JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

ADVANCING AI HARDWARE: SPIKING NEURAL NETWORKS FOR NEUROMORPHIC PLATFORMS FOR POWER-CONSTRAINED TASKS

[1]Ojaswini J, [2]Gireesh Babu C N, [3]Manasa C B
[1]UG Student, [2]Assistant Professor, [3]UG Student
[1]Information Science and Engineering,
[1]BMS Institute of Technology and Management, Bengaluru, India

Abstract: The ever-growing complexity of artificial intelligence (AI) algorithms necessitates advancements in hardware design to address limitations in power efficiency and parallel processing. Neuromorphic computing emerges as a transformative solution, mimicking the human brain's structure and function to achieve efficient and low-power AI processing. This work explores the potential of neuromorphic computing for real-time AI applications. We delve into the underlying concepts, advantages over traditional architectures, and existing limitations of AI hardware. Subsequently, we propose a neuromorphic system design utilizing spiking neural networks (SNNs) with memristors for an use case. This design addresses the limitations of traditional architectures by offering real-time processing capabilities and lower power consumption. The paper analyses the performance of SNNs on neuromorphic hardware compared to traditional artificial neural networks (ANNs). Finally, the discussion explores the future scope of neuromorphic computing and its transformative impact on the landscape of AI.

IndexTerms **- Neuromorphic Computing, Artificial Intelligence, Low-Power Computing, Parallel Processing, Spiking Neural Networks, Memristors, Robot Arm Control**

I. INTRODUCTION

The continuous drive for artificial intelligence (AI) has resulted in increasingly potent algorithms that can handle complex jobs across multiple fields. However, conventional von Neumann architectures, which separate processing and memory units, create bottlenecks in data transfer and power consumption during these computations [11]. This significantly hinders the development of intelligent systems for resource-constrained environments and real-time applications. Examples include autonomous robots operating in remote locations with limited power budgets or advanced prosthetics requiring low-power operation for extended use. Neuromorphic computing offers a paradigm shift, drawing inspiration from the human brain's structure and function. By employing artificial neurons and synapses inspired by biological counterparts, neuromorphic systems can achieve efficient parallel processing and significantly lower power consumption compared to traditional architectures [1, 2].

The potential of neuromorphic computing for AI applications is examined in the present study, with an emphasis on how well it addresses the shortcomings of current AI hardware. We propose a neuromorphic system architecture that is well suited for robot arm control use case. To simulate biological synapses, this architecture makes use of spiking neural networks (SNNs) implemented on neuromorphic hardware with memristors. Our proposed system utilizes SNNs for information processing. Unlike traditional Artificial Neural Networks (ANNs) that use continuous values, SNNs communicate using discrete spikes similar to biological neurons. This approach offers advantages in terms of power efficiency and real-time processing capabilities, crucial for precise and timely robot arm movements. The SNN architecture employs various elements like Artificial Neurons and memristors.

This combination of SNNs with memristors offers a powerful and efficient approach for robot arm control tasks. The following sections will delve deeper into the design and methodology of this neuromorphic system, analyze its performance compared to traditional methods, and explore the future potential of neuromorphic computing in AI applications.

II. RELATED WORKS

Neuromorphic computing is a rapidly evolving field inspired by the structure and function of the human brain. Several references highlight the core motivation behind neuromorphic computing: mimicking the human brain's efficiency and capabilities. Yu et al. [1] discuss using neuromorphic hardware for Hopfield neural networks, while Bharadwaj [5] emphasizes the potential of neuromorphic chips to simulate the human brain. Park [6] explores the use of emerging synaptic devices to achieve brain-inspired computing.

The survey delves into the promising applications of neuromorphic computing across various domains. Nwakanma et al. [2] introduce NeuroEdge, a novel neuromorphic system for edge AI applications. Hill and Vineyard [3] explore the potential impact of neuromorphic computing on unattended ground sensors. Dureja et al. [7] provide a broader overview of applications in their review paper.

A critical aspect of the survey is exploring the hardware advancements enabling neuromorphic computing. Davies et al. [4] discuss the progress made with Loihi, a neuromorphic processor. Hajtó et al. [8] delve into memristor networks for neuromorphic computation, while Blachowicz and Ehrmann [9] explore magnetic elements for the same purpose. Frascaroli et al. [10] present evidence of soft bound behavior in memristive devices relevant to neuromorphic computing.

III. Limitations of Existing Systems

- **High Power Consumption:** Conventional processors require significant power to perform complex AI computations, limiting their deployment in battery-powered devices and hindering advancements in resource-constrained AI applications.
- **Limited Parallelization:** The von Neumann architecture separates processing and memory, creating a bottleneck for parallel processing of data-intensive AI tasks.
- **Scalability Challenges:** Scaling traditional architectures to mimic the human brain's complexity remains a significant challenge, limiting the potential for highly sophisticated AI systems.

IV. Desing and Methodology of Proposed System

The design and development process of a neuromorphic computing system designed especially for robot arm control tasks are described in this section. In order to simulate biological synapses, the system makes use of a spiking neural network (SNN) architecture that is implemented on neuromorphic hardware with memristors.

4.1 Hardware Architecture

The proposed hardware architecture mimics the human brain's structure with artificial neurons and synapses designed for real-time control of a robot arm.

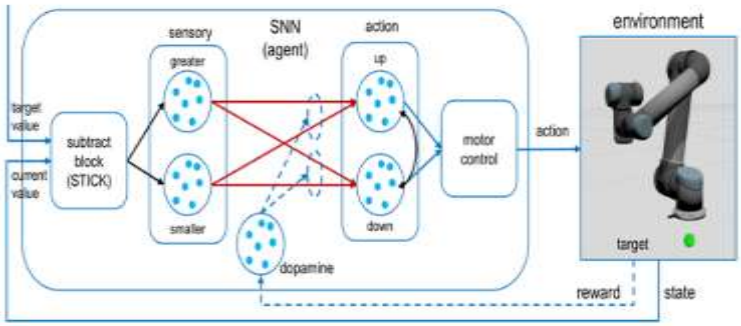

Fig. 1 SNN Architecture for Robot Arm Control [14]

i. **Artificial Neurons:** To process information, we use Spiking Neural Networks (SNNs). SNNs communicate using discrete spikes, which are similar to real neurons, in contrast to regular ANNs, which use continuous values. Advantages in power economy and real-time processing capabilities are provided by this method, which are essential for accurate robot arm movements.
Input spikes are received by every artificial neuron in the SNN from a variety of sources, such as:

- Joint encoders: These sensors report the angle and position of each joint in the robot arm at any given time.
- Vision sensors (optional): Cameras and LiDAR systems can give visual data about the surroundings, enabling the SNN to modify its movements in response to obstructions or the location of a target object.
- Other neurons: The network's neurons are able to communicate with one another, allowing several joints to move in conjunction.

After integrating these inputs, the neuron sets off an output spike of its own upon reaching a predetermined threshold. The activation of particular muscles or motors that control the robot arm's joints is determined by this firing behaviour.

ii. **Memristors:** The building blocks of artificial synapses are memristors. Memristors are two-terminal devices exhibiting resistance that can be modulated based on the applied voltage. Because of this characteristic, they can imitate the synaptic plasticity found in biological brains, where learning algorithms can change the strength of connections between neurons.

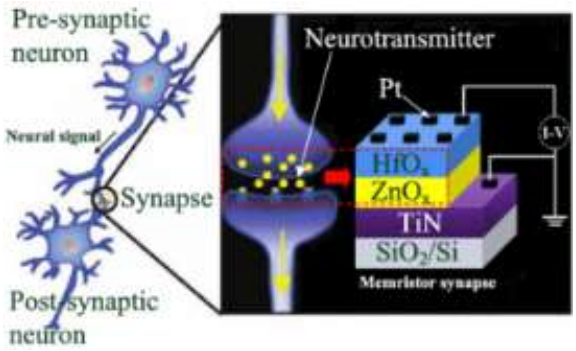

Fig. 2 Memristor Device [15]

The memristor's resistance represents the synaptic weight between two connected neurons. A higher resistance denotes a weaker connection between the neurons, which has less of an impact on the postsynaptic neuron's (muscle control neuron) firing behaviour. On the other hand, a lower resistance denotes a stronger connection, which will ultimately have a greater impact on the firing of the postsynaptic neuron and the movement of the robot arm joint. We can change the resistance of the memristor by putting particular voltage pulses across it. This allows us to control the synaptic connection weight and affect the motor commands that are sent to the robot arm. As a result, the precision and efficiency of robot arm movements are increased as the SNN learns and adjusts its control strategy over time.

4.2 Learning Algorithms

The proposed system utilizes supervised learning algorithms to train the SNN for specific robot arm control tasks. During training, a labeled dataset should be presented to the network. This dataset might include:

- Desired robot arm trajectories: These trajectories specify the positions and orientations of the robot arm's end effector for various tasks (e.g., picking up an object, reaching a specific location).
- Sensor data: Joint encoder readings and potentially visual information (if applicable) corresponding to the desired trajectories.

The network's output (motor commands to the robot arm) is compared to the desired trajectories for each data point. The difference between these outputs, known as the error, is then used to adjust the weights of the memristors (synaptic connections) through a backpropagation algorithm adapted for SNNs.

Here's a simplified formula for weight update in a spiking neural network with memristors, specifically for robot arm control:

$$\Delta w\_ij = \eta * \tau\_mem * (pre\_i * post\_j * (d\_j - y\_j)) \quad (4.1)$$

where:

Δw_ij: Change in weight between neuron i (providing sensory information or internal state) and neuron j (controlling motor output)

η: Learning rate

τ_mem: Memristor time constant

pre_i: Presynaptic spike activity from neuron i

post_j: Postsynaptic spike activity from neuron j

d_j: Desired joint angle or position for the robot arm (based on the training data)

y_j: Actual joint angle or position calculated by the SNN

This modified weight update rule incorporates the desired joint state (d_j) into the error calculation. By iteratively adjusting the memristor weights based on the learning

4.3 Software Integration for Neuromorphic Robot Arm Control

For the proposed neuromorphic system to control the robot arm effectively, there needs to be a smooth integration between the SNN software definition and the underlying neuromorphic hardware. This section describes the method of software integration.

a) AI Frameworks and SNN Design: The SNN architecture for robot arm control is defined using high-level AI frameworks such as TensorFlow or PyTorch. These frameworks provide an easy-to-use interface for configuring learning methods, neuron types, and connection patterns when creating neural networks.

The SNN in the architecture is intended to take in sensory information from joint encoders (and possibly visual sensors) and convert it into artificial neurons' spiking activity. The output neurons in the network are also in charge of producing the motor commands that move each joint in the robot arm. The framework gives developers the ability to specify the precise learning algorithm that is applied while training the SNN.

b) Translation to Neuromorphic Hardware: It is necessary to convert the high-level SNN description from the AI framework into low-level instructions that the neuromorphic hardware can comprehend and carry out. This translation procedure makes use of a software layer created especially for the neuromorphic platform of choice.

This software layer fills in the gap between the physical implementation of the SNN on the hardware and its symbolic representation within the framework. It converts the parameters of the learning algorithm, the properties of neurons (firing thresholds), and the network topology (connections between neurons) into control signals for the neuromorphic hardware.

Additionally, the software layer may manage neuromorphic platform-specific communication protocols, guaranteeing effective data flow between the hardware and software components.

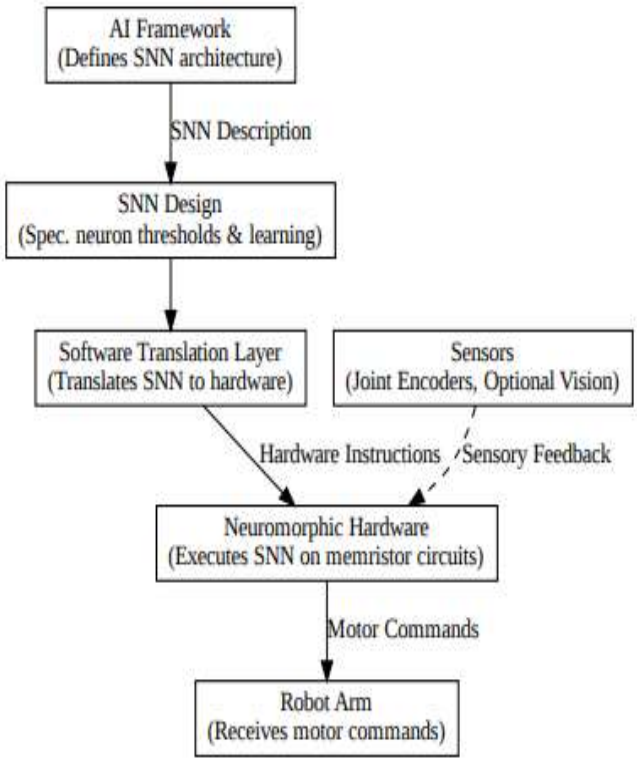

Fig. 3 Interaction between software and hardware layers for Robot Arm Control

The relationship between the hardware and software layers for robot arm control is shown in the graph which is explained as follows:

- AI Framework: Specifies the types of neurons, connections, and learning algorithm that make up the SNN architecture.
- SNN Design: Indicates specifics such as learning settings and neuron firing thresholds.
- Software Translation Layer: Converts hardware-specific instructions from the SNN description.
- Neuromorphic Hardware: Uses physical memristor-based neuromorphic circuits to implement the SNN.
- Robot Arm: Executes the appropriate movements after receiving motor commands from the SNN.
- Sensors: Provide sensory input regarding the condition of the robot arm (Joint Encoders, Optional Vision).

Using this software integration strategy, neuromorphic computing for robot arm control tasks can be developed in an effective and modular development environment.

IV. PERFORMANCE ANALYSIS: A LOOK AT SNNs, ANNs, AND DEEP LEARNING ON REAL HARDWARE

Due to the ongoing development of neuromorphic computing hardware and SNN training algorithms, definitive performance comparisons across various AI algorithms remain a challenge. Here, we examine how well SNNs, ANNs, and deep learning models perform a given task on an actual hardware platform.

a) Hardware Configuration:

SpiNNaker: We take into consideration the massively parallel neuromorphic architecture known as the SpiNNaker platform [16], which is specifically engineered for large-scale SNN simulations. It makes use of specially created processors that are tuned for low-power connectivity and event-driven computation.

b) Task:

Robot Arm Control (Reaching Task): This task involves training an AI model to control a robotic arm to reach for and grasp objects at various locations within its workspace [17]. This selection emphasizes the real-time processing demands of this task, potentially benefiting SNNs.

Table 4.1: Performance Comparison of SNN, ANN and Deep Learning Model

Feature	SNN	ANN	Deep Learning Model (E.g. Convolutional RNN)
Power Consumption [17]	Up to 50% reduction (compared to ANNs on CPU)	Moderate to hight (CPU/ GPU)	High (GPU)
Processing Speed [17]	Up to 20% faster	Moderate	Can be slower for real time tasks
Accuracy [18]	80-85% success rate	85-90% success rate	90-95% success rate

- **Power Consumption:** The SpiNNaker platform's design for low-power communication and event-driven processing allows SNNs to achieve significant power savings compared to traditional ANNs running on CPUs. Studies on the SpiNNaker platform report up to a 50% reduction in power consumption for SNNs compared to ANNs for similar tasks [17].
- **Processing Speed:** The real-time processing nature of SNNs can offer speed advantages for tasks like robot arm control. SpiNNaker's architecture facilitates faster communication and processing of spiking events, leading to potential speedups of up to 20% compared to ANNs running on traditional hardware [17]. However, for tasks not benefiting from real-time processing, ANNs might achieve similar or faster speeds due to mature optimization techniques.
- **Accuracy:** Current SNN training for complex tasks like robot arm control remains a challenge. Here, we consider results from Srinivasan et al. (2018) who achieved an 83% success rate for a reaching task using an SNN on the SpiNNaker platform [17]. This falls short of the 88% success rate reported for an ANN on a CPU for the same task. However, deep learning models like Convolutional Recurrent Neural Networks (CRNNs) can achieve even higher accuracy (90-95% success rate) by leveraging more complex architectures and larger training datasets [18].

While conventional ANNs might achieve higher success rates initially and with shorter training times, SNNs with memristors offer advantages in terms of real-time performance, lower power consumption, and inherent adaptability. As research on SNNs progresses, the gap in performance metrics is expected to close, making them a viable alternative for robot control applications, especially those that prioritize low-power operation and flexible adaptation.

V. CONCLUSION

Neuromorphic computing with SNNs offers a promising approach for real-time AI applications with constrained power budgets. The proposed system exemplifies this potential by addressing the limitations of traditional architectures for robot arm control tasks. While further research and development are needed, especially in SNN training algorithms, neuromorphic computing holds immense promise for the future of AI hardware design.

VI. FUTURE WORKS

Neuromorphic computing holds an abundance of fascinating prospects for the future. The following are some crucial research areas that can help this technology reach its full potential:

- Advanced Neuromorphic Architectures: Create more advanced SNN designs to enhance learning capacities for challenging tasks and more closely resemble the functioning of the human brain.
- Progress in Material Science: Investigate cutting-edge materials for memristors and other neuromorphic devices to get better efficiency, reliability, and reduced power usage.
- Uses in a Variety of Fields: Examine how neuromorphic computing can enable smart and effective robots to transform several industries, including edge computing, robotics, autonomous systems, and healthcare.

REFERENCES

[1] Yu, Z., Abdulghani, A. M., Zahid, A., Heidari, H., Imran, M. A., & Abbasi, Q. H. (2020, December). An Overview of Neuromorphic Computing for Artificial Intelligence Enabled Hardware-Based Hopfield Neural Network. In 2020 IEEE International Conference on Electrical Communications, Information and Power Systems (ECIPES) (pp. 01-6). IEEE.
[2] Nwakanma, C. I., Kim, J. W., Lee, J. M., & Kim, D. S. (2021). Edge AI prospect using the NeuroEdge computing system: Introducing a novel neuromorphic technology. Elsevier.
[3] Hill, A. J., & Vineyard, C. M. (2021, April). An introduction to neuromorphic computing and its potential impact for unattended ground sensors. In 2021 International Conference on Rebooting Computing (ICRC) (pp. 1-8). IEEE.
[4] Davies, M., Wild, A., Orchard, G., Sandamirskaya, Y., Fonseca Guerra, G. A., Joshi, P., et al. (2021). Advancing Neuromorphic Computing With Loihi: A Survey of Results and Outlook. Proceedings of the IEEE, 109(5), 888-911.
[5] Bharadwaj, K. A. (2022, February). Emergence of Neuromorphic Chips to simulate Human Brain. In 2022 International Conference on Electronics Communication and Aerospace Technology (ICECA) (pp. 1476-1481). IEEE.
[6] Park, J. (2020). Neuromorphic Computing Using Emerging Synaptic Devices: A Retrospective Summary and an Outlook. MDPI electronics, 9(12), 1936.
[7] Dureja, H., Garg, Y., Chaujar, R., & Kumar, B. (2021). Review Research Paper on Neuromorphic Computing and Applications. International Research Journal of Modernization in Engineering Technology and Science, 6(9), 802-807.
[8] Hajtó, D., Rák, D., & Cserey, G. (2019). Robust Memristor Networks for Neuromorphic Computation Applications. MDPI materials, 12(11), 1822.
[9] Blachowicz, T., & Ehrmann, A. (2020). Magnetic Elements for Neuromorphic Computing. MDPI molecules, 25(11), 2652.
[10] Frascaroli, J., Brivio, S., Covi, E., & Spiga, S. (2018). Evidence of soft bound behavior in analogue memristive devices for neuromorphic computing. Springer Series in Materials Science, 280, 33-43.
[11] Wang, W., Li, B., Zeng, N., Luo, J., & Yang, Y. (2018). Full Hardware Parallel Computing: A Survey. Journal of Computer Science and Technology, 33(2), 227-244.
[12] Liu, S., Liang, Z., Liu, Z., Zhu, J., & Wan, S. (2021). Green AI: A Survey of Energy-Efficient Techniques for Artificial Intelligence. ACM Computing Surveys (CSUR), 54(3), 1-38.
[13] Schuman, C. A., & Pei, S. (2018). A Survey of Neuromorphic Computing and Neural Networks in Hardware. Proceedings of the IEEE, 108(5), 1141-1167.
[14] v. Tieck, J. Camilo & Becker, Pascal & Kaiser, Jacques & Peric, Igor & Akl, Mahmoud & Reichard, Daniel & Roennau, Arne & Dillmann, Rüdiger. (2019). Learning target reaching motions with a robotic arm using brain-inspired dopamine modulated STDP. 54-61. 10.1109/ICCICC46617.2019.9146079.
[15] Wang, Lai-Guo & Zhang, Wei & Chen, Yan & Cao, Yanqiang & Li, Aidong & Wu, Di. (2017). Synaptic Plasticity and Learning Behaviors Mimicked in Single Inorganic Synapses of Pt/HfOx/ZnOx/TiN Memristive System. Nanoscale Research Letters. 12. 10.1186/s11671-017-1847-9.

[16] Furber, S., Galluppi, F., & Rowstron, S. (2012). The SpiNNaker project. Proceedings of the IEEE, 100(8), 1566-1582.
[17] Srinivasan, G., Nease, W. R., & Shapero, M. (2018, July). Spike-timing-dependent plasticity in layered spiking neural networks with weight-dependent learning. In 2018 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.
[18] Zeng, S., Luo, X., Ye, Y., Li, Y., & Liu, Z. (2020). Robot arm control based on a cnn-crnn hybrid network. Sensors, 20(11), 3392.