Fault Tolerance Technique Using Bidirectional Hetero-Associative Memory for Self-Reconfigurable Programmable Matter

Abdallah Makhoul and Jad Bassil
University of Franche-Comté
FEMTO-ST Institute, CNRS, Montbéliard, France
{first}.{last}@femto-st.fr

***Abstract*—Programmable Matter (PM) based on modular robots is a material which can be reprogrammed to have different shapes and to change its physical properties on demand. It can be deployed in several domains and has a variety of applications in construction, surgery, environmental science, space exploration, etc. PM is composed of a big number of limited resources connected robots called modules or particles to form its shape. These modules communicate with each other and move around each other dynamically in order to switch from one configuration to another. Due to the limited resources of modules and the high number of packets that transit within the system, it is very challenging to ensure packet delivery with high reliability. In this paper, we are using a Bidirectional Hetero-Associative Memory (BHAM) networks to improve the reliability and fault tolerance in PM. The idea is to let modules sending packets with smaller size without loosing any information. Furthermore, this model is also capable to remove noise from received packets. The proposed approach is tested on a real programmable matter blinky blocks platform as well as via simulations. We studied two versions of artificial neural networks based on storage capacity. The experimental results show that the studied approach is efficient in reducing the size of packets that transit in the system thus reducing energy consumption and it is capable to detect and remove noise and correct noisy packets.**

***Index Terms*—Modular robots, programmable matter, artificial neural networks, reliability, energy saving**

I. Introduction

The vision for programmable matter (PM) [1] is to create a material which can be reprogrammed to have different shapes and to change its physical properties. PM could be deployed in different domains while promising to have a variety of applications in construction, surgery, environmental science, space exploration, etc. For example, imagine a material that has been pre-programmed so that it can transform itself in complicated ways in response to environmental events. Examples of exciting future applications are robotic ensembles monitoring hostile environments (e.g., nuclear), delivering drugs in the human body, educational robots and a new set of robotic toys [1], [2].

There are various ways to implement programmable matter. One is to build it as a huge modular self-reconfigurable robot composed of a large set of independent micro-robots connected to each other. The connections to one another form the overall shape of the system and called modular robot. These micro-robots can have different forms (spherical, cubic, etc.). Beyond sensing, processing and communication capabilities, a modular robot includes actuation and motion capabilities that allow it to reconfigure its shape by rearranging connections between modules [3] [4]. They must be able to stick to each other and move around their neighbours. Hence, the main task of a modular robots system is to reconfigure its shape in order to accommodate for variable conditions that need to be met in order to complete a given final goal. For example, one can program the robots so that, starting from an initial configuration without any holes, they can self-reconfigure into a line containing all the robots, without ever breaking the connectivity of the system [4], [5]. In figure 1 we show the self-reconfiguration of modular robots from the initial shape to the goal which is done in a distributed manner.

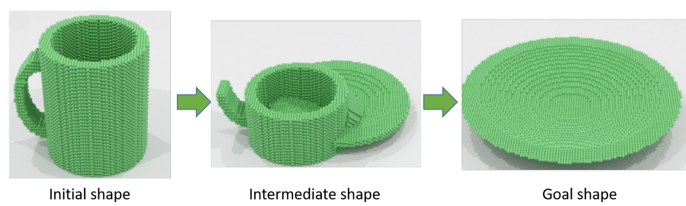

Fig. 1. An example of self-reconfiguration of Modular Robots.

Modular robotic systems are distributed systems composed of several thousands of particles or modules with low computational and energy resources. Furthermore, the connections between modules are made via connectors that maintain connections between them while exerting shear forces. As it turns out, the strength of the connectors directly influences the robustness of the structure of the system and may lead to a noisy data transmission. Therefore, one of the main challenges in this system is to detect faulty packets while minimizing energy consumption. Saving energy in a modular robot can be made by two ways, the first one by reducing the number of movements when it changes its shape from an initial configuration to a final configuration [6]. The second one is to reduce the size of the packets/data transmitted in the system [7]. On the other hand, due to the connector fragility and the system dynamics, a packet may not be successfully transmitted to the destination and this can lead to several dysfunctions.

In this paper, an artificial neural network model is proposed based on Bidirectional Hetero-Associative Memory (BHAM) [8] learning model allowing a modular robot system to correct erroneous packets and send packets with less size. The idea is that the modules send packets with small size to save energy. Then, once these packets are received, the destination node will regenerate the original packets using the BHAM and the association model. Indeed, the BHAM is capable of recalling the associations between packets. This model is tested on a real blinky blocks [9] programmable matter platform and via simulations. The obtained results showed the efficiency of the proposed approach in reducing energy consumption of the modules and detecting and correcting erroneous packets.

In the literature several compression methods with error detection have been proposed to low resource networks (e.g. Wireless sensor networks) [10], [11], [12], [13], [14]. Unfortunately, these compression techniques still very complex, require high processing capabilities and need additional communication for programmable matter based modular robots with very low computational resources. Therefore, in this paper we propose a new technique dedicated to PM based Bidirectional Hetero-Associative Memory with low complexity.

The reminder of this paper is organized as follows. Section II presents a background about Bidirectional Hetero-Associative Memory Networks. In Section III the methodology of using BHAM for programmable matter and modular robots is presented. Section IV is dedicated to experimental and simulation results. Section V concludes the paper and gives some directions for future work.

II. Bidirectional Hetero-Associative Memory (BHAM) Network

Bidirectional Hetero-Associative Memory (BHAM) [15], [8] is a supervised neural network model with the ability to memorize the associations between input and output patterns without need of continuous learning. These associations will be stored only once and recalled by the BHAM for patterns recognition. These networks are called hetero-associative memory, where for a given input pattern, it returns another output pattern of a different size. This property is very useful for programmable matter (PM) where the objective is to reduce the size of the transmitted packets. The main objective of this work is to retrieve a packet (pattern) given a packet with less size and even noisy packets.

The architecture of a BHAM is presented in Figure 2. The idea is to use this network in order to encode the associations between two sets of vectors A and B. It is a bidirectional network, in other terms if the input is a vector of the Set A with dimension n then the network recalls the corresponding vector of dimension m in the Set B ; Similarly, if the input is a vector of the Set B with dimension m then the network recalls the corresponding vector of dimension n in the Set A. In this case we consider $m < n$.

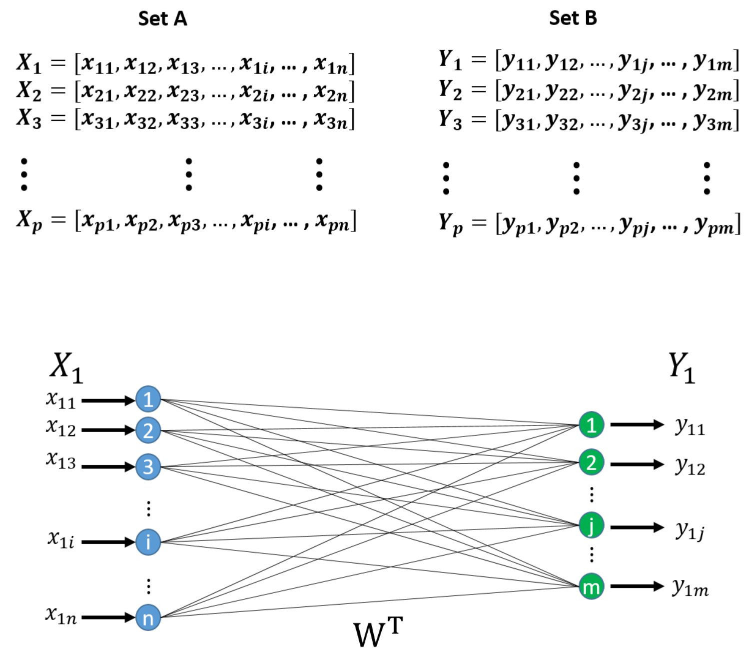

Fig. 2. BHAM Architecture.

The learning step of BHAM is mainly the computation of a weight matrix W between the p pairs of patterns. This matrix is calculated by following Equation 1.

$$W = \sum_{l=1}^{p} X_l Y_l^T \tag{1}$$

where, X and Y are bipolar vectors with elements values are of 1's or -1's.

After the calculation of the matrix W, the following equations will be used in order to recall Y_l for corresponding X_l and recalls X_l for corresponding Y_l.

$$X_l = SGN(WY_l) \tag{2}$$

$$Y_l = SGN(W^T X_l) \tag{3}$$

where SGN is a bipolar threshold function similar to the classical threshold function and defined as follows:

$$SGN(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{if } x < 0 \end{cases} \tag{4}$$

In this paper we propose to use the BHAM model in order to construct associations between two sets of packets of different sizes m and n. The idea is to let a module transmits packets of size m and the destination module will retrieve the original packet of size n by recalling the BHAM associations, where $m < n$ in order to save energy and increase reliability and fault tolerance.

III. BHAM FOR PROGRAMMABLE MATTER : METHODOLOGY

There is a set of actions encoded as packets that can be used in a modular robots. In Figure 3 we give an example of encoding some actions in PM as bipolar vectors. For example in a scaffold [4] topology, several actions have been defined in order to achieve the goal shape (e.g. rotate: asks the modules to turn around its neighbor, Idle modules: should wait to be called by other modules, Free Agent: when a module is called to enter the reconfiguration scene, Coordinators packets: means that the module is docked in the root position of a tile, etc.). In this context, we consider that a module in a modular robot must send a packet from a set P of packets as shown in 3 where each packet has a size of n bits. To reduce the energy consumption of the modules, our objective is to reduce the size of the messages/packets exchanged between modules. Furthermore, it is more likely to have errors, noise and faulty packets in packets with larger size. Therefore, one solution is to reduce the size of the transmitted packets while associating them to the original large-size packets using the BHAM model. This ensures the optimisation of the energy consumption and better fault tolerance and reliability in modular robots and programmable matter.

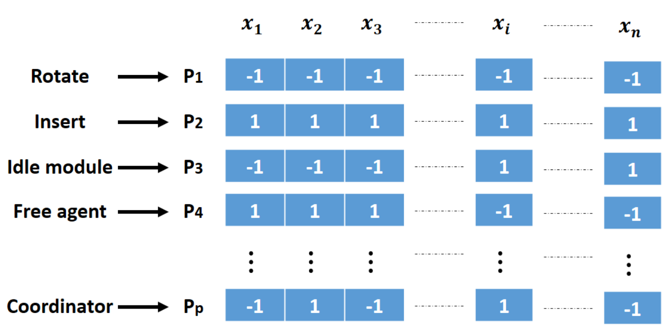

Fig. 3. An example of encoding PM actions (packets) into Bipolar 2D-Patterns Map.

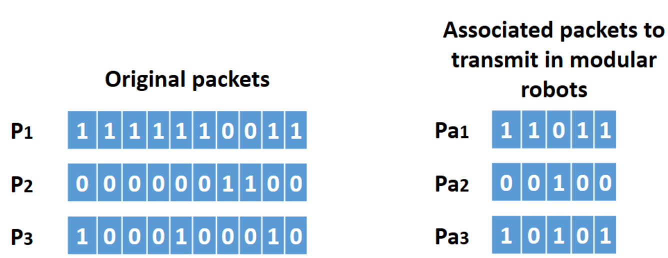

Fig. 4. An example of packets associations.

The BHAM model learns the associations of packets/vectors in order to associate each packet of size n to a less size packet of size m as presented in the example of Figure 4. In this figure we show an example of original packets of size 10 bits that will be replaced by packets of size 5 bits to be transmitted in the network of modular robots. In a second step, these vector/packets will be transformed to bi-polar vectors X and Y as presented in figure 5. In this figure, the BHAM learns the associations encoded into the two vectors X and Y. We notice that both, input and output have the same number of vectors/packets. However, as BHAM is hetero-associative memory model, the sizes of input and output vectors are different. In the example of Figure 5 it is shown that the size of the input packets is 10, while the size of the output packets is 5.

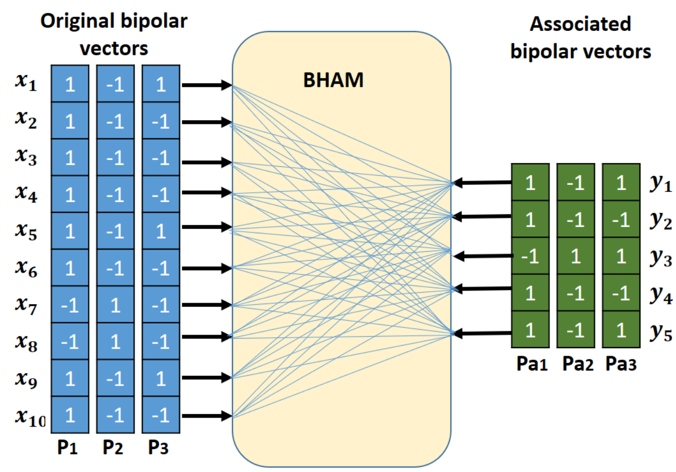

Fig. 5. Bipolar vector associations using BHAM.

A. Illustrative Example

Let us consider the example presented in Figure 5. Instead of transmitting the packets P_1, P_2, and P_3 the modules send their associated packets P_{a1}, P_{a2}, and P_{a3}. Following Equation 1 the weight matrix W is calculated:

$$W = \begin{pmatrix} 3 & 1 & -1 & 1 & 3 \\ 1 & 3 & -3 & 3 & 1 \\ 1 & 3 & -3 & 3 & 1 \\ 1 & 3 & -3 & 3 & 1 \\ 3 & 1 & -1 & 1 & 3 \\ 1 & 3 & -3 & 3 & 1 \\ -3 & -1 & 1 & -1 & -3 \\ -3 & -1 & 1 & -1 & -3 \\ -3 & -1 & 1 & -1 & -3 \\ 3 & 1 & -1 & 1 & 3 \\ 1 & 3 & -3 & 3 & 1 \end{pmatrix}$$

To retrieve P_1 (corresponding bipolar vector X_1) from the transmitted packet P_{a1} (corresponding bipolar vector Y_1) and following Equation 2 we compute $X_1 = SGN(WY_1)$ as follows:

$$SGN(\begin{pmatrix} 3 & 1 & -1 & 1 & 3 \\ 1 & 3 & -3 & 3 & 1 \\ 1 & 3 & -3 & 3 & 1 \\ 1 & 3 & -3 & 3 & 1 \\ 3 & 1 & -1 & 1 & 3 \\ 1 & 3 & -3 & 3 & 1 \\ -3 & -1 & 1 & -1 & -3 \\ -3 & -1 & 1 & -1 & -3 \\ -3 & -1 & 1 & -1 & -3 \\ 3 & 1 & -1 & 1 & 3 \\ 1 & 3 & -3 & 3 & 1 \end{pmatrix} \times \begin{pmatrix} 1 \\ 1 \\ -1 \\ 1 \\ 1 \end{pmatrix}) = SGN \begin{pmatrix} 9 \\ 11 \\ 11 \\ 11 \\ 9 \\ 11 \\ -9 \\ -9 \\ 9 \\ 11 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \end{pmatrix}$$

So the corresponding vector/packet X_1 is found. The same operations can be done to find the other associations between the original and corresponding packets/vectors while recalling the BHAM network.

B. Recall of faulty packets using BHAM

Let us consider in this example that the packet P_{a1} is received by the modules with errors. Assume that 2 of 5 bits (the first and third bits in the vector) of the bipolar vector X_1 are distorted due to noise and then the received bipolar vector at one of the modules of the modular robot for P_{a1} is $Y_1^{'} = ([-1, 1, 1, 1, 1])$ instead of $([1, 1, -1, 1, 1])$. Then to retrieve the received packet, we compute the product $WY_1^{'}$ as follows:

$$SGN(WY_1^{'}) = SGN(W \times \begin{pmatrix} -1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}) = SGN(\begin{pmatrix} 1 \\ 3 \\ 3 \\ 3 \\ 1 \\ 3 \\ -1 \\ -1 \\ 1 \\ 3 \end{pmatrix}) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \end{pmatrix}$$

This result shows that the corresponding packet or vector of the received noisy vector $Y_1^{'}$ is X_1. Therefore, if the received vector is corrupted the BHAM network will be able to recall and retrieve the correct vector. Furthermore, BHAM networks are known to be more reliable with fault packets due to missing bits than of erroneous bits.

C. Size of the associated vectors/packets

In this section, we will discuss how to choose the size m of the packets/vectors P_{ai} to transmit within the network. Indeed, the size of these packets will depend on the total number of different packets (vectors) corresponding to the events that occur in a modular robots systems (e.g. rotate, idle module, free agent, coordinator, etc.). Even if the original packet size is large and the total number of different packets to handle/store is small then the size of the associated packets P_{ai} will be small. Let assume the size of the original packet is n (e.g $n > 100\ bits$) and the number of different packets/events is p (e.g. 10 events), then the size of the associated packets will be less or equal to p (e.g. $m \leq 10\ bits$) and this can ensure fault tolerance and reliability in the system. In other terms, the size of the packets to transmit should not exceed the number of associated pairs of packets.

D. Memory capacity

The memory capacity of a BHAM network is defined as the minimum between the number of inputs in the first layer (the size of the original vector or packet) and the number of output in the second layer (the size of the vector or packet to be transmitted by the modules). Therefore, the limit of storage of BHAM is linear with this minimum and then BHAM can store only a small number of packets [16]. This can be sufficient in some cases or algorithms dedicated to Programmable Matter but not in all types of applications. On the other hand, it has been shown that the exponential nonlinearity content addressable memory has a high storage capacity which is exponential with the number of bits and can ensure a higher error correcting capabilities [17]. The authors in [17] present a neural network model for the BHAM with exponential nonlinearity called Exponential BHAM (eBHAM) in order to have more memory capabilities. Hence, the eBHAM model has higher capacity for packets pair or vector pair storage than the BHAM networks. For the programmable matter applications and depending on the needs for each application we can use eihter BHAM as explained before or the eBHAM model. The using of eBHAM will be explained in the next section.

1) Exponential BHAM (eBHAM) for Programmable Matter:

The eBHAM model can be used in the context of Programmable Matter. For this, let us consider the example presented in Figure 5. We consider we have several combination pairs given by $\{(P_1,\ P_{a1}),\ (P_2,\ P_{a2}),\ (P_3,\ P_{a3}),\ ...,\ (P_p,\ P_{ap})\}$ with their respective bipolar vectors $\{(X_1,\ Y_1),\ (X_2,\ Y_2),\ (X_3,\ Y_3),\ ...,\ (X_p,\ Y_p)\}$. The sizes of the original vectors and the transmitted vectors are respectively n and m where $m < n$. In order to retrieve the original vector $X(x_1, x_2, ..., x_n)$ while receiving the transmitted vector $Y(y_1, y_2, ..., y_n)$ with an eBHAM model, we use the following equation for X :

$$X = SGN(\sum_{i=1}^{p} X_i \times \alpha^{Y_i.Y}) \tag{5}$$

where $(X.Y)$ denotes the inner product of vectors X and Y and α is a number greater than one.

For example, to retrieve P_1 (corresponding bipolar vector X_1) from the transmitted packet P_{a1} (corresponding bipolar vector Y_1) using eBHAM and following Equation 5 as follows:

First we compute the inner products as follows:

$$< Y_1.Y_1 >= 5,\ \ < Y_1.Y_2 >= -5\ and\ \ < Y_1.Y_3 >= -1 \tag{6}$$

Then suppose that $\alpha = 2$, we compute:

$\alpha^{Y_i.Y_1}$ for $i = 1, 2, 3$ then we find $(2^5, 2^{-5}, 2^{-1}) = (32, 1/32, 1/2)$ then the retrieved vector is

$$
\begin{aligned}
&SGN(32 \times X_1 + 1/32 \times X_2 + 1/2 \times X_2) \\
&= SGN(32.4, 31.4, 31.4, 31.4, 32.4, 31.4, -32.4, -32.4, 32.4, 31.4) \\
&= SGN(1, 1, 1, 1, 1, 1, -1, -1, 1, 1) \\
&= X_1
\end{aligned}
$$

So the corresponding vector/packet X_1 is found. The same operations can be done to find the other associations between original and corresponding packets/vectors while recalling eBHAM network. Furthermore, eBHAM can be used to recall faulty packets with high capability to correct them. In the literature, we can find several variations of eBHAM that try to outperform the performance of eBHAM in some situations [18], [19], [20]. The use of these networks is very similar to the networks presented in this paper.

IV. EXPERIMENTAL RESULTS

To show the efficiency of the proposed approach, a Python based simulator was developed. Furthermore, a real experimentation was done using the blinky blocks [1] [9] platform. Blinky Blocks are 4 cm cubic modules that connect to form a modular robot. Each module has an ARM Cortex M0 32-bit controller and can be attached with up to 6 neighbors using magnets. They can communicate neighbor-to-neighbor using serial links present on the 6 block faces. A special module as shown in Figure 6 is connected to a power supply and shares the power with the ensemble using dedicated pins. The same module can be connected to a computer to receive and transmit the program to be executed to all the modules. All Blinky Blocks execute the same program. They can light up in colors using RGB leds and can be programmed to change their color.

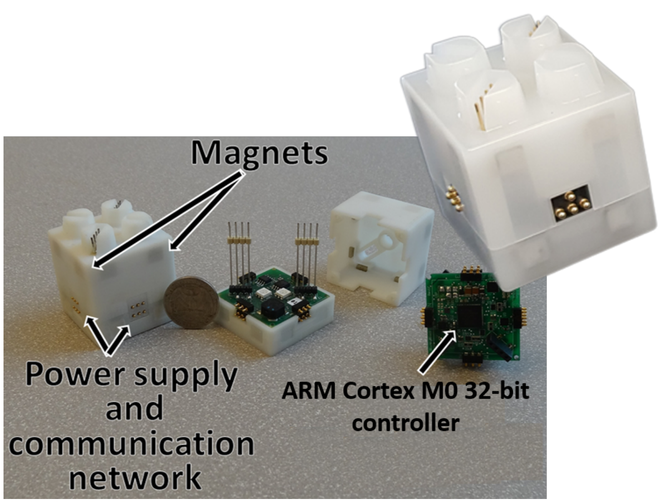

Fig. 6. A Blinky Block module.

We conducted experiments on 24 Blinky Blocks placed in a 6x4 rectangle and we varied the size of the BHAM network (i.e. the size of the bipolar vectors as well as the number of the associated packets/patterns) randomly. For each run of simulation or experimentation, after the selection of packets, the weight matrix was computed. Then the blinky blocks modules send low size packets where the destination modules use the BHAM model to recall orginal packets. The objective of our simulations and experimentation is to show the ability of BHAM used for programmable matter to reduce energy consumption and to correct faulty/noisy packets. All the results presented after are the average of several executions (5 at least).

A. Original packets recalling without noise

The objective of these series of experimentation was to test the proposed approach in recalling non-faulty packets. In other words, the idea was to test the capability of the network to recall original large-size packets from transmitted low-size packets in order to reduce the energy consumption and the probability of errors. For each execution of simulation or experimentation, after computing the weight matrix, the proposed approach was tested to recall non-noisy packets. In all these simulations and experimentation, it was clear that BHAM was able to recall original packets with 100% of success.

B. Original packets recalling from faulty-packets using BHAM

In these series of experimentation, the performance of the proposed approach was tested on noisy packets. The objective is to show the ability of a BHAM network in recalling the correct associated packet from a noisy received packet. We induced errors in the sent packets by flipping b bits at random positions. We varied b from 1 to 3 when the packet's size is 5, from 1 to 5 when the packet's size is 10 and from 1 to 7 when the packet's size is 15. We also varied the percentage of erroneous packets transmitted in the network in the same run of the algorithm from 10% to 40%. For each number of erroneous packets and number of erroneous bits in a packet, we calculated the average number of modules which was not able to recall the correct original packet. Furthermore, the number of associated packets pairs was varied.

1) Percentage of correct recall of BHAM:

The objective of this section is to show the efficiency of BHAM in correcting faulty received packets while varying the sizes of the original packets (n) and the associated packets to be transmitted (m). For these series of experimentation we considered three pairs of packets association. The modules of the modular robot exchanged the packets with errors. Moreover, the percentage of faulty packets transmitted in the network was varied. Figures 7, 8 and 9 present the obtained results while varying the couple (m, n) as follows $(5, 10)$, $(10, 20)$, and $(15, 30)$ respectively.

From these results we can clearly notice the ability of BHAM to recall original packets from noisy packets. For instance, for one bit or two bits errors the BHAM can recall almost 100% correct original packets from faulty packets. Moreover, in the worst case where $5O\%$ of the bits in the faulty packets are erroneous and 40% of the packets transmitted in the modular robots system are erroneous, the BHAM was able to correct up to 50% of these erroneous packets. On the other side, we can notice in these figures that when

[1] https://www.programmable-matter.com/technology/blinky-blocks

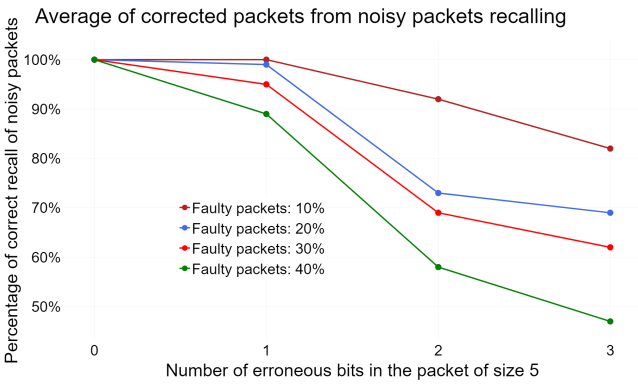

Fig. 7. size of packets: $m = 5$, $n = 10$.

the size of the BHAM (m, n) increases for the same number of the packets pairs association, the percentage of corrected packets increases also. For instance, for 3 erroneous bits in the transmitted packets and 30% of the transmitted packets are noisy, the percentage of corrected packets by BHAM networks of sizes $(5, 10)$, $(10, 20)$ and $(15, 30)$ are respectively 69%, 80% and 98%.

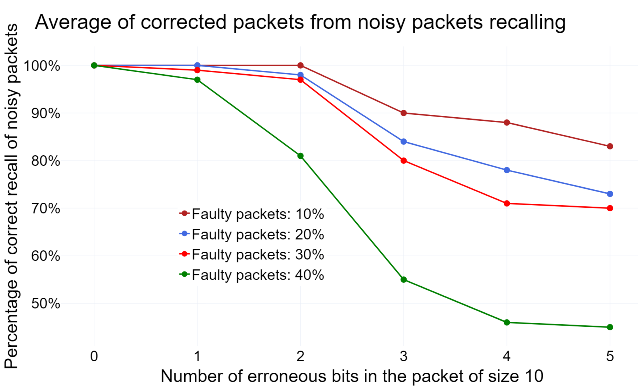

Fig. 8. size of packets: $m = 10$, $n = 20$.

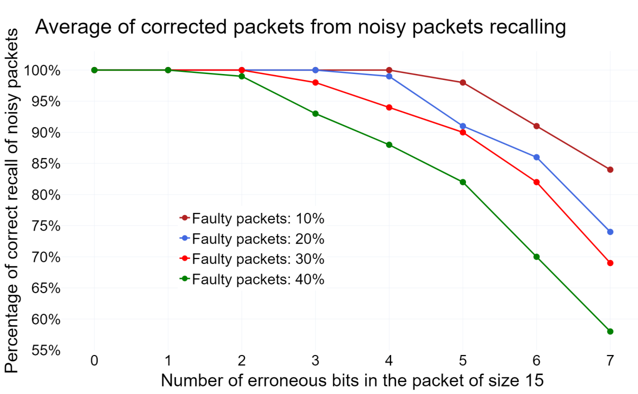

Fig. 9. size of packets: $m = 15$, $n = 30$.

2) Number of packets pairs association: Our objective in this experimentation is to show the behavior of the BHAM network when the number of associated packets pairs increases while the size of the network (m, n) remains unchanged. In Figure 10 we varied this number between 3 and 5 for the same size $(15, 30)$. It is shown in this result that when the number of associated packets pairs increases, the percentage of corrected packets decreases but not drastically. For instance, for the same network size and for packets with 5 erroneous bits, the percentage of corrected packets was behind 100% for three packets pairs and decreases for 4 and 5 associated packets. As mentioned before, to have a reliable programmable matter or modular robot system the size of the transmitted between modules should not exceed the number of associated pairs of packets. Therefore, it is important to choose the size of the packets to transmit according to the number of associated pairs of packets.

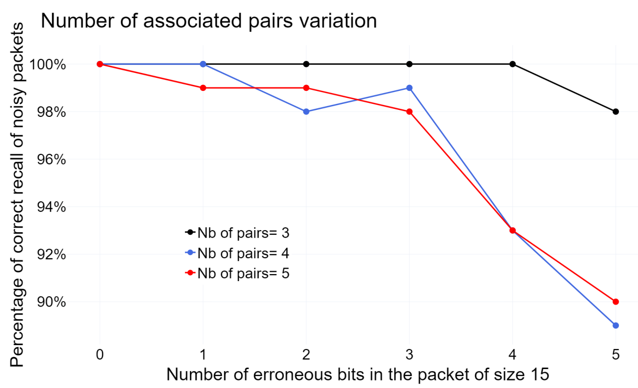

Fig. 10. size of packets: $m = 15$, $n = 30$.

C. Original packets recalling from faulty-packets using eBHAM

The aim of these experimentation and simulations is to show the efficiency of eBHAM in recalling correct packets from faulty received ones. The eBHAM was implemented and tested on the Blinky Blocks platform composed of 24 modules. We consider the size of the eBHAM network as $m = 15$ and $n = 30$. The obtained results are shown in Figure 11. We can clearly see the efficiency of eBHAM in correcting received erroneous received packets. For instance and in the worst case where more than $5O\%$ of the bits in the faulty packets are erroneous and 40% of the packets transmitted in the modular robots system are erroneous, the eBHAM was able to correct up to 60% of these erroneous packets. Furthermore when comparing the two networks eBHAM and BHAM, it is shown that eBHAM has higher capabilities than BHAM in recalling faulty packets and memory capacity.

V. Conclusion and Future Work

In this paper we have proposed an approach that exploit the Bidirectional Hetero-Associative Memory (BHAM) model in order to ensure reliability, fault tolerance and energy saving in programmable matter based on modular robots system. This model is one iteration trained neural network which is capable to storing and recalling original large size packets while receiving transmitted packets with smaller size. This

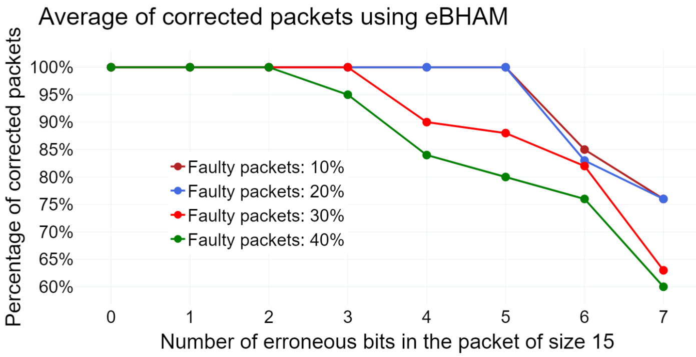

Fig. 11. Packets recalling using eBHAM.

model is based on binary or bipolar vectors association. We also introduced exponential BHAM model that has larger storage capacity.

To show the efficiency of the proposed approach, both simulation and experimentation were done while using a real platform composed of modular robots based on blinky blocks. The obtained results showed that BHAM can recall correct packets from low sizes transmitted packets. Furthermore, we studied the behaviour of the model when packets or vectors are altered by different level of noise (e.g. flipping randomly from 10% to 50% of the bits in the transmitted packets). The obtained result was satisfactory.

Although we obtained acceptable results, it was deduced that recalling correct packets for various incomplete or faulty packets requires a BHAM network of huge sizes, and big amount of memory cells. Therefore, one of our future work will consist on studying a new variation of BHAM networks to improve the storage capacity and fault tolerance. It can be coupled with another recurrent neural networks as a second phase to recover noisy packets.

Furthermore, a second step in our future work is to propose a new BHAM that is work not only with binary values but also with real one. Then a blinky block platform can be modeled as a bidirectional associative memory neural network in a distributed and parallel manner where each module will correspond to one neuron. The idea is to resolve more complex problems like 2D image reconstruction and 3D object recognition by the modular system itself.

VI. Acknowledgment

This work is partially funded by the EIPHI Graduate School (contract “ANR-17-EURE-0002”).

References

[1] S. C. Goldstein, J. D. Campbell, and T. C. Mowry, “Programmable matter,” *Computer*, vol. 38, no. 6, pp. 99–101, 2005.
[2] P. Thalamy, B. Piranda, and J. Bourgeois, “Engineering efficient and massively parallel 3d self-reconfiguration using sandboxing, scaffolding and coating,” *Robotics Auton. Syst.*, vol. 146, p. 103875, 2021.
[3] E. Hourany, C. Stephan, A. Makhoul, B. Piranda, B. Habib, and J. Bourgeois, “Self-reconfiguration of modular robots using virtual forces,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2021, Prague, Czech Republic, September 27 - Oct. 1, 2021*. IEEE, 2021, pp. 6948–6953.
[4] P. Thalamy, B. Piranda, and J. Bourgeois, “Distributed self-reconfiguration using a deterministic autonomous scaffolding structure,” in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS ’19, Montreal, QC, Canada, May 13-17, 2019*, 2019, pp. 140–148.
[5] M. Gerbl and J. Gerstmayr, “Self-reconfiguration of shape-shifting modular robots with triangular structure,” *Robotics and Autonomous Systems*, vol. 147, p. 103930, 2022.
[6] Y. Dai, C.-F. Xiang, Z.-X. Liu, Z.-L. Li, W.-Y. Qu, and Q.-H. Zhang, “Modular robotic design and reconfiguring path planning,” *Applied Sciences*, vol. 12, no. 2, p. 723, 2022.
[7] J. Bassil, M. Moussa, A. Makhoul, B. Piranda, and J. Bourgeois, “Linear distributed clustering algorithm for modular robots based programmable matter,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2020, Las Vegas, NV, USA, October 24, 2020 - January 24, 2021*. IEEE, 2020, pp. 3320–3325.
[8] R. A. Vázquez, H. Sossa, and B. A. Garro, “A new bi-directional associative memory,” in *Mexican International Conference on Artificial Intelligence*. Springer, 2006, pp. 367–380.
[9] B. T. Kirby, M. Ashley-Rollman, and S. C. Goldstein, “Blinky blocks: a physical ensemble programming platform,” in *CHI’11 extended abstracts on human factors in computing systems*, 2011, pp. 1111–1116.
[10] Y. Zhang, J. Wang, C. Huang, and L. Li, “Application of compressed sensing based on logistic chaotic sequence in acoustic signal,” in *2022 4th International Conference on Image, Video and Signal Processing*, 2022, pp. 139–142.
[11] N. K. Pathak and R. Kumar, “Energy-intensive data compression in order to extend the life span of the network,” in *Electronic Systems and Intelligent Computing*. Springer, 2022, pp. 663–669.
[12] Y. Yang, H. Liu, and J. Hou, “A compressed sensing measurement matrix construction method based on tdma for wireless sensor networks,” *Entropy*, vol. 24, no. 4, p. 493, 2022.
[13] C. Habib, A. Makhoul, R. Darazi, and R. Couturier, “Real-time sampling rate adaptation based on continuous risk level evaluation in wireless body sensor networks,” in *13th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, WiMob 2017, Rome, Italy, October 9-11, 2017*. IEEE Computer Society, 2017, pp. 1–8.
[14] A. Makhoul and C. Pham, “Dynamic scheduling of cover-sets in randomly deployed wireless video sensor networks for surveillance applications,” in *2nd IFIP Wireless Days, WD 2009, Paris, France, December 15-17, 2009*. IEEE, 2009, pp. 1–6.
[15] R. A. de los Monteros, J. H. S. Azuela *et al.*, “A bidirectional hetero-associative memory for true-color patterns,” *Neural Processing Letters*, vol. 28, no. 3, pp. 131–153, 2008.
[16] V. Folli, M. Leonetti, and G. Ruocco, “On the maximum storage capacity of the hopfield model,” *Frontiers in Computational Neuroscience*, vol. 10, 2017.
[17] Y.-J. Jeng, C.-C. Yeh, and T.-D. Chiueh, “Exponential bidirectional associative memories,” *Electronics Letters*, vol. 26, pp. 717–718, 1990.
[18] M. Wang and S. Chen, “Exponential bidirectional associative memory based on small-world architecture,” in *Third International Conference on Natural Computation (ICNC 2007)*, vol. 1, 2007, pp. 391–397.
[19] S. Chen, H. Gao, and W. Yan, “Improved exponential bidirectional associative memory,” *Electronics Letters*, vol. 33, no. 3, pp. 223–224, 1997.
[20] W. Baoyun, Z. Qing, and H. Zhenya, “New exponential bidirectional associative memory,” *Journal of Electronics (China)*, vol. 13, no. 1, pp. 56–60, 1996.