Neural Network-based Autonomous Navigation for a Homecare Mobile Robot

ByungSoo Ko, Ho-Jin Choi
School of Computing
KAIST, Daejeon, Korea
{kobiso, hojinc}@kaist.ac.kr

Chansol Hong, Jong-Hwan Kim
School of Electrical Engineering
KAIST, Daejeon, Korea
{cshong, johkim}@rit.kaist.ac.kr

Oh Chul Kwon, Chang D. Yoo
School of Electrical Engineering
(Division of Future Vehicle)
KAIST, Daejeon, Korea
{zelot11, cd_yoo}@kaist.ac.kr

***Abstract*—By the number of people aged 60 or over and people with disabilities growing, homecare mobile robot draws increasing attention. However, there are challenges of autonomous navigation for homecare robot such as frequent changes of environment, obstacles and goal position. In this paper, we focus on verifying potential of neural network-based autonomous navigation for homecare mobile. And we compare recurrent neural network with multilayer perceptron in the navigation of an autonomous mobile robot. The result suggested that the recurrent neural network can do better robot navigation because of its capability to handle the temporal dependency of a data sequence. Also, it shows that neural network-based navigation can be a good alternative since it has decent generalization ability for new environment, obstacles and goals.**

***Keywords*—neural network; autonomous navigation; homecare mobile robot**

I. Introduction

The number of people aged 60 or over is growing fast as medical technology have grown significantly and life expectancy rises. The population division of United nations predicted 21.59% of world population will be 60 or older by 2050 [1]. Since the population are ageing, the number of people with disabilities is increasing as well. There are more than a billion people having some form of disability and it is about 15% of the world population [2]. In order to help the elderly and disabled people to live safe and independent, home mobile robot was introduced and has been playing a big role.

One of the homecare mobile robots for social interaction is SocialRobot [3]. They proposed a SocialRobot which providing a practical interaction robotic solution to improve the quality of life of elderly people. The system it used is a mobile robot navigation with onboard novel micro-spectrometer system to recognize unexplored environment and unknown medicines. In [4], they proposed a mobile robot platform for socially assistive care applications that was developed under consideration of functionalities, user acceptance, and costs. Other homecare mobile robots include the intelligent Navigation and Micro-Spectrometer Content Inspection System [5], Linda [6] and PR2 [7].

Many of traditional homecare mobile robot focused on performing in well-known and structured environment. This can be limitation of homecare system since home environment can be changed easily by furniture movement and random obstacles. Also, the target position will change frequently and it can be a challenge for a mobile robot. It should navigate to the target position by identifying terrain and avoiding many obstacles. These problems are the major challenges of navigation system of homecare mobile robot.

Fig. 1. Example of homecare mobile robot: Care-O-bot.

In this paper, we are going to verify the potential of developing neural network-based autonomous navigation for homecare mobile robot. We mainly focus on the challenges that homecare mobile robots have and examine whether neural network can be a possible way to handle them. In determining the navigation direction, either a multilayer perceptron or a recurrent neural network is used. We compare the performance of the two networks in the robot navigation problem to figure out which kind of neural network is more suitable for homecare mobile robot.

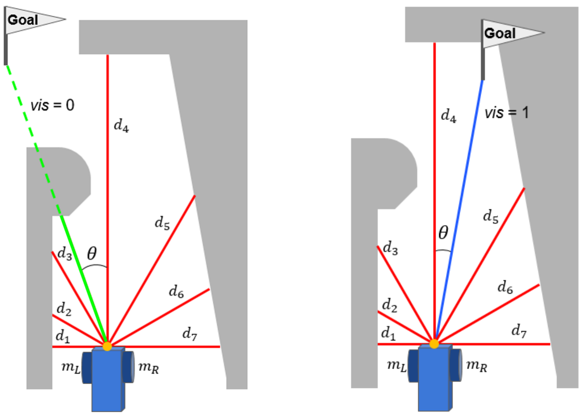
II. Implementation

A. Robot Navigation Simulator

In order to conduct the robot navigation experiment, we implemented robot navigation simulator in MATLAB. The simulator consisted of robot kinematics implementation, sensor readings, creation of different maze environments based on external files, basic methods for robot navigation without neural networks, and simulation of the robot's behavior in the environment. On the simulator, we additionally introduced the concept of a goal, capability to handle more sensors, and a new navigation algorithm that can lead the robot toward the goal without collision.

B. The Robot

The robot implemented in the experiment had seven obstacle sensors, one goal sensor, two separately controlled differential wheels with fixed velocity, and one caster wheel.



(a) Goal is not visible. (b) Goal is visible.

Fig. 2. Description of robot environment.

1) Robot Lasers: The robot had seven laser sensors for obstacle detection and one sensor for locating the goal. Figure 2 shows how the robot's sensors read the environment. There are seven laser sensors located at (-90, -60, -30, 0, 30, 60, 90) from the front of the robot. Each laser measures the distance to the nearest wall/obstacle in each direction to have seven distance measures (d_1, d_2, d_3, d_4, d_5, d_6, d_7). Also, one goal sensor tracks the direction of the goal relative to the robot's heading direction, θ, and checks if the goal is visible (1) or not (0), vis [8], [9]. From these sensors, robot could make total of nine readings (d_1, d_2, d_3, d_4, d_5, d_6, d_7, θ, vis).

2) Robot Wheels: For the robot movement, two differential wheels were used. For simplicity, the wheels could operate with a fix velocity. Each wheel could either be on (1) or off (0). The robot wheel configuration could be described as (m_L, m_R) where m_L is the configuration for the left wheel and m_R is the configuration for the right wheel. Thus, the wheel configuration of (1, 0) would make robot turn right and (0, 1) would make robot turn left.

III. Methods

In this section, the method for gathering the robot navigation dataset and two neural network structures used for learning the robot navigation are described.

A. Robot Navigation Dataset Collection

For collecting robot navigation path dataset, manual control of the robot to navigate through each maze was inefficient. For efficiency, we implemented an algorithm that could navigate through an environment while avoiding obstacles until the robot reaches the goal. Figure 3 is an example path created using the navigation algorithm. As can be seen, the robot wandered around the maze until it saw the goal. As soon as it saw the goal, the robot headed directly toward the goal.

B. Elman-type Recurrent Neural Network

As the decision module for the robot navigation, we used an Elman-type recurrent neural network (RNN). The Elman-type

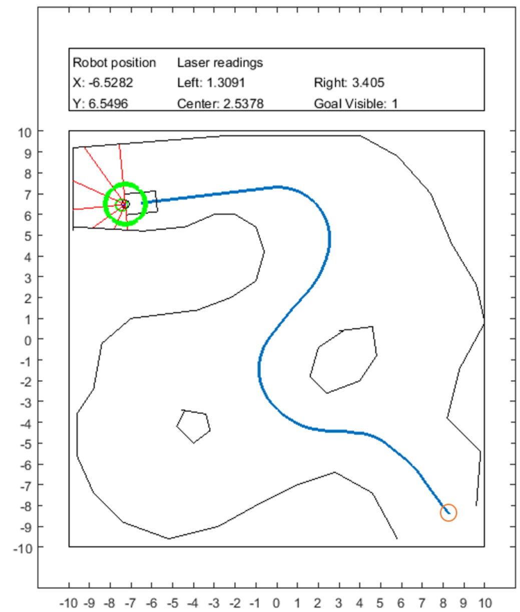


Fig. 3. An example of navigated path.

RNN, known as the simple RNNs along with the Jordan-type RNN, is a type of the RNN that shows the recurrency in the neural network from the hidden layer to the input layer [10]. This reccurency allows the network to handle the sequential data such as sensor readings along a path. Figure 4 shows the detailed structure of the Elman-type RNN we used in the experiments.

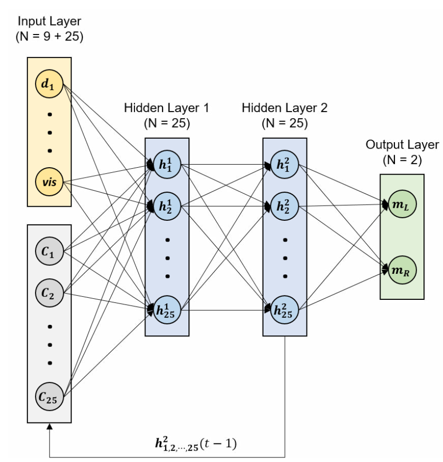


Fig. 4. The structure of the Elman-type RNN.

As seen in the figure, the nine sensory readings gathered from the robot navigation algorithm were used as the data entering the first nine nodes in the input layer of the RNN. The remaining 25 nodes, known as the context nodes, in the input layer were responsible for the recurrency. In each time step, the context node values were the exact copies of node values in the last hidden layer at the previous time step. This enabled the network to hold some information about the previous time steps' data in deciding current motor activations. Since there

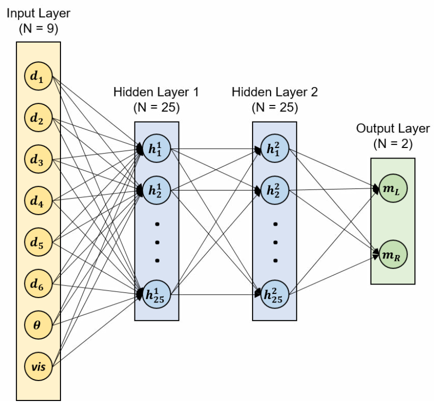


Fig. 5. The structure of the MLP.

is no previous time step for the first step, the context node values were initialized to random values in range [0,1].

Then, the input layer values would propagate through two hidden layers with 25 nodes per each to reach the output layer. In the output layer, two nodes represented the activation of the left and right wheel motors of the robot, respectively. The threshold for the motor activation was 0.5, meaning that when an output node's value was above 0.5, the wheel motor corresponding to that node would move in the next time step. Thus, four different motor configurations could be made from the output of the RNN: (m_L, m_R) = $(0, 0)$ - 'Stop,' $(0, 1)$ - 'Turn Left,' $(1, 0)$ - 'Turn Right,' or $(1, 1)$ - 'Go Straight.' As the activation function, the conventional sigmoid function shown below was used.

$$y = \frac{1}{1 + e^{-x}} \tag{1}$$

In training the Elman-type RNN, the backpropagation through time (BPTT) algorithm was used along with segmented path data. Since the number of time steps for each path was around thousands, the network was not able to learn the whole path at once that the path data had to be segmented. For each path data, only a fixed-length segment of the whole path was used in training the network. In each epoch, different segment of each path was used so that all paths were used for training the network when many epochs were passed. Table I shows the parameters of the Elman-type RNN training in the experiments.

TABLE I
RNN TRAINING PARAMETERS

Parameter	Value
Learning Rate, η	0.0005
Length of a Path Segment	50
Number of Epochs	100000

C. Multilayer Perception

We implemented Multi-Layer Perceptron (MLP) model and used it to compare its performance with the recurrent neural network's performance in the robot navigation problem. Figure 5 shows the detailed structure of the MLP we used in the experiments. As seen in the figure, the structure was very similar to the Elman-type RNN described above. The only difference was that the 25 input nodes responsible for the recurrency were removed. Also, since MLP cannot consider any sequential dependency, data of each time step acted as a sample for MLP. Thus, all path data in the dataset were concatenated to form a dataset of many samples. The input scaling and output configuration settings were identical to the Elman-type RNN case.

In training the MLP, a method similar to backpropagation with stochastic gradient method was used. The difference with the conventional method was that only a small number of samples were used for training in each epoch instead of using all samples in the dataset. In this way, the training could avoid the local minima at the cost of slow convergence. Because of this, the network was trained with many epochs. Table II shows the parameters of the MLP training in the experiments.

TABLE II
MLP TRAINING PARAMETERS

Parameter	Value
Learning Rate, η	0.05
Number of Samples per Epoch	650
Number of Epochs	100000

IV. EXPERIMENT AND RESULT

In this section, we evaluate the performances of the MLP and the Elman-type RNN in the robot navigation problem. In order to conduct experiments, we used 17 mazes for collecting the training dataset and testing the trained networks. We generated training data which was collected in 13 different mazes. We used the fixed initial and default goal positions per maze when we collected the training dataset. Then, we trained the RNN and MLP models with the dataset. We tested the networks with three different test cases: test with the default goal positions in the 13 mazes, test with 2 other goal positions in each of the 13 mazes, and test with four new mazes. When we tested the models, the initial position of the robot varied little.

A. Test with the Default Goal Positions in the 13 Mazes

TABLE III
RESULT OF THE FIRST TEST

	Maze													
Model	1	2	3	4	5	6	7	8	9	10	11	12	13	Total
RNN	8	6	0	6	10	0	0	0	0	5	0	1	0	36
MLP	0	0	0	0	1	8	10	0	0	10	4	0	0	33

We tested the models with the default goal positions on the 13 maze that had been used for collecting the training dataset. Table III shows the number of successful trials on each maze out of 10 trials per maze. The RNN worked better than the MLP in 5 mazes while the MLP worked better than the RNN in 4 mazes. The performance of two algorithms did not differ significantly in this case.

B. Test with 2 Other Goal Positions in Each of the 13 Mazes

In this case, we used the two other goal positions, Goal 1 and Goal 2, that are not used for collecting the training dataset. We tested the RNN and MLP models with the two other goal positions in all 13 mazes in order to check if the navigation worked well even when the goal position was different from those in the training dataset.

TABLE IV
RESULT OF THE SECOND TEST WITH GOAL 1S

	Maze													
Model	1	2	3	4	5	6	7	8	9	10	11	12	13	Total
RNN	2	5	4	4	10	3	0	10	10	10	4	5	10	77
MLP	0	0	0	0	10	0	2	0	0	1	0	10	10	33

TABLE V
RESULT OF THE SECOND TEST WITH GOAL 2S

	Maze													
Model	1	2	3	4	5	6	7	8	9	10	11	12	13	Total
RNN	3	9	0	0	5	5	10	3	0	0	0	10	10	55
MLP	3	0	0	0	1	0	10	0	0	0	0	0	0	14

Table IV and Table V display the results for navigation toward Goal 1s and Goal 2s, respectively. The RNN worked better than the MLP in 15 cases while the MLP worked better than the RNN only in 2 cases. The performance of two network models differed significantly in this test. This experiment showed that the RNN had a better generalization ability for different goal positions.

C. Test with 4 New Mazes

In this case, we tested the models in 4 new mazes not used for collecting the training dataset to check if the navigation worked well even when the environment was totally different.

TABLE VI
RESULT OF THE THIRD TEST

	Maze				
Model	14	15	16	17	Total
RNN	8	7	5	8	28
MLP	0	0	0	0	0

Table VI shows the number of successful trials on each of the four mazes. In this case, the RNN worked better than the MLP in all four mazes and the MLP could not reach the goal in any of four mazes. Same with the previous experiment, the performance of two network models differed significantly and the RNN had a better generalization ability for different mazes.

V. DISCUSSION AND CONCLUSION

In this paper, we conducted experiments to verify if neural network-based autonomous navigation could be an alternative of traditional homecare mobile robot. In order to do this, we used the Elman-type RNN and the MLP for autonomous navigation problem. The performances of the networks varied on each maze and the RNN model could even navigate toward the goals that had not been used in training. Also, the RNN could navigate even when the robot was in totally new environments. In all experiments, the RNN showed overall better performance than the MLP did. Since the path data were time-series data, the navigation problem could better be solved by the RNN. Also, the experiments showed that the RNN had a better generalization ability for different mazes and goals than the MLP did. The RNN's ability to handle arbitrary temporal dynamics and robustness for noisy environment seems to make the network competitive for controls in real dynamical systems.

From the result of the experiments, neural network-based autonomous navigation is a good option for a homecare mobile robot, especially recurrent neural network. Since it has decent generalization ability for new environment and goals, homecare robot can deal with flexible situation and environment change. Also, it can save memory space by storing only a network model rather than storing complex navigation algorithm or map details. We made the navigation tests stop if the robot was bumped into wall in order to check the precise navigation ability. When the neural network is used in the homecare mobile robot, algorithm for navigation should continue to navigate even after crashing into wall. Furthermore, the navigation algorithm can be implemented by using different types of network model or integrated with other solution for future work.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. 2010-0028631).

REFERENCES

[1] "World Population Prospects: The 2015 Revision, Key Findings and Advance Tables," *United Nations Department of Economic and Social Affairs/Population Division*, 2015.
[2] "World Report on Disability," *World Health Organization 2011*, 2011.
[3] A. A. Author, B. B. Author, and C. Author, "Title of article," *Title of Journal*, vol. 10, no. 2, pp. 49–53, 2005.
[4] M. Merten, A. Bley, C. Schröter, and H.-M. Gross, "A mobile robot platform for socially assistive home-care applications," in *Robotics; Proceedings of ROBOTIK 2012; 7th German Conference on*. VDE, 2012, pp. 1–6.
[5] R. Lee, F.-H. S. Chiu, C. W. de Silva, and C.-Y. A. Shih, "Intelligent navigation and micro-spectrometer content inspection system for a homecare mobile robot," *Int. J. Fuzzy Syst*, pp. 389–399, 2014.
[6] T. Krajník, J. M. Santos, and T. Duckett, "Life-long spatio-temporal exploration of dynamic environments," in *Mobile Robots (ECMR), 2015 European Conference on*. IEEE, 2015, pp. 1–8.
[7] T. L. Chen, M. Ciocarlie, S. Cousins, P. Grice, K. Hawkins, K. Hsiao, C. C. Kemp, C.-H. King, D. A. Lazewatsky, A. E. Leeper *et al.*, "Robots for humanity: Using assistive robots to empower people with disabilities," 2013.
[8] C. Hong, C. W. Park, and J.-H. Kim, "Evolutionary dual rule-based fuzzy path planner for omnidirectional mobile robot," in *Fuzzy Systems (FUZZ-IEEE), 2016 IEEE International Conference on*. IEEE, 2016, pp. 767–774.
[9] P. G. Zavlangas, S. G. Tzafestas, and K. Althoefer, "Fuzzy obstacle avoidance and navigation for omnidirectional mobile robots," in *European Symposium on Intelligent Techniques, Aachen, Germany*, 2000, pp. 375–382.
[10] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.