Implementation of RAM Based Neural Networks on Maze Mapping Algorithms for Wall Follower Robot

Ahmad Zarkasi¹, Huda Ubaya², Cora Deri Amanda³, Reza Firsandaya⁴

¹, ², ³, ⁴ Department of Computer Engineering, Faculty of Computer Science, Universitas Sriwijaya, Indonesia
¹zarkasi98@gmail.com, ²huda_ubaya@yahoo.com, ³corawaa17@gmail.com, ⁴reza.firsandaya@gmail.com

Abstract. A wall follower robot will recognize its environment by knowing the position of the surrounding wall. The goal is that the robot has a good safe navigation system without damaging the walls. For navigation systems, robots use the Maze Mapping method with the left-hand rule, while to study the wall distance pattern is a RAM-based artificial neural network method. The neural network has 3 RAM nodes to process the received environmental patterns. The left sensor, the right sensor, and the front sensor have 8 bits of the input pattern, the pattern will be optimized into 4 bits stored in the RAM node, so that the computing process in the robot becomes faster and simpler [1][4]. RAM discriminator design has 3 RAM nodes. Each RAM node has 4 bit words (x = 4), with a total of 12 input vector bits (n = 12). Each RAM discriminator will receive 48 binary input patterns. The results of environmental polo testing, the wall follower robot can complete the labyrinth path which has 8 intersections and a dead end with a distance of 595 cm, then the next 4 intersections with a distance of 360 cm.

1. Introduction
In the early 80s, mobile robot technology had begun to replace industrial robots or manipulator robots. In the development of technology, a robot is getting smarter in term of navigation and can recognize the environment around it autonomously according to the program embedded on the robot. In several studies [2][3] discussed about robots that are able to follow the maze. Not only able to trace a maze, but robot is also able to get out of the maze with programs on the robot brain. There are several methods of finding solutions to maze problems, one of which is the maze mapping algorithm. Maze Mapping or path mapping that applies the concept of left / right hand rule or wall follower rule in the search for a way out of the maze. This study will apply the maze mapping algorithm to robot wall follower. Robot wall followers who navigate through the wall must study the position of the surrounding wall so that they can navigate safely without damaging the walls. Position distance wall learning will use RAM-based artificial neural network methods [4].

2. Overview
2.1 RAM Based Neural Network
RAM-based neural networks, also known as n-tuple categorizers or weightless neural networks (WNNs), do not store knowledge on their connections but in Random Access Memories (RAM) in network nodes or neurons. Neurons operate in binary input values and using RAM as lookup tables: the synapses of each neuron collect a bit vector from the network input that is used as RAM address, and
the value stored at this address is the neuron output \([4][5]\). Figure 1 below is the architecture of the WNNs. In its development, the WNNs has been accepted and received more attention in research as a learning machine that has more capabilities and has a better pattern classification. This is because the WNNs have several advantages such as a short and simple learning scheme, fast execution time and ready to be implemented into hardware.

![Architecture of WNNs](image)

Figure 1. Architecture of WNNs

2.2 Maze Mapping Algorithm

The algorithm of wall followers is generally used when the target position is unknown. In the wall follower algorithm, the robot will see the right or left wall and navigate along the maze until it finds a target. This algorithm is recognized to be very efficient for mazes that connect walls to targets \([2]\). There are 2 types of wall follower algorithms: left-hand rule and right-hand rule. The workings of these two algorithms are the same, only differing in prioritizing the direction when meeting intersections, depending on the type chosen. Right hand rule prioritizes turn right. When the robot applies the right-hand rule algorithm, the robot will turn right whenever it meets the right turn. If there is no right turn, the robot keeps going straight ahead. If the right and front are detected obstruction, the robot will turn left. And the robot will store the intersections and dead ends that are encountered. Whereas left hand rule prioritizes turn left. Figure 2 is maze mapping algorithm.

![Maze mapping algorithm](image)

Figure 2. Maze mapping algorithm: (a) Right hand rule (b) Left hand rule

After exploring and storing the entire maze and knowing the finish position of the maze, the entire route is analyzed so that it can be optimized. The process of simplifying the route is by identify a deadlock code on the route that has been saved. If there is a deadlock code, the code will be simplified.

3. System Implementation

3.1 Framework

This The research methodology is divided into several stages. The steps taken follow the research framework that has been designed. Chart of research framework can be seen in Figure 3.
When active, the robot will process the mapping by giving a code every time it encounters an intersection and a dead end. The codes are stored in the robot's memory and will be arranged continuously every time the robot encounters an intersection and a dead end until the robot reaches the finish position. The codes used when mapping is:

- "L" means left. This code indicates that the robot has turned left because it passed the intersection.
- "F" means forward or continue. This code indicates that the robot keeps going because it meets intersection three with the right or right turn.
- "0" This code means that the robot encounters a dead end and walks back to the last intersection

### 3.2 RAM Based Neural Network Design

There are 3 RAM nodes namely RAM 1 in address 21H to store the pattern received from the left sensor, RAM 2 in the address 22H to store the received pattern from the right sensor, and RAM 3 in the 23H address save the pattern received from the front sensor. 8 bits of input pattern received will be optimized so that only 4 bits are stored in the RAM node. The optimization process is done by shifting 2 bits to the right. This is intended to make the input pattern becomes more optimal because there are invisible patterns. So the computing process is simpler. The following in Figure 4 is a design for optimization on RAM nodes.
open space, corridor, left corner, right corner, u-shape, left wall, right wall and front wall. The output from the discriminator will determine the winner class.

### 3.3 Maze Mapping Design

The maze mapping algorithm is used as a route determinant so that robots can get out of the maze. The algorithm used is left hand rule, where the robot will prioritize turn left. This system uses 8 RAM addresses to store the intersections encountered by robots, namely 41h, 42h, 43h, 44h, 45h, 46h, 47h and 48h. After the robot explores the entire maze and exits the maze using the left-hand rule, the robot will return to the maze to the starting line with the shortest path. Figure 5 shows the maze used and the conversion results.

![Figure 5. Code giving and conversion](image)

4. **Result and Analysis**

#### 4.1 RAM Based Testing

Testing input data patterns were done to ensure that the distance data received by the AT89C55 microcontroller is stored in the correct RAM node address. This test is done by reading the distance data sent by the ATmega8535 microcontroller via PORT2 AT89C55 microcontroller and the data is stored in a RAM node address that has been initialized. Distance data addressing is done using 2 bits MSB. Table 1 is sensor Addressing. In the table, it is explained that the bits used as chip select are 2 bits of MSB, the 7th and 6th bits. When the 7th bit is 0 and the 6th bit is 1, the data received by the AT89C55 microcontroller is the left sensor data and the data will be stored at the 21H address. When the 7th bit is 1 and the 6th is 0, the data received is the right sensor data and the data will be stored at the 22H address. When the 7th and 6th bits are 1, the data received is front sensor data and the data will be stored at address 23H.

| Sensor | RAM Address | Chip Select | Data |
|--------|-------------|-------------|------|
|        | M1/D7       | M0/D6       | D5   | D4   | D3   | D2   | D1   | D0   |
|        | Bit-7       | Bit-6       | Bit-5| Bit-4| Bit-3| Bit-2| Bit-1| Bit-0|
| Left   | 21H         | 0           | 1    | x    | x    | x    | x    | x    | X    |
| Right  | 22H         | 1           | 0    | x    | x    | x    | x    | x    | X    |
| Front  | 23H         | 1           | 1    | x    | x    | x    | x    | x    | X    |

Table 1 Sensor Addressing
Table 2 Reading sensor data

| Data | Bit |
|------|-----|
| 11   | 0 0 0 0 1 0 1 1 |
| 12   | 0 0 0 0 1 1 0 0 |
| 13   | 0 0 0 0 1 1 0 1 |
| 14   | 0 0 0 0 1 1 1 0 |
| 15   | 0 0 0 0 1 1 1 1 |
| 16   | 0 0 0 1 0 0 0 0 |
| 17   | 0 0 0 1 0 0 0 1 |
| 18   | 0 0 0 1 0 0 1 0 |
| 19   | 0 0 0 1 0 0 1 1 |
| 20   | 0 0 0 1 0 1 0 0 |

4.2 Maze Mapping Testing

Figure 7 is the initial route that the robot passes to get out of the maze. Starting from the start position, then there is left corner and the robot turns left and saves the 20H code at address 41H. Then the robot encounters the left wall and a straight robot and stores the 80H code at the 42H address. The robot is straight up to meet u-shape and rotates back to the previous junction and stores the F0H code at 43H address. After turning back to the previous intersection, the robot reads the right wall and turns left and stores the 20H code at the 44H address. Then there is a front wall and a robot turn left until you find u-shape and rotate back and save the 20H code at the address 45H and F0H at address 46H. After turning back, the robot reads the left wall and chooses straight and stores the 80H code at the 47H address. Then the robot encounters the left corner and turns left and saves the 20H code at address 48H. Finally, the robot encounters an open space pattern at the finish line and stores the stop code or C0H at the 49H address. From Figure 6, it can be seen that the robot has been able to get out of the labyrinth according to the algorithm left hand rule. After the robot explores and exits the maze, the robot will simplify the stored code. The result of simplification when meeting the first dead end is to turn right. The result of simplification when meeting the second dead end is to turn right. So after exploring the

![Figure 6. Maze mapping testing](image)

Test results of the algorithm left hand rule are displayed using Top View Simulator. In Figure 8 it is shown that the robot has finished exploring the maze and storing the intersection and deadlock that is found in the 41H to 48H RAM address. After exploring and exiting the maze, the robot will simplify the stored code. After path optimization, 8 intersections are simplified into 4 intersections with a distance of 360 cm.
5. Conclusion
Wall follower robot was able to carry out their duties well, which is to get out of the maze using the left-hand rule algorithm. Data analysis on RAM also runs as desired displayed in data simulation. The robot can come out faster than the maze with 8 intersections or dead ends with a distance of 595 cm and 4 intersections with a distance of 360 cm. Simplified pattern codes are left, right, right, then left. The time needed by the robot wall follower to get out of the labyrinth before simplifying the code 3 x longer, than after using a simplified code

Acknowledgement
This work was supported by The Faculty of Computer Science, Intelligence System Research Group faculty of Computer Science, and Laboratory of Robotic and Control System of Computer Science, Universitas Sriwijaya, Palembang, Indonesia.

References
[1] Siti Nurmaini, Zarkasi Ahmad, 2015, “Simple Pyramid RAM-Based Neural Network Architecture for Localization of Swarm Robots”. Jurnal of Information Processing Systems, Vol.11, No.3, pp.370–388.
[2] A. Bakar and S. Saman, 2013 “Solving a Reconfigurable Maze using Hybrid Wall Follower Algorithm,” International Journal of Computer Applications (0975–8887), vol. 82, no. November, pp. 22–26.
[3] A. F. De Souza, F. Pedroni, E. Oliveira, P. M. Ciarelli, W. Favoreto, L. Veronese, and C. Badue, 2009. “Neurocomputing Automated multi-label text categorization with VG-RAM weightless neural networks,” vol. 72, pp. 2209–2217.
[4] Ahmad Zarkasi, Aciiek Ida Wuryandari, 2013 “Multilayer Processing Architecture of RAM Based Neural Network with Memory Optimization for Navigation System,” Proceedings of the Joint International Conference on Rural Information & Communication Technology and Electric-Vehicle Technology (r-ICT-ICEV2013) no. 1943, p. 4799.
[5] R. E. Ak and E. Iskandar, 2016 “Implementasi Sistem Navigasi Behavior Based dan Kontroler PID pada Manuver Robot Maze.” Jurnal SIMETRIS, Vol 7 No 2.