Internet of Things 26 (2024) 101207

Contents lists available at ScienceDirect

Internet of Things

journal homepage: www.elsevier.com/locate/iot

Efficient event-based robotic grasping perception using hyperdimensional computing

Eman Hassan [a,1], Zhuowen Zou [b,1], Hanning Chen [b], Mohsen Imani [b,*], Yahya Zweiri [c], Hani Saleh [a], Baker Mohammad [a,*]

[a] Department of Electrical and Computer Engineering and System-on-Chip (SoC) Center, Khalifa University, Abu Dhabi, United Arab Emirates
[b] Department of Computer Science, University of California, Irvine, USA
[c] Advanced Research and Innovation Center (ARIC), Khalifa University, Abu Dhabi, United Arab Emirates

ARTICLE INFO

Dataset link: https://github.com/emfhasan/GraspHD

Keywords:
Artificial intelligence
Robotics
Hyperdimensional computing
Dynamic vision sensor
Object grasping
Neuromorphic vision

ABSTRACT

Grasping is fundamental in various robotic applications, particularly within industrial contexts. Accurate inference of object properties is a crucial step toward enhancing grasping quality. Dynamic and Active Vision Sensors (DAVIS), increasingly utilized for robotic grasping, offer superior energy efficiency, lower latency, and higher temporal resolution than traditional cameras. However, the data they generate can be complex and noisy, necessitating substantial preprocessing. In response to these challenges, we introduce GraspHD, an innovative end-to-end algorithm that leverages brain-inspired hyperdimensional computing (HDC) to learn about the size and hardness of objects and estimate the grasping force. This novel approach circumvents the need for resource-intensive pre-processing steps, capitalizing on the simplicity and inherent parallelism of HDC operations. Our comprehensive analysis reveals that GraspHD surpasses state-of-the-art approaches in terms of overall classification accuracy. We have also implemented GraspHD on an FPGA to evaluate system efficiency. The results demonstrate that GraspHD operates at a speed 10x faster and offers an energy efficiency 26x higher than existing learning algorithms while maintaining robust performance in noisy environments. These findings underscore the significant potential of GraspHD as a more efficient and effective solution for real-time robotic grasping applications.

1. Introduction

Robotics' potential applications require advanced perception capabilities to enable robots to tackle various physical activities [1]. The rapid advancement of the Internet of Things (IoT) and the 4.0th industrial revolution have led to a new era of robotics, characterized by enhanced interaction between robots and their environments [2]. Central to this transformation is the capability of robots to perform grasping and manipulation tasks, fundamental actions that enable diverse applications from industrial automation to assistive technologies [3,4]. Grasping is one of the primitive manipulations and essential skills that robots need to master during interaction with static or dynamic environment. For example, industrial robots can perform tasks that appear tedious to humans, such as pick-and-place or sorting tasks, and assistive robots can help special needs and the elderly in their day-to-day grasping tasks. Therefore, The majority of robots in today's operations are equipped with parallel grippers or a complex end-effectors, which qualifies them to perform human-like behavior of simple to dexterous grasping [4,5].

* Corresponding authors.
E-mail addresses: m.imani@uci.edu (M. Imani), baker.mohammad@ku.ac.ae (B. Mohammad).
[1] Authors contributed equally.

https://doi.org/10.1016/j.iot.2024.101207
Received 1 February 2024; Received in revised form 7 April 2024; Accepted 26 April 2024
Available online 4 May 2024

The high grasping capabilities of human inspired researchers in the 1980s to come up with attractive ideas for dexterous grasping utilizing perspective in physics and geometry [4,6]. However, lacking advanced vision assistance and well-developed grasping algorithms posed many challenges and limitations on the grasping performance. Several challenges have historically impeded the progress in robotic grasping techniques [7]. These challenges include the diversity in object shapes, sizes, and materials, the variability of environmental conditions, and the limitations of sensor technologies in accurately capturing the tactile and visual signals necessary for effective manipulation. Moreover, traditional haptic sensing methods often employ visual sensors like Gelsight or DIGIT, which deduce object properties upon direct contact with a gel-based pad [8,9]. This contact-based process captures complex details and enhances grasping quality. However, these systems are constrained by their latency and sampling frequency, capturing observations at a frequency of 15–30 HZ with a total latency ranging between 50–250 ms [10]. Furthermore, while recent advancements in tactile sensing, such as the GradTac spatio-temporal gradient-based tactile sensor, have shown promise in overcoming low Signal to Noise Ratios (SNR) inherent in fluid-based sensors like BioTac, the search for more responsive and accurate sensing solutions remains [11,12]. While these methods have enabled significant progress, they come with inherent limitations that can hinder their effectiveness in real-world scenarios. Moreover, the computational complexity of processing sensor data and making real-time decisions further complicates robust grasping solutions.

Conversely, with the advancement in vision techniques, and AI surge, grasping system moved to a higher level. Several methods have been addressed to improve grasping behavior in robots applications for different object shape, size and textures [13–15]. Specifically, with image processing and optical technology progression, robotics grasping based on a deep learning framework obtained a high grasping success rate for many tasks [14,16]. However, these advanced vision-based systems also encounter challenges in real-world settings, such as occlusions, lighting variations, and dynamic environmental changes, which can significantly affect their effectiveness [17]. Furthermore, processing data from traditional cameras demands substantial computational resources and time, potentially limiting the responsiveness and adaptability of robotic systems

In contrast, neuromorphic vision sensors, exemplified by the Dynamic and Active Vision Sensor (DAVIS), represent a paradigm shift in robotic perception [18]. Unlike conventional sensors, DAVIS operates on principles that mimic the human retina, offering remarkable responsiveness to changes in light and motion through its event-driven data acquisition [19,20]. This approach significantly reduces latency, data processing requirements, enhance dynamic environment adaptability and precision in unpredictable settings, which aligns closely with the IoT's focus on efficient, real-time data handling.

By integrating DAVIS within the robotic system, we aim to enhance the robot's ability to perceive, interpret, and interact with its environment in a manner that is both more efficient and adaptable. This is achieved without direct physical contact between the sensor and the objects, paving the way for non-invasive, dynamic interaction strategies that have yet to be explored in the domain of robotic grasping [21].

While state-of-the-art AI computing approaches, including those utilizing DAVIS for robotic grasping systems, have demonstrated significant capabilities, they come with their own set of challenges. These systems often rely on algorithms that demand high computational power, extensive memory bandwidth, and substantial energy consumption. Moreover, the dependency on large datasets for training not only exacerbates the resource demands but also introduces practical challenges in data acquisition. For instance, the effort required to train a robot to grasp various objects, as seen in [22], involved more than 700 h of operation to collect 50,000 trials, highlighting the intensive nature of current training methodologies. In contrast, this work introduces a novel, brain-inspired approach that utilizes Hyperdimensional Computing (HDC) to overcome these limitations. Drawing from the remarkable efficiency with which the human brain processes complex and diverse information streams, HDC employs principles such as redundancy, distributed representation, and robust parallel processing. This high-dimensional operational framework not only reflects the brain's adeptness at handling information but also significantly reduces the computational demands typical of deep neural network (DNN) methods. Distinctly, the HDC algorithm diverges from conventional DNN and machine learning algorithms by its simplicity in training, capacity for parallel operations, and robustness against noise. Crucially, unlike traditional approaches, HDC does not necessitate large training datasets to achieve effective results, presenting a more efficient and practical solution for real-world robotic applications. This advantage positions HDC as a uniquely suitable method for advancing robotic grasping capabilities, especially in environments where adaptability and efficiency are paramount.

In this study, our primary objective is to explore how we can effectively leverage the unique capabilities of a neuromorphic vision sensor to determine object properties such as hardness [23] and size [24] and accurately estimate the optimal grasping force for a robotic grasping system. In order to do that, we propose a novel approach, the GraspHD framework, which capitalizes on the distinctive advantages of event-based cameras. In addition, The GraspHD framework operates directly on raw neuromorphic data, employing hyperdimensional computing (HDC) properties to encode the spatiotemporal information from the event-based data. This encoded data is then utilized to classify object properties and estimate the appropriate grasping force. In this context, the term 'force' refers to the force applied by the robotic gripper during the grasping process. It is not a property intrinsic to the object but a parameter the system needs to optimize for stable grasping.

The main contributions of this work are:

1. We devise a bio-inspired approach, GraspHD, that utilizes HDC learning for classifying the object prior knowledge. The work's main focus is to classify objects' hardness, size, and grasping force through a novel neuromorphic vision-based framework.
2. Inspired by our previous work, a modified version of the Event-Based Object Grasping (EBOG) dataset in [13] is derived and analyzed.
3. To the authors' best knowledge, GraspHD is the first framework utilizing the raw stream of events from the DAVIS camera to analyze the prior understanding of the grasped object's attributes via HDC. This eliminates the need for signal conditioning and filtering, which results in low latency and reduce computing complexity.

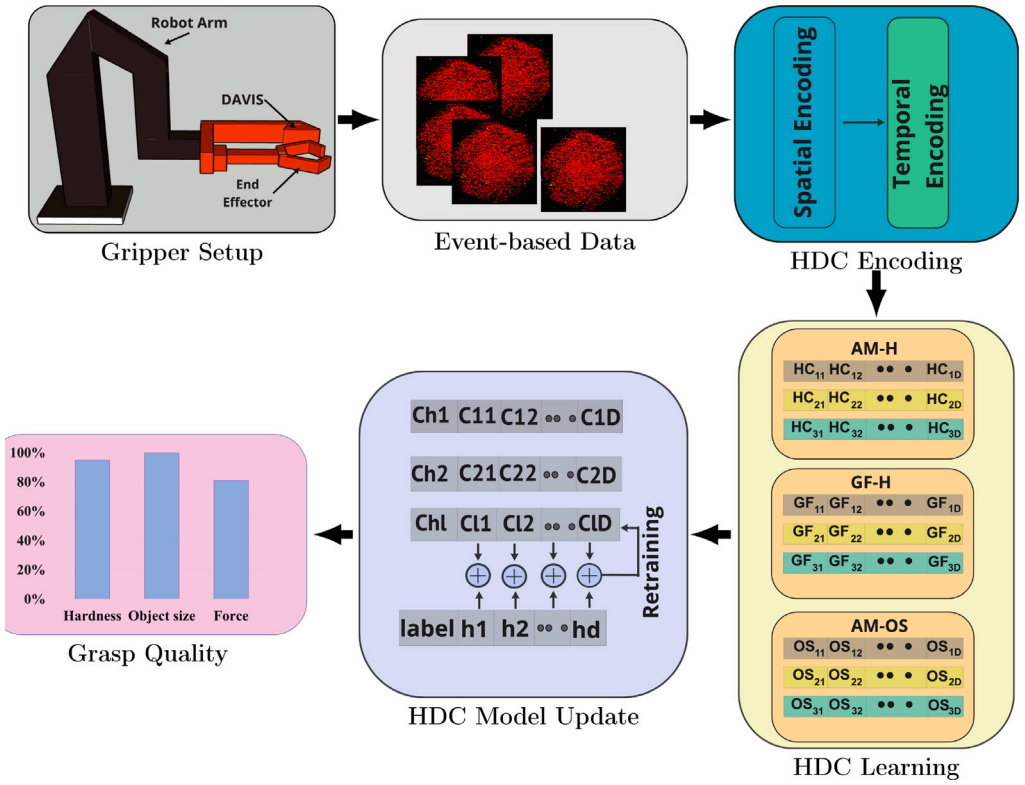

Fig. 1. GraspHD system overview. 'Gripper Setup' shows the arm and sensor configuration. 'Event-based Data' depicts sensory events during grasp. 'HDC Encoding' details spatial/temporal streams. 'HDC Learning' models for hardness (AM-H), force (AM-GF), and size (AM-OS). 'HDC Model Update' presents retraining with classes hypervectors (Ch1, …, Chl). 'Grasp Quality' indicates output metrics.

4. We rigorously compare the accuracy and efficiency of GraspHD with three of state-of-the-art (SOTA) solutions for grasping using DAVIS output. We demonstrate that GraspHD provides comparable accuracy compared to these grasping algorithms, and the FPGA implementation of GraspHD enables (1) throughput improvement of ~ 10x and (2) computation reduction by ~ 26x compared to DNN [25].

The rest of the paper is organized as follows: Section 2 provides an overview of current grasping systems using both standard and neuromorphic vision-based sensors. In Section 3, the neuromorphic grasping framework adopted in this work is introduced. Sections 4 & 5 presents the Hyperdimensional Computing paradigm (HDC), including its inspiration, main characteristics, properties, and the pipeline specifically designed for classification tasks. The HDC model for grasping objects, using raw events from DAVIS, illustrated in Fig. 1 and is further explained in Section 6. Results and performance metrics are thoroughly analyzed in Section 7. Finally, the conclusions, current challenges and potential avenues for future work are discussed in Section 8 & Section 9 .

2. Vision-based robotic grasping system

The field of vision-based robotic grasping has seen a variety of approaches, which can be grouped based on multiple criteria. These approaches can be divided into analytical or data-driven methods, with analytical approaches typically analyzing the shape of the target object to identify a suitable grasp pose [26]. Additionally, approaches can be categorized as model-based or model-free, depending on whether specific knowledge about the object is used. With the development of neuromorphic vision in grasping, a new approach has emerged that can be categorized as standard or neuromorphic vision-based [27]. In this section, we will focus on the advantages of the neuromorphic vision-based approach compared to the standard vision-based approach.

2.1. Standard vision-based robotic grasping system

The SOTA robotic grasping detection systems predominantly employ conventional machine vision, utilizing sensory inputs such as RGB images, RGB-D, and depth images [3,26,28–30]. Most of these vision-based grasping systems necessitate prior knowledge about an object's geometry to execute the intended task effectively [26]. Various works have leveraged large-scale deep neural networks (DNNs), training robots on extensive datasets to enable them to understand and decide how to grasp objects of different shapes, sizes, and textures. For instance, the Dexterous Network (DexNet) is a method that learns policies for a given set of grippers [31,32].

The Grasp Pose Detection (GPD) method, widely used for detecting grasp poses in point clouds, is another well-known approach in this domain [33,34]. In [35,36], a DNN approach is designed and combined with cloud point segmentation to enhance 3D object grasping estimation for a bin-picking task. Similarly, a vision-based grasping system was proposed in [37], where the neural network uses the point cloud of the grasp to classify the quality of the grasping, utilizing 3D scans and RGB-D pictures of known objects. Work in [38] proposed a novel grasping framework for sorting bottles in a complex environment, exploiting the Region Proposal Network (RPN) for object recognition and pose estimation [39]. Moreover, an intelligent grasping approach was proposed in [40] for picking objects from the floor using the method of highly accumulated features (HAF).

Furthermore, numerous haptic sensing techniques have been developed to enhance the perception capabilities of robotic systems. These methods have focused on various aspects of object detection, such as tactile perception, texture assessment, and in-hand manipulation. For instance, work in [8] utilized a GelSight tactile sensor with a Kinect sensor to infer the properties of an object through touch. They processed the tactile data using Convolutional Neural Networks (CNN) for perception tasks related to clothing materials. Similarly, [9] introduced TouchRoller, a roller-based optical tactile sensor designed to efficiently measure large surface areas, providing high-resolution tactile images for texture assessment. Meanwhile, [41] demonstrated a novel design for a low-cost, high-resolution tactile sensor called DIGIT. This sensor is intended for in-hand manipulation applications and captures the deformation of a soft elastomeric gel skin to infer object properties such as shape and texture. In [42], a tactile feedback-based grasping is exploited to detect and corrects slippage for a stable grasp without prior knowledge of object properties. These innovations have made significant strides in haptic sensing. However, they still face limitations due to reliance on traditional frame-based cameras. These cameras require intense image processing, introducing a delay in data processing and subsequent robot actions, and present issues of motion blur and low sampling rate even with high-speed cameras [43]. These restrictions have hindered the broader application of these technologies, particularly in grasping applications in structured and unstructured environments.

2.2. Neuromorphic vision-based robotic grasping

To address challenges encountered in the conventional frame-based camera, the neuroscience and computer vision community have conceived a bio-inspired sensor, often referred to as “neuromorphic”. This sensor imitates the neuro-biological architecture of the human retina [44,45]. The conventional neuromorphic vision sensor consists of a silicon retina that mimics the retina in flying organisms. This innovation has led to the development of sensors such as Dynamic Vision Sensors (DVS) [46], asynchronous time-based image sensors (ATIS) [47], and dynamic and active-pixel vision sensors (DAVIS) [18], marking significant strides in integrated CMOS technology applications, particularly in the industrial domain. The event-driven nature of neuromorphic sensors captures per-pixel illumination changes as events. It asynchronously detects brightness changes per pixel, outputting them as a stream of events defined by a microsecond-level timestamp, spatial coordinates of the changing pixel, and the sign of the changes [44,45]. Unlike conventional frame-based sensors, the neuromorphic sensor [18] encompasses unique properties, rendering them indispensable for robotics applications. Their ability to provide sparse data with rich temporal information about changes in scene allows for accurate and fast detection within dynamic environments. This facilitates the development of motion-based vision assistance for various tasks [44,48].

Despite the promising capabilities of neuromorphic sensors to redefine robotic grasping, the transition from concept to application poses considerable challenges Work in [49] utilized a neuromorphic vision sensor for robotic grasping, employing the DNN algorithm. A grasping dataset titled “Event-Grasping Dataset” was created and annotated automatically. The main features in the reconstructed images were extracted to achieve an optimal grasping pose. The system annotated good and bad grasping poses using LED light markers, tracked with a practical filter algorithm. The design could grasp various object sizes and shapes using multiple feature maps, achieving high accuracy at a 1 kHz sampling rate. However, it comes with its challenges, where numerous synchronization steps between annotation event data and object event data were necessary, and preprocessing steps required substantial memory bandwidth and high-speed processors, rendering real-time performance impractical. Further investigations in [50] introduced a multimodal neural network leveraging both event streams and RGB images for robotic grasp pose estimation, enhancing accuracy compared to using single-modal data. This approach showed strong performance in varied and cluttered environments. However, challenges include shadows and dense event areas affecting object recognition and grasp accuracy. In other studies, [5,43] employed event-based vision sensors for stable grasping and real-time slippage prevention. The authors developed a novel technique for incipient slip detection during object holding, discussing two main approaches based on temporal coding. The baseline approach processed raw events and sampled the threshold empirically, while the feature-based method utilized detector surface of active events (SAE) and spatially adaptive e-Harris algorithms to detect informative events. Both techniques were successful at a 2 kHz sampling rate under varying illumination. Furthermore, [51] employed image processing techniques to reconstruct contact areas and analyze event distribution for slippage detection and proper grasping. Various ML algorithms were explored in [52] to estimate contact force and classify object materials during grasping, employing event-based tactile sensors. Similarly, [13] used DAVIS contact level grasping information to classify object sizes and hardness and estimate contact force for sorting applications, achieving accurate predictions with low latency. For multi-object grasping, study in [27] proposed a fast planning and generation approach using neuromorphic vision in pick-and-place applications. Techniques for object segmentation, localization, cloud point processing, and feature extraction were combined to ensure an optimal grasping stage. Despite the high performance of the above approaches in grasping various unknown objects and under low lighting conditions, and in real-time environments, these approaches required prior knowledge for grasping objects and faced challenges with perception.

Despite the advancements in neuromorphic data for robotic grasping, significant obstacles persist, particularly in resource-limited and challenging environments. The task of deriving highly informative features from the raw neuromorphic events demands

complex perception algorithms, substantial computational power, and extensive storage, all of which contribute to elevated energy consumption in embedded systems. This issue is further exacerbated when dealing with dynamic, moving objects instead of static ones, heightening the complexity of accurate object detection and grasp execution. Moreover, the quality and comprehensiveness of existing datasets for neuromorphic grasping still need to be improved, necessitating enhancements for improved detection accuracy. Expanding and refining these datasets and conducting thorough analyses is essential to ensure their effectiveness in training detection models. In light of these challenges, there is an imperative need to develop more efficient computational strategies. These strategies must not only align with the real-time processing demands of robotic systems but also achieve a balance between computational efficiency and the high accuracy established by contemporary methods, all while maintaining robustness in the face of environmental noise and variability.

3. Neuromorphic grasping framework

3.1. System setup

In this work, the experimental setup for the neuromorphic grasping system using a Baxter robot includes several key components, as illustrated in Fig. 2a. These include the Baxter robot itself, whose original grippers have been modified to facilitate stable grasping with its parallel plate grippers (PPG). The PPG consists of two main parts: a metallic part that moves in a direction parallel to the table surface, and a transparent acrylic part used for grasping objects. An adjustable camera holder is attached to the metallic part of the PPG to ensure stable grasping, reduce vibration, and enable adjustment of the position and direction of the DAVIS vision sensor, which is attached to the right side of the PPG. Fig. 2b shows a top view of the PPG and its metallic and acrylic components. The distance between the DAVIS sensor and the PPG was determined through trial and error to ensure that the entire surface of the object could be covered during grasping.

Additionally, a semi-transparent silicon material is attached to the inner side of the right gripper to enhance the grippers' flexibility. This allowed the sensor to capture detailed event-based information through deformations of the silicon in-contact and ensure precise control over the grasping process, as demonstrated in Fig. 2f.

To further enhance the neuromorphic grasping system's functionality, we adjust the silicon material's hardness on the gripper to various levels. This adjustment mimics the hardness of different objects the system might encounter, enhancing its ability to adapt to and effectively grasp objects with varied hardness levels. Such a feature improves the grasp's stability and accuracy and aids the system in accurately estimating the required forces for each grasp. Consequently, this leads to a more precise and stable grasping performance across diverse object properties.

Fig. 2(c–d)illustrate the main stages of each experiment, which comprise grasping, holding and releasing states.

For this study, we selected hexagonal nuts as our objects of interest illustrated in Fig. 2e. Nuts are essential components in a variety of industrial machines and products. Despite their simplicity, identifying, picking up, and sorting these nuts present substantial challenges for robotic systems. Given their widespread use in industrial and commercial settings, hexagonal nuts are the primary focus of our study. The nuts screws are categorized by size into small (11 mm), medium (13 mm), and large (17 mm) diameters. These sizes were chosen to challenge the system's adaptability to various grasping complexities. To simulate different material properties, silicon wafers with thicknesses corresponding to small (4 cm), medium (7 cm), and large (10 cm) degrees of hardness were used. This variety in object properties tests the system's capability to adjust grasping force, which is set to 10%, 50%, and 100% of the gripper's maximum capacity, accordingly. In addition to these variables, it is pertinent to note that the nuts' metallic surface are utilized in the experiment. It provides a uniform texture that presents its own set of challenges for robotic grasping, such as reflectivity and slipperiness under certain conditions. Therefore, our dataset, incorporates objects that not only vary in size and hardness but also present consistent surface characteristics representative of common industrial materials.

Additionally, a force/torque (F/T) sensor is mounted on the left gripper to track the movement of the PPG during the grasping, holding, and releasing phases. This sensor is used to measure the contact force of the grasped object and serves as a ground truth for the grasping process's performance. A PC with a custom interface is used to control the Baxter robot and receive data from the DAVIS sensor and force sensor. Overall, the system is calibrated to ensure accurate event-based information and precise grasp control, and it is designed to allow the integration of the neuromorphic vision sensor and the PPG for the robot to grasp objects effectively and maintain stability during the grasping process. The position and orientation of the DAVIS sensor are carefully chosen to facilitate this integration. A similar setup has been used and validated by several recent studies in the field [5,13,27,49,53]. In this study, our experimental setup, particularly integrating the neuromorphic vision sensor with the robotic system, is conceptualized as a prototype. This prototype is helpful in demonstrating the preliminary application of our methodology, specifically how the raw data from the DAVIS sensor can be leveraged to enhance grasping quality. Conducting the above experiment on a robot for smart grasping under the above conditions is costly and time-consuming. The setup itself requires a lot of time and resources, making it expensive. As a result, we have a limited number of samples that took quiltly long time to collect and label.

3.2. Event data formulation for grasping system

Building upon the system setup, we formulated event data to encapsulate the dynamic interactions between the robot grippers and the objects. This data formulation is critical for understanding how the system responds to various object properties, including size, hardness, and estimate the grasping force. This work built a dataset using three different object sizes and three hardness levels.

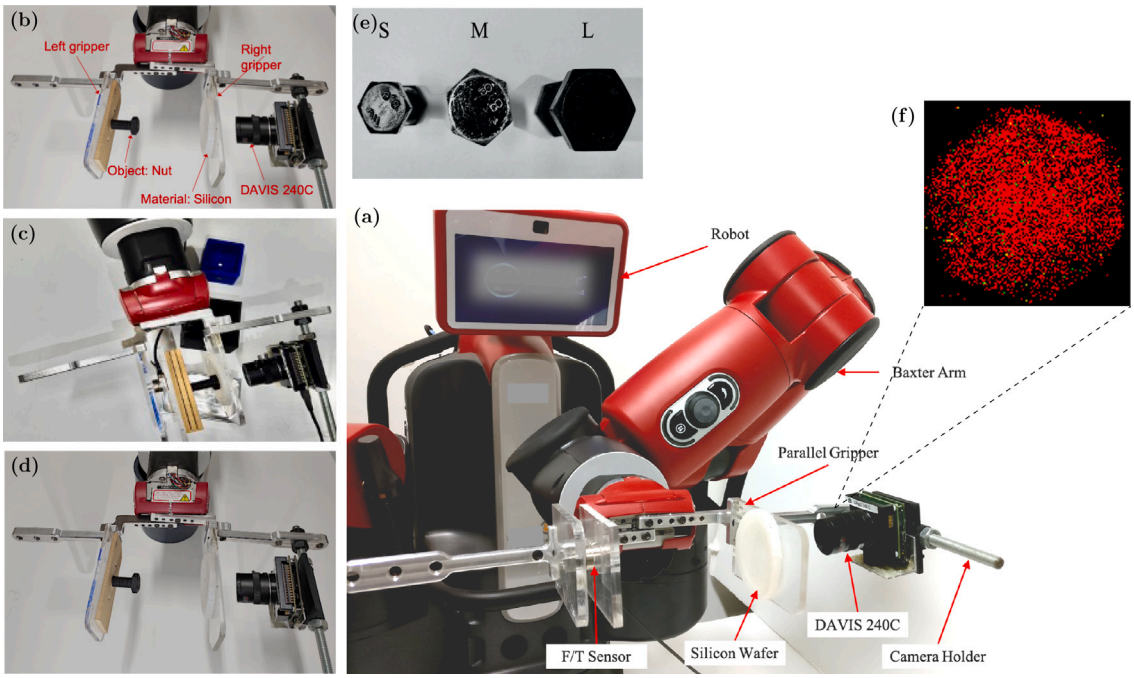

Fig. 2. (a) Neuromorphic grasping system overall Setup. (b) the top view of parallel plate grippers (PPG) for the right grasp cage. (c, d) the top view during grasping and releasing the object. (e) The object's size and shapes used during the experiment, (f) DAVIS scene during the grasping stage.

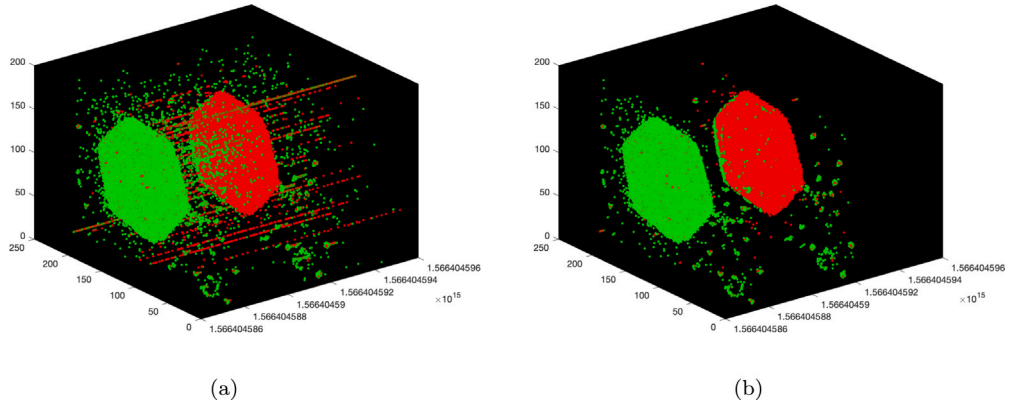

Fig. 3. (a) An example of the hexagon shape used for the GraspHD, and (b) the effect of filtering out the noise on the grasped events.

The experimental protocol involved conducting 11 repetitions for each combination of object size, hardness, and estimated force, capturing the grasping, holding, and releasing phases. These repetitions ensure a robust dataset that reflects the system's performance across spectrum of conditions. An example of the triggered events accumulated over a 1 ms time interval for the three stages is shown in Fig. 3(a).

To mitigate the impact of external noise, such as gripper vibration and lighting conditions, the events were grouped over 1 ms interval, focusing on the most informative aspects of each grasp. In total, our dataset includes 243 samples spanning 27 different combinations of object properties and serves as a comprehensive testbed for evaluating the proposed neuromorphic grasping system [13]. This careful formulation of event data, grounded in a well-defined experimental setup, allows for an in-depth analysis of the neuromorphic grasping system's performance. It highlights the system's adaptability to a range of object properties and its potential for real-world application, particularly in industrial settings where precision and adaptability are paramount

Raw Events: The raw events data from the DAVIS vision sensor can be represented as a tuple, $(t_k, (x_k, y_k), p_k)$. The variable k is the index of the kth event in the events stream. For each triggered event, t_k represents the event timestamp in microseconds, (x_k, y_k) represents the pixel coordinate in the scene, and p_k represents the polarity of the light intensity change, with a value of "on" or "off".

The polarity is determined by the sign of the difference between the current light intensity I at the triggered event and a pre-defined threshold C:

$$\Delta I_k = I_k - C \tag{1}$$

where I_k is the light intensity at the kth event. The polarity is then defined as:

$$p_k = \begin{cases} \text{“on”} & \text{if } \Delta I_k > 0 \\ \text{“off”} & \text{if } \Delta I_k < 0 \end{cases} \tag{2}$$

If the brightness at the pixel increases beyond the threshold ($\Delta I_k > 0$), the event's polarity is “on”, while a decrease in brightness below the threshold ($\Delta I_k < 0$) results in a polarity of “off”. Each event is triggered asynchronously, resulting in a non-uniform distribution of event timestamps.

States and Transitions: The state of a pixel at a given time is its current brightness condition, and can be represented as $s_k \in \{+, -\}$. A new event at a particular pixel location will only change the state if it results in a different brightness condition. This transition can be represented as a positive transition (event) p when the state changes from $-$ to $+$, and a negative transition n when the state changes from $+$ to $-$.

4. Hyperdimensional computing

Hyperdimensional computing (HDC) is a proposed solution for efficient and robust learning that is inspired by the way the human brain processes data [54]. In HDC, data is represented and manipulated using low-precision random vectors in thousands of dimensions, known as hypervectors. These vectors are composed of elements that are independently and identically distributed (i.i.d). HDC is motivated by the extensive circuits in the human brain. In HDC, each item or entity is assigned a seed vector in high dimensions to represent its features for the desired application, and all seeds are stored in an item memory. A dense seed hypervector consists of elements that are binomially distributed with equal probability. For example, in a binary representation, the number of 0 s and 1 s in the vector are the same, with a probability of 1/2.

The general representation of HDC has the following characteristics [55]:

- HDC uses high-dimensional representations to encode data, similar to the way the human brain uses tens of billions of neurons.
- HDC uses randomness in its learning mechanisms, similar to the “random origins” of the human brain.
- HDC uses a holistic representation that is independent of position and is optimized for robustness.
- HDC is tolerant to component failure thanks to its redundant representation and error-correcting mechanisms.

Recently, HDC has been designed to represent symbolic and compositional structures utilizing a set of well-designed operations. In particular, three operators have been identified as the building blocks of representing data structures: addition, multiplication, and permutation. In most HDC literature [56–58], they have been shown to represent sets [57], variable binding [59], and sequences ordering:

- Bundling $\oplus$: For a set $S = \{x, y\}, \subset \mathcal{X}$, $h_S = h_x \oplus h_y$, where (x, y) are entities in the original space, (h_x, h_y) are the randomly-generated representative HD vector in high-space, and $\oplus$ represents the element-wise addition for h_i vectors in HD space.
- Binding $\odot$: For an order-less tuple $(x, y) \in \mathcal{X}^2$, $h_{(x,y)} = h_x \odot h_y$. It is usually done using XOR or element-wise multiply operations. The output hypervector is orthogonal to hypervectors being multiplied, and
- Permutation ρ: For an element $y \in \mathcal{X}$, we can represent “y is in ith place” as $h_{y[i]} = \rho^i(h_y)$, where ρ is a random permutation over the hyperdimensions. Incidentally, $h_{y[0]} = h_y$.

Therefore, HDC has been used in a variety of fields in recent years, including classification [60], activity recognition [61], biomedical signal processing [62,63], multimodal sensor fusion [64], security [65], and robotics [66]. One of the major advantages of HDC is its ability to be trained quickly and accurately with a limited number of examples. In terms of performance, HDC has been shown to be comparable or better than other methods such as support vector machines(SVM), gradient boosting, and convolutional neural networks(CNN), and also requires less energy to run on embedded processors compared to these other methods.

5. HDC for classification

This section explains using HDC for supervised classification tasks adopted in this work. The main classification steps will also be introduced, including single-pass training and adaptive learning from labeled data.

5.1. HDC learning for supervised learning

Initial single-pass training: After mapping the input neuromorphic data into HD space , HD learning begins by finding the universal property for the training dataset. This involves using a trainer module to linearly combine the hypervectors belonging to each class in the dataset. The most common training method for HD learning is single-pass training, in which the trainer module combines all the hypervectors for a given class in a single pass [57,67]. Once the hypervectors have been combined, the resulting per-class accumulated hypervectors, called class hypervectors, are treated as the acquired model. This model can then be used to make predictions on new data inputs.

Adaptive training (multi-class updates): Instead of naively combining input HD vectors to generate the class hypervector, the proposed HDC learning approach includes adaptive training, which is adopted in this work [68]. Adaptive training is a technique used in machine learning to improve the performance of a model by adjusting its parameters during training. This approach can help to prevent the saturation of the class hypervectors, which is a common issue when dealing with high-dimensional data. In adaptive training, the model is updated during training based on the similarity of a new training data point with the existing class hypervectors. In this case, the similarity is measured using the cosine similarity between a new data point H and a class hypervector C_l is defined as: $\delta_l = \frac{H \cdot C_l}{|H||C_l|}$

If the similarity is high, indicating that the data point is already represented in the model, it is added to the model with a small weight in order to prevent saturation. If the similarity is low, indicating that the data point contains new information, it is added to the model with a weight that is proportional to the inverse of the similarity. Mathematically, the update rule for the model during adaptive training can be written as follows:

$$C_{correct} = C_{correct} + \eta\delta \times H_i \tag{3}$$

$$C_{wrong} = C_{wrong} - \eta\delta \times H_i \tag{4}$$

where η is the learning rate and δ is a value that varies according to the physical distance between the classes hypervectors C_l and the input hypervector H. This method ascertains an adaptive update for the model hypervectors, and ensures that noise spikes in the data input will not lead to a prediction.

Inference: The inference process is carried out in two steps. First, the test dataset is mapped to HD space using the same encoding method applied during the training stage. Secondly, the similarity δ between the test hypervector Q and all classes of hypervectors is measured, and the winning class is the one with the highest similarity with the test hypervector.

Iterative Re-training: for accurate learning tasks. GraspHD supports retraining to enhance the quality of the model. Instead of starting from a naive initial model, GraspHD retraining begins from the initial adaptive model, which takes into account the weight of each input data point during single-pass training as discussed above. This enables GraspHD to retrain the model with a much lower number of iterations, resulting in fast convergence. for the iterative retraining step, the class hypervectors are updated based on the similarity between the input hypervector H_i and the class hypervector C_K. The update rule is given by: $C_K \leftarrow C_K + \eta(1 - \delta(H_i, C_K)) \times H_i$, where H_i is the hypervector for the input that is labeled l, and $\delta(H_i, C_K)$ is the cosine similarity between H_i and C_K. This update rule ensures that input hypervectors that are less similar to the class hypervector are given more weight, and prevents biasing towards any particular input in the class HD vector.

6. GraspHD framework overview

In this section, we present an end-to-end framework for efficient learning utilizing the HDC for the neuromorphic vision-based grasping application "GraspHD". Unlike the previous works for grasping systems that operate over preprocessed neuromorphic data, GraspHD is the first HDC framework that operates directly over the raw neuromorphic data. Fig. 1 gives an overview of the GraspHD framework. It includes spatial and temporal encoding schemes to map event-based grasping data into high-dimensional space. Given the sparsity and spatio-temporal nature of the DAVIS output, we leverage hyperdimensional computing mathematics to symbolically encode spiking events into high-dimensional vectors. Besides, it utilizes the encoded data to acquire a prior knowledge about the object while grasping by classifying the object hardness(H), size(S) and estimate the appropriate grasping force(GF). The output of the DAVIS camera consists of a set of time series encoded as a (1) timestamp of $E_k = (t_k, \mathbf{x}_k, p_k)$, where the kth event signals an illumination change upward or downward p_k in time t_k and (2) pixel location given its coordinate in the Euclidean space $\mathbf{x}_k = (x_k, y_k)$ [13]. For simplicity, we first explain how the proposed encoder preserves the spatial correlation of spikes in holographic high-dimensional space. Then, we add the temporal locality as a memorization term to the spatial encoder. Therefore, GraspHD preserves the temporal and spatial correlation directly between the input events. This encoder aims to represent event streams in a holographic representation; thus, a single event in original data induces a pattern of neural activity (changes) represented in high-dimensional space. And the information of each original spike will be evenly distributed over all dimensions of the encoded hypervector. In addition, given that our encoding is purely event-based, it can also be operated in an asynchronous setting by reacting to DAVIS events, thus preserving synchrony.

6.1. HDC symbolic encoding for event-based data

In HDC space, and before mapping the features from the original dimension to hyperspace, HDC begins by choosing the appropriate seeds that will be used later to encode the data to HD space [57]. The main symbolic elements for the event data used in grasping are: *event timestamp* t_k, *pixel coordinate* $\mathbf{x}_k = (x_k, y_k)$, and *event states* p_k. In the following, we cover the GraspHD seed generation procedure in high-dimensional space and demonstrated in Fig. 4a.

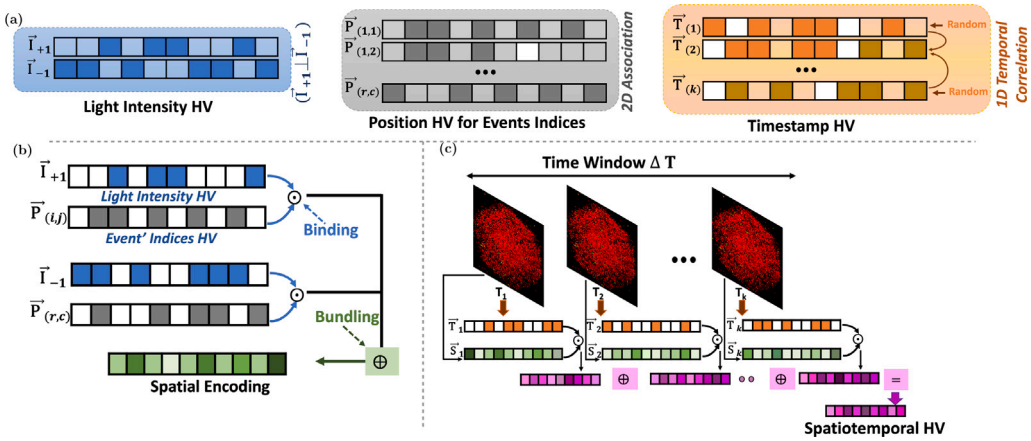

Fig. 4. (a) GraspHD seeds generation in high dimensional space that represents light intensity, active events' indices, and timestamp hyperdimensional vectors for triggered events. (b) 2D spatial association of active events position and their corresponding light intensity at a particular timestamp. The spatial encoding preserves the triggered event's location and their polarity as a single hypervector in high space. (c) temporal encoder keeps information correlated in continuous time domain utilizing binding and bundling operation in high space.

6.1.1. Seed generation

- **Light Intensity (Polarity) Hypervectors**: in DAVIS, the polarity of each event has two states; increase or decrease depending on the brightness conditions. These values $I \in \{1, -1\}$ can be represented in HD space using a hypervector generated randomly, where each entry h_i in the vector is sampled from the bipolar distribution $h_i \sim Uniform\{-1, 1\}$, and the hypervector for $I = 1$ and $I = -1$ are additive inverses of each other: $h_{I_{+1}} = -h_{I_{-1}}$.
- **2D Position Hypervectors**: Drawing inspiration from the work in [69], we use a set of position hypervectors $P_{0,0}, P_{0,1}, \ldots, P_{r-1,c-1}$ to represent the locations of active events, where r, c indices correspond to the row and column positions in the active pixel sensor frame. These 2D position HD are designed to preserve the spatial correlation between neighboring events, meaning that if two pixels are physically close in location, their corresponding HD should be highly correlated. Here is the general approach to generating these position hypervectors:

  1. **Frame Division**: We first divide the DAVIS sensor frame into non-overlapping windows of size $k \times k$.
  2. **Corner Hypervectors**: For each $k \times k$ window, we assign random HD to the pixels located at the corners. For instance, when $k = 2$, all pixels at the corners of the windows $P_{0,0}, P_{0,2}, P_{2,0}, P_{2,2}, \ldots, P_{2i,2j}$ are assigned random hypervectors. This process is repeated for all $k \times k$ windows. As these vectors are generated randomly with high dimensions and sampled from a normal distribution, they are likely to be nearly orthogonal [54].
  3. **Intermediate Hypervectors**: For the pixels located between the corners within the same ith window, we generate their HD by combining dimensions from the four corners hypervectors. The proportion of dimensions taken from each corner is determined by the physical distance of the pixel from the corners. For example, with a kernel size of $k = 2$, the hypervector for $P_{0,1}$, which is on the edge of the window, will take 50% of its elements from the right and left corners $P_{0,0}, P_{0,2}$ respectively. However, for $P_{1,1}$, which is in the center of the window, 25% of the elements will be taken from each corner.

  This approach ensures that the resulting 2D position hypervectors preserve the spatial correlation between the positions of events in the scene.
- **Timestamp Hypervectors**: Incorporating timestamps in the representation of events is crucial to maintain the notion of continuous-time. Similarly, to represent time in the hyperdimensional (HD) space, we divide the experiment's time into t-sized sub-windows. The size of the sub-window is chosen empirically, and random hypervectors are generated to represent the timestamps at the borders of the t time windows. Later, a 1D interpolation is implemented for the intermediate timestamps to preserve the correlated temporal information using the two hypervectors encompassing the time window.
  For example, and for simplicity, assume that the time window size is K. A set of random hypervectors representing sub-windows of time intervals are $T_0, T_{2K}, T_{3K}, \ldots, T_{tK}$. The main function of the temporal encoder is to determine the neighboring hypervectors $T_{iK}, T_{(i+1)K}$ that are used for interpolation. To encode an intermediate timestamp, the proposed algorithm uses $(1 - \frac{i}{K} \cdot D)$ elements from T_{iK} to form the new associated HD vector, where i is the index of the current time sub-window and D is the dimension of the hypervector, and the remaining elements of the hypervector are from $T_{(i+1)K}$. This process allows for the representation of time in HD space while preserving the correlated temporal information.

6.1.2. 2D-spatial encoding: Event-position and brightness association

The primary objective of the spatial encoder is to map neuromorphic data into a high-dimensional space using pre-generated seed vectors, which are stored in an item memory (IM). At any given timestamp, our encoder associates the position of each active event

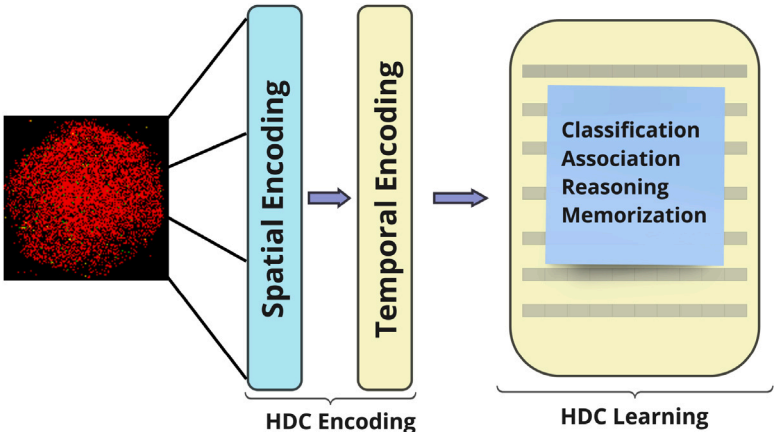

Fig. 5. GraspHD Framework: Hyperdimensional learning from DAVIS data. It includes the spatial and temporal encoders for mapping neuromorphic sensor 2D data to high space, for performing classification over the encoded data.

with its corresponding light intensity. This is achieved in the hyperdimensional (HD) space using the binding operation, as illustrated in Fig. 4b. For instance, if an event is triggered at a pixel coordinate $P_{p,m}$ with a specific transition (polarity), the encoder ties this information using the binding operation $P_{p,m} \odot I_I$, where I_I is either I_{-1} or I_{+1}. The resulting hypervector is nearly orthogonal to the bounded HD vectors of the event position and polarity. This process is repeated for all triggered events within a pre-defined time interval.

Subsequently, all HD vectors for the encoded events are combined using the addition operation, forming a unique hypervector that preserves the spatial information between the activated events within the specific interval. This hypervector is expressed as $H = \sum_1^r \sum_1^c (P_{p,m} \odot I_{I(p,m)})$, where (p, m) are the indices of the active event at a particular timestamp. This hypervector serves as the "**memory**" of the events, preserving the association between location and light intensity for all pixels within the given time window.

6.1.3. Association-based temporal encoding

The proposed temporal encoder, shown in Fig. 4c, is a method for preserving the temporal relationships between events in a neuromorphic system. The encoder uses pre-generated hypervectors that represent timestamps to bind with spatial-encoded events, creating a combined representation of the event's spatial location and time of occurrence. This allows the encoder to preserve the spatial–temporal information of the events.

To illustrate the use of the temporal encoder, suppose we have spatial-encoded events and want to combine them over time. We can use the pre-generated time-based correlated hypervectors, which are explained above. For example, for an event happening at time t_i, the encoder can preserve the spatio-temporal information using the following equation: $\mathbf{E} = P_{p,m} \odot I_I \odot t_i$.

Here, $P_{p,m}$ represents the spatial encoding of the event, I_I is the pre-generated hypervector that represents the timestamp, and t_i is the actual timestamp of the event. For a group of events occurring over a time period of n time-steps, the temporal encoder can preserve the relationships between these events using the bundling operation, as shown in the following equation: $\mathbf{E} = \sum_{i=1}^{n} \mathbf{E}_i$ The temporal encoder described above is a way to preserve the temporal relationship between events in a neuromorphic system, which is useful for tracking the temporal evolution of events in a neuromorphic system.

6.2. GraspHD: Learning for smart grasping

The proposed GraspHD model uses the spatio-temporal encoding technique to encode the raw events from a neuromorphic camera. The goal is to train the model to learn the attributes of the main grasped object, such as its hardness (H), size (OS), and to estimate the appropriate grasping force (GF) while holding it. The system architecture is shown in Fig. 5. Unlike previous works that pre-process the neuromorphic data to extract features for detecting grasped object characteristics [13] or grasping pose [26]. Our proposed encoder uses the **raw** events to predict the object's contact and ensure grasping stability using the HDC properties. In this work, two different mapping models (encoders) are proposed. The first one generates a classification model for each possible label (H, OS, and GF). The size of each model depends on the number of possible combinations for each label. In this case, we have three labels, each with three different values, resulting in a total of three models, each with three classes of hypervectors. The second model transforms the multi-label task into a single-label problem by generating a new class for every label combination. The implementation for each mapping model is discussed in detail in the following sections.

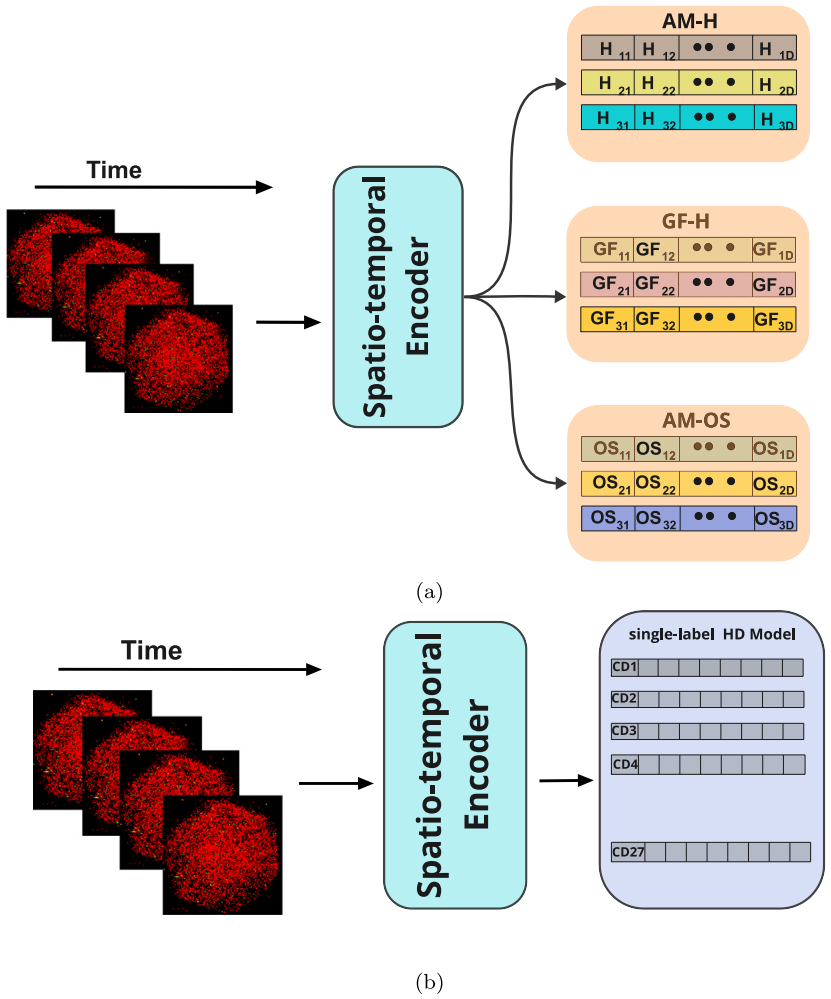

Fig. 6. Multilabel vs. Single label GraspHD models. (a) GraspHD multi-label model: An overview of the three learning modules used to classify object's Hardness (H), Grasping Force (GF), and Object Size (OS). In muti-model HD, an independent classification models for every label are created and evaluated.(b) GraspHD single-label model: each set of labels is mapped to a single label. For this work, the number of label set for classes HD model is 27.

6.2.1. *Smart grasping multi-label classification*

The grasping dataset used in this work contains highly nonlinear relations among the variables H, GF, and OS. Furthermore, the size of the data set is small, which limits the number of encoded samples that can be used to train HD models [49]. To take advantage of the event information from the event-based vision sensor, we propose developing three encoding modules for the object hardness (H), grasping force (GF), and object size (OS). These modules are illustrated in Fig. 6(a). The method used for building the HD model for each label is similar to the problem transformation method [70] which involves transforming a multi-label classification problem into multiple binary classification problems, one for each label. In this case, each event is encoded and added to multiple classes in each of the three models. For instance, suppose we have an encoded spike with label [H:"low", GF:"medium", OS:"small"], the data point is added to H model in Fig. 6(a) for class H_1 hypervector for first label, as the "low" label is for the first label in H model. It is then additionally added to GF model for class GF_2 hypervector for the second label, as the "medium" label is for the second class in GF model, and it is also added to class OS_1 hypervector for the third label, as the "small" label is for the first class for OS model. During inference, we feed the query data into all three models in parallel and check for the existence of each label. This allows us to take full advantage of the event-based information and improve the accuracy of our grasping predictions.

6.2.2. *Smart grasping single-label classification*

In this approach, we transform a multi-label dataset into a single-label classification problem by creating a new class for each possible combination of labels. The transformation method used in this work is PT3, which combines each set of labels into a single label [70]. For this particular dataset, there are three possible labels for each sample and each label has three possible values. Therefore, the number of new label sets for the HD model is $3^3 = 27$ as shown in Fig. 6(b). We refer to this resulting model as the

Algorithm 1 Smart Grasping Multi-Label Classification using HDC

Input: Neuromorphic data
Output: Multi-label classification

Initialize the System:
Generate Seed Hypervectors for light polarity I_i , 2D position $P_{p,m}$, and timestamp T_i with pre-defined dimension.
Define learning rate η and other hyperparameters.

Encoding: Convert Event-Based Data to HD Space
Spatial Encoding:
for each triggered event at pixel coordinate $P_{p,m}$ with polarity I_i **do**
    $H_{\text{spatial}} \leftarrow \sum_{r1} \sum_{c1} (P_{p,m} \otimes I_{I(p,m)})$
end for
Temporal Encoding:
for each spatial-encoded event H_{spatial} at time t_i **do**
    $E \leftarrow P_{p,m} \otimes I_i \otimes t_i$
    $E_{\text{temporal}} \leftarrow \sum E_i$
end for

Training Phase:
Initialize Class Hypervectors for each label (H, GF, OS).

Adaptive Training:
for each new data point E_i **do**
    compute similarity $\delta_l = \frac{E_i \cdot C_l}{|E_i||C_l|}$
    Update $C_{\text{correct}} \leftarrow C_{\text{correct}} + \eta\delta_l \times E_i$
    Update $C_{\text{wrong}} \leftarrow C_{\text{wrong}} - \eta\delta_l \times E_i$
end for

Inference:
Map test dataset to HD space
for each class hypervector C_l **do**
    Compute similarity $\delta_l = \delta_l(Q, C_l)$
end for
Determine winning class $C_l^* = \arg\max_l \delta_l$

Iterative Re-Training (Optional):
Starting from the initial adaptive model:
for each input hypervector E_i **do**
    Update class hypervectors:

$$C_K \leftarrow C_K + \eta(1 - \delta(E_i, C_K)) \times E_i$$

end for

Result Interpretation:
Determine the grasping properties (H, GF, OS) of the classified objects.
Guide the robotic grasping.

J model, with "J" standing for "joint labels". It is important to note that for J model, the number of labels increases exponentially with the number of labels, which can impact the complexity of the generated HD model [71].

6.2.3. Threshold rate for noise calibration

The main challenge of using the vision-based DAVIS sensor for grasping tasks is noise, which can come from sources such as the gripper's vibrations, sensor temperature, and lighting conditions. This noise can lead to a high signal-to-noise ratio (SNR) and negatively affect the system's performance. To address this issue, a proposed calibration method samples the noise and improves grasping quality. It is noticed that every single experiment has three stages: grasping, holding, and releasing. The two prominent peaks in Fig. 7a correspond to the grasping and releasing phases, respectively. Additionally, we analyze the number of events triggered during these stages for all samples in the training dataset, and a threshold value is determined to differentiate between noise and informative events, as demonstrated in Fig. 7b. This process is used to determine a global threshold value that can be used to filter out noise in the event data and improve grasping performance. The number of events per bin is observed to be in the range of $(200 - 300)$ spikes, 97% of the time. This value is used as a threshold to keep track of the noise rate for each bin. If the rate goes above the threshold, the model considers the event informative and encodes it to the HD space. In other words, for a window

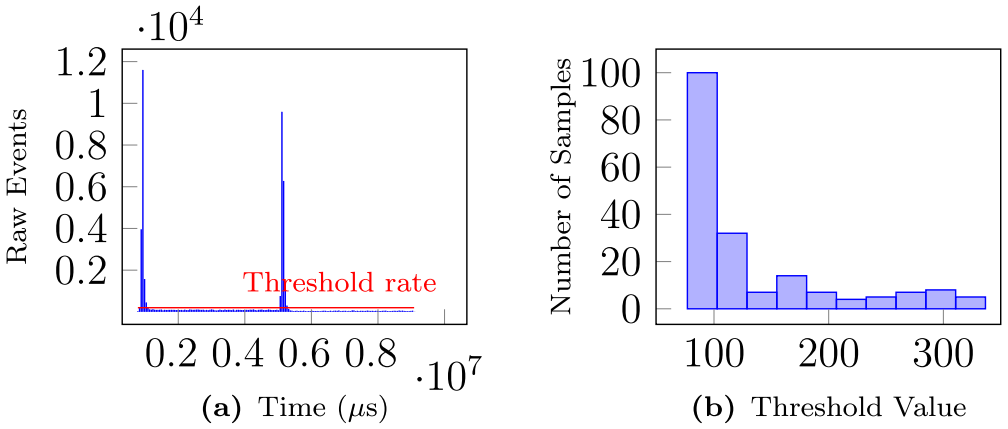

Fig. 7. Threshold analysis for noise calibration. (a) An example for the histogram of events during grasping, holding and releasing stages reveals that 95%–97% number of events/bin is in the range of 200–300 event. (b) Threshold value examined globally for all samples in the training dataset and used to mitigate the noise effect during encoding events to the high space.

of pixels, if the difference between the current time slot t and the previous time slot $(t - \Delta t)$ in the number of triggered events is more than the threshold value, the events are mapped to HD space; otherwise, they are considered as a noise and ignored. Fig. 3(b) shows the effect of filtering out the noise on the encoded events. The impact of noise reduction on improving the model's accuracy and efficiency will be further discussed in the Section 7.

7. GraspHD evaluation metrics

In this section, we evaluate the GraspHD system utilizing the events captured during the grasping and releasing phases and collected in EBOG dataset [13].

7.1. Workload

GraspHD training and testing, with a ratio of 80/20, is verified. The accuracy and efficiency of GraspHD are assessed using the event-based dataset for grasping, which is collected through DAVIS mounted on the robot hand as depicted in Section 3. The dataset is designed to classify the object contact level, H, GF, and OS based on DAVIS data. Our experiments correspond to DAVIS240C cameras with 240 × 180 pixel resolution. For ground truth, the gripper is equipped with an F/T sensor to track the force value during the grasping. Furthermore, we implement GraspHD using software and hardware. In software, we verified GraspHD training and testing using a python implementation. Its design is configured to have hyperdimension vectors of length $D = 4000$, a kernel size of $k = 3$ for spatial correlation, and a time window size of $t = 50000$ μs across all experiments, as it leads to the best average performance. In hardware, we practically tested the design's functionality on Xilinx Zynq Ultrascale+ ZCU104 Evaluation Kit. Fig. 8 shows the overall hardware design. Xilinx Zynq platform includes the processing system part(ARM CPU) and the programming logic part(FPGA) [72]. Specifically, ARM CPU provides DAVIS data and random seed to FPGA. The GraspHD model is implemented on FPGA, which reads DAVIS data from the CPU via AXI interconnect IP(AXI_I IP) and return classification result. The results are reported along three axes: accuracy, robustness, and energy efficiency, and we compare our approach against existing works [13,49].

7.2. GraspHD quality and comparative evaluation

In our objective to advance the domain of robotic grasping, we present Table 1, which demonstrates the performance of GraspHD in contact level classification. This performance is benchmarked against ML methodologies that harness event-based sensor data for similar tasks. Our metric of choice, accuracy, serves the dual purpose of measuring the ability to determine object size (OS) and hardness (H) and assuring the grasp's overall reliability and stability. GraspHD demonstrates performance on par with the notable work in [13] when identifying object hardness. However, we noted a slight decrease in precision when identifying object size and estimating grasping force. These findings are crucial to accurately assessing GraspHD's current capabilities. This nuanced performance reflects our commitment to operating with raw sensor data amidst the inherent noise of a realistic environment. This is marked as a shift from the noise-tuned, pre-processed data approach utilized in [13]. Our paradigm actively forgoes the sequential classification model, which first resolves hardness, then force and size and is adopted in [13]. GraspHD, with its innovative asynchronous classification, simplifies the learning process, elevating both the speed and efficiency of the system. This agility is particularly advantageous in fast-paced industrial scenarios, such as pick-and-place operations, where prompt processing is not just a benefit but a necessity. It is crucial to acknowledge the marked improvement in model performance post noise calibration, a testament to the system's capacity for noise mitigation. The interaction between the DAVIS sensor and mechanical vibrations of the

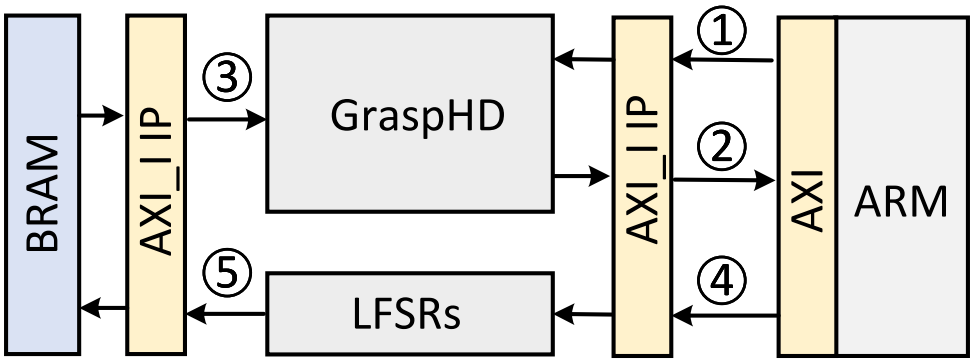

Fig. 8. GraspHD implementation on FPGA. 1. ARM CPU(host) loads DAVIS data into FPGA(kernel) via AXI interface. 2. Kernel GraspHD IP returns inference result back host Arm CPU. 3. GraspHD IP load timestamp Hypervectors from on-chip BRAM. 4. Host ARM CPU gives linear feedback shift registers(LFSRs) random seed. 5. LFSRs IP writes timestamp random hypervector into on-chip BRAM.

Table 1
Comparison of GraspHD learning with SOTA methods using accuracy metric and additional comparison criteria. GraspHD achieves a comparable accuracy with work in [13] for object hardness (H). However, the accuracy level is decreased for object size (OS) and for estimating the grasping force (GF) compared to the SVM results. Notably, SVM and KNN approaches require pre-processed events, feature extraction, and denoising, while GraspHD operates on neuromorphic-based end-to-end raw data. The table also compares whether each approach filters the noise, analyzes robustness to data variation, and requires data pre-processing. When considering these additional factors, GraspHD's performance is superior to the ML methods, particularly in handling noise and data variation.

Method	Accuracy			Noise	Robustness	Data processing
	H	G	S			
GraspHD[a]	**98.1**%	**81.5**%	**100**%	✓	✓	x
GraspHD[b]	84.6%	71.9%	81%	x	✓	x
SVM [13]	81.5%	77.8%	88.9%	x	x	✓
KNN [13]	74.1%	72.2%	88.9%	x	x	✓
DNN [49]	–	–	93%	✓	x	✓

[a] After noise calibration.
[b] Before noise calibration.

grippers, along with fluctuations in sensor temperature and ambient lighting are the primary sources of noise. GraspHD successfully addresses this challenge.

While GraspHD showcases promising results in object contact-level classification, demonstrating competencies comparable to state-of-the-art methods, it is essential to shed light on the framework's limitations and sensitivities. Our experiments are conducted under highly structured conditions to ensure the results' repeatability and mitigate external variables that could impact the model's performance. However, it has been observed that GraspHD's accuracy in detecting correct object attributes is susceptible to environmental noise and lighting variations. Our research has revealed that under various lighting conditions, there is a degradation in the system's ability to accurately identify object attributes. This sensitivity is attributed to the nature of the DAVIS camera and the model's processing capabilities, which could be more adept at filtering out the inherent noise by employing advanced filtering techniques and handling the complexity of light-dynamic environments. Similar challenges arise with more complex objects, where the complex details and variable surface textures can impact the model's classification accuracy. These findings, based on thorough and rigorous experimentation, provide a clear understanding of GraspHD's limitations. The Conclusion and Discussion section will elaborate more on the planned expansions and the strategies to address the identified limitations.

Interestingly, in most scenarios, our model achieves a 100% accuracy rate in predicting the size of objects. This high level of accuracy is likely attributable to several key factors: the cloud of events generated around objects upon contact, the effective preservation of these events' spatial positions through 2D spatial encoding, and the bundling operation. Together, these elements enable GraspHD to recall object sizes more accurately than other variables. This observation aligns with and reinforces the conclusions of our previous research [13].

Subsequent sections delve deeper into the robustness of GraspHD, rigorously discussing its resilience to data variability and responsiveness, setting the stage for a comprehensive understanding of its operational prowess.

Multi-label vs. single-label class: We evaluate the GraspHD model accuracy for two configurations: multi-label classification and single-label classification. Under the same parameters as multi-label GraspHD in Table 1, the GraspHD model for the single-label achieves an accuracy of 85%. This value is lower than the accuracy for the multi-labels model for hardness and object size, though it is higher than the prediction for grasping force. This implies that the single-label model is worse than the multi-label individual models overall. The weak behavior of the single-label model is due to the limited number of samples used to train the single label, as

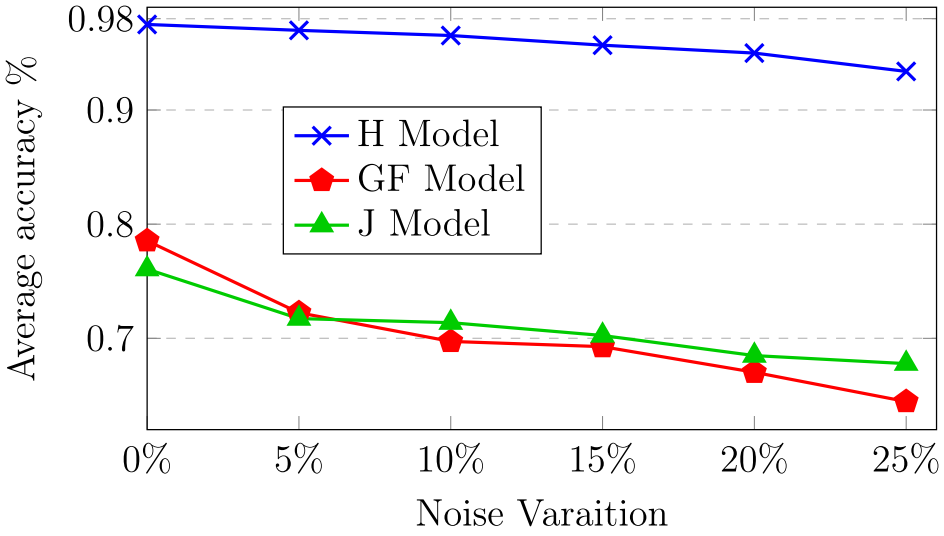

Fig. 9. GraspHD models robustness to variation. The H model is highly robust against potential variation in data. However, the GF and J model are sensitive to the added noise, as the accuracy drops about 8% for 5% fluctuation in data. The OS model accuracy does not change with data fluctuation and therefore has not added to the figure.

only nine samples were used for each label combination for training. In addition, as stated in Section 6.2.2, due to the exponential increase in class hypervectors, the multi-labels model is faster, more efficient, and requires less energy than the single-label model.

Temporal encoding & multi-class updates:

In GraspHD, the temporal correlation within hypervectors is preserved through association-based time encoding with correlative hypervectors [73]. This approach mitigates the sensitivity to scaling and shifting commonly found in methods that rely on the order of time through permutation [62].

For encoding the time information, GraspHD employs both static and dynamic approaches. In the static approach, $\frac{t}{k}$ random vectors are generated, where t represents the experiment time, and k is the sub-window time with $k \gg D$. This selection ensures that consecutive timestamps translate into highly similar hypervectors in hyperspace, providing reasonable time resolution. The static approach is further expanded into a dynamic method that tracks only two hypervectors in real-time. The multi-class iterative refinements allow the simultaneous update of two neighboring class hypervectors, enhancing model robustness against noise.

The learning rate, indicated by (η), plays a critical role in these updates. It is computed based on the distance between the query vector Q and the associated class hypervector: The class hypervectors are then updated using the following rules:

$$C_{\text{correct}} = C_{\text{correct}} + \eta \cdot Q \tag{5}$$

$$C_{\text{wrong}} = C_{\text{wrong}} - \eta \cdot Q \tag{6}$$

These updates adapt to the distance between the query hypervector Q and the class hypervector. Significant distance leads to major changes in the class hypervectors, while marginal mispredictions result in smooth and minor adjustments. By leveraging η, GraspHD achieves a nuanced adaptation to the underlying data structure. It maintains the temporal correlation's integrity and facilitates an effective learning process that aligns with the dynamic nature of the data.

7.3. Temporal correlation and GraspHD performance

To further investigate the importance of temporal correlation, we analyzed the GraspHD performance when the time effect is ablated while conserving the other variables during the encoding process. Then, we compared the performance of GraspHD when using both spatial and temporal encodings to the version when only using spatial encoding. We found that ignoring temporal encoding impacted the performance of GraspHD significantly. In particular, we observe a significant decrease in the system's ability to estimate the grasping force (GF) as the accuracy decreases by 10%. One possible explanation for these results is that the object's properties change over time. That is because the object's hardness varies over time due to factors such as deformation or external forces acting on it, which may not be accurately reflected in the encoded data if the temporal relationship between events is not considered. That can reduce accuracy when estimating the grasping force, as the appropriate grasping force depends on the object's current hardness. Additionally, we observed that the model requires many iterations to learn the task. In the GraspHD framework, the temporal correlation is considered a memorization term in the encoding scheme. Besides, the temporal relationship between events provides important information that may be lost if the temporal correlation is not considered. As a result, more retraining steps are needed to compensate for the time information, accurately classify the object's properties, and estimate the appropriate grasping force.

Table 2
The resource utilization of HDC model acceleration on Xilinx UltraScale+ MPSoC ZCU 104. Here FPGA kernel frequency is 200 MHz and the power consumption is 7.1 W.

	LUT	FF	DSP	BRAM	UltraRAM
Total	157.8 K	125.2 K	7	56	0
Available	230.4 K	460.8 K	1728	624	96
Utilization	68.4%	27.1%	~0%	8.9%	0%

7.4. GraspHD robustness

As we discussed earlier, the main issue in DAVIS is its sensitivity to the noise coming from various sources in the vicinity. Up to date, the ML and DNN methods utilizing events data require complex algorithms to filter the noise in the dataset, which increase the learning time, and degrade its performance [74,75]. However, in GraspHD, the data representation in high dimensional space is robust due to the redundancy and holography of the hypervectors: in HD space, hypervectors are generated randomly with i.i.d components so every hypervector stores the information across its elements holographically and there is no one component is responsible for storing a single piece of information more than another. In addition, the encoding module keeps the triggered events' spatial and temporal information correlated, rendering the representation shift-resistant. To test the GraspHD ability to response to noisy data, we simulate the noise by flipping the polarity for a percentage of triggered events, selected randomly. Fig. 9 shows the quality of grasping for GraspHD after injecting the noise. The evaluation shows that the model for the object hardness is highly robust against potential variation in data. It provides the maximum accuracy even for 5% noisy data. In contrast, the model for the estimated grasping force is more sensitive to the added noise, as the accuracy drops about 8% for 5% fluctuation in data. It is remarkable to mention that the object size model is highly tolerant to the noisy data, as the accuracy remains at its maximum level even after 25% variation in input data. From the results above, we deduce that exploiting the spatio-temporal encoder for the cloud of events related to object hardness and size preserve the spike's triggering time and location in a way that enable GraspHD model to memorize events even when noise is added to input data. Besides, the holographic representation for spikes in HD space, making GraspHD rigid against minor variation in HD components as long this variation does not dominate the entire space. In contrast, the SVM and KNN method used in [13] are sensitive to any changes in input data, which degrade the grasping quality and reduce the performance of the whole grasping system.

7.5. Feasibility of GraspHD for real-time implementation

In addition to its accuracy, the GraspHD framework has also been evaluated for its running time at inference. We found the time required to encode query input into HD space and compare it with the hypervector classes for object hardness, size, and estimate the grasping force in the associative memory is around 2.2 s. This time is comparable to the response time of the robot, which is measured at 2.5 s. These results suggest that the GraspHD framework is suitable for use in real-time grasping applications, as it can process data and make decisions within the time frame of the robot's response. In order to optimize the running time of the GraspHD model, steps such as parallelization or hardware realization may be considered in the future. In addition, it is important to consider the response time of the robot to the tasks it can perform, as well as any limitations it may have. Overall, the performance trade-offs of the GraspHD model in terms of running time and accuracy can be carefully balanced to achieve the desired results in practical scenarios.

7.6. FPGA implementation

As mentioned above in Section 7.1 and shown in Fig. 8, we tested the GraspHD functionality on Xilinx Zynq Ultrascale+ ZCU104 Evaluation Kit. In Table 2 we present the FPGA resource utilization. To provide a comprehensive illustration of our FPGA accelerator implementation overhead, we present look-up table (LUT), flip-flops (FF), digital signal processing (DSP), block random-access memory (BRAM), and Ultrascale random access memory (UltraRAM) ultilizations in Table 2. In the FPGA implementation of the GraspHD model, a detailed allocation of hardware resources underscores the system's efficiency. The utilization of Look-Up Tables (LUTs) stands at 68.4% of the total 230.4 K available, serving as the cornerstone for the logic functions critical to GraspHD's operations. Similarly, Flip-Flops (FFs) are utilized at 27.1% of the available 460.8 K, highlighting the model's adeptness at managing temporal data. Notably, the Digital Signal Processors (DSPs) see nearly negligible usage, indicative of GraspHD's streamlined approach that minimizes dependence on complex arithmetic operations. Moreover, only 8.9% of the available 624 Block Random-Access Memories (BRAMs) units are employed, demonstrating the model's efficient memory management capabilities crucial for storing and accessing temporary data during processing. The non-utilization of UltraRAM further emphasizes the model's operational efficiency, performing within the constraints of the available BRAM without compromising on processing speed or accuracy. This detailed account of resource utilization showcases GraspHD's efficient use of FPGA hardware and its potential for scalability and adaptability in broader applications, supporting its viability as an effective solution for real-time robotic grasping tasks.

The GraspHD FPGA accelerator demonstrates exceptional performance and efficiency, processing single images within approximately 11.12 μs, enabling real-time data analysis essential for robotic grasping and manipulation tasks. Moreover, its operation at 7.6 W of power consumption marks a substantial leap towards energy-efficient computing in robotic systems. This efficiency

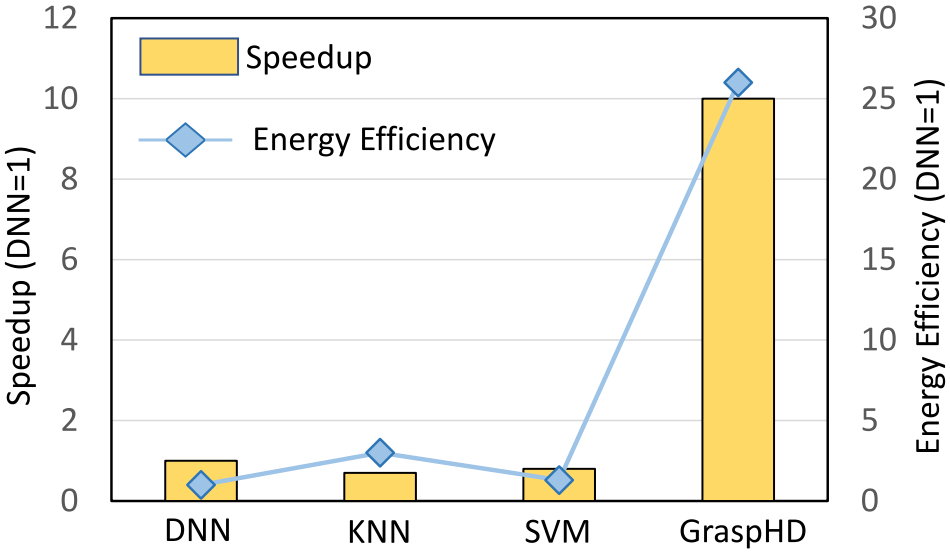

Fig. 10. Comparison of GraspHD efficiency as compared to state-of-the-art DNN, KNN, and SVM on FPGA.

contrasts with traditional DNN implementations on FPGA platforms [76]. We compare GraspHD efficiency and robustness to existing SOTA solutions [76,77]. The results are for the total processing throughput and energy efficiency, including both preprocessing and learning. Fig. 10 compares GraspHD computation efficiency with the existing SOTA solutions running on FPGA. The results are reported for both training and inference phases. For DNN, we used DNNWeaver V2.0 [78] for the inference and FPDeep [25] for training implementation on a single FPGA device. For SVM and KNN, we referred previous design [76,77], and implemented on FPGA by ourselves. All FPGA implementations are optimized to maximize performance by utilizing FPGA resources. All results, in Fig. 10, are relative to DNN throughput and energy efficiency. The latency of DNN acceleration on FPGA is over 120 μs with power consumption over 15 W. GraspHD achieves, on average, 10× faster and 26× more energy-efficient computation as compared to FPGA-based DNN implementation, respectively. The high efficiency of GraspHD in training comes from GraspHD capability in: (i) creating an initial model that significantly lowers the number of required retraining iterations and (ii) eliminating the costly gradient for the model update. This results in higher GraspHD efficiency, even in terms of a single training iteration. For inference, the main computation efficiency comes from eliminating the costly preprocessing step and replacing it with HDC encoding.

8. Conclusion

In this work, we have introduced GraspHD, a pioneering end-to-end framework designed to enhance the efficiency and robustness of robotic grasping through the innovative application of hyperdimensional computing. GraspHD's novel approach to encoding DAVIS spiking events into high-dimensional vectors captures the data's inherent sparsity and spatiotemporal characteristics, ensuring the preservation of vital spatial and temporal correlations among the events. This capability allows for precise classification of object contact levels, paving the way for robots to acquire prior knowledge about an object's attributes, such as hardness, size, and the required grasping force, directly from DAVIS sensor data.

The contributions and advantages of GraspHD are manifold and significant when benchmarked against state-of-the-art methodologies in the field [13,49]. Firstly, GraspHD's remarkable ability to process raw spiking events for learning without necessitating any pre-processing represents a substantial leap forward, eliminating the complexities and computational overhead associated with data preprocessing. Secondly, the framework's streamlined architecture not only simplifies the learning process but also substantially reduces the network size and the number of parameters that need to be learned. This efficiency is particularly advantageous in contexts where data availability is limited, yet GraspHD still manages to deliver remarkable learning outcomes. Furthermore, the FPGA implementation of GraspHD highlights its superior performance, demonstrating a processing speed that is 10× faster and energy efficiency 26× greater than DNN algorithms. These contributions underscore GraspHD's potential to revolutionize robotic grasping learning, offering a more efficient and robust solution that aligns with the evolving needs of robotics applications. By harnessing the power of HDC, GraspHD opens new avenues for future research and development in the field.

9. Discussion

Challenges and future works

For the rest of this section, we will outline limitations and challenges that our GraspHD framework still needs to address, as well as potential avenues for future research.

Noisy Environment: A primary limitation is GraspHD's vulnerability to high signal-to-noise ratios (SNR) in data from vision-based DAVIS cameras, which can potentially detract from system performance. The system's inability to filter out noise effectively may obscure informative events, necessitating the development of advanced noise reduction techniques or incorporating supplementary sensors to bolster grasping accuracy in noisy environments.

Hardware Integration : Another challenge is the hardware integration of GraspHD with other robotic systems and sensors, using hardware accelerators such as FPGA and evaluate its performance in a real-world scenarios.

Scalability and generalization: GraspHD's current model has yet to be rigorously tested in large-scale applications or across a wide range of object classes, potentially limiting its complexity handling and efficiency. Furthermore, the model's training under constrained conditions raises concerns about its generalizability to new objects and its adaptability to environmental changes, such as lighting variations or alterations in object shape, size and texture. These factors could impact GraspHD's reliability and applicability in diverse real-world settings.

Potential avenues for future work could encompass, but are not limited to, the following: (1) **Integration with Robotic Manipulators**: Coupling the DAVIS with robotic hands for enhanced real-time application promises to broaden GraspHD's operational scope, facilitating more sophisticated tasks like dexterous object manipulation. (2) **Dexterous Grasping**: Exploring GraspHD's utility for a broader range of grasping tasks, such as object manipulation and dexterous grasping [79]. This could extend the algorithm's capabilities, enabling it to handle more intricate tasks. (3) **Comprehensive Performance Improvement**: Conducting comprehensive experiments with a broader array of objects – varying in size, shape, and texture – is essential to heighten our understanding of GraspHD's strengths and limitations, guiding targeted enhancements to the system's design and algorithms. (4) **Online Learning Enhancement**: Develop GraspHD's online learning capabilities to allow real-time model updates based on new data and feedback acquired during its operation. This will enable GraspHD to dynamically adapt to new objects and environments, continually improving its accuracy and efficiency.

In addition, many generalizations can be made to integrate HDC approaches to extend our methods to alternative tasks involve but not limited to the neuromorphic data. Analyzing each of its layers:

1. Hypervector substrates: for the efficiency of our approach, we have selected bipolar hypervectors as our basis of operation. While this is suited particularly to our purpose, Hyperdimensional computing is also defined over real numbers and complex numbers, which consequently possess more expressive power at the price of computational complexity. In particular, there is a natural connection between spiking neurons and hyperdimensional computing with complex vectors, which renders the latter a favorable companion for neuromorphic hardware.
2. Atomic concepts and encoding: our selection of atomic concept, i.e. components of data that are directly mapped to the hyperspace, is simply all the parameters within each event (timestamp, coordinates, and polarity), and we have proposed simple methods for projecting them to hyperspace such that their similarity conforms to a natural metrics. Alternative selection and more advanced encoding methods, including kernel-inspired approaches, can be applied such that the similarity between the hypervectors is more suitable for certain tasks. This is certainly necessary in the case where the parameters of the data change.
3. Composition schema for complex objects: to represent an event from the atomic hypervectors, we perform essentially the binding of all atomic hypervectors corresponding to its time, space, and polarity components; for representation of the memory of events within a certain window, we applied bundling over all events within that window. In other words, we have selected certain algorithms to compose complex concepts from simpler and atomic ones; we call these algorithms *schemas* for those concepts. While we have demonstrated the effectiveness of our schemas, alternative designs can be made that affect the similarity between compound hypervectors correspondingly. The design of schema is an active part of HDC methodology and we encourage the practitioner to attend to them when learning and working with HDC.

Neuromorphic computing acceleration on FPGA

Previous studies have demonstrated significant achievements in the domain of neuromorphic acceleration on FPGA platforms [80]. Notably, there has been extensive research focused on the acceleration of event-based spiking neural networks (SNNs) on computing platforms [81]. Specifically, we delve into two state-of-the-art (SOTA) FPGA acceleration works concerning SNNs [82,83]. The first work [82] focuses on the implementation of fault-tolerant, context-dependent learning FPGA acceleration solutions at a large scale, while the second work [83] centers around the acceleration of the CerebelluMorphic network on an FPGA. Both SNNs and the CerebelluMorphic network are event-based models designed to closely mimic the functioning of the human brain.

Additionally, it is noteworthy to mention that hyperdimensional computing (HDC) models have also been influenced by advancements in neural science [84]. In comparison to neural network and similar models (such as CNN and SNN), HDC models exhibit greater hardware compatibility and are more amenable to achieving parallelism on domain-specific accelerators [85–87]. Furthermore, with the integration of an event-based mechanism, the HDC model can effectively simulate the functionality of the human brain, approaching a level of resemblance comparable to that of SNNs. Consequently, the HDC model represents an alternative solution to SNNs in the context of event-based neural network acceleration.

Integrating GraspHD with security measures

As we transition towards the quantum computing era, robotic systems, including those powered by frameworks like GraspHD, become increasingly integral to various sectors, and the need for robust security protocols become essential. GraspHD, primary focuses on operational efficiency and perception accuracy, anticipates the adoption of security considerations are part of its roadmap for future enhancements. The emergence of Post-Quantum Cryptography (PQC) and the shift towards efficient, low-energy cryptographic solutions present new challenges and opportunities for enhancing the security of these systems.

Recent advancements in fault diagnosis schemes and error detection mechanisms offer promising avenues for securing cryptographic operations within the realm of PQC [88–91]. Specifically, the development of CRC-based error detection methods [89] and the assessment of error detection schemes in FPGA environments [90] highlight the importance of resilient cryptographic systems in an era increasingly defined by quantum computing capabilities.

Furthermore, the push towards lightweight cryptography [92,93] and the exploration of secure, energy-efficient architectures [94] are crucial for ensuring the adaptability and sustainability of robotic systems in resource-constrained environments. These considerations are not only suitable to the robustness of GraspHD but also its potential integration within larger, potentially vulnerable networks.

As we look towards the future of GraspHD and similar robotic frameworks, integrating advanced security measures, informed by the latest research in PQC and lightweight cryptography, will be essential. This includes exploring reliable architectures for finite field multipliers [93], optimizing ECC implementations [92], and addressing the challenges posed by side-channel attacks and fault attacks [95,96]. By aligning GraspHD with emerging security standards and considerations, we can ensure that the next generation of robotic systems is efficient, adaptable but also secure, and resilient against evolving cyber threats.

CRediT authorship contribution statement

Eman Hassan: Writing – review & editing, Writing – original draft, Validation, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Conceptualization. **Zhuowen Zou:** Writing – review & editing, Methodology, Investigation, Formal analysis. **Hanning Chen:** Writing – review & editing, Methodology, Investigation, Formal analysis. **Mohsen Imani:** Supervision. **Yahya Zweiri:** Supervision, Data curation. **Baker Mohammad:** Writing – review & editing, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The datasets analyzed during the current study are event-based data collected in our lab utilizing the DAVIS sensor for robot grasping applications. The data for this paper is available at https://github.com/emfhasan/GraspHD.

Acknowledgments

This publication is based upon work supported by the System-on-Chip Center, Khalifa University Award No. [RC2-2018-020] and Competitive Internal Research Award (CIRA) under Award No. [CIRA-2019-026]. Further support from Advanced Research and Innovation Center (ARIC), Khalifa University, Abu Dhabi, UAE. Also, this work was supported in part by DARPA Young Faculty Award, National Science Foundation #2127780 and #2312517, Semiconductor Research Corporation (SRC), Office of Naval Research, grants #N00014-21-1-2225 and #N00014-22-1-2067, the Air Force Office of Scientific Research, United States under award #FA9550-22-1-0253, and generous gifts from Xilinx, AMD, and Cisco.

References

[1] Y. Ge, Z. Li, X. Yue, H. Li, Q. Li, L. Meng, IoT-based automatic deep learning model generation and the application on empty-dish recycling robots, Internet Things (2023) 101047.
[2] G. Aceto, V. Persico, A. Pescapé, A survey on information and communication technologies for industry 4.0: State-of-the-art, taxonomies, perspectives, and challenges, IEEE Commun. Surv. Tutor. 21 (4) (2019) 3467–3501.
[3] G. Du, K. Wang, S. Lian, K. Zhao, Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review, Artif. Intell. Rev. 54 (3) (2021) 1677–1734.
[4] H. Duan, P. Wang, Y. Huang, G. Xu, W. Wei, X. Shen, Robotics dexterous grasping: The methods based on point cloud and deep learning, Front. Neurorobotics 15 (2021) 73.
[5] R. Muthusamy, X. Huang, Y. Zweiri, L. Seneviratne, D. Gan, Neuromorphic event-based slip detection and suppression in robotic grasping and manipulation, IEEE Access 8 (2020) 153364–153384.
[6] R. Monica, J. Aleotti, Point cloud projective analysis for part-based grasp planning, IEEE Robot. Autom. Lett. 5 (3) (2020) 4695–4702.
[7] J. Jiang, S. Luo, Robotic perception of object properties using tactile sensing, in: Tactile Sensing, Skill Learning, and Robotic Dexterous Manipulation, Elsevier, 2022, pp. 23–44.
[8] W. Yuan, Y. Mo, S. Wang, E.H. Adelson, Active clothing material perception using tactile sensing and deep learning, in: 2018 IEEE International Conference on Robotics and Automation, ICRA, 2018, pp. 4842–4849.

[9] G. Cao, J. Jiang, C. Lu, D.F. Gomes, S. Luo, TouchRoller: A rolling optical tactile sensor for rapid assessment of textures for Large Surface Areas, Sensors 23 (5) (2023) 2661.

[10] S. Weiss, D. Scaramuzza, R. Siegwart, Monocular-SLAM–based navigation for autonomous micro helicopters in GPS-denied environments, J. Field Robotics 28 (6) (2011) 854–874.

[11] K. Ganguly, P. Mantripragada, C.M. Parameshwara, C. Fermüller, N.J. Sanket, Y. Aloimonos, GradTac: Spatio-temporal gradient based tactile sensing, Front. Robot. AI 9 (2022) 898075.

[12] F. Veiga, B. Edin, J. Peters, Grip stabilization through independent finger tactile feedback control, Sensors 20 (6) (2020) 1748.

[13] X. Huang, R. Muthusamy, E. Hassan, Z. Niu, L. Seneviratne, D. Gan, Y. Zweiri, Neuromorphic vision based contact-level classification in robotic grasping applications, Sensors 20 (17) (2020) 4724.

[14] M. Graña, M. Alonso, A. Izaguirre, A panoramic survey on grasping research trends and topics, Cybern. Syst. 50 (1) (2019) 40–57.

[15] Q. Lei, J. Meijer, M. Wisse, A survey of unknown object grasping and our fast grasping algorithm-C shape grasping, in: 2017 3rd International Conference on Control, Automation and Robotics, ICCAR, IEEE, 2017, pp. 150–157.

[16] C. Bousquet-Jette, S. Achiche, D. Beaini, Y.L.-K. Cio, C. Leblond-Ménard, M. Raison, Fast scene analysis using vision and artificial intelligence for object prehension by an assistive robot, Eng. Appl. Artif. Intell. 63 (2017) 33–44.

[17] R. Qin, H. Ma, B.-B. Gao, D. Huang, RGB-D grasp detection via depth guided learning with cross-modal attention, in: 2023 IEEE International Conference on Robotics and Automation, ICRA, 2023, pp. 8003–8009.

[18] C. Brandli, R. Berner, M. Yang, S.-C. Liu, T. Delbruck, A 240 × 180 130 dB 3 μs latency global shutter spatiotemporal vision sensor, IEEE J. Solid-State Circuits 49 (10) (2014) 2333–2341.

[19] Y. Li, Y. Kim, H. Park, T. Geller, P. Panda, Neuromorphic data augmentation for training spiking neural networks, in: European Conference on Computer Vision, Springer, 2022, pp. 631–649.

[20] X. Huang, K. Sanket, A. Ayyad, F.B. Naeini, D. Makris, Y. Zweir, A neuromorphic dataset for object segmentation in indoor cluttered environment, 2023, arXiv preprint arXiv:2302.06301.

[21] Y. Li, F. Guo, M. Zhang, S. Suo, Q. An, J. Li, Y. Wang, A novel deep learning-based pose estimation method for robotic grasping of axisymmetric bodies in industrial stacked scenarios, Machines 10 (12) (2022) 1141.

[22] L. Pinto, A. Gupta, Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours, in: 2016 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2016, pp. 3406–3413.

[23] S. Li, W. Sun, Q. Liang, C. Liu, J. Liu, Assessing fruit hardness in robot hands using electric gripper actuators with tactile sensors, Sensors Actuators A 365 (2024) 114843.

[24] H.X. Huynh, B.H. Lam, H.V.C. Le, T.T. Le, N. Duong-Trung, Design of an IoT ultrasonic-vision based system for automatic fruit sorting utilizing size and color, Internet Things (2023) 101017.

[25] T. Geng, T. Wang, A. Sanaullah, C. Yang, R. Xu, R. Patel, M. Herbordt, FPDeep: Acceleration and load balancing of CNN training on FPGA clusters, in: 2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines, FCCM, IEEE, 2018, pp. 81–84.

[26] K. Kleeberger, R. Bormann, W. Kraus, M.F. Huber, A survey on learning-based robotic grasping, Curr. Robot. Rep. 1 (4) (2020) 239–249.

[27] X. Huang, M. Halwani, R. Muthusamy, A. Ayyad, D. Swart, L. Seneviratne, D. Gan, Y. Zweiri, Real-time grasping strategies using event camera, J. Intell. Manuf. (2022) 1–23.

[28] A. Ramisa, G. Alenya, F. Moreno-Noguer, C. Torras, Learning RGB-D descriptors of garment parts for informed robot grasping, Eng. Appl. Artif. Intell. 35 (2014) 246–258.

[29] U. Viereck, A. Pas, K. Saenko, R. Platt, Learning a visuomotor controller for real world robotic grasping using simulated depth images, in: Conference on Robot Learning, PMLR, 2017, pp. 291–300.

[30] J. Cai, H. Cheng, Z. Zhang, J. Su, Metagrasp: Data efficient grasping by affordance interpreter network, in: 2019 International Conference on Robotics and Automation, ICRA, IEEE, 2019, pp. 4960–4966.

[31] E. Solowjow, I. Ugalde, Y. Shahapurkar, J. Aparicio, J. Mahler, V. Satish, K. Goldberg, H. Claussen, Industrial robot grasping with deep learning using a Programmable Logic Controller (PLC), in: 2020 IEEE 16th International Conference on Automation Science and Engineering, CASE, 2020, pp. 97–103.

[32] J. Mahler, M. Matl, V. Satish, M. Danielczuk, B. DeRose, S. McKinley, K. Goldberg, Learning ambidextrous robot grasping policies, Science Robotics 4 (26) (2019) eaau4984.

[33] H. Xu, Y. Sun, Q. Sun, M. Yang, J. Chen, B. Qiang, J. Wang, 3D grasp pose generation from 2D anchors and local surface, in: The 18th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry, 2022, pp. 1–7.

[34] A. ten Pas, C. Keil, R. Platt, Efficient and accurate candidate generation for grasp pose detection in SE(3), in: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2021, pp. 5725–5732.

[35] K.-T. Song, Y.-H. Chang, J.-H. Chen, 3D vision for object grasp and obstacle avoidance of a collaborative robot, in: 2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM, IEEE, 2019, pp. 254–258.

[36] D. Yang, T. Tosun, B. Eisner, V. Isler, D. Lee, Robotic grasping through combined image-based grasp proposal and 3d reconstruction, in: 2021 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2021, pp. 6350–6356.

[37] N. Singh, Z. Blum, N. Renjith, Point cloud grasp classification for robot grasping, 2018.

[38] C. Zhihong, Z. Hebin, W. Yanbo, L. Binyan, L. Yu, A vision-based robotic grasping system using deep learning for garbage sorting, in: 2017 36th Chinese Control Conference, CCC, IEEE, 2017, pp. 11223–11226.

[39] F. Mou, H. Ren, B. Wang, D. Wu, Pose estimation and robotic insertion tasks based on YOLO and layout features, Eng. Appl. Artif. Intell. 114 (2022) 105164.

[40] P. De La Puente, D. Fischinger, M. Bajones, D. Wolf, M. Vincze, Grasping objects from the floor in assistive robotics: Real world implications and lessons learned, IEEE Access 7 (2019) 123725–123735.

[41] M. Lambeta, P.-W. Chou, S. Tian, B. Yang, B. Maloon, V.R. Most, D. Stroud, R. Santos, A. Byagowi, G. Kammerer, et al., Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation, IEEE Robot. Autom. Lett. 5 (3) (2020) 3838–3845.

[42] K. Ganguly, B. Sadrfaridpour, P. Mantripragada, N.J. Sanket, C. Fermüller, Y. Aloimonos, Grasping in the dark: Zero-shot object grasping using tactile feedback, 2020, arXiv preprint arXiv:2011.00712.

[43] R. Muthusamy, A. Ayyad, M. Halwani, D. Swart, D. Gan, L. Seneviratne, Y. Zweiri, Neuromorphic eye-in-hand visual servoing, IEEE Access 9 (2021) 55853–55870.

[44] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, et al., Event-based vision: A survey, 2019, arXiv preprint arXiv:1904.08405.

[45] F. Liao, F. Zhou, Y. Chai, Neuromorphic vision sensors: Principle, progress and perspectives, J. Semicond. 42 (1) (2021) 013105.

[46] P. Lichtsteiner, C. Posch, T. Delbruck, A 128× 128 120 dB 15 μs latency asynchronous temporal contrast vision sensor, IEEE J. Solid-State Circuits 43 (2) (2008) 566–576.

[47] C. Posch, D. Matolin, R. Wohlgenannt, A QVGA 143 dB dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS, IEEE J. Solid-State Circuits 46 (1) (2011) 259–275.

[48] L. Steffen, D. Reichard, J. Weinland, J. Kaiser, A. Roennau, R. Dillmann, Neuromorphic stereo vision: A survey of bio-inspired sensors and algorithms, Front. Neurorobotics 13 (2019) 28.

[49] B. Li, H. Cao, Z. Qu, Y. Hu, Z. Wang, Z. Liang, Event-based robotic grasping detection with neuromorphic vision sensor and event-stream dataset, Front. Neurorobotics 14 (2020).
[50] H. Cao, G. Chen, Z. Li, Y. Hu, A. Knoll, NeuroGrasp: multimodal neural network with Euler region regression for neuromorphic vision-based grasp pose estimation, IEEE Trans. Instrum. Meas. 71 (2022) 1–11.
[51] A. Rigi, F. Baghaei Naeini, D. Makris, Y. Zweiri, A novel event-based incipient slip detection using dynamic active-pixel vision sensor (DAVIS), Sensors 18 (2) (2018) 333.
[52] F. Baghaei Naeini, A.M. AlAli, R. Al-Husari, A. Rigi, M.K. Al-Sharman, D. Makris, Y. Zweiri, A novel dynamic-vision-based approach for tactile sensing applications, IEEE Trans. Instrum. Meas. 69 (5) (2020) 1881–1893.
[53] F.B. Naeini, S. Kachole, R. Muthusamy, D. Makris, Y. Zweiri, Event augmentation for contact force measurements, IEEE Access 10 (2022) 123651–123660.
[54] P. Kanerva, Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors, Cogn. Comput. 1 (2) (2009) 139–159.
[55] M. Imani, Hyperdimensional neural computation, Biophys. J. 121 (3) (2022) 270a–271a.
[56] R.W. Gayler, Vector symbolic architectures are a viable alternative for Jackendoff's challenges, Behav. Brain Sci. 29 (1) (2006) 78–79.
[57] E. Hassan, Y. Halawani, B. Mohammad, H. Saleh, Hyper-dimensional computing challenges and opportunities for AI applications, IEEE Access (2021).
[58] D. Kleyko, A. Rahimi, D.A. Rachkovskij, E. Osipov, J.M. Rabaey, Classification and recall with binary hyperdimensional computing: Tradeoffs in choice of density and mapping characteristics, IEEE Trans. Neural Netw. Learn. Syst. 29 (2018) 5880–5898.
[59] M.N. Jones, D.J.K. Mewhort, Representing word meaning and order information in a composite holographic lexicon, Psychol. Rev. 114 (1) (2007) 1–37.
[60] P. Kanerva, J. Kristoferson, A. Holst, Random indexing of text samples for latent semantic analysis, in: Proceedings of the Annual Meeting of the Cognitive Science Society, Vol. 22, 2000, pp. 1–2.
[61] Y. Kim, M. Imani, T.S. Rosing, Efficient human activity recognition using hyperdimensional computing, in: Proceedings of the 8th International Conference on the Internet of Things, 2018, pp. 1–6.
[62] A. Rahimi, S. Benatti, P. Kanerva, L. Benini, J.M. Rabaey, Hyperdimensional biosignal processing: A case study for EMG-based hand gesture recognition, in: 2016 IEEE International Conference on Rebooting Computing, ICRC, IEEE, 2016, pp. 1–8.
[63] A. Rahimi, S. Datta, D. Kleyko, E.P. Frady, B. Olshausen, P. Kanerva, J.M. Rabaey, High-dimensional computing as a nanoscalable paradigm, IEEE Trans. Circuits Syst. I. Regul. Pap. 64 (9) (2017) 2508–2521.
[64] E.-J. Chang, A. Rahimi, L. Benini, A.-Y.A. Wu, Hyperdimensional computing-based multimodality emotion recognition with physiological signals, in: IEEE International Conference on Artificial Intelligence Circuits and Systems, AICAS, IEEE, 2019, pp. 137–141.
[65] F. Yang, S. Ren, On the vulnerability of hyperdimensional computing-based classifiers to adversarial attacks, in: International Conference on Network and System Security, Springer, 2020, pp. 371–387.
[66] A. Mitrokhin, P. Sutor, C. Fermüller, Y. Aloimonos, Learning sensorimotor control with neuromorphic sensors: Toward hyperdimensional active perception, Science Robotics 4 (30) (2019).
[67] M. Imani, S. Bosch, S. Datta, S. Ramakrishna, S. Salamat, J.M. Rabaey, T. Rosing, QuantHD: A quantization framework for hyperdimensional computing, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. (2019).
[68] A. Hernandez-Cane, N. Matsumoto, E. Ping, M. Imani, Onlinehd: Robust, efficient, and single-pass online learning using hyperdimensional system, in: 2021 Design, Automation & Test in Europe Conference & Exhibition, DATE, IEEE, 2021, pp. 56–61.
[69] J. Kim, H. Lee, M. Imani, Y. Kim, Efficient brain-inspired hyperdimensional learning with spatiotemporal structured data, in: 2021 29th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, MASCOTS, IEEE, 2021, pp. 1–8.
[70] T. Durand, N. Mehrasa, G. Mori, Learning a deep convnet for multi-label classification with partial labels, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 647–657.
[71] J. Morris, Y. Hao, S. Gupta, R. Ramkumar, J. Yu, M. Imani, B. Aksanli, T. Rosing, Multi-label HD classification in 3D flash, in: 2020 IFIP/IEEE 28th International Conference on Very Large Scale Integration, VLSI-SOC, IEEE, 2020, pp. 10–15.
[72] J. Lee, H. Chen, J. Young, H. Kim, RISC-V FPGA platform toward ROS-based robotics application, in: 2020 30th International Conference on Field-Programmable Logic and Applications, FPL, IEEE, 2020, p. 370.
[73] P. Poduval, Z. Zou, X. Yin, E. Sadredini, M. Imani, Cognitive correlative encoding for genome sequence matching in hyperdimensional system, in: 2021 58th ACM/IEEE Design Automation Conference, DAC, 2021, pp. 781–786.
[74] S. Gupta, A. Gupta, Dealing with noise problem in machine learning data-sets: A systematic review, Procedia Comput. Sci. 161 (2019) 466–474.
[75] T. Ching, D.S. Himmelstein, B.K. Beaulieu-Jones, A.A. Kalinin, B.T. Do, G.P. Way, E. Ferrero, P.-M. Agapow, M. Zietz, M.M. Hoffman, et al., Opportunities and obstacles for deep learning in biology and medicine, J. R. Soc. Interface 15 (141) (2018) 20170387.
[76] A. Saidi, S.B. Othman, M. Dhouibi, S.B. Saoud, FPGA-based implementation of classification techniques: A survey, Integration 81 (2021) 280–299.
[77] S. Afifi, H. GholamHosseini, R. Sinha, FPGA implementations of SVM classifiers: A review, SN Comput. Sci. 1 (3) (2020) 1–17.
[78] H. Sharma, J. Park, D. Mahajan, E. Amaro, J.K. Kim, C. Shao, A. Mishra, H. Esmaeilzadeh, From high-level deep neural models to FPGAs, in: 2016 49th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO, IEEE, 2016, pp. 1–12.
[79] R. Muthusamy, J.M. Indave, P.K. Muthusamy, E.F. Hasan, Y. Zweiri, V. Kyrki, D. Gan, Investigation and design of robotic assistance control system for cooperative manipulation, in: 2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems, CYBER, IEEE, 2019, pp. 889–895.
[80] Q. Wang, Y. Li, B. Shao, S. Dey, P. Li, Energy efficient parallel neuromorphic architectures with approximate arithmetic on FPGA, Neurocomputing 221 (2017) 146–158.
[81] J. Han, Z. Li, W. Zheng, Y. Zhang, Hardware implementation of spiking neural networks on FPGA, Tsinghua Sci. Technol. 25 (4) (2020) 479–486.
[82] S. Yang, J. Wang, B. Deng, M.R. Azghadi, B. Linares-Barranco, Neuromorphic context-dependent learning framework with fault-tolerant spike routing, IEEE Trans. Neural Netw. Learn. Syst. 33 (12) (2021) 7126–7140.
[83] S. Yang, J. Wang, N. Zhang, B. Deng, Y. Pang, M.R. Azghadi, CerebelluMorphic: large-scale neuromorphic model and architecture for supervised motor learning, IEEE Trans. Neural Netw. Learn. Syst. 33 (9) (2021) 4398–4412.
[84] P. Kanerva, Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors, Cogn. Comput. 1 (2) (2009) 139–159.
[85] S. Salamat, M. Imani, B. Khaleghi, T. Rosing, F5-hd: Fast flexible fpga-based framework for refreshing hyperdimensional computing, in: Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 2019, pp. 53–62.
[86] M. Imani, Z. Zou, S. Bosch, S.A. Rao, S. Salamat, V. Kumar, Y. Kim, T. Rosing, Revisiting hyperdimensional learning for fpga and low-power architectures, in: 2021 IEEE International Symposium on High-Performance Computer Architecture, HPCA, IEEE, 2021, pp. 221–234.
[87] H. Chen, M. Issa, Y. Ni, M. Imani, DARL: Distributed reconfigurable accelerator for hyperdimensional reinforcement learning, in: Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design, 2022, pp. 1–9.
[88] M.M. Kermani, S. Bayat-Sarmadi, A.-B. Ackie, R. Azarderakhsh, High-performance fault diagnosis schemes for efficient hash algorithm blake, in: 2019 IEEE 10th Latin American Symposium on Circuits & Systems, LASCAS, IEEE, 2019, pp. 201–204.
[89] A.C. Canto, M.M. Kermani, R. Azarderakhsh, CRC-based error detection constructions for FLT and ITA finite field inversions over GF (2 m), IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 29 (5) (2021) 1033–1037.

[90] A.C. Canto, A. Sarker, J. Kaur, M.M. Kermani, R. Azarderakhsh, Error detection schemes assessed on FPGA for multipliers in lattice-based key encapsulation mechanisms in post-quantum cryptography, IEEE Trans. Emerg. Top. Comput. (2022).
[91] J. Kaur, A.C. Canto, M.M. Kermani, R. Azarderakhsh, Hardware constructions for error detection in WG-29 stream cipher benchmarked on FPGA, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. (2023).
[92] M.B. Niasar, R. Azarderakhsh, M.M. Kermani, Optimized architectures for elliptic curve cryptography over Curve448, Cryptol. ePrint Arch. (2020).
[93] A. Cintas-Canto, M.M. Kermani, R. Azarderakhsh, Reliable architectures for finite field multipliers using cyclic codes on FPGA utilized in classic and post-quantum cryptography, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 31 (1) (2022) 157–161.
[94] A. Cintas-Canto, J. Kaur, M. Mozaffari-Kermani, R. Azarderakhsh, ChatGPT vs. Lightweight security: First work implementing the NIST cryptographic standard ASCON, 2023, arXiv preprint arXiv:2306.08178.
[95] M.M. Kermani, R. Azarderakhsh, J. Xie, Error detection reliable architectures of Camellia block cipher applicable to different variants of its substitution boxes, in: 2016 IEEE Asian Hardware-Oriented Security and Trust, AsianHOST, IEEE, 2016, pp. 1–6.
[96] A. Aghaie, M.M. Kermani, R. Azarderakhsh, Fault diagnosis schemes for low-energy block cipher Midori benchmarked on FPGA, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 25 (4) (2016) 1528–1536.