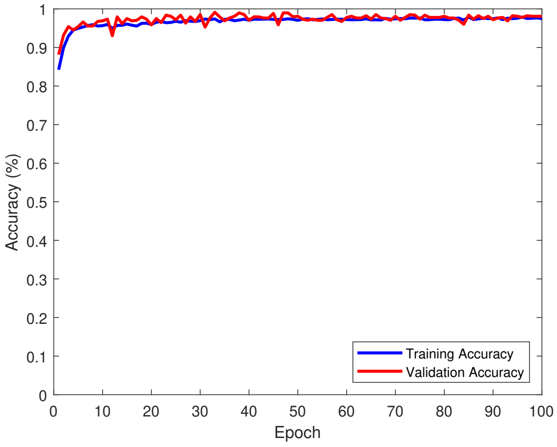


Fig. 6: Training accuracy across training and validation sets

the TUM RGB-D dataset and was able to notably improve the accuracy of the estimated robot trajectory, compared to semantic SLAM estimates.

In the future, the approach proposed in this paper can be integrated with semantic SLAM in an online manner which will facilitate error minimization in real time.

ACKNOWLEDGEMENTS

This publication is based upon research work supported by the Khalifa University of Science and Technology under Award No. RC1-2018-KUCARS

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. E. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. [Online]. Available: https://doi.org/10.1038/nature14539

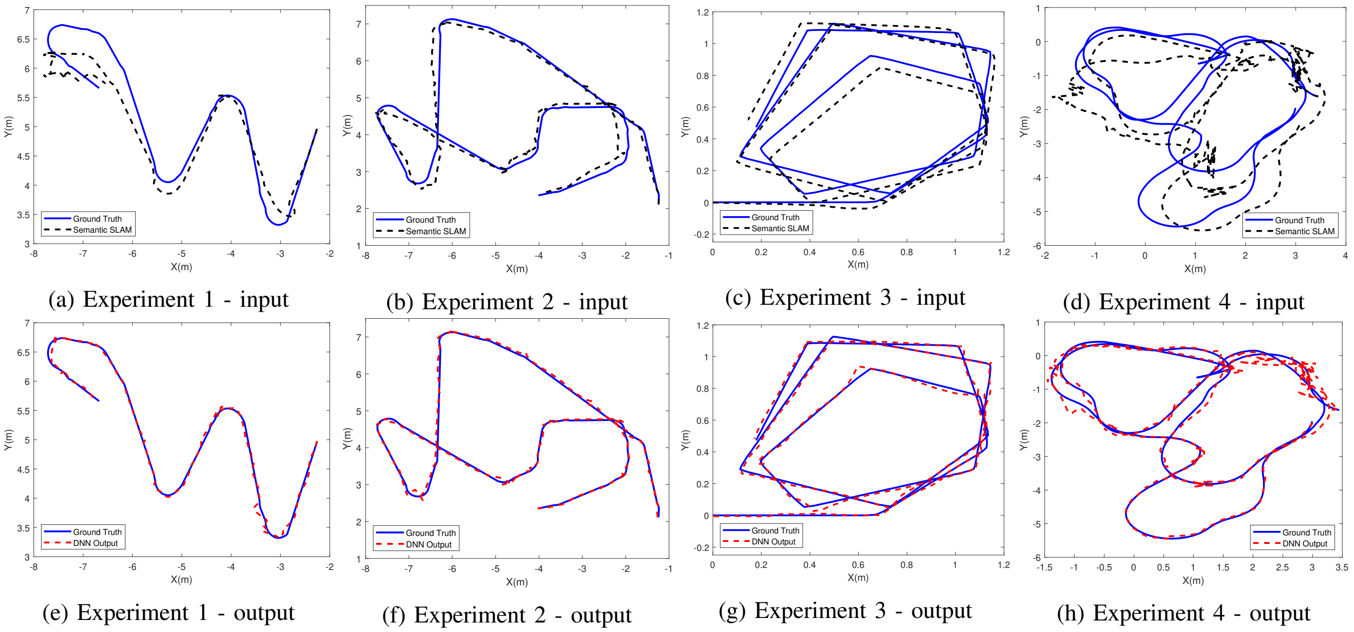
[2] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504 – 507, 2006.

[3] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006. [Online]. Available: http://dx.doi.org/10.1162/neco.2006.18.7.1527

[4] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proceedings of the 19th International Conference on Neural Information Processing Systems*, ser. NIPS'06. Cambridge, MA, USA: MIT Press, 2006, pp. 153–160. [Online]. Available: http://dl.acm.org/citation.cfm?id=2976456.2976476

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, p. 2012.

[6] S. Wang, R. Clark, H. Wen, and N. Trigoni, "End-to-end, sequence-to-sequence probabilistic visual odometry through deep neural networks," *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 513–542, 2018. [Online]. Available: https://doi.org/10.1177/0278364917734298



(a) Experiment 1 - input

(b) Experiment 2 - input

(c) Experiment 3 - input

(d) Experiment 4 - input

(e) Experiment 1 - output

(f) Experiment 2 - output

(g) Experiment 3 - output

(h) Experiment 4 - output

Fig. 7: Experimental validation of the proposed approach over different simulated and real-time datasets

[7] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, Oct 2017.

[8] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 834–849.

[9] R. Gomez-Ojeda, F. A. Moreno, D. Scaramuzza, and J. G. Jiménez, "PL-SLAM: a stereo SLAM system through the combination of points and line segments," *CoRR*, vol. abs/1705.09479, 2017. [Online]. Available: http://arxiv.org/abs/1705.09479

[10] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, Dec 2016.

[11] G. Costante, M. Mancini, P. Valigi, and T. A. Ciarfuglia, "Exploring representation learning with cnns for frame-to-frame ego-motion estimation," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 18–25, Jan 2016.

[12] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 2650–2658.

[13] C. Cadena, A. R. Dick, and I. D. Reid, "Multi-modal auto-encoders as joint estimators for robotics scene understanding," in *Robotics: Science and Systems*, 2016.

[14] A. Kendall, M. Grimes, and R. Cipolla, "Convolutional networks for real-time 6-dof camera relocalization," *CoRR*, vol. abs/1505.07427, 2015. [Online]. Available: http://arxiv.org/abs/1505.07427

[15] V. Peretroukhin and J. Kelly, "Dpc-net: Deep pose correction for visual localization," *CoRR*, vol. abs/1709.03128, 2017. [Online]. Available: http://arxiv.org/abs/1709.03128

[16] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, "Probabilistic data association for semantic slam," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 1722–1729.

[17] B. Mu, S. Y. Liu, L. Paull, J. Leonard, and J. P. How, "Slam with objects using a nonparametric pose graph," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 4602–4609.

[18] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *CoRR*, vol. abs/1506.02640, 2015. [Online]. Available: http://arxiv.org/abs/1506.02640

[19] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," *CoRR*, vol. abs/1512.02325, 2015. [Online]. Available: http://arxiv.org/abs/1512.02325

[20] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," *CoRR*, vol. abs/1809.10790, 2018. [Online]. Available: http://arxiv.org/abs/1809.10790

[21] N. Sünderhauf, O. Brock, W. J. Scheirer, R. Hadsell, D. Fox, J. Leitner, B. Upcroft, P. Abbeel, W. Burgard, M. Milford, and P. Corke, "The limits and potentials of deep learning for robotics," *CoRR*, vol. abs/1804.06557, 2018. [Online]. Available: http://arxiv.org/abs/1804.06557

[22] P. Ozog and R. M. Eustice, "On the importance of modeling camera calibration uncertainty in visual slam," in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 3777–3784.

[23] J.-H. Park, Y.-D. Shin, J.-H. Bae, and M.-H. Baeg, "Spatial uncertainty model for visual features using a kinect$^{\text{TM}}$ sensor," *Sensors*, vol. 12, no. 7, pp. 8640–8662, 2012. [Online]. Available: http://www.mdpi.com/1424-8220/12/7/8640

[24] X. Dong, J. Shen, D. Yu, W. Wang, J. Liu, and H. Huang, "Occlusion-aware real-time object tracking," *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 763–771, April 2017.

[25] J. Toledo, J. D. Piñeiro, R. Arnay, D. Acosta, and L. Acosta, "Improving odometric accuracy for an autonomous electric cart," *Sensors*, vol. 18, no. 1, 2018. [Online]. Available: http://www.mdpi.com/1424-8220/18/1/200

[26] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," *CoRR*, vol. abs/1710.05941, 2017. [Online]. Available: http://arxiv.org/abs/1710.05941

[27] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85 – 117, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608014002135

[28] J. Brownlee, *Machine Learning Algorithms from Scratch: With Python*. Jason Brownlee, 2017. [Online]. Available: https://books.google.ae/books?id=ZTq2tAEACAAJ

[29] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 573–580.