J Inf Process Syst, Vol.11, No.3, pp.370~388, September 2015
http://dx.doi.org/10.3745/JIPS.01.0008

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Simple Pyramid RAM-Based Neural Network Architecture for Localization of Swarm Robots

Siti Nurmaini* and Ahmad Zarkasi*

Abstract

The localization of multi-agents, such as people, animals, or robots, is a requirement to accomplish several tasks. Especially in the case of multi-robotic applications, localization is the process for determining the positions of robots and targets in an unknown environment. Many sensors like GPS, lasers, and cameras are utilized in the localization process. However, these sensors produce a large amount of computational resources to process complex algorithms, because the process requires environmental mapping. Currently, combination multi-robots or swarm robots and sensor networks, as mobile sensor nodes have been widely available in indoor and outdoor environments. They allow for a type of efficient global localization that demands a relatively low amount of computational resources and for the independence of specific environmental features. However, the inherent instability in the wireless signal does not allow for it to be directly used for very accurate position estimations and making difficulty associated with conducting the localization processes of swarm robotics system. Furthermore, these swarm systems are usually highly decentralized, which makes it hard to synthesize and access global maps, it can be decrease its flexibility. In this paper, a simple pyramid RAM-based Neural Network architecture is proposed to improve the localization process of mobile sensor nodes in indoor environments. Our approach uses the capabilities of learning and generalization to reduce the effect of incorrect information and increases the accuracy of the agent's position. The results show that by using simple pyramid RAM-base Neural Network approach, produces low computational resources, a fast response for processing every changing in environmental situation and mobile sensor nodes have the ability to finish several tasks especially in localization processes in real time.

Keywords

Localization Process, RAM-Based Neural Network, Swarm Robots

1. Introduction

Robots can be used for exploration in an unknown environment, especially in dangerous ones. It is common to employ an advanced robot for such tasks. However, the robot is susceptible for solving task in such environment because a failure of the robot means the failure of the entire mission. An emerging approach to robotics research is to employ a group of simple robots named swarm robots, which can collectively achieve a demanding task. The failure of one robot should not affect the overall mission. When swarm robots work in unknown environment, localization considers the most important

Manuscript received March 3, 2014; accepted October 3, 2014; onlinefirst August 10, 2015.
Corresponding Author: Siti Nurmaini (sitinurmaini@gmail.com)
* Dept. of Computer Engineering, Faculty of Computer Science, Sriwijaya University, Sumatera, Indonesia (sitinurmaini@gmail.com, zarkasih98@gmail.com)

performance aspect. Localization is the process of determining the positions of robots or targets in their entire environment. Without determining the robot's position and orientation within an environment, no trajectory can be generated and swarm robots cannot achieve a target [1].

In swarm robots research with low cost processor and simple sensors is suffice to initially localization, particularly in a highly dynamic environment. The imprecise environmental perception of sensors is a result of the fact that many sensors provide a relatively accurate measurement of the distance from an object, but they provide poor information about its exact location. To overcome imprecision detection in sensor system is a difficult problem, due to it need large computational resources. Otherwise, it can produce inaccurate measurement [2].

Correctly estimating about swarm location is a prior work to accomplish several tasks in swarm robots. A single sensor, like GPS, provides global position estimations, but it is restricted to outdoor environments. It has an inherent imprecision of a few meters while more accurate estimations are obtained by using sensor fusion between the laser and camera [3,4]. Especially, in an indoor environment, laser and camera sensors can be used for conducting pose estimation [5,6]. However, they require hard computation to process complex algorithms and a limited field of view, which makes the localization task harder. Another possibility is to use an odometer sensor, which provides useful information in some cases [7,8]. Nevertheless, when odometer sensor is used in real systems, it has an incremental error that usually invalidates.

Advances in sensor technologies have created low-cost, low-power, multi-functional miniature sensor devices. These devices make up hundreds or thousands of ad hoc tiny sensor nodes spread across a geographical area [9]. A sensor network (SN) can provide access to information anytime and anywhere by collecting, processing, analyzing, and disseminating data. SNs are widely available in indoor and outdoor environments and allow for efficient global localization. They require relatively low computing resources.

Other advantages of this approach are scalability, robustness, and independence of specific features in the environment [9]. Recently, the concept of mobility was introduced to SNs by dispatching sensor nodes conducting various missions with multi-robots, named swarm robots. Nevertheless, the inherent instability in the wireless signal does not allow for it to be directly used for accurate position estimations of swarm robots.

One machine learning technique that could reduce the instability of the signals of the SNs is the artificial neural network (ANN), given its capability of learning from examples, and its ability to generalize and adapt to outputs. Many applications utilize this technique, but in the learning process they require approximation, prediction, classification, and implementation of the network operational use of the multiplier unit [10]. It can be problematic for simple swarm robots' applications with a small memory, due to the large size of resources because training the generalized data requires both the forward propagation phase and backward propagation phase [11].

A weightless neural network (WNN) is a type of ANN but it does not store knowledge in ANN multiplier unit connections but in the Random Access Memory (RAM) inside the neurons. These neurons operate in binary input values and use the RAM as lookup tables. The synapses of each neuron collect a vector of bits from the inputs array of the RAM address. Training is carried out in one shot and basically consists of storing the desired output in the address associated with the input vector of the neuron [12].

The WNN is very effective as pattern recognition tools, offering fast training and testing, in addition to being easy to implement [12,13]. However, if the network input is too large, the memory size becomes prohibitive, since it must be equal to $2n$, where n is the input size. An n input RAM node has $2n$ memory locations, addressed by the n-bit string $a = (a_1\ a_2\ ....\ a_n)$. A binary signal $x = (x_1\ x_2\ ....x_n)$ on the input lines will access only one of these locations resulting in $y = C\ [x]$ [11,12]. The challenging task would be to design an algorithm to fill the RAM during training for obtaining the best possible generalization, especially in an application with a small memory resource.

2. Proposed Method

2.1 RAM-Based Neural Network Architecture

ANNs have been successfully applied to many fields. However, their application in real world situations still faces many challenges. One of these challenges is that training requires the repeated presentation of training data and often results in a long amount of learning time [14]. Finding the best architectural configuration for ANNs is difficult even when the ideal number of output classes is known. The problem is increased exponentially when there are multiple output sets [14].

WNNs can operate on more simplistic hardware. The key difference between WNNs and ANNs modify stored look-up tables instead of complex weightings. The architecture of neurons and the interconnection of layers are the important elements of WNNs. However, this architecture can go one step higher because the network can consist of a multiplicity of neurons. One of the biggest improvements in WNNs, is use architecture as small as possible to solve complex problems.

A single layer of RAM-nodes uses most of the architecture for a WNN. A system formed into various RAM-discriminators is called WiSARD [15]. Each RAM-discriminator is trained in a particular class of patterns, and classification by the multi-discriminator system is performed in the following way: Kan and Aleksander [16] introduced a multilayer or pyramid architecture, which is formed by new nodes called probabilistic logic nodes (PLNs). It stores a 2-bit value each of all of these memory locations: "0," "1," and "u" randomly, before the learning phase. However, pyramid architecture is saturated very easily and it fails to learn new patterns before the training set has been completely presented [13].

In [17], the author has proposed a natural extension of the PLN—the m-state PLN (MPLN). In this new model, a wider discrete range of values can be stored at each s-memory location. However, new information is acquired after different steps, as incorrect information is only thrown away after a certain number of errors [13]. An evolution of this model is proposed by Taylor [18] called a probabilistic RAM (pRAM). In this architecture values belonging to [0, 1] can be stored in the memory locations with continuous probability. Given a certain input, the contents of memory locations represented the probability that the value of 1 is produced as an output.

The Goal Seeking Neuron (GSN) has been developed with the aim of preserving the corruption and error information in the PLN [19]. GSN architecture can input, store, and output 0's, 1's, and u's. Two different locations are addressed in a GNS if the input contains a u. GRAMs were introduced by Aleksander [20] at the node level by including a spreading phase in the learning algorithm just after a training pattern is stored, in order to increase the generalization of WNNs. However, there is no generalization property in the RAM neurons. There are several extensions of RAM neurons in which

they try to smooth the non-generalization problems of RAM neurons, such as PLN, GSN, and G-RAM [21].

The implementation of a WNN is limited by the amount of variable memory since RAM neurons are assembled as a binary vector, addressed by the inputs and is stored in the system's memory [11,12]. Therefore, the architecture must overcome the memory problem when WNNs is implemented. RAM-based Neural Network architecture on hardware implementation produce fast testing and training, thus it is ideally suited for mobile robots. Some of the research that has been carried indicates that this technique is successful. Hannan et al. [22] have presented that a RAM-based Neural Network is considered as the heart of the controller and is implemented on FPGAs for collision free robot navigation. Botelho et al. [23], have proposed a simple RAM-based network that has been developed to control a mobile robot and they used a high-speed neural network to detect obstacles and navigation control. Yao et al. [24] have described a RAM-based Neural Network for a mobile robot using a simple microprocessor system. Their method allows for the robot to detect and avoid obstacles in real time. McElroy and Howells [25] have introduced a technique for the identification of rooms or locations in the absence of complex and succinct information for mobile robot localization. Coraggio and De Gregorio [26] have described a hybrid WiSARD and NSP approach for solving a robot global localization problem in an office-like environment. The global localization problem deals with the estimation of the robot position when its initial pose is unknown. De Gregorio [27] has presented the possible use of virtual neural sensors that is implemented by means of weightless neural systems as active or reactive sensors in robotic vision.

2.2 Proposed Simple Architecture

A RAM-based Neural Network (RAM-bNN) is composed of a class of neural networks characterized by their potential for hardware realization. It has fast processing during the training phase. The memory capacity required is not too large, because it is only used to store training data. However, if the size of the training data sets increases and the memory storage is not big enough, the network can suffer from saturation. In order to avoid this problem, we are proposing pyramid architecture for increasing the process training and avoiding the memory saturation.

This research is focused on the development of efficient RAM-bNN architecture based on a simple microprocessor system. The proposed architecture is shown in Fig. 1. The input data in the form of a bit vector coming from sensor arrays (s_1,…, s_n). The groups of sensor arrays are connected to a discriminator. In this research, sensor values were distributed to each neuron in the input layer. The training process of RAM-bNN does not use weight matrices, it only indicates that the results of the outputs are correct or incorrect. Training only determines the value of a single distance parameter, such as being far or near.

Each discriminator is trained as a sample of its labeled class for specific classification. Several output classes are required to construct the appropriate number of layers. The architecture with neuron data processing is divided into two layers for learning and recalling. Output pattern from decision layer to determine the winning class.

The number of layers depends upon the input vector size and the memory size. The network is built of a group of layers, whereby each of these layers has a set number of neurons. The number of neurons in each layer is defined by the size of the network input.

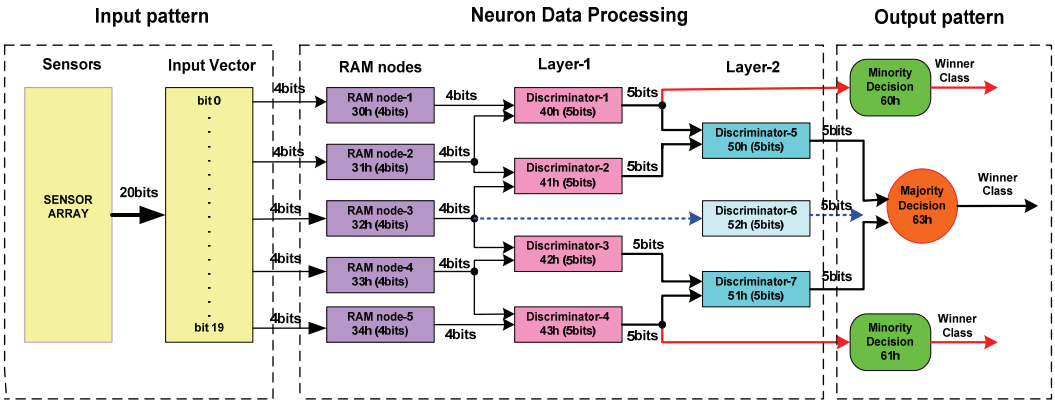


Fig. 1. Proposed simple pyramid RAM-Based Neural Networks architecture.

The proposed pyramid architecture is comprised of layers of input and output modules. All neuron connections are chosen at random and each sub-network is determined by the system. There are two working modes of RAM-bNN architecture that are known as learning and recalling. For the initial condition, each address location is signed as a memory register and all registers in the memory location are set to zero. This zero is interpreted as a *u* value. During the learning mode, the counter register may become positive or negative. A positive value is interpreted as 1 and a negative value is interpreted as 0, as far as that memory location value is concerned. If the desired output is 1, the address location is incremented. Otherwise, the address location is decremented.

This technique involves calculating the address vectors for the next layer, moving towards the output. On the previous layers from each connecting line to the next layer, the input vector to each memory location is constructed by invoking the recalling mode. During training, the contents of any memory location may be zero, due to undefined values (*u*), in that vector. These positions are filled with the desired output value for that pyramid before applying the vector to the next layer. The newly constructed address vector is then applied to the memory inputs and the memory location contents are again modified. This procedure is continued until the output is reached. In the recalling mode, the content of each addressed memory location, starting from the input layer, is interpreted as 0 (negative), 1 (positive), or as *u* (undefined). The output of each location memory may move an undefined output forward.

2.3 Memory Optimization

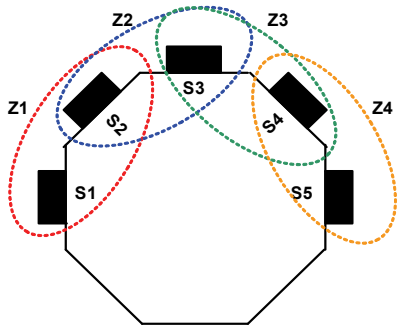


Fig. 2. RAM discriminator classification.

In this research, every mobile robot uses five sensors for environmental detection. Five sensors can be formed into four groups of RAM discriminator patterns, which are called RAM discriminator classifications, as shown in Fig. 2. Each pattern composed of memory locations or neurons addressed by the RAM address line in the discriminator. The initial input patterns of the RAM discriminator represents the classes that have been determined based on the positions of the sensors.

For data optimization, the memory only uses 4-bits of the MSB. Each RAM discriminator has 2 RAM nodes, each RAM node has 4-bits of a word or $x = 4$ and then the total input vector is 8-bits or $n = 8$. Therefore, each RAM discriminator can receive 32-bits of input patterns. Based on Fig. 2, the input for RAM discriminator Z1 is a combination of data from S1 and S2 or Z1 = S1 + S2. This means that the sensor data, S1 (4-bits) is added with sensor S2 (4-bits). Then, the result is stored in the RAM discriminator. This process is repeated for other discriminators Z2, Z3, and Z4.

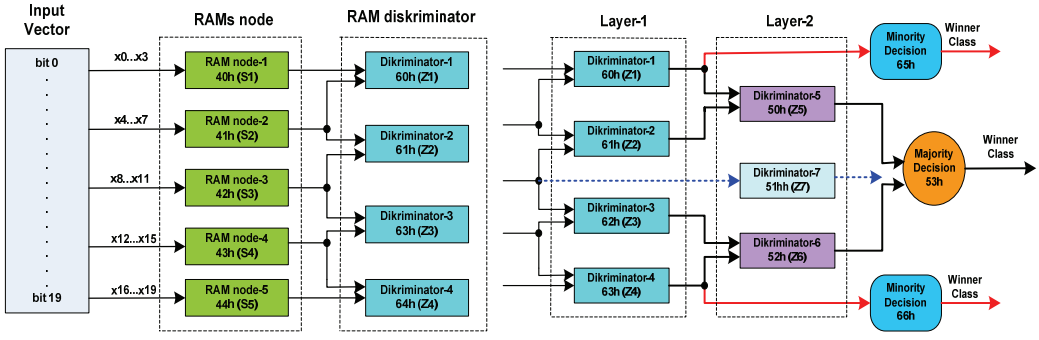


Fig. 3. Process of memory optimization in proposed architecture. (a) RAM discriminator and (b) multi-layer processing.

In this research, multi-layer processing is used with two layers, as shown in Fig. 3. Layer-1 is a RAM discriminator layer, which consists of four parts of RAM discriminators, Z1, Z2, Z3, and Z4, with addresses 60h, 61h, 62h, and 63h, respectively. Layer-2 has three discriminators, Z5, Z6, and Z7, with addresses 51h, 52h, and 53h, respectively. These discriminators are classes for determining the result of the desired pattern or winning class in multilayer processing architecture.

These classes are representatives of the three final output patterns of different address. Each output pattern from the discriminator classes will be compared to a threshold value (threshold). The RAM discriminator on Layer-1 receives the minimum data is 00100b and the maximum data is 10000b. These values as the input pattern are obtained from RAM classification. The RAM discriminator Z1 and Z2 can directly determine the final outcome or winning class on the condition of there being only one discriminator that has a minimum value (minority decision). The final results are utilized for determining the direction of the movement of the swarm robots and target localization.

3. Research Method

3.1 Swarm Robots Design

The implementation of pyramid RAM-bNN architecture in swarm robots is shown in Fig. 4. They

contain two microcontrollers, which implies the implementation of a robust communication mechanism between modules. For this platform, the architecture can be conceptually seen as the driver module, processing module, and the communication module. All modules are connected to each other via an 8-bits of data. A general purpose high-performance controller for decision-making is connected to the parallel bus. There are five infrared sensors located at the front, left, and right side of the mobile robot for navigating. Two gas sensors, one humidity sensor, and one temperature sensor are employed for environmental monitoring and target localization.

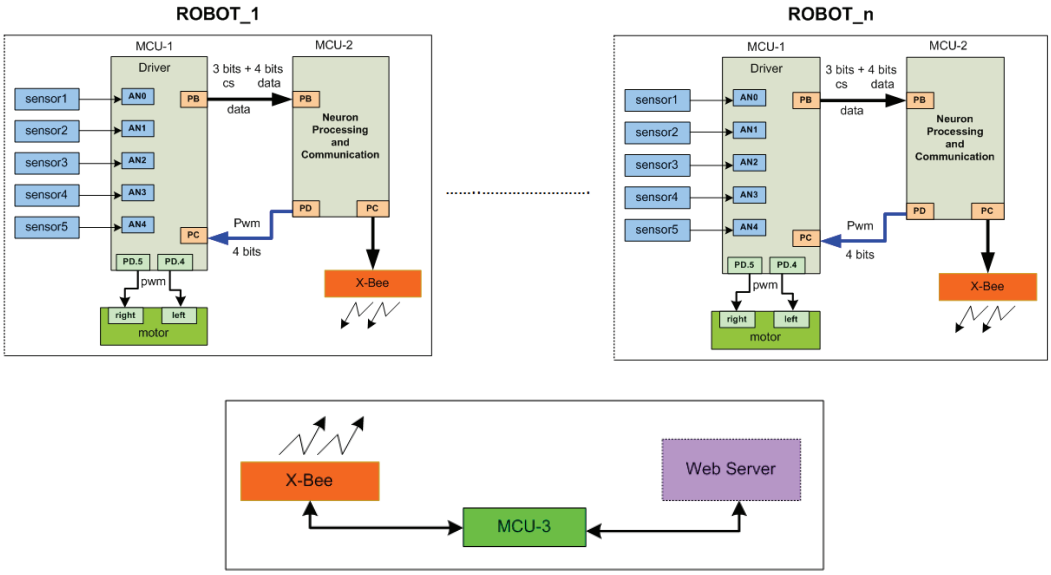


Fig. 4. Swarm robots implementation with communication system.

Swarm robots can communicate each other as mobile sensor node uses X-Bee-Pro OEM/ZigBee. The radio frequency transceiver and receiver provides full duplex communication at frequency IEEE 802.15.4 2.4 GHz. ZigBee is targeted at radio-frequency applications that require a low data rate, long battery life, and secure networking. ZigBee is a low-cost, low-power, wireless mesh networking standard. The low cost allows the technology to be widely deployed in wireless control and monitoring applications. The hardware and software development are summarized. Our research required two modules of X-Bee for communicating between the web server, static nodes, and swarm robots as mobile nodes.

3.2 Input Data

The idea of this research is to apply RAM technology for applications with a limited number of inputs. In this way, a pyramid RAM device is utilized for addressing all possible input vectors. This is done so as to avoid excessively large tables. For example, 16 input bits correspond to a table with 216 = 65,536 entries (64K). Doubling the number of input bits to 32 requires a table with 232 entries, or 4 Gigabytes entries. This situation must be avoided in our work, due to the fact that swarm robots only use a simple microcontroller with small memory resources.

As stated previously, the sample problem domain was data that we gathered from the infrared

sensors attached to a robot and initial data collect from infrared sensors, as shown in Table 1. Sensor data must be encoded into a binary value for determining the data threshold. Sensor 1 (S1), sensor 3 (S3), and sensor 5 (S5) have the same value as the sensor distance range of about 2.5 cm. The closest distance is 10 cm or 80h or 1000 0000b and the farthest distance is 12.5 cm or 70h or 0111 0000b. Sensor 2 (S2) and sensor 4 (S4) have the same value as the sensor distance range of 7.5 cm. The closest distance is 10 cm or 80h or 1000 0000b and the farthest distance is 17.5 cm or 50h or 0101 0000b.

Table 1. Example data from the sensors

Sensor	Binary data	Distance data	Range of sensor
S1	(0111 0000) to (1000 0000)	(12.5 cm) to (10.0 cm)	2.5 cm
S2	(0101 0000) to (0110 0000)	(17.5 cm) to (15.5 cm)	2 cm
	(0110 0000) to (0111 0000)	(15.5 cm) to (12.5 cm)	3 cm
	(0111 0000) to (1000 0000)	(12.5 cm) to (10.0 cm)	2.5 cm
S3	(0111 0000) to (1000 0000)	(12.5 cm) to (10.0 cm)	2.5 cm
S4	(0101 0000) to (0110 0000)	(17.5 cm) to (15.5 cm)	2 cm
	(0110 0000) to (0111 0000)	(15.5 cm) to (12.5 cm)	3 cm
	(0111 0000) to (1000 0000)	(12.5 cm) to (10.0 cm)	2.5 cm
S5	(0111 0000) to (1000 0000)	(12.5 cm) to (10.0 cm)	2.5 cm

Once all of the input data has been generated, it is placed into a RAM node, as shown in Table 2. Because 8-bits of data produce high computational cost. Therefore, we only used 4-bits of MSB data. Every 1-bit of data equals to one change of environmental pattern, all inputs are 15 patterns of the networks.

In this work, the input data from five infrared sensors, such as S1, S2, S3, S4, and S5, are stored in the RAM address at 40h, 41h, 42h, 43h, and 44h, respectively, as a RAM node address for each neuron. The range area of sensors is divided into two groups, the first group area is the left sensor, which is S1; the front sensor is S3; and the right sensor is S4. For this group the farthest distance from the sensor to the obstacle is about 14 cm or 0111b and the nearest distance from the sensor to the obstacle is about 10 cm or 1000b. The second group area is the left-front sensor S2 and the right-front sensor is S4. For this group the farthest distance from the sensor to the obstacle is about 22 cm or 0100b and the nearest distance from the sensor to the obstacle is about 10 cm or 1000b. That sensor values are a function of the activation value or threshold for RAM-bNN. If the sensor value does not correspond with any activation value, it is indicates there are no obstacles and the input neuron to the RAM node is 0 or 0000b.

3.3 Environmental Classifications

The sensors database, which represents a variety of possible situations in the environment, should be built beforehand to classify the entire environment into one of the prototype patterns. The appropriate RAM node contained in the proposed architecture trains every prototype environmental pattern. To identify the environmental patterns, five infrared sensors are used in swarm robots. The detection range of infrared sensors is set at 10–30 cm. The values of the sensor outputs are used as the input to the RAM node, in order to generate one of the classified environments. In this research, eight classes of the environmental patterns are utilized for obtaining information about the initial patterns of the

environmental situation, such as open space, front wall, U-shape, corridor, left wall, right wall, left corner, and right corner, as shown in Fig. 5.

Although only a few of the environmental patterns are used to train the RAM node, as shown in Fig. 5, in the training processes it must be generalized to all kinds of environmental patterns. All of the patterns, including scattered obstacles in the environment, that may be encountered when a mobile robot moves.

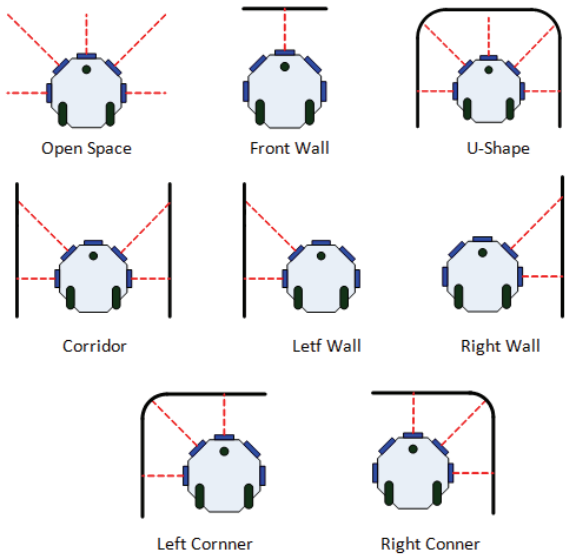


Fig. 5. Sensory patterns for eight environmental features.

Table 2. Input patterns in RAM node

RAM nodes	Input data					
	8 bits			4 bits		
	Max	Min	Address	Max	Min	Address
RAM node-1	1000 0000	0111 0000	80h–70h	1000	0111	08h–07h
RAM node-2	1000 0000	0101 0000	80h–50h	1000	0101	08h–05h
RAM node-3	1000 0000	0111 0000	80h–70h	1000	0111	08h–07h
RAM node-4	1000 0000	0101 0000	80h–50h	1000	0101	08h–05h
RAM node-5	1000 0000	0111 0000	80h–70h	1000	0111	08h–07h

Table 2 shows the threshold data in the RAM node. These neurons operate in binary input values. Each neuron collects a vector of bits from network inputs that are used as the RAM address. The 8-bits of data from the sensor arrays directly connects to the address lines of the memory. The calculation of this value is based on the distance from an obstacle to the robot. In this research, the vector input from 8-bits only uses 4-bits of MSB for memory optimization. Therefore, the entire amount of data is 20-bits. Tables 3 and 4 show the results of memory optimization. The discriminators process only a maximum of 5-bits of data, including 4-bits of data from sensors and 1-bit of data for unseen patterns. If 8-bits of data is used, memory allocation for processing data is 9-bits and then the number of patterns that can be processed are 192 patterns. But if 5-bits of data is used, the number of patterns that can be processed are 12 patterns. From these results, the percentage of memory optimization for all RAM nodes is achieved at about 93.75%.

Table 3. Percentage of RAM memory optimization

RAM node	8 bits		4 bits		Optimization (%)	Unseen pattern
RAM node-1	80h–70h	128–112=16	08h–07h	8–7=1	93.75	15
RAM node-2	80h–50h	128–79=49	08h–05h	8–4=3	93.75	45
RAM node-3	80h–70h	128–112=16	08h–07h	8–7=1	93.75	15
RAM node-4	80h–50h	128–79=49	08h–05h	8–4=3	93.75	45
RAM node-5	80h–70h	128–112=16	08h–07h	8–7=1	93.75	15

Table 4. Simple discriminator optimization

Neuron classification from sensor array	RAM node (4 bits MSB)			RAM node (4 bits MSB)			Neuron (5 bits)	
	Max	Min	Address	Max	Min	Address	Max	Min
Z1 (S1+S2)	0111	1000	10fh–04fh	0101	1000	10h–04h	00101	10000
Z2 (S2+S3)	0101	1000	10fh–04fh	0111	1000	10h–04h	00101	10000
Z3 (S3+S4)	0111	1000	10fh–04fh	0101	1000	10h–04h	00101	10000
Z4 (S4+S5)	0101	1000	10fh–04fh	0111	1000	10h–04h	00101	10000

4. Results and Analysis

4.1 Generalization

Table 5. Generalization process

Input patterns	Layer-1	Obstacle position	Layer-2
	Discriminator relation		Class
(S1=1), (S2, S3=0), "(S4, S5=0)	(Z1>Z2)&(Z2=Z3)&(Z3=Z4)	Left	Z5
(S1=1), (S2, S3=0), "(S4=0, S5=1)	(Z1>Z2)&(Z2=Z3)&(Z1>Z4)	Left, right	Z5
(S1=1), (S2, S3=0), "(S5=0, S4=1)	(Z1>Z2)&(Z3=Z4)&(Z1>Z3)	Left, right-oblique	Z5
(S1=1), (S2=0, S3=1), "(S4,S5=0)	(Z1>Z2)&(Z2=Z3)&(Z3>Z4)	Left, front	Z5
(S1, S2=1), (S3=0), "(S4, S5=0)	(Z1>Z2)&(Z2>Z3)&(Z3=Z4)	Left-oblique	Z5
(S1, S2 =1), (S3=0), "(S4=0, S5=1)	(Z1>Z2)&(Z2>Z3)&(Z1>Z4)	Left, left-oblique, right	Z5
(S2=1), (S1, S3=0), "(S4, S5=0)	(Z1=Z2)&(Z2>Z3)&(Z3=Z4)	Left-oblique	Z5
(S2=1), (S1, S3=0), "(S4=0, S5=1)	(Z1=Z2)&(Z2>Z3)&(Z2>Z4)	Left-oblique, right	Z5
(S2=1), (S1, S3=0), "(S5=0, S4=1)	(Z1=Z2)&(Z2>Z3)&(Z3=Z4)	Left-oblique, right-oblique	Z5
(S2=1), (S1=0, S3=1), "(S4, S5=0)	(Z2>Z1)&(Z2>Z3)&(Z3>Z4)	Left-oblique, front	Z5
(S3=1), (S1,S2=0), (S4,S5=0)	(Z2=Z3)&(Z2>Z1)&(Z3>Z4)	Front	Z7
(S5=1), (S4, S3=0), "(S2, S1=0)	(Z4>Z3)&(Z3=Z2)&(Z2=Z1)	Right	Z6
(S5=1), (S4, S3=0), "(S2=0, S5=1)	(Z4>Z3)&(Z3=Z2)&(Z4>Z1)	Right, left	Z6
(S5=1), (S4, S3=0), "(S1=0, S2=1)	(Z4>Z3)&(Z2=Z1)&(Z4>Z2)	Right, left-oblique	Z6
(S5=1), (S4=0, S3=1), "(S2,S1=0)	(Z4>Z3)&(Z3=Z2)&(Z2>Z1)	Right, front	Z6
(S5, S4=1), (S3=0), "(S2, S1=0)	(Z4>Z3)&(Z3>Z2)&(Z2=Z1)	Right-oblique	Z6
(S5, S4 =1), (S3=0), "(S2=0, S1=1)	(Z4>Z3)&(Z3>Z2)&(Z4>Z1)	Right, right-oblique, left	Z6
(S4=1), (S5, S3=0), "(S2, S1=0)	(Z4=Z3)&(Z3>Z2)&(Z2=Z1)	Right-oblique	Z6
(S4=1), (S5, S3=0), "(S2=0, S1=1)	(Z4=Z3)&(Z3>Z2)&(Z3>Z1)	Right-oblique, left	Z6
(S4=1), (S5, S3=0), "(S1=0, S2=1)	(Z4=Z3)&(Z3>Z2)&(Z2=Z1)	Right-oblique, left-oblique	Z6
(S4=1), (S5=0, S3=1), "(S2, S1=0)	(Z3>Z4)&(Z3>Z2)&(Z2>Z1)	Right-oblique, front	Z6

In this research, the generalization process is the ability to correctly classify patterns that are not included in the training set. This generalization skill is inherent in the structure of neural networks models, including the RAM-based network. However, there is no generalization property in the RAM neurons. The generalization aspect is related to the ability to generalize at the pattern recognition level. This means that from an unknown pattern the most similar trained pattern can be recalled. Therefore, an unknown pattern is considered as a noise version of a trained pattern.

The performance of RAM-bNN architecture is evaluated in several experiments. The numbers of neurons are determined based on the patterns formed in the environment with walls and obstacles in different shapes and sizes. It is created as the mobile robot-working domain. To evaluate this performance of the approach, 20-bits of neuron is chosen with 4-bits per neuron in two microcontrollers. The training process of all of the patterns is presented in increasing order. In this research, 20 input patterns are defined to describe every environmental situation. Starting from the class of zeros until 20 patterns of all the class. The generalization process for unseen patterns is described in Table 5.

4.2 Single Robot Experiments

The performance of the proposed pyramid RAM-based neural network architecture is evaluated in several experiments involving mobile robot movement. Experiments are conducted to demonstrate the ability of the single mobile robot and swarm robots to recognize various unknown environments. In the single mobile robot experiment, a maze environment area containing walls is created as the mobile robot-working domain. Obstacles in different shapes and sizes were also used as walls to configure the environmental patterns. The RAM-based neural network is learnt in parallel because many physical RAMs are used. As such, processing can take place faster and it increases the accuracy in identifying environmental patterns as compared to the monolithic architecture and modular architecture proposed by do Valle Simoes [12]. In this work every group of neurons in accordance with the sensor group. Therefore, the number of neurons can be minimized and the memory is not filled even when adequate training has been achieved. This could enhance the generalization process to correctly classify patterns that are not included in the training set.

Table 6. Pulse width modulation (PWM) control

PC7	PC6	PC5	PC4	Data	PD4	PD5	Notation
Left motor		Right motor		Hexa number	Left	Right	
0	0	0	0	00h	00h	00h	Stop
1	1	0	1	0dh	20h	50h	Turn right with ¼ PWM
1	1	1	0	0eh	30h	50h	Turn right with ½ PWM
0	1	1	1	07h	48h	20h	Turn left with ¼ PWM
1	0	1	1	0bh	48h	30h	Turn left with ½ PWM
1	1	1	1	0fh	48h	50h	Forward

The experiments were conducted to utilize a single mobile robot operating in the maze with some obstacles, as shown in Fig. 6. The results show that mobile robots try to avoid any existing obstacles and to reach the destination point. The mobile robot avoids the obstacles in the environment and its movement was adapted to the real-time PWM data. The PWM data for controlling the mobile robot

actuator is shown in Table 6. The results obtained, which the mobile robot managed properly in past response to any given obstacle and able to reach the end goal.

The mobile robot managed to avoid an obstacle in the front of it and was able to detect the presence of the obstacle as close as 20 cm, as shown in Fig. 6. The infrared sensor readings were taken from the front left and back left side. The results show that the mobile robot was successfully able to follow the left wall and right wall without hitting the walls. It was able to maintain a minimum distance of 10 cm from the wall, as shown in Fig. 6. In a concave environment the mobile robot was capable of avoiding a 45° obstacle angle. Considering the four experiments above, the mobile robot was able to move and maneuver to avoid the obstacles in front of it and to follow the wall. There was no situation where the mobile robot was not able to see the obstacles.

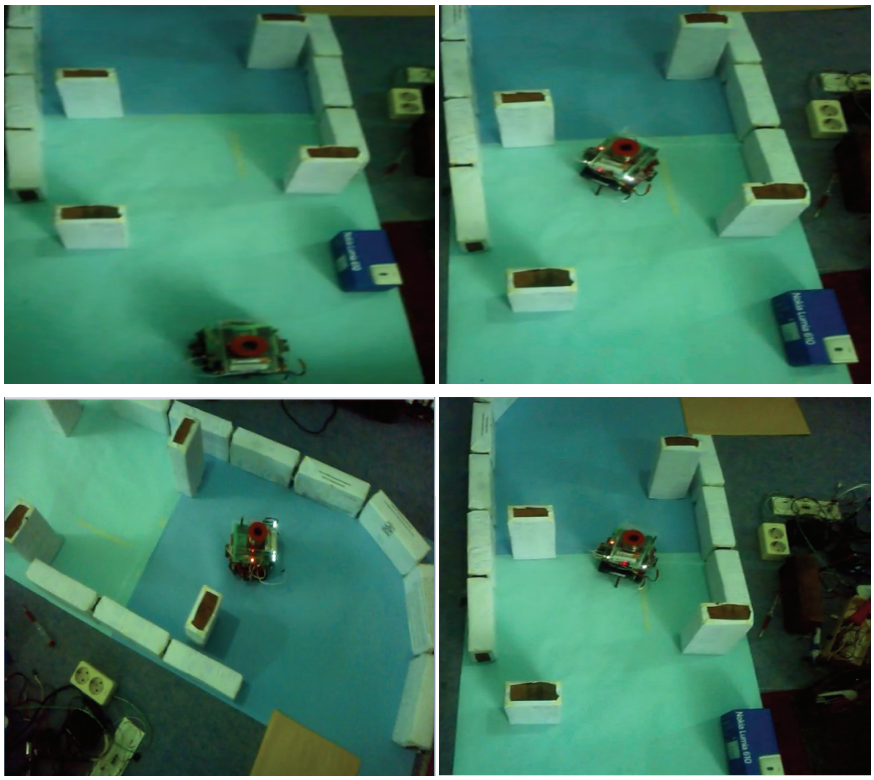


Fig. 6. Experiment with single robot.

The variation of data output for controlling the mobile robot actuator according to the pulse width modulation (PWM) regulation is shown in Fig. 7. The PWM value changed under three conditions such as maximum, medium, and minimum. When the mobile robot is working at full speed, its maximum speed value is 4 cm/sec when no obstacle is present. If the obstacle is far away, the mobile robot's at medium speed is 2 cm/sec or the speed value at ½ PWM. If the obstacle is near, the mobile robot's minimum speed is 1 cm/sec or the speed value at ¼ PWM. Fig. 7(a)-(c) respectivelly, show the speed control of a single mobile robot when it moves in three environmental situation, such as turn right, turn left, and open space. The change of speed in a stable condition and a swarm robot's move in smooth trajectories.

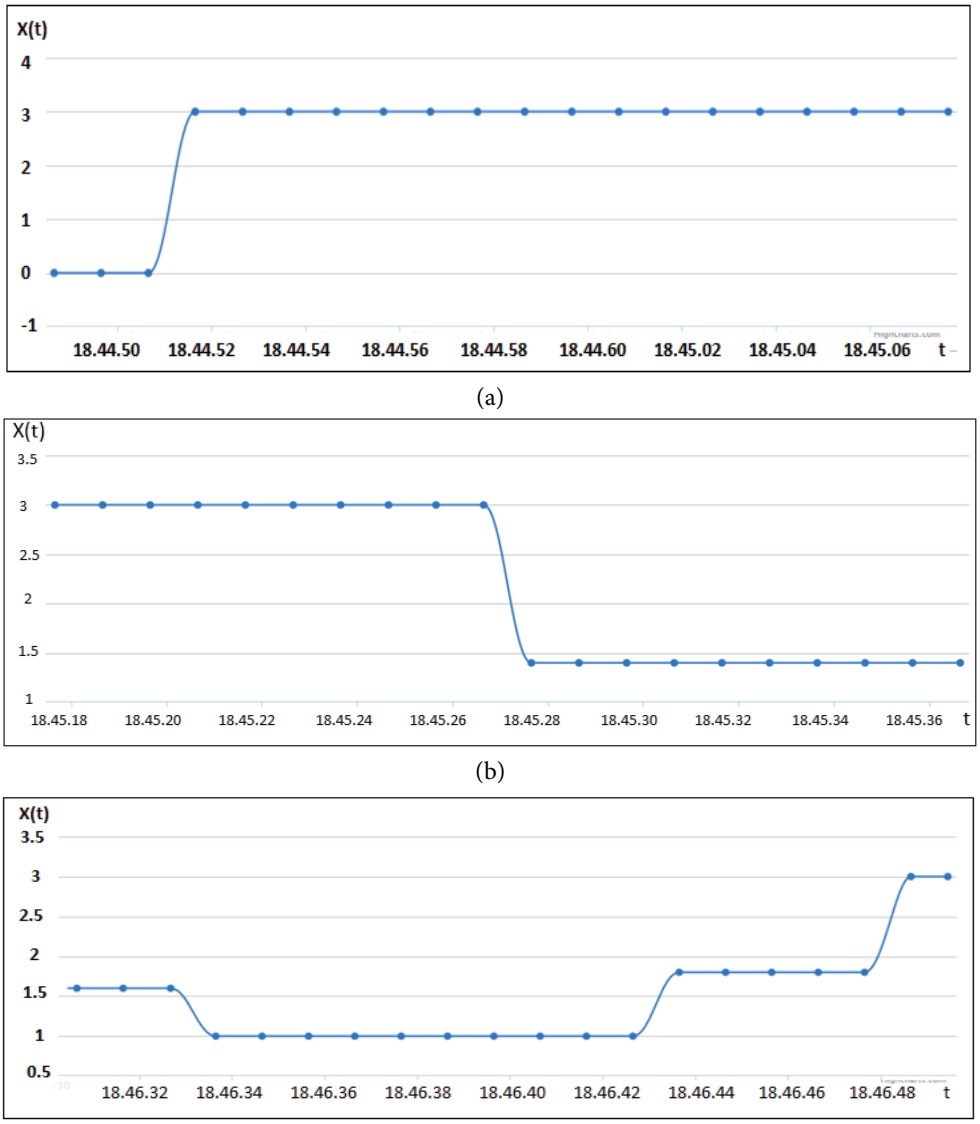


Fig. 7. Single mobile robot movement. (a) Turn right, (b) turn left, and (c) open space.

In the previous experimental results proposed by Nurmaini and Tutuko [11] show that using the RAM-bNN approach has yielded promising results, which indicate that the mobile robot is able to recognize the environmental pattern and to achieve robust performance with a lower computational cost. Furthermore, it successfully performed in several environments and outperformed other controllers, including logic function and fuzzy logic, as shown in Fig. 8. As seen from all of the experimental results, this technique could improve single robot recognition level about the environmental pattern, since it allows for fast processing. However, in previous architecture the generalization and memory optimization has not been discussed and the processing data utilized 8-bits. Therefore, the mobile robot moved more slowly because it processes 120-bits per each neuron and had a computational cost of 5-kbytes of neurons. In this research, the architecture becomes a multi-layer

structure in the processing module with generalization ability and memory optimization. Because only 4-bits of MSB data and 1-bit for the unseen pattern are used, then the proposed RAM-bNN approach processes only 20-bits per each neuron. The process speed is increased and produces less computational resources by only 3-kbytes.

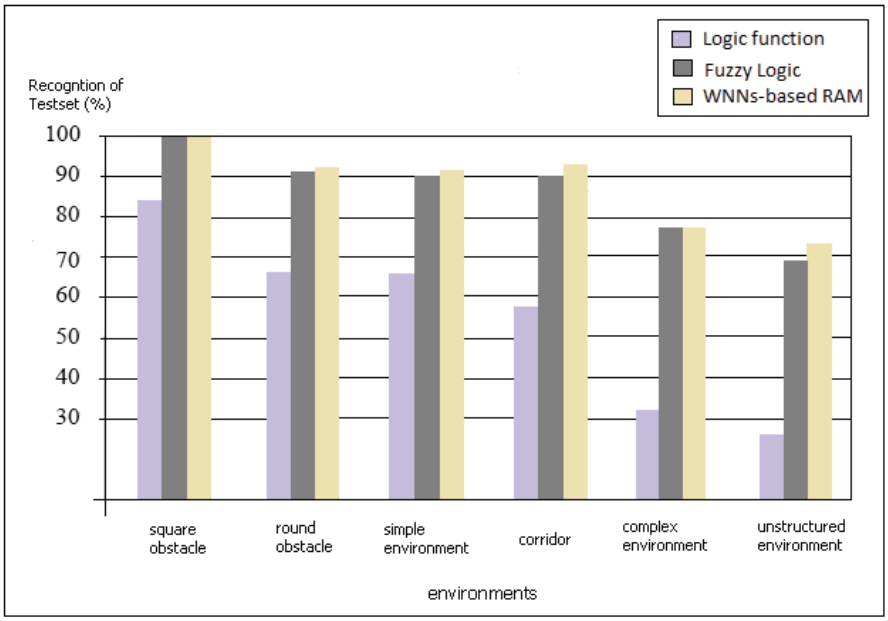
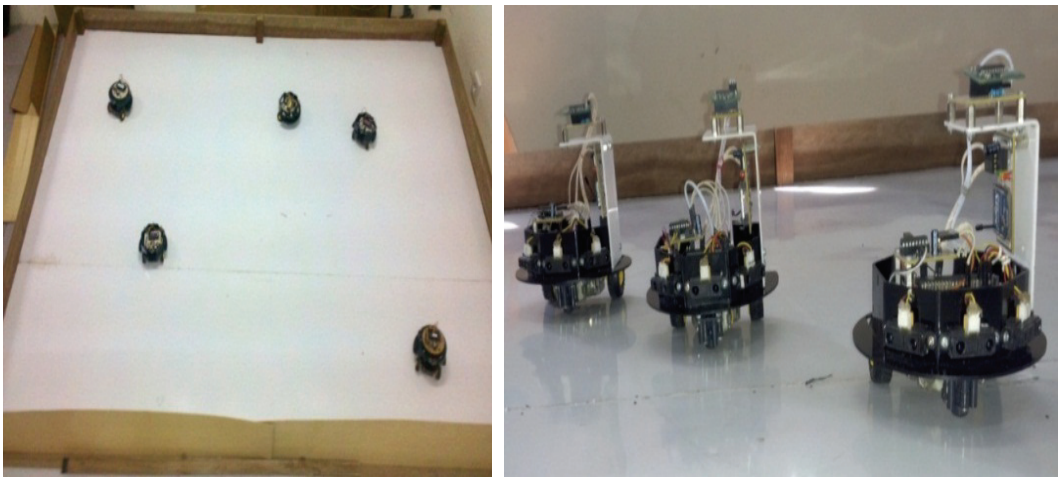


Fig. 8. Comparison of recognition rate for several environments [11].

4.3 Swarm Robots Implementation

A prototype of swarm robots are designed and implemented by using low cost mobile robots, as shown in Fig. 9(a) and (b), respectively. The low cost robot platform uses lithium polymer batteries, low-cost infrared sensors, a TGS 2600 gas sensor, simple actuator, and an ATmega16 microcontroller. For swarm robots communication, X-Bee Pro OEM/ZigBee is utilized. Our design uses an expandable processing board and sensor board with multi-sensing capabilities, which provides a flexible basis if experimental needs in the future, such as complex task with several sensors. Later on, various locomotion and control modules were added to the proposed architecture. Therefore, they can move in all directions.

Communication systems between swarm robots as a mobile sensor node on two directions from point to multipoint or vice versa. Hence, X-Bee is used as a means of communication between the mobile node and local server. In the implementation, sending and receiving 8-bits of data from swarm robots to the local server conduct the process of the communication system, which can be seen in Table 7. In this research, the transmission data of the X-Bee system must use 8-bits of data. This is because when transmission data is above 8-bits, the data error become 100%. For example, when data transmission is 50, 100, 150, and 200, respectively, the measurement error is 0%. But if data transmit is 260, the communication system produces a data error of about 100%. This is due to the fact that X-Bee performs only 8-bits of data transmission, while the 260 exceeds 8-bits or 255h.



(a) (b)

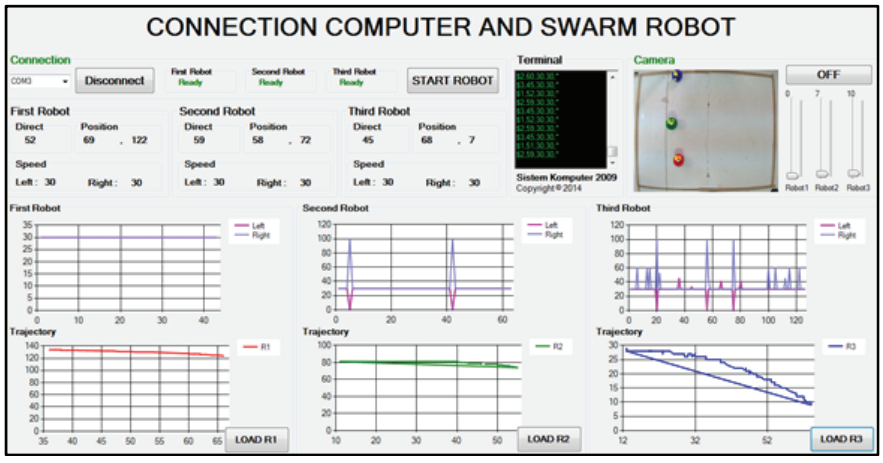
Fig. 9. Experiment with swarm robots. (a) Five mobile robots and (b) swarm robots sample.

The swarm robots must traverse the entire path between the current waypoint to the next waypoint since the distance between the current positions to the next waypoints can vary for each robot. The communication range of the robots is usually limited. This is because the area to be searched can be large and the robots can exit the communication range of their neighbors, which may result in an indefinite wait and stalling by the system. During this situation, the communication system of the swarm robots work in a unidirectional approach and they only continuously detect the environment. They just broadcast their information and send it to the base station as the local server.

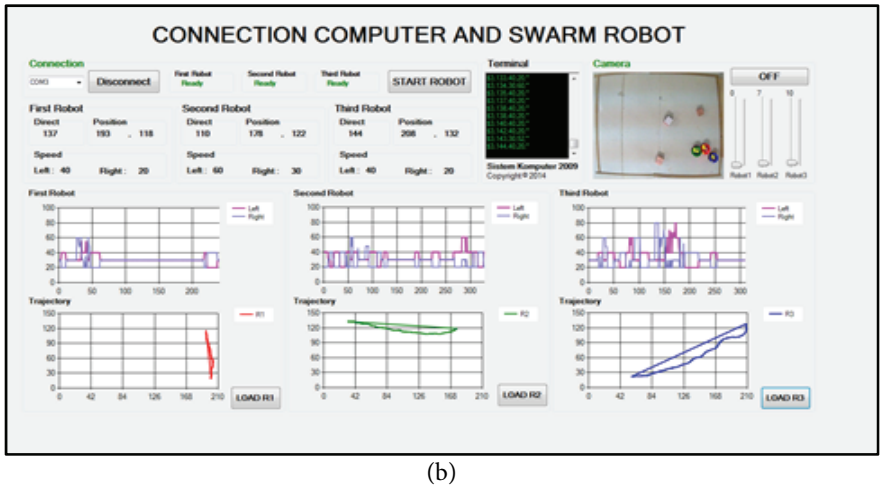
Initial robot positions and their orientations are generated randomly as shown in Fig 10(a)-(c). The current coordinates of the robot are calculated by using the on-board compass. Therefore, the initial swarm robot's position in the environment achieves the new position in the whole swarm. For the first step of the experiment, the swarm robots were placed in a random position. When they move, every position and motion is recorded. Data collection utilizes a computer as a local server and using X-Bee generates all data such as speed, position, and other parameters. In some experiments, such data encounter failures, including a robot being separated from the flock, or losing direction occur. This is mainly due to external factors such as wheel slippage, hardware/software failure, or simply due to sensor mishaps.

Table 7. X-Bee communication results

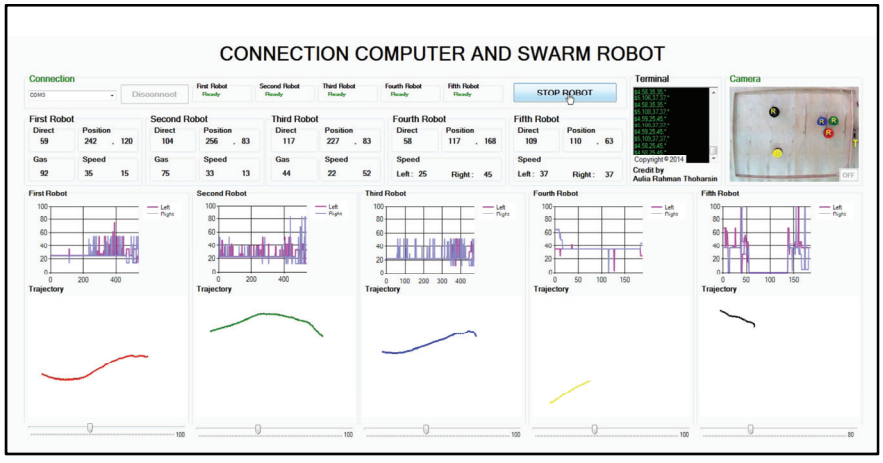
No.	Transmitter	Receiver 1	Receiver 2	Results
1	50	50	50	Right
2	100	100	100	Right
3	150	150	150	Right
4	200	200	200	Right
5	260	4	4	Wrong



(a)



(b)



(c)

Fig. 10. Swarm robots trajectory in simple environment. (a) Three robots and no obstacle, (b) three robots with obstacles, and (c) five robots no obstacle.

5. Conclusion

In this paper, the swarm robots localization system based on a pyramid RAM-based Neural Network has been presented. Several experiments for validating such architecture have been proposed. Our system produces fast responses and is compact in size. Our strategy allows for a mobile robot to learn and move around in an unknown environment. The results show that the mobile robots have the ability to detect and avoid obstacles and are capable of localizing the target in real time.

We have proposed a pyramid RAM node model. Its architecture differs in the way it accesses the contents. Due to this model has generalization of the original RAM. The proposed architecture does not only access the addressed content in order to calculate the neuron output, but also the generalization region of this content. The implementation of pyramid RAM-based Neural Network architecture only needs a certain amount of RAM and produces a simple control algorithm. This architecture is easily implemented in hardware, like a microcontroller, for improving the computational performances.

The main circumstances of choosing a simple RAM-based Neural Network architecture for the localization of swarm robots are shown, as is their importance to the environmental recognition level. The advantages of the employing the microcontroller are the great facilities of the hardware implementation of the neural network. This fact increases the swarm robot's performance while it permits neural network appliances in real time environmental localization. In the future, the Particle Swarm Optimization technique will be combined with the pyramid RAM-based Neural Network to improve the performance of the swarm robots in a dynamic environment.

Acknowledgement

This research is supported by the Robotic and Control Research Lab of the Computer Engineering Department in the Faculty of Computer Science at Sriwijaya University. This research was funded by Competitive Research Grants from Sriwijaya University.

References

[1] B. McElroy, M. Gillham, G. Howells, S. Spurgeon, S. Kelly, J. Batchelor, and M. Pepper, "Highly efficient localisation utilising weightless neural systems," in *Proceedings of 2012 European Symposium on Artificial Neural Networks*, Bruges, Belgium, 2012, pp. 543-548.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA: MIT Press, 2005.

[3] M. Buehler, K. Iagnemma, and S. Singh, *The 2005 DARPA Grand Challenge: The Great Robot Race*. Berlin: Springer, 2007.

[4] M. Buehler, K. Iagnemma, and S. Singh, *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*. Berlin: Springer, 2009.

[5] H. Lang, Y. Wang, & C. W. De Silva, "Mobile robot localization and object pose estimation using optical encoder, vision and laser sensors," in *Proceedings of IEEE International Conference on Automation and Logistics (ICAL2008)*, Qingdao, China, 2008, pp. 617-622.

[6] A. Napier, G. Sibley, and P. Newman, "Real-time bounded-error pose estimation for road vehicles using vision," in *Proceedings of 2010 13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Funchal, Portugal, 2010, pp. 1141-1146.

[7] A. Martinelli, "The odometry error of a mobile robot with a synchronous drive system," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 3, pp. 399-405, 2002.

[8] K. S. Chong and L. Kleeman, "Accurate odometry and error modelling for a mobile robot," in *Proceedings of 1997 IEEE International Conference on Robotics and Automation,* Albuquerque, NM, 1997, pp. 2783-2788.

[9] Y. Sun, J. Xiao, and F. Cabrera-Mora, "Robot localization and energy-efficient wireless communications by multiple antennas," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2009)*, St. Louis, MO, 2009, pp. 377-381.

[10] M. Conforth and Y. Meng, "An artificial neural network based learning method for mobile robot localization," in *Robotics Automation and Control*. Vienna: i-TECH, 2008, pp. 103-112.

[11] S. Nurmaini and B. Tutuko, "A new classification technique in mobile robot navigation," *Telkomnika*, vol. 9, no. 3, pp. 453-464, 2011.

[12] do Valle Simoes, "An embedded evolutionary controller to navigate a population of autonomous robots," in *Frontiers in Evolutionary Robotics*. Vienna: i-TECH, 2008, pp. 439-464.

[13] I. Aleksander, M. De Gregorio, F. M. G. Franca, P. M. V. Lima, and H. Morton, "A brief introduction to Weightless Neural Systems," in *Proceedings of European Symposium on Artificial Neural Networks (ESANN)*, Bruges, Belgium, 2009, pp. 299-305.

[14] S. Nurmaini, S. Z. M. Hashim, and D. N. A. Jawawi, "Modular weightless neural network architecture for intelligent navigation," *International Journal of Advances in Soft Computing and its Applications*, vol. 1, no. 1, pp. 1-18, 2009.

[15] I. Aleksander, W. V. Thomas, and P. A. Bowden, "WISARD: a radical step forward in image recognition," *Sensor Review*, vol. 4, no. 3, pp. 120-124, 1984.

[16] W. K. Kan and I. Aleksander, "A probabilistic logic neuron network for associative learning," in *Neural Computing Architectures*. Cambridge, MA: MIT Press, 1989, pp. 156-171.

[17] I. Aleksander, "From WISARD to MAGNUS: a family of weightless virtual neural machines," in *RAM-Based Neural Networks*. Singapore: World Scientific, 1998, pp. 18-30.

[18] J. G. Taylor, "Spontaneous behaviour in neural networks," *Journal of Theoretical Biology*, vol. 36, no. 3, pp. 513-528, 1972.

[19] R. G. Bowmaker and G. G. Coghili, "Improved recognition capabilities for goal seeking neuron," *Electronics Letters*, vol. 28, no. 3, pp. 220-221, 1992.

[20] I. Aleksander, "Ideal neurons for neural computers," in *Parallel Processing in Neural Systems and Computers*. Amsterdam: Elseriver, 1990, pp. 225-228.

[21] A. F. De Souza, F. Pedroni, E. Oliveira, P. M. Ciarelli, W. F. Henrique, L. Veronese, and C. Badue, "Automated multi-label text categorization with VG-RAM weightless neural networks," *Neurocomputing*, vol. 72, no. 10, pp. 2209-2217, 2009.

[22] M. A. Hannan Bin Azhar and K. R. Dimond, "Design of an FPGA based adaptive neural controller for intelligent robot navigation," in *Proceedings of Euromicro Symposium on Digital System Design*, Dortmund, Germany, 2002, pp. 283-290.

[23] S. S. Botelho, E. do Valle Simões, L. F. Uebel, and D. Barone, "High speed neural control for robot navigation," in *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, Beijing, China, 1996, pp. 1956-1959.

[24] Q. Yao, D. Beetner, D. C. Wunsch, and B. Osterloh, "A RAM-based neural network for collision avoidance in a mobile robot," in *Proceedings of the International Joint Conference on Neural Networks*, Portland, OR, 2003, pp. 3157-3160.

[25] B. McElroy and G. Howells, "Automated adaptation of input and output data for a weightless artificial neural network," *International Journal of Database Theory and Application*, vol. 4, no. 3, pp. 49-58, 2011.

[26] P. Coraggio and M. De Gregorio, "WiSARD and NSP for robot global localization," in *Nature Inspired Problem-Solving Methods in Knowledge Engineering.* Heidelberg: Springer, 2007, pp. 449-458.

[27] M. De Gregorio, "Active and reactive use of virtual neural sensors, in *Proceedings of European Symposium on Artificial Neural Networks (ESANN)*, Bruges, Belgium, 2008, pp. 349-354.

Siti Nurmaini http://orcid.org/0000-0002-8024-2952

Dr. Siti Nurmaini was born in Palembang, August 2, 1969. Currently, she is a lecturer in department of computer engineering, Faculty of Computer Science, University of Sriwijaya, Indonesia. She is graduated from department of electrical engineering-UNSRI, Master of Engineering from Institute Technology Bandung (ITB) in majoring of control system and computer. She received a Ph.D. degree from Universiti Teknologi Malaysia-UTM majoring of intelligent control. She is very interesting in research area of the soft computing, control system, embedded system and robotic.

Ahmad Zarkasi

Ahmad Zarkasi, M.T., was born in Palembang, August 25, 1979. He received Master of Engineering from Institute Technology Bandung (ITB) in majoring of computer engineering, in 2013. His research interests are in the area of Microprocessors, SoC (System-On-Chip), embedded system, and robotics. They include topics such as WNNs in robotic system, Pattern Recognition for robotic mapping.