Neuromorphic Computing and Vision for Interactive Robotics

Muhammad Aitsam

A thesis submitted in partial fulfilment of the requirements of

Sheffield Hallam University

for the degree of Doctor of Philosophy

September 2025

Abstract

Neuromorphic computing and event-based vision seek to replicate the brain's sparse, spike-driven information flow, offering a route to robots that see and react with the speed and parsimony of their biological counterparts. This thesis pursues that goal through a single narrative arc that begins with computation, advances through perception, and culminates in adaptive interaction. It starts by turning frame-based deep networks into spiking models and deploying them on the SpiNNaker neuromorphic platform. Careful comparisons of conversion algorithms show how timing precision, power consumption, and accuracy interplay, establishing practical guidelines for real-time deployment. With spiking computation in place, the thesis next addresses perception. A dynamic-attention mechanism realised as a recurrent spiking neural network keeps track of multiple moving objects in asynchronous event streams, granting robots the ability to prioritise salient targets while ignoring distractors. An accompanying data pipeline exploits the microsecond resolution of event cameras, supporting both hand-gesture recognition and vibration-based machinery monitoring. In tests, these perception modules maintain high accuracy under challenging lighting and rapid motion, confirming the advantages of event-level sensing. The final stage connects machine perception to human intent. Event-based vision is fused with physiological and behavioural indicators to infer the cognitive load of a human partner, enabling the robot to adjust its actions to the user's real-time mental state. This multimodal loop closes the gap between low-level spikes and high-level collaboration, demonstrating how neuromorphic methods can underpin more intuitive human–robot teamwork. Taken together, the thesis charts a coherent path from energy-efficient spiking computation, through event-driven attention and sensing, to cognitively aware interaction, illustrating how each layer supports the next in building responsive, robust and scalable neuromorphic robotic systems.

I hereby declare that:

1. I have not been enrolled for another award of the University, or other academic or professional organisation, whilst undertaking my research degree.

2. None of the material contained in the thesis has been used in any other submission for an academic award.

3. I certify that this thesis is my own work. The use of all published or other sources of material consulted have been properly and fully acknowledged. I confirm that I have sought and obtained copyright permission for any third-party materials included in this thesis. I used AI at AITS 2 (AI for Shaping) of the Artificial Intelligence Transparency Scale (AITS). I acknowledge the use of ChatGPT (https://chat.openai.com) to assist with refining the clarity of written text and ensuring consistency in British English spelling and grammar throughout the preparation of this thesis.

4. The work undertaken towards the thesis has been conducted in accordance with the SHU Principles of Integrity in Research and the SHU Research Ethics Policy, and ethics approval has been granted for all research studies in the thesis, as shown in the table below.

Ethics Review ID	**Title of Study**	**Approval date**
2024-126 of 2024/04/26 (Bielefeld University)	Cognitive Load and Behaviours tracked with Event Camera Pose Estimation	29/04/2024
2024-244 of 2024/08/30 (Bielefeld University)	Multi-Modal Cognitive Load Estimation during Human Robot Interaction	03/09/2024
ER70783970	Multi-Modal Cognitive Load Estimation during Human Robot Interaction	07/09/2024

5. The word count of the thesis is 41,410

Name: Muhammad Aitsam

Award: Doctor of Philosophy

Date of Submission: September 2025

College: Business, Technology and Engineering

Director(s) of Studies: Prof. Alessandro Di Nuovo

Supervisors: Dr. Sergio Davies, Dr. Alejandro Jimenez Rodriguez

Acknowledgment

First and foremost, I would like to express my deepest gratitude to my supervisory team. I am profoundly thankful to Prof. Alessandro Di Nuovo, whose vision, mentorship, and unwavering belief in my work have shaped every step of this journey. I also wish to extend my heartfelt thanks to Dr. Sergio Davies and Dr. Alejandro Jimenez Rodriguez for their continuous guidance, insightful feedback, and support throughout this research.

This thesis would not have been possible without the lifelong support of my parents, who instilled in me the values of perseverance and integrity. Their sacrifices, prayers, and unconditional love are the foundation of everything I have achieved.

To my beloved wife, Maira Ahmad, words will never be enough to capture the depth of my gratitude. You have been my constant companion in every sense, sharing in the late nights, the setbacks, and the triumphs with patience and grace. Your encouragement carried me through moments of doubt, and your resilience reminded me of the bigger picture when the journey felt overwhelming. You sacrificed your own comfort and embraced the demands of this path as if they were your own. More than a partner, you have been a true co-traveller in this PhD journey, and it is no exaggeration to say that this work belongs to you as much as it does to me. I dedicate this thesis to you with all my heart and deepest appreciation.

To my son, Muhammad Izhaan Aitsam, you may be too young to understand the meaning of this work, but your presence has been the greatest source of motivation and joy. Every smile, every small moment of happiness with you, reminded me why perseverance mattered and gave me the strength to finish with purpose.

I am equally grateful to my sisters, Maha Farhat and Rimsha Midhat, for their love, encouragement, and belief in me. Alongside them, my nephews and nieces, Aarish, Muhammad, Ayat, and Abrish, brought joy and light into my life during the most demanding phases of this journey.

To the wonderful members of the Smart Interactive Technologies (SIT) Research Lab, thank you for creating an environment of curiosity, collaboration, and friendship. A very special thanks goes to Imene Tarakli and Kavyan Zoughalian; more than

colleagues, they are my dearest friends. Their unwavering support, thoughtful conversations, and constant encouragement played a pivotal role in shaping this thesis. I am incredibly lucky to have shared this journey with them.

I also extend my gratitude to Gaurvi Goyal and Chiara Bartolozzi for their generous mentorship during my research visit to IIT Genoa in 2024. The time I spent there was deeply enriching, and I am thankful for the opportunity to learn from such brilliant minds. I am also grateful for the fruitful collaborations that enriched my research. My sincere thanks go to Dimitri Lacroix and Samiulhaq Chardiwall, whose ideas and perspectives added valuable depth to my work.

This journey was made possible through the generous support of the MSCA Horizon 2020 PERSEO programme, which gave me the opportunity to pursue a PhD in an area I have long been passionate about. Lastly, I extend my sincere thanks to the School of Computing and Digital Technologies at Sheffield Hallam University (SHU) for their academic support and for fostering a vibrant research culture that allowed this work to flourish.

Research Contributions

The research contributions presented in this thesis have resulted in the following peer-reviewed publications:

1. **Muhammad Aitsam**, Sergio Davies, and Alessandro Di Nuovo, "Neuromorphic Computing for Interactive Robotics: A Systematic Review," in IEEE Access, vol. 10, pp. 122261-122279, 2022 [6].

2. **Muhammad Aitsam**, Samiulhaq Chardiwall, Alejandro Jimenez Rodriguez, and Alessandro Di Nuovo. “Benchmarking ANN-to-SNN Conversion: Dataset-dependent Analysis of Accuracy, Latency, and Spike Efficiency”. In: The 14th International Conference on Biomimetic and Biohybrid Systems, Living Machines 2025.

3. **Muhammad Aitsam**, and Alessandro Di Nuovo. "Energy Efficient Personalized Hand-Gesture Recognition with Neuromorphic Computing." CONCATENATE Workshop, ACM/IEEE International Conference on Human-Robot Interaction (HRI), 2023 [13].

4. **Muhammad Aitsam**, Sergio Davies, and Alessandro Di Nuovo, "Event-driven dynamic attention for multi-object tracking on neuromorphic hardware". In Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR) Workshops, pages 5055–5062, 2025 [7].

5. **Muhammad Aitsam**, Alejandro Jimenez Rodriguez and Alessandro Di Nuovo. "Efficient data processing pipeline for event-based vision datasets: techniques and insights." Engineering Research Express, volume 6, 2024 [10].

6. **Muhammad Aitsam**, Sergio Davies, and Alessandro Di Nuovo. "Event Camera-Based Real-Time Gesture Recognition for Improved Robotic Guidance," 2024 International Joint Conference on Neural Networks (IJCNN), Yokohama, Japan, pp. 1-8, 2024 [8].

7. **Muhammad Aitsam**, Gaurvi Goyal, Chiara Bartolozzi and Alessandro Di Nuovo. Vibration Vision: Real-Time Machinery Fault Diagnosis with Event

Cameras. “European Conference on Computer Vision (ECCV) 2024” - Workshop on Neuromorphic Vision: Advantages and Applications of Event Cameras, Milan, Italy, 2024 [9].

8. **Muhammad Aitsam**, Dimitri Lacroix, Gaurvi Goyal, Chiara Bartolozzi and Alessandro Di Nuovo. Measuring Cognitive Load Through Event Camera Based Human-Pose Estimation. Human-Friendly Robotics (HFR). Springer Proceedings in Advanced Robotics, vol 35. Springer, 2024 [11].

9. **Muhammad Aitsam**, Kavyan Zoughalian, Dimitri Lacroix, and Alessandro Di Nuovo. Multimodal Cognitive Load Estimation in Human-Robot Interaction: Integrating Physiological and Behavioral Dynamics. In: IEEE Transactions on Cognitive and Developmental Systems. [under review]

List of contributions that are not part of this thesis:

1. **Muhammad Aitsam**, Samiulhaq Chardiwall and Alessandro Di Nuovo. Differentially Private Spiking Neural Networks: Enhancing Privacy and Robustness in Social Robotics. In: Proceedings of the 16th International Conference on Global Security, Safety and Sustainability (ICGS3), November 2024 [5].

2. Mehboobeh Dorafshanian, **Muhammad Aitsam**, Mohamed Mejri, Alessandro Di Nuovo. "Beyond Data Collection: Safeguarding User Privacy in Social Robotics," 2024 IEEE International Conference on Industrial Technology (ICIT), Bristol, United Kingdom, 2024, pp. 1-6, 2024 [101].

"The world is its own best model."

Rodney A. Brooks, Intelligence Without Representation *(1991)*

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Biological intelligence has long captivated researchers who seek to understand how living organisms memorize, think, perceive, and act. Humans, in particular, stand out for their remarkable ability to make rapid, well-informed decisions in the face of incomplete or ambiguous information. This ability underlies the complex behaviours necessary for survival in constantly changing environments. Recent advances in computational and behavioural neuroscience, along with progress in embodied cognitive systems, have propelled interdisciplinary work in robotics, where scientists are investigating how the brain, sensors, and actuators can collaborate to perform intricate tasks in real-world settings [47]. For a robot to operate autonomously, it must be able to perceive its surroundings in real time, process sparse information efficiently under tight latency constraints, and adapt to fluctuating conditions through self-learning.

The rise of powerful computing technologies and sophisticated sensing systems has greatly accelerated the field of machine learning, leading to notable successes in both science and commerce. Deep learning methods, which build upon the hierarchical organization of the human visual system, have demonstrated particular promise [216]. However, these models still struggle to match human performance in tasks that require precise motor coordination, swift responsiveness, and robust adaptability, and they frequently encounter scalability issues. A striking example of the gap between human and artificial intelligence is evident when considering energy usage: simulating a human-scale brain on a standard clock-based computer might demand

around 12 gigawatts of power, whereas the human brain itself operates on just about 20 watts [115]. This discrepancy can be largely attributed to the traditional reliance on clock-driven processing, which requires hardware to operate at high frequencies to handle continuous data flow [317].

In contrast, biological organisms rely on spike-based strategies to process information, allowing them to perceive and act with extraordinary efficiency. Reproducing this compact yet powerful neuro-synaptic design in machines stands as a key challenge for achieving human-like intelligence. To overcome the limits of clock-driven models, researchers have turned to neuromorphic computing an approach that merges insights from neuroscience, computing, and electronics to create architectures modeled on the brain's event-driven, energy-efficient communication methods.

1.1 Neuromorphic Computing and Spiking Neural Networks

Neuromorphic computing aims at replicating the remarkable capabilities of the human brain in electronic systems, with a particular focus on low energy consumption, adaptability, and parallel processing [133]. The human brain, which operates on only about 20 watts of power [423], is known for feats that conventional computers struggle to match, such as context-aware decision-making, continuous learning, and adaptability to unfamiliar situations. These strengths stem from key biological principles, including high degrees of connectivity among neurons, extremely efficient spike-based communication, and the co-location of memory and computational units [130, 73].

In the nervous system, billions of neurons form a massively parallel, three-dimensional network connected by trillions of synapses. Each neuron integrates incoming spikes from thousands of other neurons and, if the cumulative input surpasses a threshold, generates its own spike to transmit onward [167]. This event-driven communication stands in stark contrast to the clock-based, frame-driven methods typically found in modern computing hardware. Because neurons only fire

when necessary, energy usage is minimized, and computation is inherently sparse [423]. Another crucial aspect of the brain's efficiency lies in synaptic plasticity, which allows connection strengths between neurons to change based on their firing patterns. Repeated co-activation of two neurons reinforces the synapse between them, forming the basis for Hebbian learning [159]. As a result, brains can learn and adapt without explicitly requiring large, labelled datasets or extensive offline training. Instead, they continuously adjust themselves by interacting with the surrounding environment, a process known as associative learning [90].

Neuromorphic systems aim to capture these features by designing specialized electronic architectures that mirror neural networks, synaptic connections, and plasticity mechanisms [133, 299]. Unlike the von Neumann architecture, where memory and processing elements are physically separated, neuromorphic chips often place these units in close proximity. This co-location dramatically cuts down on data transfer overhead—an inefficiency that has long plagued conventional computers. Additionally, neuromorphic designs support event-based or spike-based processing, so they activate only when relevant signals arrive, which further conserves power. Figure 1.1 shows the conceptual comparison of the traditional Von Neumann architecture versus neuromorphic architecture. Many of today's neuromorphic platforms rely on analog-programmable non-volatile memory devices (including phase change memory, resistive RAM, and other emerging technologies) to realise synapse-like behaviours [274, 83, 322, 67]. These materials exhibit gradually adjustable resistance states, enabling them to store synaptic weights and perform in-memory computations. As a result, hardware built on these principles can execute large-scale neural computations efficiently and in real time [382].

Spiking Neural Networks form the computational backbone of most neuromorphic systems, taking direct inspiration from the way neurons communicate through timed pulses or "spikes." In an SNN, each artificial neuron accumulates weighted inputs from incoming spikes and triggers its own spike once a threshold is reached [275, 386]. The precise timing of these spikes, rather than only their average firing rate, carries critical information, enabling temporally rich and biologically plausible computation compared to traditional Artificial Neural Networks (ANNs). While SNNs share

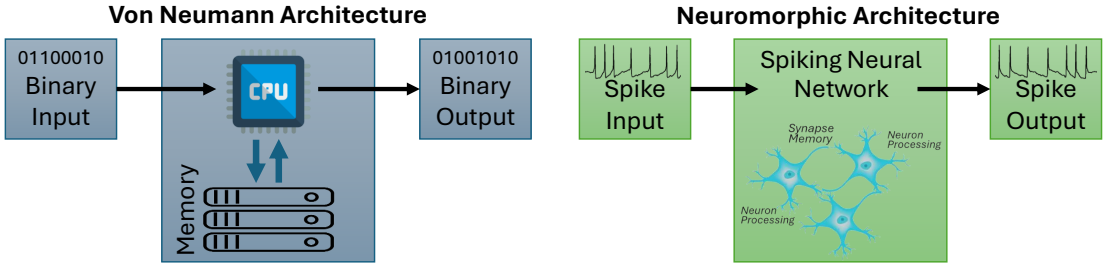


Figure 1.1: Conceptual comparison of the traditional Von Neumann architecture, which processes binary inputs and outputs with separate CPU and memory units, versus the neuromorphic architecture, where spiking neural networks integrate memory and computation to handle spike-based signals.

structural similarities with conventional deep networks, such as layered architectures and weighted synaptic connections, their event-driven dynamics fundamentally change how learning is performed. Traditional ANNs are typically trained using gradient-based backpropagation with continuous activations, requiring large labelled datasets and extensive offline optimisation. In contrast, SNNs often rely on spike-based learning mechanisms, including biologically inspired local rules such as spike-timing-dependent plasticity (STDP), where synaptic updates depend on the relative timing of pre- and post-synaptic spikes [423, 130, 73]. More recently, surrogate-gradient methods and ANN-to-SNN conversion approaches have been developed to bridge the gap between high-performance deep learning and spike-based computation, enabling efficient training while preserving the energy benefits of neuromorphic execution. This combination of sparse event-driven processing and alternative learning paradigms significantly reduces redundant computation, making SNNs particularly well suited for robotics and edge computing applications where low power consumption, real-time responsiveness, and on-device adaptation are essential.

1.2 Neuromorphic Vision

Building on the principles behind neuromorphic engineering and SNNs, researchers have begun investigating advanced sensory modalities that further reduce the gap between biological and engineered perception. One particularly promising approach is event-based vision sometimes called neuromorphic vision which leverages event-driven

data capture to boost efficiency and speed. Event-based cameras, such as Dynamic Vision Sensors (DVS) [231], depart from the traditional frame-based paradigm by operating asynchronously, meaning they capture data only when changes occur rather than at fixed intervals. Rather than capturing complete images at regular intervals, DVS devices register changes in brightness at individual pixels. This yields a continuous stream of events with exceptionally high temporal resolution, minimizing unnecessary data and slashing power requirements. When coupled with neuromorphic processors, event-based sensors can closely emulate the human visual system's ability to interpret complex, fast-moving scenes in real-time [331]. Integrating event-based vision in robotic systems has already demonstrated significant improvements in navigation, obstacle detection, and motion tracking factors that critically enhance autonomy and adaptability [136].

Dynamic Vision Sensors can be regarded as large-scale implementations of non-uniform sampling. Because each pixel produces output only when it detects a sufficient change in brightness, overall data rates remain inherently sparse. In a simple scenario where only one object is in motion, DVS sensors only transmit outputs from the pixels covering that object, while traditional active pixel sensors (APS) must read out and process data from every pixel in each frame. This approach significantly lowers both energy consumption and latency during data processing. One limitation of event-based cameras, however, is that they do not encode static scenes: pixels in an unchanging environment produce no output, whereas APS systems still transmit full frames. A practical solution is to combine the strengths of both techniques incorporating event-based functionality at the pixel level alongside APS features [136]. By merging the advantages of sparse sampling with the ability to register static scenes, such hybrid sensors open new possibilities for smart vision systems that more closely approximate the efficiency and versatility of human sight [128]. Figure 1.2 illustrates how to merge APS with DVS.

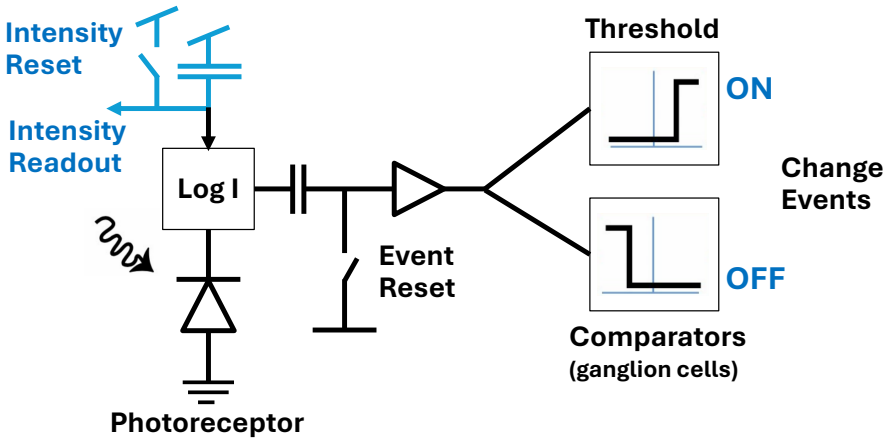


Figure 1.2: Hybrid pixel combining APS read-out and DVS change detection, producing both frame intensity and asynchronous ON/OFF events from the same photoreceptor. This figure is adapted from [55]

1.3 Thesis Contribution and Organization

This thesis explores the development and application of neuromorphic computing methods, spanning hardware and software implementations for spiking neural networks and event-based data processing. The work addresses challenges from foundational preprocessing, such as efficient data pipelines for event-based vision, to advanced applications, including gesture recognition, machinery fault detection, and cognitive load estimation in HRI. It integrates biologically inspired approaches, such as real-time attention models, with engineering-driven solutions for deploying SNNs on neuromorphic hardware like SpiNNaker. The primary focus is on enhancing the efficiency, robustness, and applicability of neuromorphic systems in diverse domains, particularly human-robot interaction. The rest of this thesis presents these contributions in detail.

Chapter 2: Literature Review and Theoretical Background. This chapter provides an overview of neuromorphic computing and vision for interactive robotics, highlighting key challenges and opportunities. It includes two systematic reviews of neuromorphic computing for interactive robotics and neuromorphic vision for interactive robotics, synthesizing trends, findings, and research gaps to establish the foundation for the contributions in subsequent chapters. Some parts of this chapter have been published in:

- **Muhammad Aitsam**, Sergio Davies, and Alessandro Di Nuovo, "Neuromorphic Computing for Interactive Robotics: A Systematic Review," in IEEE Access, vol. 10, pp. 122261-122279, (2022) [6].

Chapter 3: Neuromorphic Computing and Spiking Neural Network (SNN) Implementations. This chapter focuses on the development and deployment of SNNs on neuromorphic hardware. It begins with a theoretical overview of SNNs, their training methods, and ANN-to-SNN conversion techniques. It then presents the systematic evaluation of conversion approaches, discusses the deployment of SNNs on neuromorphic hardware such as SpiNNaker. Then, it introduces event-driven dynamic attention for multi-object tracking with SpiNNaker. The content of this chapter has been published in:

- **Muhammad Aitsam**, Samiulhaq Chardiwall, Alejandro Jimenez Rodriguez, and Alessandro Di Nuovo. “Benchmarking ANN-to-SNN Conversion: Dataset-dependent Analysis of Accuracy, Latency, and Spike Efficiency”. In: The 14th International Conference on Biomimetic and Biohybrid Systems, Living Machines 2025.

- **Muhammad Aitsam**, and Alessandro Di Nuovo. "Energy Efficient Personalized Hand-Gesture Recognition with Neuromorphic Computing." CONCATENATE Workshop, ACM/IEEE International Conference on Human-Robot Interaction (HRI), (2023) [13].

- **Muhammad Aitsam**, Sergio Davies, and Alessandro Di Nuovo, "Event-driven dynamic attention for multi-object tracking on neuromorphic hardware". In Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR) Workshops, pages 5055–5062, (2025) [7].

Chapter 4: Applications of Event Cameras in Robotics. This chapter explores the applications of event cameras in robotic systems. It begins by introducing a data processing pipeline for event-based vision and demonstrates its utility in two key applications: real-time gesture recognition for human–robot interaction, machinery fault

diagnosis. Each application is detailed with its methodology, implementation, and performance analysis, showcasing the practical impact of event-driven neuromorphic approaches. The content of this chapter has been published in:

- **Muhammad Aitsam**, Alejandro Jimenez Rodriguez and Alessandro Di Nuovo. "Efficient data processing pipeline for event-based vision datasets: techniques and insights." Engineering Research Express, volume 6, (2024) [10].
- **Muhammad Aitsam**, Sergio Davies, and Alessandro Di Nuovo. "Event Camera-Based Real-Time Gesture Recognition for Improved Robotic Guidance," 2024 International Joint Conference on Neural Networks (IJCNN), Yokohama, Japan, (2024) [8].
- **Muhammad Aitsam**, Gaurvi Goyal, Chiara Bartolozzi and Alessandro Di Nuovo. Vibration Vision: Real-Time Machinery Fault Diagnosis with Event Cameras. "European Conference on Computer Vision (ECCV) 2024" - Workshop on Neuromorphic Vision: Advantages and Applications of Event Cameras, Milan, Italy, (2024) [9].

Chapter 5: Cognitive Load Estimation in Human–Robot Interaction (HRI). This chapter investigates methods for cognitive load estimation in HRI. It starts with an event-based human-pose estimation study to infer cognitive load and then introduces a multi-modal framework combining physiological signals, behavioural indicators, and other metrics. The findings emphasize the role of cognitive load monitoring in improving robotic adaptability and collaboration with humans. The content of this chapter has been published in:

- **Muhammad Aitsam**, Dimitri Lacroix, Gaurvi Goyal, Chiara Bartolozzi and Alessandro Di Nuovo. Measuring Cognitive Load Through Event Camera Based Human-Pose Estimation. Human-Friendly Robotics (HFR) 2024. Springer Proceedings in Advanced Robotics, vol 35, (2024) [11].
- **Muhammad Aitsam**, Kavyan Zoughalian, Dimitri Lacroix, and Alessandro Di Nuovo. Multimodal Cognitive Load Estimation in Human-Robot Interaction:

Integrating Physiological and behavioural Dynamics. In: IEEE Transactions on Cognitive and Developmental Systems. [under review]

Chapter 6: Conclusions and Future Work. The final chapter summarizes the key contributions of this thesis, including the advancements in neuromorphic computing, SNNs, event-based vision, and adaptive HRI. It also discusses potential directions for future research to further enhance the capabilities of neuromorphic and event-driven systems.

Chapter 2

Literature Review and Theoretical Background

2.1 Introduction

This chapter presents a systematic review of neuromorphic computing and vision, emphasizing their role in interactive robotics. It covers key concepts like spiking neural networks (SNNs) and event-based vision, exploring their integration into robotic systems. Additionally, the chapter traces the evolution of neuromorphic architectures, outlining their benefits and existing challenges.

Novelty & Impact

Novelty: Presents two systematic reviews on neuromorphic computing and vision for interactive robotics, synthesising key trends and clearly identifying research gaps.

Impact: Establishes a structured knowledge base, guiding the research contributions of subsequent chapters and clearly positioning the work within the state-of-the-art.

2.2 Neuromorphic Computing Landscape

Neuromorphic computing represents a transformative leap in artificial intelligence by enabling energy-efficient, adaptive, and real-time processing systems that closely resemble the workings of the human brain. Unlike conventional AI models that require extensive computational resources, neuromorphic systems excel in handling complex, unstructured data while consuming significantly less power. This efficiency arises from their ability to process information asynchronously and in parallel, much like biological neural networks. Additionally, neuromorphic architectures demonstrate exceptional robustness, as they can function effectively in uncertain and dynamic environments, making them ideal for real-world applications where traditional AI struggles. Another key advantage of neuromorphic computing is its capacity for on-chip learning and adaptation, reducing dependency on cloud-based processing and enabling faster decision-making at the edge. This capability is particularly beneficial for embedded systems and autonomous agents that need to operate with minimal latency. In the last decade many neuromorphic computing systems have been developed. Figure 2.1 (taken from [208]) illustrates the historical evolution of neuromorphic computing systems up to the present. The foundational architectures of these systems mark significant milestones in complexity, adaptability, and diversity. However, several key challenges persist across all layers of the stack, which must be overcome to enable large-scale, practical deployment and broader adoption of neuromorphic computing.

The interaction between humans and machines is also of great relevance to both neuromorphic computing and robotics. Utilizing neuromorphic technologies in robotics, from perception to motor control, presents a promising approach to creating robots that can seamlessly integrate into society. In neuromorphic computing and robotics, bio-inspired sensors efficiently encode sensory signals, allowing robots to perceive their surroundings in a manner similar to biological organisms. These neuromorphic sensors, such as event-based vision systems and bio-inspired tactile sensors, enhance a robot's ability to process sensory inputs efficiently while adapting to different environmental conditions. By integrating multiple sensory

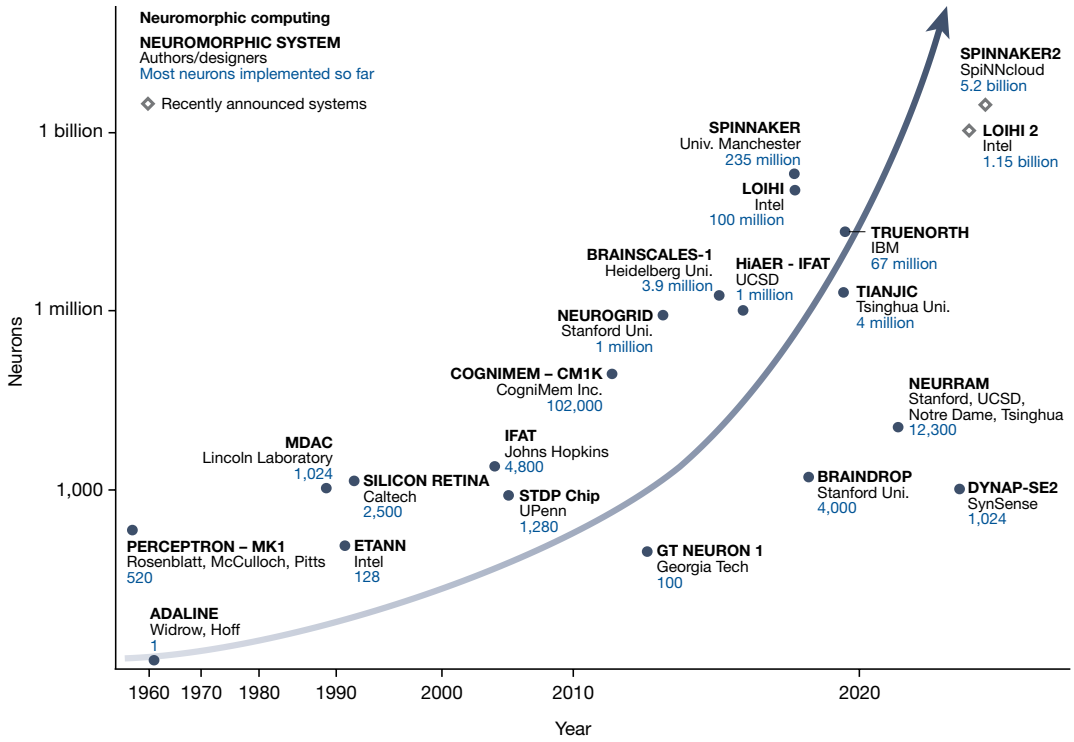


Figure 2.1: Progression of neuromorphic computing systems. This figure is taken from [208].

inputs and leveraging event-based computation, robots can respond dynamically to their surroundings, significantly improving their autonomy and efficiency. Figure 2.2 summarises the landscape of neuromorphic computing and interactive robotics. Hardware and software simulators use specific neuron and synaptic models according to the desired applications.

Despite significant advancements in neuromorphic computing and interactive robotics, a comprehensive and systematic review of the field has yet to be conducted. Except few recent articles like [208], most of the research efforts have primarily addressed specific challenges, offering incremental improvements within the conventional machine learning paradigm rather than introducing transformative changes. To support future research in interactive neurorobotics, this chapter explores the integration of neuromorphic computing, spiking neural networks (SNNs), and event-based vision in interactive robotic systems. It also examines the impact of using SNN models in conjunction with neuromorphic hardware. Our investigation covers both theoretical and practical contributions, including hardware and software platforms essential for

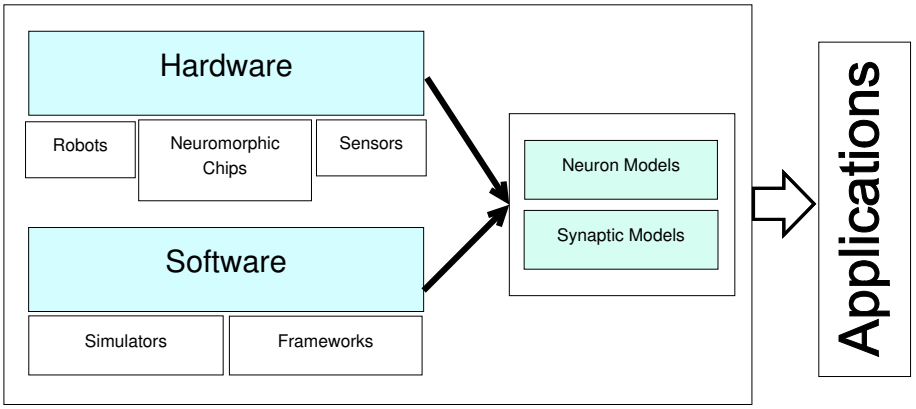


Figure 2.2: Summary of neuromorphic computing and robotics landscape. Hardware and software simulators use specific neuron and synaptic models according to the desired applications.

neurorobotics development. The following sections are structured as follows: We begin by introducing spiking neural networks, along with neuron and synapse models. Section 2.5 outlines a systematic literature review methodology on neuromorphic computing, SNNs, and event-based vision for interactive robotics. The systematic review for Neuromorphic Computing for Interactive Robotics is conducted in Section 2.6. The systematic review for Neuromorphic Vision for Interactive Robotics is conducted in Section 2.7. Section 2.8 presents the summary of this chapter.

2.3 From Artificial Neural Network (ANN) to Spiking Neural Network (SNN)

Neural networks are generally categorized into three generations, each reflecting certain aspects of the multilayer structure of the human brain. However, the way neurons function varies significantly across these generations [386]. The first generation features neurons with binary outputs (0 or 1), determined by applying a threshold to the weighted synaptic input. In 1943, McCulloch and Pitts demonstrated that artificial neural networks based on this principle could perform mathematical and logical computations [255].

Over time, researchers introduced the backpropagation technique for multilayer perceptron networks, addressing the limitations of earlier perceptron models. This

2.3 From Artificial Neural Network (ANN) to Spiking Neural Network (SNN)

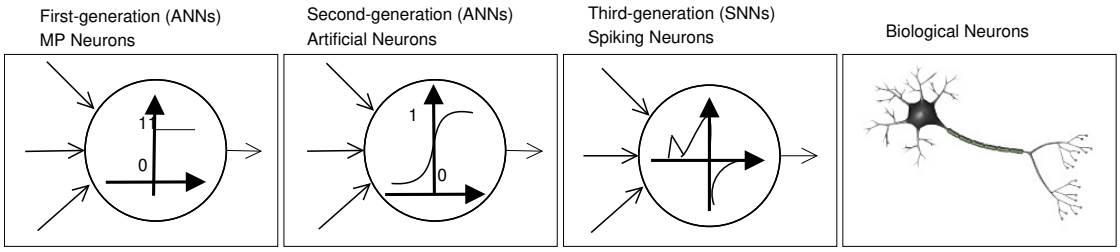


Figure 2.3: Three generations of neural networks. Starting from the McCulloch Pitts (MP) neuron model in the first generation, then artificial neural networks in the second generation, and spiking neural networks in the third generation, which mimic the behaviour of biological neurons.

method, which remains widely used in deep learning today, led to the development of the second generation of neural networks, commonly known as Artificial Neural Networks (ANNs). Unlike the first generation, ANNs produce real-valued outputs by applying a transfer function, typically sigmoidal, to the weighted sum of inputs. The weights are determined using machine-learning algorithms, ranging from basic linear regression to complex classification tasks. With modern computational hardware readily available, researchers continue to explore advanced neural network architectures and learning mechanisms [386].

The first two generations of neural networks progressed from the binary McCulloch–Pitts model to analogue artificial neural networks with continuous activation functions, improving pattern-recognition accuracy but still lacking a faithful, time-continuous representation of neural activity. Biological brains encode information through the precise timing of spikes, whereas recurrent ANNs model time only as discrete iteration steps. Spiking Neural Networks (SNNs) address this gap by allowing neurons to integrate membrane potentials and emit spikes once a threshold is crossed, passing event-timed signals to post-synaptic units and thus capturing the event-driven coding seen in real cortex. Figure 2.3 contrasts these three generations of network models.

SNNs offer several advantages, including temporal plasticity, lower computational complexity, and improved neural interface compatibility [176]. Their growing popularity in recent years has led to the development of models for tasks such as image classification and object recognition. SNNs are well-suited for various computer vision

and robotics applications, including classification, clustering, and pattern recognition. They have been applied in converting sensor data directly into spike-based representations [86][182][247], controlling robotic manipulators in intelligent systems [382][17], and enhancing robotic functionalities [34][292][414]. Additionally, SNNs have been used for detection and recognition tasks [210][117], as well as numerical data processing using the Neural Engineering Framework (NEF) [333][81][364].

2.4 Introduction to Event-Based Vision

Event cameras represent a fundamental shift in visual sensing, capturing per-pixel brightness changes asynchronously with microsecond-level precision. Their unique architecture supports extremely low latency and minimal power consumption, as only meaningful changes in the visual field are processed. One of their most significant advantages is their high dynamic range (HDR) of up to 140 dB, which far exceeds the 60 dB typical of conventional cameras. This capability enables robust performance under challenging lighting conditions such as high glare or low illumination, making them particularly well-suited for dynamic environments.

The origins of event-based vision lie in neuromorphic engineering, specifically the development of biologically inspired sensors such as the Silicon Retina, which paved the way for contemporary event cameras. This lineage is rooted in efforts to replicate the efficiency and reactivity of biological perception systems. Large-scale initiatives such as the Human Brain Project and the U.S. BRAIN Initiative have further accelerated advancements in neuromorphic sensing technologies. Today, companies including Samsung and Prophesee are actively engaged in scaling these technologies for commercial use, recognising their value in fields ranging from autonomous robotics to augmented reality and artificial intelligence [331].

2.4.1 Operating Principles of Event Cameras

Event cameras operate on the principle of detecting changes in brightness, encoding this information asynchronously as discrete events rather than capturing entire

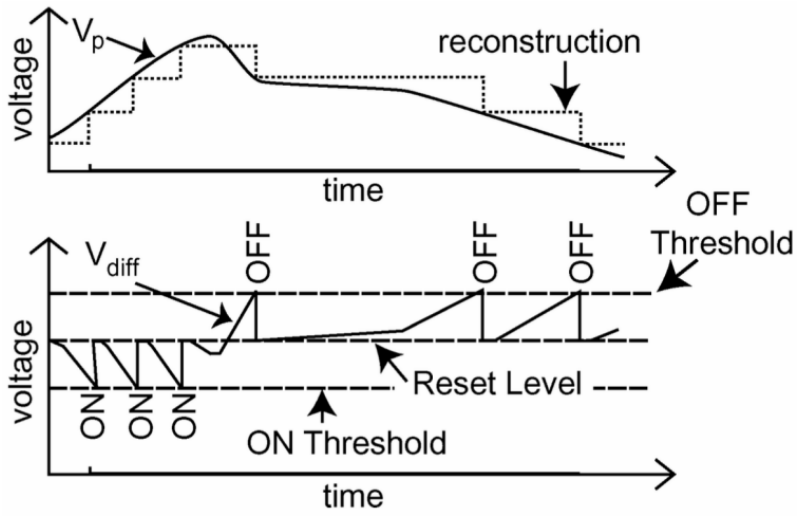


Figure 2.4: Top: pixel-voltage traces cross fixed positive and negative thresholds, emitting ON or OFF events when brightness rises or falls. Bottom: Events accumulated over time outline the moving object, showing that the sensor records only brightness changes. This figure is taken from [331].

frames. Each pixel in the camera continuously monitors the brightness of the scene, and when the change surpasses a predefined threshold, it generates an event. The attached figure illustrates this process, showing how the log intensity of light at a pixel is tracked over time. When the brightness increases or decreases beyond the "ON" or "OFF" thresholds, respectively, an event is triggered, and the brightness level is reset to a new reference point, preparing the pixel to detect subsequent changes.

This encoding mechanism is inspired by the biological spiking nature of visual pathways and is highly efficient in adapting to dynamic scenes. The output of the camera, as shown in Figure 2.4, is a sequence of events, each encoded with the precise time of the change, the pixel location, and the polarity (indicating whether the brightness increased or decreased). This event-driven approach not only minimizes data redundancy but also ensures a high temporal resolution, enabling the detection of fast-moving objects with minimal latency and motion blur. By capturing only meaningful changes in the scene, event cameras reduce computational overhead while delivering highly responsive and efficient vision sensing. Figure 2.4 also highlights the reset level mechanism, which prevents the continuous accumulation of brightness changes and ensures accurate monitoring of future changes. This capability to respond

in real-time to visual stimuli makes event cameras uniquely suited for applications requiring low latency and high-speed processing, such as robotics and augmented reality. By leveraging this asynchronous operation, event cameras provide a robust solution for capturing and processing dynamic visual data.

2.5 Systematic Review Methodology

We divided the systematic review into two parts to gain an in-depth understanding of the field and identify its shortcomings. The first part focuses on (i) Neuromorphic Computing for Interactive Robotics, while the second part explores (ii) Neuromorphic Vision for Interactive Robotics. We conducted our search using three widely used academic databases: *IEEE-Xplorer*, *Scopus* and *Web-of-Science*. Titles, abstracts, and keywords were extracted for an initial screening. This metadata was then added to Rayyan.ai [289], an online tool for systematic reviews. We labelled all relevant publications by searching for our keywords within the title, abstract, and keyword sections. Since this review specifically examines (i) and (ii), we shortlisted publications that presented neuromorphic computing or vision techniques designed to implement new behaviours for human-robot interaction. In cases where the metadata was ambiguous, we also reviewed the full text of the articles. The review process was conducted independently by three authors. An article was shortlisted only if at least two authors agreed on its relevance.

2.6 Neuromorphic Computing for Interactive Robotics

The keywords that we searched were selected according to our goal to identify the scientific articles on the intersection of neuromorphic computing and interactive robotics. To this end, we identified two groups of keywords with similar meanings, respectively: 1) "neuromorphic computing", "spiking neural network" and "brain-inspired computing"; 2) "interactive robotics", "social robotics", and "humanoid

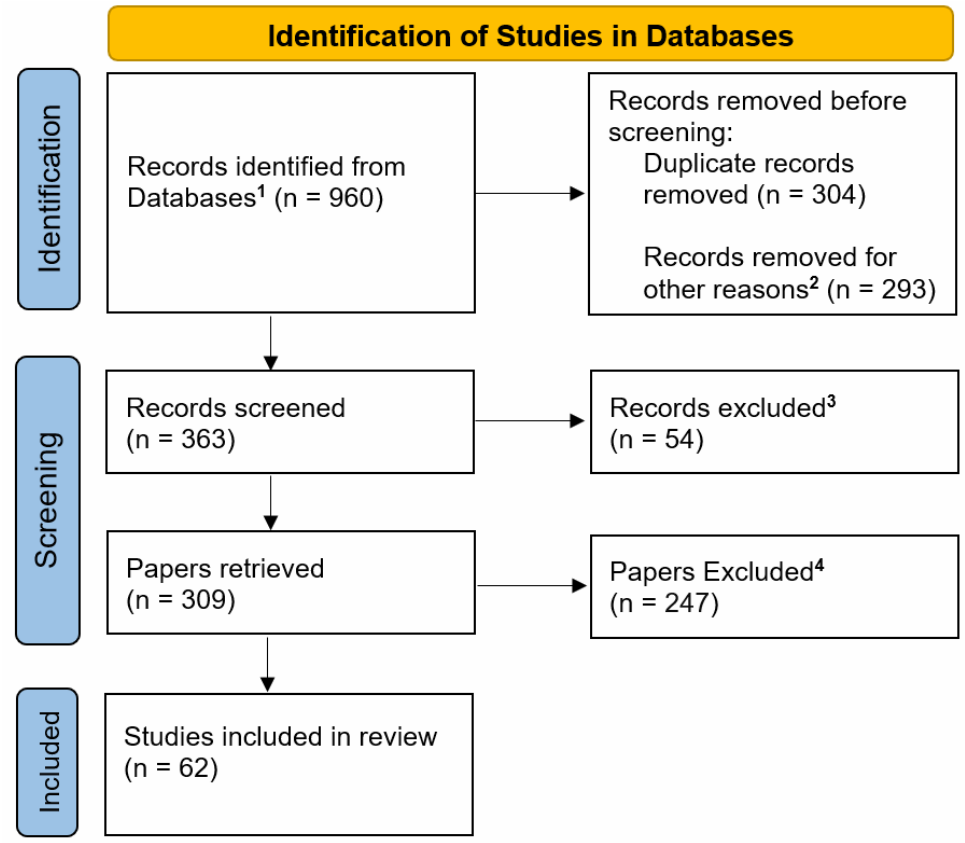


Figure 2.5: *1)* Records retrieved in each database: IEEE-Xplorer = 117, Scopus = 594, Web of Science = 249. *2)* This review considers papers from 2017 onward, so all publications before 2017 are removed. *3)* By reading the meta-data, we found 51 publications that were either focusing on hardware implementation or didn't consider robotics and SNNs. *4)* While screening, we labelled all the publications according to their focus, then we excluded all the papers not related to robotics.

robotics". In group 2, we also included "cognit*" to find cognitive models that may be used to learn and implement interactive behaviour in agents and robots. Finally, we used the pair "brain-inspired computing" AND "robot*", because we wanted to find alternative methods by using the "brain-inspired computing" keywords, but when combined with the keywords of group 2, it resulted in only 2 articles. The total number of articles we added was 960. Initially, we removed all duplicates, reducing the total count to 656. After that, we decided to limit our systematic review to articles published between 2017 and 2024. This reduced the total publication count to 363. After the initial screening, the number of publications that remained for review was reduced to 62. Figure 2.5 summarises the paper selection procedure using a PRISMA chart.

2.6.1 Neuromorphic chips/ Simulators and Frameworks

In socially interactive neurorobotics, neuromorphic hardware is one of the essential elements for the robot to perform cognitive tasks. With recent advancements in neuroscience and in the chip industry, new neuromorphic hardware is introduced for the simulation of SNNs. In the tables below, we will briefly discuss the neuromorphic chips (Table 2.1), simulators (Table 2.3), and frameworks (Table 2.2) we came across during our review. We also discuss the robots (Table 2.4) used in our selected publications. There are several other neuromorphic chips (e.g. TrueNorth [62], Braindrop [278], SyNAPSE [341], FACETS [323], NeuroMem [1] , NM500 [200] , SynSense [404]), simulators (e.g: Genesis [54], SpikeFun [340], DVS pixel simulator [192]) and humanoid robots (e.g. Pepper [292], Nao [22]) which are not discussed here as they were not used in any of the publications we reviewed. Some of the general-purpose simulators such as *Mayavi* [377] and *CSIM* [384] focus on simulating the environment for models, instead of simulating SNN models. Unlike other simulators, they do not have in-built neuron and/or synaptic models. Besides this, we also found several articles where no neuromorphic hardware is used. Input signals are obtained by sensors on the robot and SNN models are used as processing layers.

2.6.2 SNNs in Robotics Applications

The use of SNNs in robotics introduces considerable complexity when performing specific tasks [93]. In cognitive robotics, the goal is to understand the environment and compute the output. Such an approach usually returns useful insights for neural architectures and learned behaviour, especially when dedicated neural hardware is available. So far, we have briefly discussed spiking neural networks and their models. In this section, we dive deeper into the applications of SNNs and neuromorphic computing in the field of socially interactive robotics. Robots provide an interesting testbed for SNNs, yet their application requires finding solutions to many problems, such as power consumption, action duration, and output fidelity. Through our systematic review, we found 5 major directions in which contributions have been made. Before getting into the details of these applications, we summarise our analysis

Table 2.1: Neuromorphic chips used in robotic applications.

#	Neuromorphic Chips	Description	Ref.
1	**SpiNNaker**	Based on a biologically-inspired, massively parallel computing architecture, SpiNNaker can model and simulate neural networks containing up to one billion neurons and one trillion synapses in biological real-time. The System-on-Chip SpiNNaker consists of 18 ARM968 processors residing in synchronous islands, surrounded by a lightweight packet-switched asynchronous communications network. It serves as a general-purpose programmable platform enabling neuroscientists, psychologists, and brain researchers to explore brain functions using software neuronal models.	[291]
2	**Loihi**	Loihi is a 60-mm^2 chip capable of modeling spiking neural networks in silicon using Intel's 14-nm process. It features a large multicore mesh with 128 neuromorphic cores, three embedded x86 processor cores, and off-chip interfaces extending the mesh in four planar directions to other chips. Notable features include hierarchical connectivity, dendritic compartments, synaptic delays, and programmable synaptic learning rules.	[84]

Table 2.2: Description of frameworks reviewed.

#	Framework	Description	Ref.
1	**Neural Engineering Framework (NEF)**	NEF is a general methodology for building large-scale neural models for cognition. It allows users to define neuron properties and connection weights between components to perform desired functions. It efficiently handles feed-forward computations and recurrent connections. NEF has been used in modeling visual attention, inductive reasoning, reinforcement learning, and more.	[333]
2	**FenNueRobot Framework**	FaNueRobot is built on top of the NeurCube framework for modeling spatiotemporal brain data. It is a motor control framework designed for robotic and prosthetic applications, integrating an evolving SNN model of the brain with finite automata representing neuromuscular behaviour during forearm and extensor muscle movements.	[211]

Figure 2.6: General block diagram of iCub-SpiNNaker system. The I/O from the robot is converted into YARP bottles. It is then processed by a host-based EIEIO transceiver which converts the message into spikes to transmit and receive from SpiNNaker.

in Table 2.5 and Table 2.6. Here, Table 2.5 shows the robots and hardware used in the experiments, while Table 2.6 is about the software or simulation platform used to conduct experiments. These tables guide the reader through the types of robots and platforms that are typically used in experiments related to neurorobotics.

Signal Acquisition and Processing Applications

Modern robots benefit greatly from intelligent sensor integration, enabling autonomous planning and operation while adapting in real time. Image segmentation and identification, where an object's features are extracted and compared against a reference library [95], remain central to robot vision. Vision-based identification has advanced significantly, but combining vision and non-vision sensors can speed up recognition [348].

D'Angelo et al. [107] used an iCub humanoid robot and a SpiNNaker system [135, 291] to demonstrate object identification tasks. iCub, a 53-degree-of-freedom research robot, uses YARP [260] to communicate with the host PC. By enhancing networks to target behaviourally relevant goals, D'Angelo et al. showed that behaviourally relevant STDP supports positive learning. SpiNNaker relies on the EIEIO protocol [284] for real-time neuromorphic communication. In an extended publication, García et al. [138] demonstrated how neuroanatomically grounded SNNs for visual attention can also learn object names. Figure 2.6 show the block diagram of iCub-SpiNNaker system.

Research on multi-sensory integration explains how various sensor inputs converge to form a cohesive perception. Damasio [82] proposed the convergence-zone model to describe multi-modal perception in humans. Inspired by this, Al-Qaderi and Rad developed a multi-model perceptual system using concepts like fading

Table 2.3: Brief description of simulators and platforms used in robotic applications.

#	Simulator	Description	Ref.
1	Brian	Brian is a Python package used to simulate networks of spiking neurons. Users provide mathematical equations that define neuron and synaptic models, allowing the creation of neuron groups connected by synapses. Researchers can use custom models not already integrated into the simulator.	[149]
2	Nengo	Nengo is a Python library for building sophisticated spiking and non-spiking neural simulations with minimal code. It is highly extensible and flexible, allowing users to define neuron types, learning rules, and input sources.	[333]
3	NEST	NEST is an open-source simulator for spiking neural network models, focusing on neural system dynamics rather than neuron morphology. It operates as a command-line tool and can be used interactively from the Python prompt. It also supports neuron model creation, synapse definition, and connectivity with external devices.	[145]
4	CSIM	CSIM is a general-purpose discrete-event simulation environment for modeling complex systems with interdependent elements. It includes tools for creating hierarchical block diagrams and model libraries across multiple domains.	[384]
5	AI-SIMCOG	The Artificial Intelligence Simulator of Cognition (AI-SIMCOG) is an SNN simulator designed for real-time autonomous system testing. It allows users to design complex SNN architectures and test them in a virtual environment before exporting to physical robots.	[78]
6	NeMo	NeMo is a spiking neural network simulator optimized for CUDA-enabled GPUs, achieving real-time simulations of 100,000 realistically connected neurons. It supports C++, C, Matlab, Python, and is a backend for PyNN.	[169]
7	Mayavi	Mayavi is a cross-platform simulator for 2D and 3D data visualization. It features a rich user interface for interactive visualizations and a simple Python scripting interface, including ready-to-use 3D visualization tools.	[377]
8	Neuron	NEURON is a general-purpose spiking neural network simulator used for modeling individual neurons and networks. It provides an easy-to-use command-line interface and binary installers for Mac, Linux, and Windows.	[170]
9	Gazebo	Gazebo is a robot simulator capable of simulating robot populations in complex indoor and outdoor environments. It features a strong physics engine, high-quality graphics, and user-friendly interfaces for programming and visualization.	[276]

Table 2.4: Robots used in major applications.

	Robot	Description	Ref.
	iCub	A research-grade humanoid robot designed for developing and testing embodied AI algorithms. Standing 1 meter tall, it serves as a testbed for social experiments.	[372]
	Pioneer 3DX	A general-purpose differential-drive mobile robot with multiple sensors, an onboard computer, and an autonomous navigation system. Used for mapping, monitoring, and teleoperations.	[185]
	HoLLiE	A mobile, bi-manual service robot developed in the "House of Living Lab" project. Can perform basic and complex social interaction tasks.	[168]
	Hexapod	A six-legged walking robot known for stability and versatility. Used in neurorobotics for movement and obstacle avoidance experiments.	[251]
	UR3	A compact, collaborative tabletop robot for automated assembly tasks. Its small size makes it ideal for tight spaces.	[243]
	Turtlebot-2	A low-cost, open-source robot kit with customizable sensors. It supports 3D navigation and diverse applications.	[312]
	Khepera-IV	A 5.5cm small, differential-wheeled mobile robot designed for indoor lab applications. Used for mapping, teleoperation, and navigation.	[337]
	Soundman	A commercial product designed for binaural sound recording, enhancing spatial perception.	[86]
	Schunk SVH	A five-finger robotic hand with an advanced gripping system. Provides secure grip on objects of various shapes and sizes.	[351]
	Pushbot	A compact tracked robot with a front blade and onboard camera for remote navigation.	[205]

Table 2.5: Robots and hardware used in major applications related to interactive robotics.

Major Applications	Robot	Hardware
Signal Acquisition and Processing	iCub [308][138] Pioneer 3DX [16]	SpiNNaker [308][138][41] RGB Camera, Kinect Sensor [17]
Pattern Recognition	Pioneer 3DX [16] Khepera IV [45]	RGB Camera [16] Kinect Sensor [355] Raspberry Pi3 [80]
Speech Recognition	iCub [86][277] Soundman [86]	Microphones [86] iCub built-in sensors [277]
Motor Control	HoLLiE [364] Hexapod [34][220] Roving [242][413] UR3 [409] Vector [38] iRobot [283] Khepera IV [81] iCub [277]	Schunk SVH [365][366][367] sMEG Sensor, Myo Sensor [366][367] HoLLiE Arm [364] FPGA ZEM4310, Emotive Epoc Headset [34] DVS, NAS [220] UR3 with Elbow and Wrist [409] Prosthetic Hand [211] Tobii Eye Tracker, YDLIDAR G4, Raspberry Pi 3 [80]
Cognition and Learning	iCub [115] Neco [357] Turtlebot [160][103][356][160][175] Virtual [328][79][77]	Robotic Hand, Myo Armband, Epoc Headset [414][415] Loihi, Neural Computer Stick2 [160] Spartan-6, RGB Sensor [103] Raspberry Pi3, PiCamera, PiStrom [77] Loihi, LIDAR [356]

Table 2.6: Software/simulators and neuron models for major applications.

Major Applications	Software/Simulation Platform	Neuron Model
Signal Acquisition and Processing	SpiNNaker [308][138] CISM [17]	LIF [308][138][17]
Pattern Recognition	Brian [250] SIMCOG [80][81]	LIF [16][80][81]
Speech Recognition	Google ASR, SphinX [86] EDULT [277]	LIF [277] Pulse Neuron [86]
Motor Control	NEF, Nengo [365][366][367] NEST [364] MATLAB [34][211] NeuroNet [242][413] NeMo [409] Vector SDK [38] FeNeuRobot Framework [211] ROI [283] SIMCOG [80][81]	Manual [365][366][367] LIF [364][220][211][80][277] Izhikevich [409][413] aEIF [38] Pulse Neuron [283]
Cognition and Learning	SIMCOG [79][77] Nengo [160] RatSLAM [357] SNN Simulator [328] NEF [126] V-REP [175]	Created [160] LIF [79][77][115][126] Izhikevich [414][415] SKAN [103] SRM [357]

memory [247], binding criteria [371], cell assemblies [166], and top-down influences [227]. Experiments on a Pioneer 3DX robot equipped with an RGB camera, Kinect sensor, directional microphone, and sonar sensor showed that multi-model systems outperform uni-modal ones.

Pattern Recognition Applications

Visual or pattern recognition underpins many real-world robotic operations, especially those requiring human-robot interaction; failure in recognition often hinders a robot's practical utility. In cognitive robotics, visual recognition is central to complex tasks like pose estimation, grasping, and manipulation [75]. However, existing methods frequently rely on strict supervision during training, such as uncluttered views of objects or rigid metadata. To address these challenges, Fanello et al. [117] improved robot visual perception by modifying coding-pooling pipelines to enhance representation while preserving real-time performance. More recently, Mansouri-Benssassi and Ye [250] employed bio-inspired SNNs with STDP for unsupervised facial expression recognition, using Laplacian of Gaussian (LoG) filters to detect edges and contours, a Poisson-driven spike train based on pixel intensity, and online STDP [53] (see Appendix A.1, left). This approach outperformed methods like HOG features and CNN on two public datasets. Meanwhile, Al-Qaderi and Rad [16] introduced a multi-modal perceptual system for facial recognition in real-world scenarios where a robot identifies a user while in motion.

Additional experiments investigated real-time recognition of human motion through SNNs, clustering direction vectors to interpret nonverbal cues for action recognition [355]. This instructed learning occurs in three phases teaching, turn-taking, and imitation (see Appendix A.1, right). Further developments include Cyr and Theriault [80], where robots learn spatial relationships (left/right, horizontal/vertical) independently of image location or pattern details, adapting in real-time with different reward rules. In a similar vein, Cyr et al. [81] proposed a 2-bit XOR task using compound binary images for input and left/right output actions. This

design lets the robot adapt by learning simpler associative rules at runtime, with further experiments exploring the shift from 2-bit to 3-bit tasks.

2.6.3 Speech Recognition Applications

Speech recognition allows machines or robots to convert spoken language into a machine-readable format, typically by capturing voice input through a microphone, processing it on a computer, and then sending commands back to the robot. In noisy settings, Davila-Chacon et al. [86] proposed an embodied embedded cognition approach that uses Sound Source Localization (SSL) to orient the robot toward the angle that maximizes the signal-to-noise ratio (SNR) before performing Automatic Speech Recognition (ASR). A spiking neural network calculates the sound angle, and experiments on both the iCub robot and Soundman demonstrated that this SSL-driven orientation significantly enhances speech recognition accuracy compared to averaging signals from multiple channels. In future work, speech recognition may also serve as a bootstrapping mechanism to train neural layers, performing auditory grouping in both frequency and time domains [86]. Another study integrated a spiking cerebellar model into the iCub robot to handle vestibulo-ocular reflex (VOR) tasks [277]. This model functions as a feed-forward controller, combining several neural models and topologies, and uses STDP-based adaptation to generate eye motor commands that compensate for the robot's head movements. Figure 2.7 illustrates the system architecture.

2.6.4 Motor Control Applications

Research on motor control applications in neurorobotics focuses on enabling robots to mimic human cognitive and motor behaviours. One approach, proposed by Tieck et al. [364], uses a spiking neural network (SNN) to learn base motor primitives (left, right, up, down) for pointing motions. The network activates four correction primitives to generate real-time motion commands, driving a humanoid robot (HoLLiE) toward specific points on a board. The brief block diagram of the motion generation approach, along with an explanation, can be seen in the Appendix A.2. A key advantage is that

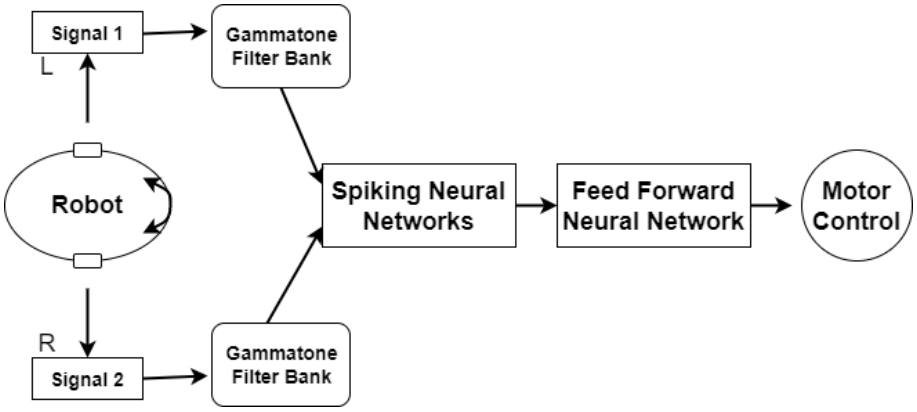


Figure 2.7: The Spike Source Localization (SSL) architecture. Sound signals are received at the left and right receiver of a robot. These sound inputs are decomposed into several frequencies with Gammatone Filterbanks. After that, it is fed to Spiking Neural Networks and then to Feed-Forward neural network for classification. This classification layer produces an output angle that is used to control the motor.

this method does not rely on any particular kinematic setup and can be extended to different robotic platforms. A related idea is introduced by Mirus et al. [262], where a mobile robot leverages similar SNN-based principles to navigate and detect objects in unknown environments.

In more advanced locomotion control, Batres-Mendoza et al. [34] proposed a real-time system for a Hexapod robot that couples bio-inspired computing with an improved Quaternion-based Signal Analysis (iQSA) [35] method, shown in Appendix A.3. Brain-Computer Interface (BCI) signals are processed by iQSA and then fed into a SNN acting as a Central Pattern Generator (CPG) [302], generating rhythmic gait patterns. Likewise, Lele et al. [220] presented another CPG-based SNN architecture, augmented by input from a Dynamic Vision Sensor (DVS), to guide a Hexapod in prey-tracking tasks with high energy efficiency. In parallel, Zahra et al. [409] designed a cerebellum-inspired controller for robotic arms, using a Differential Mapping Spiking Neural Network (DMSNN) and STDP to align spatial velocities across layers. This controller was validated on a UR3 robot arm, achieving rapid convergence with minimal motion error. For manipulative tasks, Tieck et al. [365] employed SNN-driven primitives to enable soft-grasping of various objects without complex calculations, training each grasping motion from just one example. Other

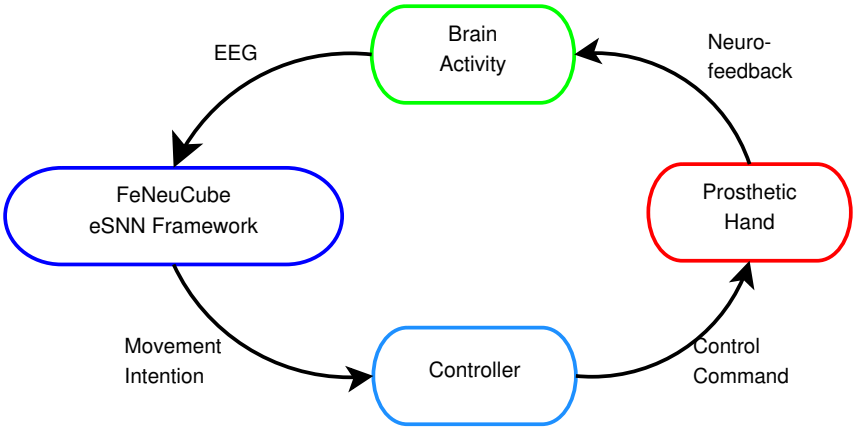


Figure 2.8: The basic block diagram of the prosthetic control through a Brain-Computer Interface (BCI). As shown, the brain provides the EEG signals to the FeNeuCube framework which, in turn, gives instructions to the controller. Finally, the controller forwards the control command to prosthetic hand [211].

work has integrated surface electromyography (sEMG) signals to trigger reflexes in robotic fingers, linking recognized muscle activations to motion primitives [366][367].

In the medical domain, Kumarasinghe et al. [211] demonstrated a proof-of-concept prosthetic hand controlled by a NeuCube evolving SNN architecture combined with finite automata theory. As depicted in Figure 2.8, the system undergoes two learning stages unsupervised STDP for evolving synaptic connections, followed by supervised learning to strengthen output-layer weights. This approach enables reliable grasping and touching movements based on recorded brain activity. Lastly, to assist the elderly or disabled, Obo et al. [283] proposed a multi-modal interface integrating seat pressure sensors with an eye-tracker. A self-organized SNN captures spatio-temporal dynamics from user inputs. By detecting subtle shifts in seating posture and gaze direction, the robot can adjust its movement and the interface's sensitivity, ensuring more comfortable teleoperation for users with limited mobility. Appendix A.4 summarizes their perception-action cycle framework.

Another interesting study focuses on solving movement issues for disabled and elderly people. In most of the existing systems, a user has to control the robot manually, which could be difficult for elderly or disabled people. Obo et al. [283] presented a multi-modal interface to control the robot remotely. Besides this, a cognitive platform to control robots based on the concept of the perception-action

Table 2.7: Learning mode, rule, and paradigm in each application.

Major Applications	Online/Offline Learning	Learning Rule	Learning Paradigm
Signal Acquisition and Processing	Online [308][138][17]	STDP [308][138]	Reinforcement [308][138][17]
Pattern Recognition	Online [80][81] Offline [250][355]	STDP [250][355]	Unsupervised [250][355] Reinforcement [250][355]
Speech Recognition	Online [277]	STDP [277]	Supervised, Feedback error learning [277]
Motor Control	Online [365][38][80] [81][277] Offline [364][220][211] [34][283][211]	STDP [409][38][413]	Supervised [364][220][211][409] Unsupervised [34][283][211][38] Feedback-error learning [277]
Cognition and Learning	Online [356][328][175][126] Offline [103]	STDP [414][415][79] [77][357]	Reinforcement [79][77][126][175] Supervised /Unsupervised [160] Feedback-error learning [277] Reinforcement [103][328] Synaptic learning [357]

cycle is also proposed. Here, a spiking neural network is used for spatio-temporal modelling of the interaction between the environment and the user. A self-organized neural network based on an unsupervised learning paradigm is used in this system. They developed a seat with pressure sensors on it and the robot's movement is dependent on the user's movement while sitting on the seat. Beside this, a stationary eye-tracker (Tobii Eye Tracker 4C) is attached to retrieve the gaze and head pose. The experimental results show that the teleoperation system can change the sensitivity of the interface according to the operation. More details about the proposed system can be found in Appendix A.5.

2.6.5 Cognition and Learning Applications

Biological systems generally have a memory, which is defined as the ability to preserve, learn and reproduce past adaptive states. As discussed earlier, synaptic plasticity is an important mechanism of memory and learning on a cellular level. Several mathematical models exist that can simulate cognitive maps, where synaptic plasticity yields the emergence of spatial memory in SNNs. Spatial memory in robots is used for the storage and retrieval of information that is used to plan a route to a desired location and to remember where an object is located or where an event occurred [242]. In this subsection, instead of discussing the classifications of learning [48] we discuss the contributions related to spatial memory, learning and using that learning to predict the next action for social robots. Table 2.7 details the learning mode, rule and paradigm in each application. More details about the learning in SNN can be found in Section 3.2.

Spatial mapping is an important component for developing Simultaneous Localization and Mapping (SLAM) in social robots. For instance, Tang and Michmizos [356] integrated a Loihi-based SNN with a 360-degree LiDAR sensor in Robot Operating System (ROS) for real-time navigation using Winner-Take-All (WTA) and competitive learning. Likewise, Cyr and Theriault [80] employed operant conditioning to enable a robot to modify its behaviour dynamically based on reward rules, while Dumesnil et al. [103] implemented classical conditioning on an FPGA using the Synapto-Dendritic Kernel Adapting Neuron (SKAN) model. Other studies highlight avoidance learning by assigning harmful regions for the robot to identify and circumvent [413, 242], and associative memory has been tested on neuromorphic hardware for energy efficiency [160]. Researchers have also explored working memory through dynamic synapses [38], episodic memory for complex tasks like serving milk tea [357], and evolutionary methods to fine-tune SNN topologies [328, 102]. Additional work examines emotional modelling to provide intrinsic motivation [175] and socio-emotional architectures on SpiNNaker, Loihi, and Braindrop [126]. Finally, to streamline the development of realistic brain–body interfaces, the Neurorobotics Platform [363, 115] offers a web-based environment where brain models connect with

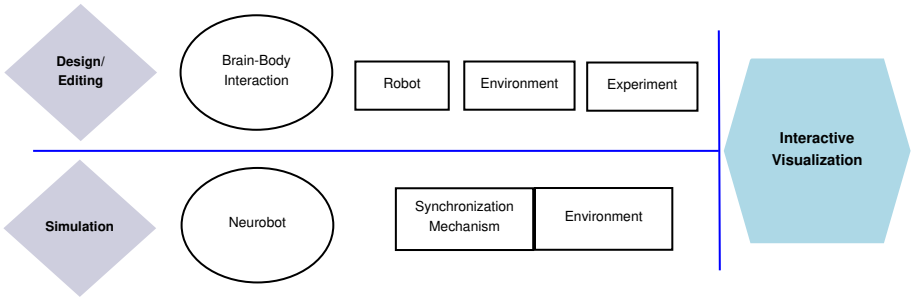


Figure 2.9: Block diagram for a functional overview of the Neurorobotics platform. With the design/editing feature, users can conduct neurorobotic experiments using the brain model and robot body which can interact in a dynamic environment. These experiments are simulated and the results can be displayed in an interactive fashion.

detailed robot simulations. This project is part of the EU flagship Human-Brain Project (HBP). The functional overview of the Neurorobotics platform is presented in Figure 2.9.

Figure 2.10 is an expanded form of Figure 2.2. It shows the sensors and chips in the hardware section. Here the red outline boxes show that the chip is not used in robotics applications yet. The software section contains a list of simulators, frameworks and platforms. It also shows the neuron models typically used in robotics applications.

2.6.6 Challenges & Future Direction

In the previous sections, contributions and advancements in SNN-based social robotics have been reviewed in terms of their focused applications. This section focuses on the critical analysis of reviewed articles. We discuss the shortcomings in this area and what are the major areas on which future research should focus.

Humanoid robots: Although many experiments are conducted in the area of neuromorphic computing for robotics, most of them are very basic when it comes to cognitive tasks [357][38][80]. Usually, they focus on robots moving around a dedicated environment, constructing the spatial map, or storing some information in its memory block. Few pieces of research considered human involvement in the experiments for detection and identification but they lack human-robot interaction in social context. Besides this, none of the experiments were done with a humanoid

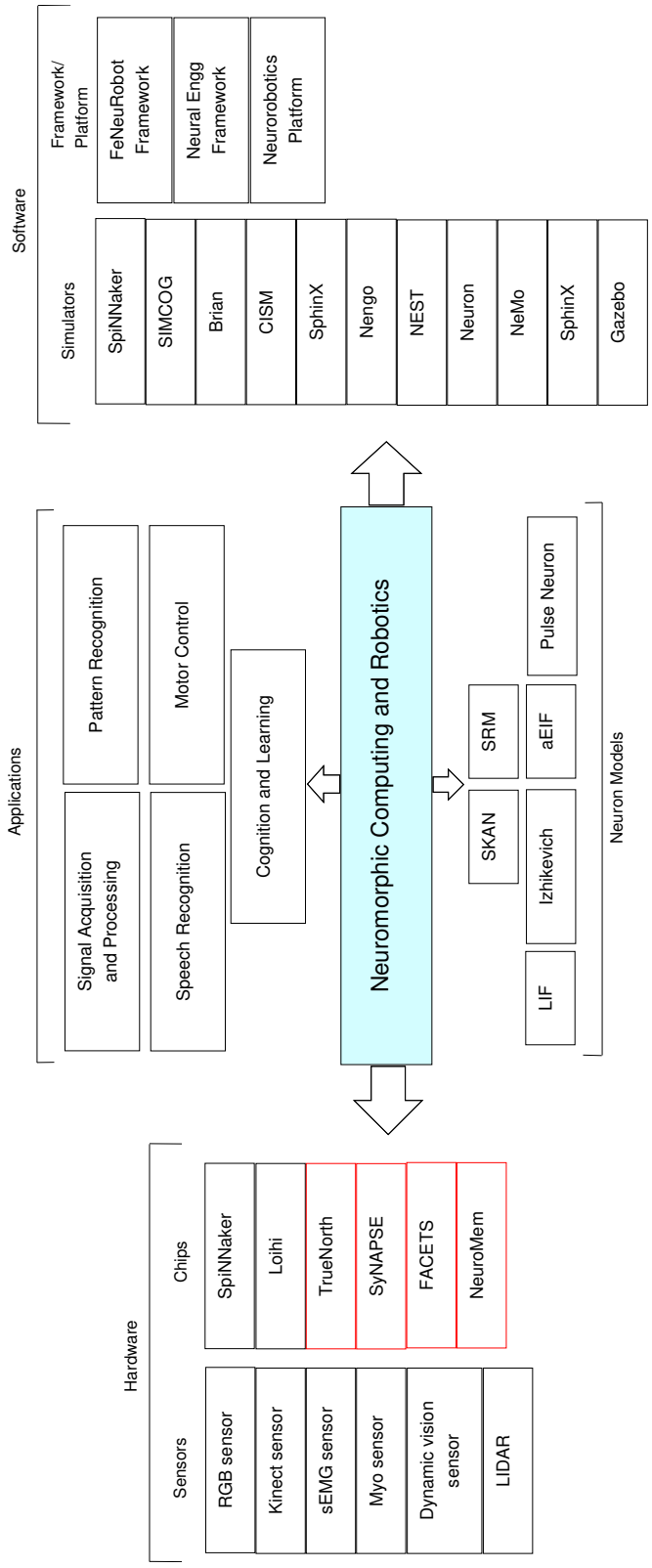


Figure 2.10: An overview figure of the landscape of neuromorphic computing and robotics. It shows major hardware, software, neuron models, and applications. The red boxes in the chips column show that they are not yet used in robotic applications.

robot. So, there is still a need to conduct experiments with a humanoid robot where the robot also interacts with humans instead of just interacting with the environment.

Neuromorphic Chips: As we can see in Table 2.1, only two types of neuromorphic chips are used in our reviewed papers. Experiments with Intel's Loihi chip are mostly related to improving the memory or spatial mapping while experiments with the SpiNNaker architecture are focused on behavioural learning and visual attention. In the experiments with SpiNNaker, the neuromorphic chip is not attached to the robot. The SpiNNaker system receives inputs from the sensors in the robot via the EIEIO protocol for processing and returns action commands. On the other hand, experiments with the Loihi chip don't contain humanoid robots. In most of the reviewed articles a turtlebot is used, which is equipped with multiple sensors and a Loihi chip. This shows that there are still several open options like using other available neuromorphic chips and conducting experiments in more complex scenarios where the humanoid robot has to make decisions in real-time.

Hardware: Many experiments conducted in our reviewed articles use non-spiking sensors (e.g. web camera is used instead of a retina camera [75]). These sensors send the signal to the processing unit, which converts it to spikes and feeds them to the neuromorphic hardware. This process could become faster and more energy-efficient on neuromorphic hardware due to its event-driven computation, which avoids redundant frame processing and enables low-latency parallel inference at substantially lower power consumption.

Personalisation: Another challenge faced by the social neurorobotics field is about personalising the robot. This can only be done through interdisciplinary research of roboticists and neuro-scientists. Usually, roboticists use simplified brain models to make real-time simulations, while neuro-scientists work on detailed brain models which are difficult to embed into the real world due to their high complexity. The community needs more solutions like the Neurorobotics Platform which provides adequate tools to model highly detailed environments, virtual robots, and complex neural networks for both roboticists and neuro-scientists.

Generalised Framework: Recent developments in neuromorphic computing have introduced comprehensive benchmark frameworks designed to standardize the

evaluation of algorithms and systems. Notably, the NeuroBench framework [187] offers a structured approach for assessing neuromorphic approaches, providing a common set of tools and methodologies for inclusive benchmark measurement. This initiative aims to deliver an objective reference framework for quantifying neuromorphic approaches in both hardware-independent (algorithm track) and hardware-dependent (system track) settings. Despite these advancements, none of the reviewed articles offer a general-purpose framework that encompasses functionalities for training and modelling. Training SNNs for deep networks remains a challenging task, necessitating further development in conversion algorithms. Moreover, there is a pressing need for improved mechanisms to evaluate computational capabilities, such as power consumption and speed, which are vital for applications in social robotics.

Ethics: Lastly, some aspects are rarely considered till now, such as ethical aspects in social neurorobotics. The development of social neurorobotics is still in its early stages, which makes it an ideal candidate for proactive and anticipatory ethical reflection. The major concern is *trust and safety* when it needs to be decided whether to use the robot in a social environment or not. Another aspect that might affect the adaptability of social neurorobotics is *data privacy*. Where and how data from the sensors of robots is being processed and how to share this data with another robot in a socially interactive environment. Therefore, research is needed to ensure that the development of neurorobotics is ethical, desirable, and socially acceptable.

2.7 Neuromorphic Vision for Interactive Robotics

The keywords we used were selected to identify scientific articles at the intersection of neuromorphic vision and interactive robotics. We identified two groups of keywords with similar meanings: "Event based," "Event Vision," and "Neuromorphic Vision," as well as "interactive robotics," "social robotics," and "humanoid robotics." We searched for these keywords in the Title, Abstract, and Keywords fields across three databases and downloaded the resulting publications for an initial screening. The total number of articles obtained was 1,604. After selecting articles published between 2017 and 2024, 1,070 remained. We then removed duplicates, reducing the total count to 776. Next, we conducted an initial screening to select articles related specifically to neuromorphic vision and interactive robotics, bringing the number down to 134. Finally, after reviewing the full papers, the count was reduced to 80. Figure 2.11 summarizes the paper selection process using a PRISMA chart.

We also analysed the companies currently manufacturing event-based vision sensors used in interactive robotics and neuromorphic perception research (Table 2.8). The market is now primarily led by the Prophesee–Sony partnership, which provides industrial-grade event cameras and development platforms widely adopted in robotics, automotive perception, and real-time embedded vision systems [305]. In parallel, CelePixel has emerged as a key manufacturer offering high-resolution event sensors and hybrid frame–event cameras for robotics and machine vision research [152]. Additionally, SynSense develops neuromorphic vision hardware integrating event-based sensing with low-power spiking processing for edge robotics and autonomous systems [179]. Together, these manufacturers represent the core commercial ecosystem driving the deployment and standardisation of event-based vision technology in contemporary robotic applications.

Through our systematic literature review we divided the studies according to the application they focus on. We ended up with six applications *Object Detection & Tracking, Gesture Recognition, Visual SLAM & Odometry, Motor Control, Autonomous Driving, Surveillance & Security.* Following is a critical analysis of studies relevant to each application.

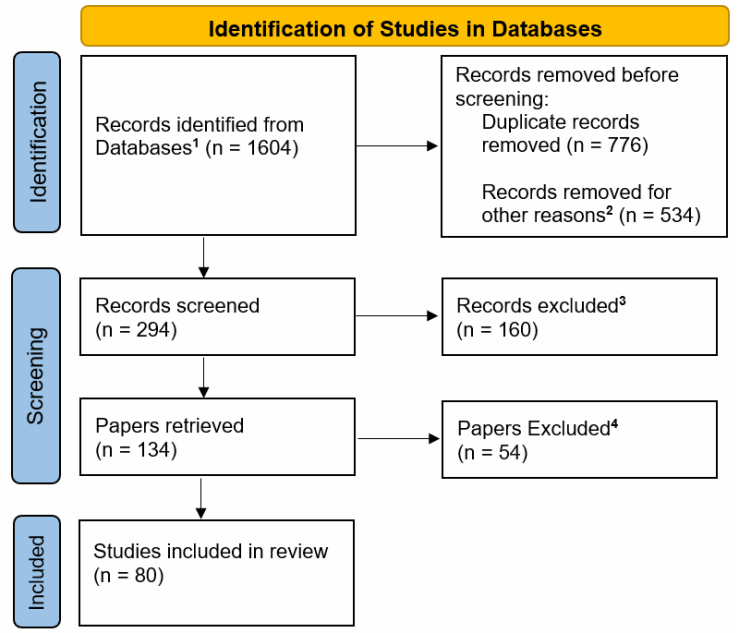


Figure 2.11: 1) Records retrieved in all databases: 1604 2) This review considers papers from 2017 onward, so all publications before 2017 are removed. 3) By reading the meta-data, we found 160 publications that were either not using event camera/vision or didn't consider robotics. 4) While screening, we labelled all the publications according to their focus, then we excluded all the papers not related to robotics.

2.7.1 Object Detection & Tracking

Neuromorphic vision has revolutionized object detection and tracking in robotics by leveraging event-based cameras (e.g., DAVIS346, DVS128) and Spiking Neural Networks (SNNs) to enable high-speed, low-latency performance with minimal power consumption [64]. Hybrid sensor fusion approaches, such as Kalman filter-based tracking for robotic air-hockey systems [397] and active LED marker integration for real-time 6-DoF pose estimation [108], have enhanced accuracy in dynamic environments, while FPGA-accelerated industrial tracking systems reduced data transmission by 85% and latency by 99ms [33]. Innovations in toolkits include a ROS-based event camera emulator for synthetic testing [59], a standardized neuromorphic perception toolbox [104], and YARP framework integration for real-time tracking in cluttered humanoid environments [147]. Advanced algorithms like event-based mean-shift clustering achieved an F-score of 0.95 with 88% computation reduction for real-world objects in an action manipulation task [32], luminance-free line detection improved efficiency in structured settings [374], and continual learning models on

Table 2.8: Current Manufacturers of Event-Based Vision Sensors (2025)

Company	Applications	Notable Products	Data Format
Prophesee–Sony Partnership	Robotics, automotive ADAS, industrial automation	GenX320, GenX series sensors, Metavision SDK	EVT (proprietary)
CelePixel (China)	Robotics, machine vision, scientific research	CeleX5, CeleX-V	Proprietary event format
SynSense (Switzerland/China)	Edge AI, robotics, neuromorphic perception	Speck, Dynap-CNN vision sensors	Event-based spike streams

Loihi chips reached 96.55% accuracy with 300× energy efficiency over CNNs [158]. Datasets such as PEDRo, with 43,259 bounding boxes for person detection [52], and UAV surveillance datasets [97] benchmarked performance in real-world scenarios, while synthetic datasets enabled algorithm validation without physical hardware [59]. Challenges persist in scalability, as seen in SNN-based obstacle memory systems limited to controlled environments [342], dependency on continuous motion for multi-robot tracking [294], and hardware constraints like high-resolution camera processing demands [108]. Future directions emphasize real-world adaptation through expanded UAV datasets [97], energy-efficient edge deployment via microsaccade-enhanced cameras for static scenes [164], and multi-sensor fusion combining LiDAR or radar [97], alongside innovations like visual stabilization for feature tracking [314] and LSTM-based trajectory prediction for human-robot interaction [23]. Together, these advancements underscore neuromorphic vision as a transformative solution, balancing speed, efficiency, and adaptability in robotic perception.

2.7.2 Gesture Recognition

Gesture recognition in interactive robotics has significantly benefited from neuromorphic computing, particularly through event-based vision sensors and Spiking Neural Networks (SNNs), enabling real-time processing of dynamic gestures with low latency and minimal computational overhead [215][85][71]. Event-based cameras such as Dynamic Vision Sensors (DVS) provide asynchronous visual data, reducing redundancy and allowing rapid motion detection, which is critical for high-speed robotic interactions. Keith et al.[215] introduced an event-based control framework utilizing

a DVS sensor for UAV tracking, integrating neuromorphic processing with real-time visual control, thereby enhancing tracking efficiency in dynamic aerial applications [215]. Aitsam et al [85] explored event-driven hand gesture recognition using SNNs, demonstrating that accumulating event data into frames instead of extracting temporal features can achieve high recognition accuracy, while also proposing an alternative dataset, DVS-Gesture-Chain (DVS-GC), to improve the understanding of temporal dynamics in gesture recognition. Expanding on neuromorphic vision, Guang et al [71] presented a radar-based hand gesture recognition system leveraging SNNs and a novel radar signal-to-spike conversion, achieving superior accuracy compared to traditional approaches and highlighting the potential of non-visual event-based sensing for gesture interpretation. These studies collectively advance gesture recognition by integrating neuromorphic vision with robotic control systems, improving robustness to occlusion and environmental variability, and reducing computational demands for real-time recognition. However, challenges remain in dataset standardization, real-world validation, and optimising neuromorphic models for embedded systems.

2.7.3 Visual SLAM & Odometry

Event-based vision has transformed Visual SLAM & Odometry by overcoming limitations of traditional systems in high-speed, low-light, and dynamic environments through high temporal resolution and asynchronous sensing. Hybrid sensor fusion approaches, such as integrating event cameras with LiDAR for agricultural navigation [423] or combining stereo event data with inertial measurements for drones [73], enhance robustness, while Evangelos et al demonstrated UAV tracking with a CelexV sensor, achieving lower latency than frame-based SLAM despite sensitivity to low-light conditions [114]. Advanced algorithms like Event-VPR, enabling sparse pixel-based place recognition for low-power localization [204], and Fast-EDI, facilitating real-time motion deblurring [234], highlight computational efficiency, while bio-inspired methods such as SNN-driven underwater navigation [407] and physics-guided neuromorphic planners for drones [325] optimise energy use. Critical datasets include the Agri-EVB-autumn dataset for agricultural SLAM [423], ground-truth optical flow

benchmarks using VICON [39], and drone racing sequences validating high-speed navigation [25], complemented by synthetic tools like ROS-based emulators [59]. Challenges persist in dependency on external sensors (e.g., IMU in stabilization frameworks [314]), scalability of neuromorphic hardware for obstacle avoidance [261], and limited real-world validation of place recognition systems [204]. Future directions emphasize adaptive algorithms for unstructured environments, as seen in quadruped odometry using adaptive time surfaces [421], unsupervised learning for feature representation [31], and multi-sensor fusion, alongside innovations like spatiotemporal registration for high-precision motion estimation [237] and real-time stereo visual odometry [416], positioning event-based vision as a cornerstone for efficient, adaptable, and robust SLAM in autonomous robotics.

2.7.4 Visuo-Motor Control

Neuromorphic computing has revolutionized motor control in robotics by enabling low-latency, energy-efficient actuation through Spiking Neural Networks (SNNs) and event-based vision, with applications spanning quadruped locomotion [130], aerial drones [61], and robotic manipulation [153]. For legged robots, event-driven policies enabling quadrupedal object catching at 15 m/s [130], while another study utilized SNNs for hexapod gait imitation, achieving bio-inspired locomotion with minimal energy [368]. Cao et al. integrated SNNs for high-speed drone control in cluttered environments [61], Antonio et al implemented neuromorphic controllers on Loihi for UAV stabilization [381], and Rajkumar et al enhanced adaptive flight using online learning [273]. Robotic manipulation saw breakthroughs with vision transformer for real-time gripper force estimation [153], event-driven grasping strategies outperforming frame-based methods in low-light [172], and a slip detection system for soft robotic fingers [120]. Adaptive control frameworks, such as FPGA-accelerated ROS 2 programming [232] and SNN-based continual learning on Loihi [158], optimised resource efficiency, while hybrid approaches like the fusion of event vision and force sensors enabled hexapod terrain adaptation [245]. Challenges include scalability in multi-limbed robots [368], dependency on synthetic datasets [392], and hardware constraints in

embedded systems [94]. Future directions emphasize multi-sensor fusion for dynamic environments [111], unsupervised learning for adaptive grasping [31], and real-world validation of neuromorphic exoskeletons [296] and prosthetics [206], positioning event-based motor control as a cornerstone for next-generation autonomous robotics.

2.7.5 Autonomous Driving

Event-based vision has redefined autonomous driving by addressing motion blur and latency in high-speed scenarios through high temporal resolution and dynamic range sensing, enabling precise detection of fast-moving objects like pedestrians [70] and robust lane detection in low-light conditions via deep learning integration [379]. Innovations include event-based perception pipeline for high-speed object detection [72], LiDAR-frame-event fusion enhancing robustness in darkness [143], and neuromorphic optical flow algorithms for low-latency velocity estimation [417]. Datasets like DSEC, offering stereo event data for illumination challenges [143], and M3ED, providing multi-sensor, multi-environment benchmarks [65], support training and evaluation, while high-resolution dataset advances motion prediction and scene understanding [173]. Challenges persist in sensor synchronization [143], computational demands of multi-modal fusion, and real-world validation of SNN-based lane-keeping systems [173]. Future directions emphasize adaptive fusion architectures, expanded datasets for diverse driving conditions [65], and edge deployment of energy-efficient neuromorphic models [379], positioning event-based vision as a critical enabler of safer, real-time autonomous navigation.

2.7.6 Surveillance & Security

Surveillance and security systems require real-time, efficient, and reliable monitoring to detect and respond to potential threats promptly. Traditional frame-based vision systems face challenges such as high latency, motion blur, and significant power consumption, particularly in dynamic or low-light environments. Event-based neuromorphic vision sensors provide a compelling alternative by capturing asynchronous changes in the scene with high temporal resolution and dynamic range,

allowing for improved detection and tracking in complex environments [315][316][299]. Several studies have explored the application of event-based vision in surveillance and security. Rodriguez et al [315] introduced an event-driven object detection framework designed for continuous monitoring in high-traffic areas, demonstrating the ability to capture motion patterns more efficiently than traditional cameras. This approach was then extended by incorporating neuromorphic processing for anomaly detection, improving system response times while minimizing false alarms [316]. Perez et al [299] focused on real-time tracking of individuals in crowded environments using event-based cameras, highlighting the potential for non-intrusive, privacy-preserving surveillance. Mario et al [326] investigated the integration of event-based sensors with deep learning architectures for facial recognition under varying lighting conditions. This work addressed the limitations of conventional systems in nighttime surveillance, improving identity verification accuracy in low-light environments. Another study explored the use of neuromorphic vision for drone-based surveillance, demonstrating enhanced performance in tracking moving targets from aerial perspectives, even under rapid camera movement and occlusions [110]. Security applications have also benefited from multimodal sensing approaches.One study introduced a hybrid framework combining event-based vision with thermal imaging, improving detection robustness in challenging weather conditions and concealed object identification [109]. It was further examined event-driven motion analysis for perimeter security, showing how asynchronous event data could be leveraged to detect unauthorized access with lower computational overhead compared to frame-based methods [97]. Table 2.9 provides a comprehensive summary of the utilization of event cameras across various neuromorphic vision applications in robotics. It highlights the specific event cameras used, whether the studies compare event-based approaches with traditional frame-based vision, the presence of event and frame fusion techniques, and the datasets utilised for evaluation.

Table 2.9: Summary of Event Camera Utilization Across Neuromorphic Applications.

Application	Event Camera	Event vs Frame	Event and Frame Fusion	Datasets
Object Detection & Tracking	DVS128 [342, 146, 147], DAVIS346 [397, 52, 108], DVXplorer [125], Prophesee Gen3 [294], Custom CMOS [33], AMI-EV [164]	Yes [397, 359, 146, 108, 190]	Yes [397, 294]	Custom air-hockey [397], PEDRo [52], M3ED [65], OptiTrack [108], Synthetic [59], Industrial [33]
Gesture Recognition	DVS128 [215], DAVIS346 [71], CenturyArk SilkyCam VGA [85]	Yes [215, 85, 71]	No	Weizmann/KTH/UCF Sports [215], EB-HandGesture [85], ALM glove [71]
Visual SLAM & Odometry	DVS240 [423, 57, 309], DAVIS346 [73, 394, 416], CelexV [114], DVXplorer [421], ATIS [39]	Yes [114, 295, 73, 394, 309, 416]	Yes [295, 73, 421, 314]	Agri-EVB-autumn [423], TartanAir/Apollo [295], MVSEC/VECtor [73], Shapes, DSEC-MOD [417], M3ED [65]
Motor Control	DVS128 [221, 273, 49], CeleX5 DVS [368], DVXplorer [130, 392], DAVIS240 [172], DAVIS346 [61, 223], Prophesee IMX636 [108]	Yes [130, 172, 273, 381]	Yes [245, 172, 61, 273, 381]	Hexapod locomotion [221], Quadruped [130], NeuroGrasp [61], RG-Event [153], Event-Stream [223], EBOG [174]
Autonomous Driving	DVS240 [72], DAVIS [70], ATIS [379], Prophesee Gen3.1 [143], DVXplorer [417]	Yes [72, 70, 379, 143, 417]	Yes [70, 143, 417]	Slasher platform [72], N-CARS [379], DSEC [143], DSEC-MOD [417], M3ED [65]
Surveillance & Security	DAVIS346 [315, 316, 299, 97], Prophesee Gen3 [326], DVXplorer [110], Custom CMOS [109], DVS128 [206]	Yes [315, 316, 299, 326, 97]	Yes [326, 97]	UAV surveillance [315, 316], Mine [326], GRIFFIN ERC [110, 109], OptiTrack [108]

2.7.7 Challenges & Future Directions

The integration of neuromorphic vision into robotics faces several cross-domain challenges. Algorithmic complexity arises from the asynchronous nature of event-based data, requiring specialized frameworks for real-time processing in applications like SLAM [309][237], motor control [130][153], and surveillance [97]. Hardware limitations, including high computational demands for high-resolution event cameras [108][97] and scalability issues in neuromorphic processors for multi-limbed robots [368][296], hinder embedded deployment. Dataset scarcity restricts generalization, particularly in autonomous driving [173][143], gesture recognition [71], and industrial tracking [125], where synthetic or small-scale datasets dominate [59][52][39]. Sensor synchronization remains critical in hybrid systems, such as LiDAR-event fusion for subterranean navigation [326] or IMU-dependent visual stabilization [314], while motion dependency plagues applications like multi-robot tracking [294] and UAV surveillance [97]. Additionally, privacy concerns in event-based surveillance and energy inefficiency in continuous SNN operation [158][381] demand urgent attention.

To address these challenges, research must prioritize multi-sensor fusion architectures combining event cameras with LiDAR [143][326], radar [71], or thermal sensors for robustness in dynamic environments [109][111]. Dataset expansion is critical, requiring large-scale, multimodal benchmarks for autonomous driving [65][173], SLAM [423][39], and industrial inspection [125], alongside synthetic tools for edge cases [59][374]. Hardware-software co-design should optimise energy-efficient neuromorphic chips (e.g., Loihi [158][381]) for real-time motor control [130][172] and aerial navigation [25][381], while adaptive algorithms like unsupervised SNNs [31] and physics-guided planners [325] could enhance generalization in unstructured terrains. Privacy-preserving frameworks for event data anonymization [97] and on-chip learning for continuous adaptation [379][111] will bolster security and autonomy. Finally, standardized evaluation metrics across applications from gesture recognition [85] to SLAM [416] will accelerate adoption, ensuring neuromorphic vision becomes a cornerstone of next-generation robotics.

2.8 Summary

This chapter presents two systematic reviews: one on neuromorphic computing for interactive robotics and another on neuromorphic vision for interactive robotics. The first review explores how neuromorphic computing, particularly spiking neural networks (SNNs) and brain-inspired architectures, enhances robotic perception, cognition, and visuo-motor control. The second focuses on event-based vision, highlighting its advantages over traditional frame-based approaches in dynamic environments. Both reviews examine the role of neuromorphic hardware, frameworks, and algorithms in advancing human-robot interaction. While these technologies offer significant potential, challenges such as hardware limitations, scalability, and real-world integration remain key areas for future research. While neuromorphic computing and vision significantly enhance interactive robotics, their deployment on specialized neuromorphic hardware is crucial for achieving real-time performance and energy efficiency. The next chapter explores how spiking neural networks (SNNs) are deployed on neuromorphic platforms, discussing learning mechanisms, conversion approaches, and practical deployment challenges.

Chapter 3

Deploying SNN Models on Neuromorphic Hardware

3.1 Introduction

This chapter focuses on deploying spiking neural network (SNN) models on neuromorphic hardware, a critical step toward efficient, real-time neuromorphic computing. It begins by reviewing various learning mechanisms in SNNs, including unsupervised, supervised, and reinforcement approaches. The chapter then systematically evaluates ANN-to-SNN conversion methods, comparing them based on accuracy, energy efficiency, and latency. In addition, it proposes dynamic attention-based multi-object tracking with event-based vision systems and the SpiNNaker platform, highlighting their potential for complex, real-world applications.

Novelty & Impact

Novelty: Provides systematic benchmarking of ANN-to-SNN conversion methods and introduces an event-driven attention mechanism for multi-object tracking on neuromorphic hardware (SpiNNaker).

Impact: Offers practical guidelines and demonstrates significant energy efficiency and real-time responsiveness, advancing the deployment of neuromorphic solutions in robotics.

3.2 Learning in Spiking Neural Networks

The fundamental principles governing spiking neural networks (SNNs) differ from those of conventional neural networks, so traditional learning techniques cannot be applied directly without substantial modification and often become considerably more difficult to use in practice. However, several training methods have been developed for SNNs. For unsupervised learning, one of the most well-known approaches is Spike-Timing-Dependent Plasticity (STDP) [217]. In STDP, synaptic weight adjustments depend on the temporal relationship between pre- and post-synaptic spikes. According to Hebb's postulate, if a pre-synaptic spike arrives before the post-synaptic neuron fires, it is deemed a causal relationship, leading to Long-Term Potentiation (LTP) and thus strengthening the synaptic connection. Conversely, if the post-synaptic spike occurs before the pre-synaptic spike, the relationship is anti-causal and results in Long-Term Depression (LTD). This timing-based mechanism is captured by the STDP function, often referred to as the learning window.

For supervised learning in SNNs, although adaptations of backpropagation have gained prominence [219], STDP can also be extended to a supervised framework by incorporating additional global feedback signals or error gradients on top of its local spike-timing–based updates. Algorithms like SpikeProp [51] and FreqProp [50] demonstrate that networks of spiking neurons, with biologically relevant time constants, can perform complex non-linear classification tasks using temporal coding. Another supervised learning method, ReSuMe [303], is applicable not only to movement control but also to tasks such as object identification and modelling non-stationary systems. In reinforcement learning, several models have been proposed for training SNNs. One such approach is the actor-critic model [304], which employs temporal-difference learning by integrating local plasticity rules with global reward signals. This model has demonstrated its ability to solve challenging grid-world tasks with sparse rewards. Another reinforcement learning technique involves modulating STDP [127], where the modulation serves as a global reward signal to facilitate reinforcement learning.

Feed-forward Neural Networks

Feed-forward neural networks (FNNs) move data strictly from input to output through one or more hidden layers, with no recurrent or skip-back connections. Each layer applies a linear transform followed by a non-linear activation, enabling the network to approximate arbitrary static functions. Convolutional neural networks (CNNs) and multilayer perceptrons (MLPs) are the two most common FNN variants: CNNs exploit weight sharing for grid-like data images, tactile arrays, and odometry maps whereas MLPs are typically applied to vector inputs. In robotics, FNNs serve mainly as perceptual front-ends, for example classifying RGB-D images for object recognition [336], detecting volatile compounds with electronic-nose arrays [328], and mapping high-resolution taxel readings to contact features [293].

Recurrent Neural Networks (RNN)

Unlike feed-forward architectures, recurrent neural networks (RNNs) incorporate feedback by passing the output (or hidden state) from a previous time step as input to the next time step. This recursive structure, often organised in a linear chain, allows RNNs to maintain internal memory and capture temporal dependencies. While some designs compute the error before feeding the output back in and others after, the defining feature remains the presence of these feedback loops, enabling RNNs to handle arbitrary input sequences effectively. Additionally, the Liquid State Machine (LSM) is a reservoir-computing model built from recurrent spiking neurons: the "liquid" reservoir has fixed, random synaptic weights, and training is performed only on a separate read-out layer—typically with linear regression or gradient-based optimisation—while the internal connectivity itself remains unchanged [195]. In robotics, RNNs are extensively applied in tasks such as speech recognition [20], control [47], and path planning [321].

Hybrid Neural Network Structures

This hybrid SNN architecture consists of neurons with both feed-forward and recurrent connections. Such hybrid structures are commonly used for end-to-end training

in SNNs, particularly for tasks like object detection and pattern recognition [210]. Experimental studies have shown that this approach yields promising results while reducing computational costs [209].

Hybrid Neural Network Architectures

In these architectures, Spiking Neural Networks (SNNs) interact with Artificial Neural Networks (ANNs). Here, the term ANN refers to second-generation, non-spiking neural networks, including conventional deep learning models such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory (LSTM) networks. These models employ continuous-valued activations and are typically trained using gradient-based backpropagation. Hybrid ANN–SNN architectures leverage the high representational capacity and mature training frameworks of ANNs alongside the sparse, event-driven computation of SNNs. Such hybrid systems can be trained jointly without requiring full ANN-to-SNN conversion, often achieving high accuracy while improving computational efficiency and reducing energy consumption compared to purely ANN-based implementations. [329].

3.2.1 Spiking Neural Network Models

Spiking Neural Networks (SNNs) are designed based on the mathematical representation of biological neurons, typically formulated using differential equations. A variety of spiking neuron models exist because they capture different aspects of neural dynamics, strike varying balances between biological realism and computational simplicity, and cater to diverse application needs. For instance, simpler models (e.g., Leaky Integrate-and-Fire) emphasise computational efficiency, while more biologically detailed models (e.g., Hodgkin–Huxley) account for complex ion-channel interactions and membrane potentials. Between these extremes, models like Izhikevich or Adaptive Exponential Integrate-and-Fire offer a compromise, incorporating key spiking behaviours without incurring excessive computational cost [180]. In robotic applications, model choice is often driven by the trade-off between simulating essential neural phenomena (such as how excitatory or inhibitory inputs shape spiking

activity) and maintaining real-time performance. In this subsection, we will explore some of the most influential spiking neuron models that are widely utilised in robotic applications.

Hodgkin-Huxley Neuron Model

The first bio-inspired neural model was developed by Sir Alan Hodgkin and Sir Andrew Huxley in 1952 [171]. Their mathematical model explains the initiation and propagation of action potentials in neurons. It consists of a set of nonlinear differential equations that approximate the electrical properties of neurons. Additionally, this model describes the ionic mechanisms responsible for generating and transmitting action potentials, specifically in the giant axons of a squid. The following differential equation of Hodgkin-Huxley model is relating the change in membrane potential to the current flowing across the membrane:

$$C_m \frac{dV_m}{dt} + I_{ion} = I_{ext} \tag{3.1}$$

The I_{ext} is an externally applied current. C_m is membrane Capacitance whereas V_m is membrane Voltage. Here the iconic current I_{ion} is the combination of three components, a sodium current, a potassium current and small leakage current[335].

Leaky Integrated-and-Fire (LIF) Neuron Model

This model is widely used due to its simplicity. In the leaky integrate-and-fire (LIF) model, each neuron has a membrane potential V_m, along with a capacitance C_m and a leaky channel that allows current to pass through the membrane with resistance R_m. The movement of charge carriers across the membrane is driven by the force voltage V_e. When the membrane potential surpasses a threshold value V_{th}, the neuron generates an action potential (spike). Following the spike, the membrane potential is reset to V_{reset}. After firing, the neuron may enter a temporary state where it cannot be excited, known as the refractory period, during which it remains unresponsive to further inputs.

The differential equation describes how the membrane potential V_m changes over time ($\frac{dV_m}{dt}$) in the face of an externally applied membrane current I_m is as follows:

$$C_m \frac{dV_m}{dt} = -G_m(V_m - V_e) + I_m \tag{3.2}$$

A simplified form of the last equation is:

$$\frac{dV_m}{dt} = \frac{-(V_m - V_e) + I_m R_m}{\tau_m} \tag{3.3}$$

Here V_e, R_m, τ_m are taken to be intrinsic properties of the cell while I_m is the external current and V_m is the membrane potential [106]. In hardware, the same dynamics can be reproduced with a compact CMOS circuit, shown in Fig. 3.1.

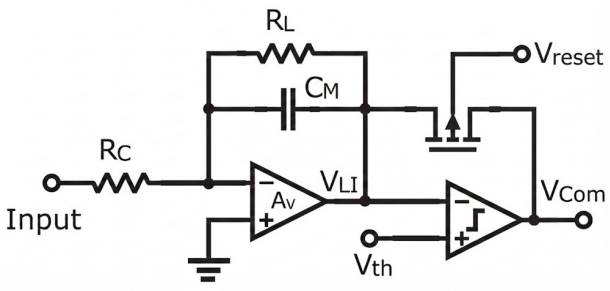


Figure 3.1: Circuit-level realisation of a leaky integrate-and-fire (LIF) neuron. The membrane capacitor C_M integrates the input current through the leak resistor R_L, generating the membrane voltage V_{LI}. A comparator continuously monitors V_{LI} and produces an output spike V_{Com} when the voltage exceeds the threshold V_{th}. The spike activates the negative-feedback amplifier stage A_V, which rapidly discharges C_M towards the reset potential V_{reset}, thereby resetting the neuron and initiating the next integration cycle. This figure is adapted from [74].

Izhikevich Neuron Model

The Izhikevich neuron model [181] balances the biological realism of the Hodgkin-Huxley model with the computational efficiency of the leaky integrate-and-fire (LIF) model. It effectively replicates the spiking and bursting behaviour observed in various types of cortical neurons [182]. This model is relatively simple, requiring only two state variables and four tunable parameters to capture the complex dynamics of cortical neurons [212].

This model can be represented as 2-D system of differential equations [181]:

$$\frac{dV_m}{dt} = 0.04V_m^2 + 5V_m + 140 - U + I_m, \tag{3.4}$$

$$\frac{dU}{dt} = a(bV_m - U), \tag{3.5}$$

$$\text{if } V_m \geq V_{\text{peak}}, \quad \begin{cases} V_m \leftarrow c, \\ U \leftarrow U + d, \end{cases} \tag{3.6}$$

Where V_m is the membrane potential, U is the membrane recovery variable I_m is injected bias current or incoming synaptic spikes. a and b are dimensionless variables that tweak the neuron's behaviour based on their values.

3.2.2 Synaptic Plasticity Models

Once a suitable neural model is chosen, the next step is to determine the synapse model that connects neurons within and across layers of the spiking neural network. Synaptic plasticity, originally proposed by Hebb in 1949 [166], serves as a fundamental mechanism for learning and memory, grounded in theoretical analysis. Based on the relationship between neural activity and synaptic plasticity, these mechanisms are generally categorized into two types: spike-based and rate-based plasticity [47].

Rate– and Spike–based

Early formulations of synaptic plasticity described weight change as a function of the *firing rate*—the average spike count in a time window [26]. Such *rate-based* rules are useful when a conventional ANN is trained with back-propagation and then converted to a spiking implementation: the ANN activation $a = g(u)$ is mapped to the steady-state firing rate $r = f(u)$ of a spiking neuron, provided $f(\cdot)$ approximates $g(\cdot)$ [88, 319, 320]. This strategy enables energy-efficient SNN deployments for robotic perception and control after an off-line ANN training phase [218, 400, 246].

Subsequent electrophysiological studies showed that weight change can also depend on the *precise timing* of individual spikes [144, 196]. These *spike-based* rules relate

closely to rate models: the BCM rule [369] expresses potentiation and depression through a firing-rate threshold, and Izhikevich & Desai [183] demonstrated how the classical BCM formulation can be obtained by integrating a pair-based spike-timing rule over time. Thus rate- and spike-based descriptions are complementary views of the same underlying plasticity phenomena; the choice depends on the temporal resolution required by the task.

Spike-Timing-Dependent Plasticity (STDP)

STDP refines spike-based plasticity by making the weight update Δw an explicit function of the pre-/post-spike interval $\Delta t = t_{\text{post}} - t_{\text{pre}}$:

$$\Delta w(\Delta t) = \begin{cases} A_+ \exp(-\Delta t/\tau_+), & \Delta t \geq 0 \ (\text{pre} \to \text{post}) \\ -A_- \exp(\ \ \Delta t/\tau_-), & \Delta t < 0 \ (\text{post} \to \text{pre}) \end{cases} \tag{3.7}$$

where A_+ and A_- set the peak magnitudes of potentiation and depression, and τ_+, τ_- are their respective time constants. Implemented in both silicon and software, STDP has enabled online learning of sensorimotor mappings in mobile and humanoid robots, as well as adaptive feature extraction in event-based vision systems.

3.3 ANN-to-SNN Conversion

Training a Spiking Neural Network (SNN) directly from scratch presents considerable challenges due to the non-differentiable nature of spike-generation functions. Traditional gradient-based backpropagation methods fail because the spike function derivative is zero almost everywhere. One effective strategy to overcome this issue is the *surrogate gradient* approach, where, during backpropagation, the exact spike function is replaced by a differentiable surrogate function, such as a smooth sigmoid, to approximate the true spike function near the threshold [401]. Although surrogate gradient and biologically inspired training methods like spike-timing-dependent plasticity (STDP) enable direct SNN training, they often require extensive computational

resources and do not consistently achieve the performance levels attained by Artificial Neural Networks (ANNs).

To mitigate these issues, ANN-to-SNN conversion has emerged as a practical alternative. This approach exploits the training efficiency and performance robustness of conventional ANN frameworks like PyTorch or TensorFlow. The ANN is first trained traditionally using gradient-based backpropagation until convergence. Afterwards, the trained ANN model is converted into an SNN by mapping continuous-valued neuron activations into discrete spike trains. This conversion involves replacing conventional activation functions with spiking neuron models, whose membrane potentials and firing thresholds are adjusted to approximate ANN neuron outputs as firing rates averaged over time. This methodology preserves ANN-level accuracy while leveraging the inherent energy efficiency and temporal sparsity advantages of spiking networks. Despite its advantages, ANN-to-SNN conversion is accompanied by several inherent challenges. Conversion typically introduces discrepancies between ANN continuous activations and discrete spike events, potentially causing accuracy loss, especially in deeper networks [93]. Additionally, conversion strategies usually result in trade-offs among accuracy, latency (inference time steps), and energy efficiency (total spike count) [401]. For instance, rate-based coding maintains accuracy at the cost of excessive spikes, whereas temporal coding achieves sparse activity at the expense of accuracy sensitivity [390]. To address these trade-offs, various strategies have been developed:

- *Rate Coding*: Converts ANN activations proportionally into spike rates, ensuring high accuracy for shallow models but producing excessive spikes in deeper networks [390].
- *Temporal Coding*: Encodes activation magnitudes through precise spike timing, improving sparsity but requiring high temporal precision [318].
- *Threshold Adjustment*: Post-conversion tuning of neuron thresholds to minimize quantization errors and reduce spike counts, though at the cost of higher latency and manual tuning complexity [161].

- *Hybrid Training*: Combines ANN pre-training with a brief surrogate gradient fine-tuning stage post-conversion, aiming to restore accuracy while preserving spike efficiency [390].

3.4 Systematic Evaluation of ANN-to-SNN Conversion Approaches

To rigorously assess ANN-to-SNN conversion, we systematically evaluated five prominent approaches: Rate Coding (SNNToolbox [320]), Temporal Coding (BrainCog [410]), Threshold Adjustment (SNNToolbox [320]), Surrogate Gradient Descent (SNNTorch [113]), and Hybrid ANN-SNN Training (BrainCog [410]). We also trained native SNNs using surrogate gradients to establish baseline performance metrics. Experiments were conducted on three widely-used datasets with increasing complexity: MNIST [37], Fashion-MNIST [398], and CIFAR-10 [100].

Our contributions include:

1. Providing a comprehensive comparison of conversion approaches in terms of accuracy, spike efficiency, latency, and complexity.

2. Empirically demonstrating how dataset complexity influences the optimal choice of conversion strategy.

3. Establishing practical guidelines for selecting conversion methods based on specific application constraints.

We focused exclusively on these core conversion techniques, excluding methods like Spike Propagation-based Training [218] and Liquid State Machines [361], to maintain clarity and relevance.

3.4.1 Experiments

We trained ANN models individually on each dataset, subsequently converting them into SNN models using the five specified approaches. The following subsections describe the training details and conversion rationale.

Table 3.1: Comparison of ANN-to-SNN Conversion Approaches

Category	Coding Scheme	Training	Key Features	Advantages	Best For
Native SNN	Temporal coding	From scratch	Biologically plausible, sparse spikes	Energy-efficient, temporal dynamics	Event-based data
Rate-Based	Rate coding	Pre-trained ANN	Simple, no temporal dynamics	Easy to implement, accurate on static data	Static image datasets
Surrogate Gradient	Rate coding w/ gradients	Fine-tuned ANN	Enables SNN training, moderate spikes	Balances accuracy & efficiency	Dynamic datasets
Temporal Coding	Temporal coding	Pre-trained ANN	Encodes magnitude into timing, sparse spikes	Energy-efficient, preserves timing	Low-power temporal data
Threshold Adjustment	Adjusted rate coding	Pre-trained ANN	Balances accuracy & efficiency, reduced spikes	Customizable thresholds	Trade-offs between accuracy & energy
Hybrid ANN-SNN	Hybrid ANN-SNN	Pre-trained ANN + SNN	Combines ANN speed & SNN efficiency, sparse spikes in SNN layers	Balanced speed, flexible architecture	Mixed data types

ANN Training

ANN architectures were specifically designed for each dataset to facilitate subsequent conversion to SNN models. The base architecture, comprising two convolutional layers (kernel size , padding=1) followed by ReLU activations and max-pooling, and two fully connected layers, was adjusted to match dataset-specific complexities, as detailed in Table 3.2.

Models were trained using the Adam optimiser with an initial learning rate of 0.001, applying the ReduceLROnPlateau strategy, which reduces the learning rate by a factor of 0.1 upon plateauing of the validation loss for ten epochs. Cross-entropy

Table 3.2: ANN architectures and training details for MNIST, Fashion-MNIST, and CIFAR-10 datasets.

Dataset	ANN Architecture	Training Details & Val Acc.
MNIST	3-layer CNN (Conv-ReLU-Pool, FC)	Adam, Cross-Entropy, 99.2% acc.
Fashion-MNIST	4-layer CNN (Conv-ReLU-Pool, FC)	Adam, Cross-Entropy, 94.1% acc.
CIFAR-10	VGG-like CNN (Conv-ReLU-Conv, FC)	SGD, Data Aug., 88.4% acc.

loss was consistently employed for optimisation. A batch size of 64 and early stopping with patience set at 15 epochs ensured efficient convergence and mitigated overfitting.

Data was normalised to zero-centred distributions. For CIFAR-10, standard augmentation techniques, including random horizontal flips and random cropping with padding, enhanced model generalisation, while simpler datasets like MNIST required no augmentation. Batch normalisation was systematically applied post-convolution to ensure stable training dynamics. All training occurred on an NVIDIA RTX 3080 GPU for a maximum of 200 epochs, selecting models based on optimal validation loss.

Native SNN Training as a Baseline

To contextualise the effectiveness of ANN-to-SNN conversion, native SNN models were trained directly using surrogate gradient methods. This baseline provides a principled point of comparison, reflecting the performance attainable by models optimised within the spiking domain under the same training conditions. It enables a fair comparative analysis, revealing the trade-offs inherent in conversion methods, such as accuracy degradation or increased latency and spike count. Surrogate gradients employed a smooth sigmoid function to approximate spiking neuron derivatives during backpropagation. Inputs from non-spiking datasets were encoded into spike trains through rate-based encoding. Membrane potential dynamics were carefully tuned during training by adjusting parameters such as neuron thresholds and membrane decay rates to optimise neuron responsiveness and spike propagation. The complexity of CIFAR-10 required additional regularisation methods and increased training epochs compared to MNIST and Fashion-MNIST, as empirically validated by observed convergence behaviour and validation accuracy stabilisation during training.

Conversion Algorithms

ANN-to-SNN conversion employs diverse encoding strategies to balance accuracy, energy efficiency, and complexity. *Rate coding* maps ANN activations directly to firing rates using Poisson-distributed spike trains. While accurate, it produces higher spike counts, leading to increased energy consumption and latency [408]. Alternatively, temporal coding encodes activations through spike timing, achieving sparse and efficient representations but requiring precise synchronization, thus being noise-sensitive [229]. Threshold adjustment reduces quantization errors by tuning neuronal thresholds after conversion, significantly decreasing spike counts without major accuracy loss, although manual calibration is required [163]. *Surrogate gradient descent* leverages ANN-trained weights as initial parameters and refines them using differentiable surrogate functions for spike generation. Despite effectively optimising accuracy and spike efficiency, this method demands careful hyperparameter tuning [19]. Lastly, hybrid ANN–SNN training combines ANN layers for early feature extraction with SNN layers for subsequent event-driven processing, trading biological plausibility for improved performance on complex datasets. Because part of the computation is performed by non-spiking ANN components, activity-based metrics such as average spikes per neuron cannot be directly compared with fully spiking SNN-only models. [345]. These strategies collectively offer flexible solutions tailored to diverse application requirements.

Algorithm 1 shows the conversion pipeline.

3.4.2 Results and Discussion

The evaluation across MNIST, Fashion-MNIST, and CIFAR-10 datasets provides a comprehensive understanding of the strengths and limitations of various ANN-to-SNN conversion methods. All average spike counts per neuron are computed over a simulation window of 25 time steps. Each model was trained and evaluated over multiple independent runs with different random initialisations and data shuffling, and results are reported as mean ± standard deviation to assess statistical robustness.

Algorithm 1 ANN-to-SNN Conversion Pipeline

```
procedure ANN_TO_SNN(ann_model, method)
    Load pre-trained ANN model ann_model
    Initialize empty SNN model snn_model
    if method = Rate-Based then
        Convert activations to firing rates
        Map weights and biases directly from ANN
    else if method = Surrogate Gradient then
        Replace ReLU activations with LIF neurons
        Fine-tune using surrogate gradients
    else if method = Temporal Coding then
        Map input magnitude to spike timing
        Calibrate SNN layers to preserve timing dynamics
    else if method = Threshold Adjustment then
        Transfer ANN weights and biases to SNN
        Dynamically adjust thresholds based on activations
    else if method = Hybrid ANN-SNN then
        Retain ANN-like layers for early processing
        Convert deeper layers to SNN with LIF neurons
    end if
    Return snn_model
end procedure
procedure EVALUATE_SNN(snn_model, dataset)
    Initialize variables: accuracy, spike count, latency
    for each batch in dataset do
        for each time step t do
            Simulate spike activity in snn_model
        end for
        Compute accuracy and total spikes
    end for
    Measure latency as total simulation time
    Return accuracy, spike count, latency
end procedure
```

For MNIST, a dataset with low spatial complexity, all conversion approaches achieved high accuracy, closely matching the baseline ANN performance of 99.2%. Temporal Coding yielded the best SNN performance with 97.92% accuracy, low average spikes per neuron (0.126), and a competitive inference latency of 1186.04 ms, where latency refers to the total time required to process all simulation steps during inference. The Hybrid Approach similarly achieved high accuracy (97.73%) and demonstrated the lowest latency (940.18 ms), indicating computational efficiency. Native SNNs, trained directly without ANN pre-training, achieved 97.68% accuracy with an average spike count of 0.105, demonstrating efficient spiking behaviour but slightly reduced accuracy compared to converted models. Rate-Based and Surrogate Gradient approaches maintained high accuracy (97.75% and 97.7%, respectively) but exhibited higher spike counts and latencies, reflecting inherent trade-offs. Threshold Adjustment resulted in the lowest accuracy (96.85%), highlighting calibration difficulties even in simpler datasets.

Fashion-MNIST, with moderate complexity, revealed clearer distinctions. The Hybrid Approach achieved the highest accuracy (93.34%) and the lowest latency (1750.32 ms) with low spike counts (0.221). Temporal Coding maintained high efficiency, achieving 92.12% accuracy with the sparsest activity (average spikes 0.184) but higher latency (2100.54 ms). Surrogate Gradient Descent demonstrated competitive performance (91.25%), though latency remained slightly elevated. Native SNNs and Rate-Based conversions underperformed relative to Hybrid and Temporal methods, while Threshold Adjustment again struggled, with the lowest accuracy (87.1%) and the highest spike count (0.265).

Table 3.3: Comparison of ANN-to-SNN conversion methods on MNIST, Fashion-MNIST, and CIFAR-10 datasets. Results are reported as mean ± standard deviation over multiple independent training runs.

Method	Accuracy (%)			Avg Spikes/Neuron			Latency (ms)		
	MNIST	F-MNIST	CIFAR-10	MNIST	F-MNIST	CIFAR-10	MNIST	F-MNIST	CIFAR-10
Native SNN	97.68 ± 0.21	89.34 ± 0.35	71.54 ± 0.48	0.105 ± 0.006	0.212 ± 0.011	0.290 ± 0.015	913 ± 42	2300 ± 95	7608 ± 210
Rate-Based	97.75 ± 0.18	88.60 ± 0.40	67.42 ± 0.52	0.130 ± 0.007	0.245 ± 0.013	0.317 ± 0.017	1190 ± 50	2700 ± 110	8308 ± 245
Surrogate Gradient	97.70 ± 0.19	91.25 ± 0.33	78.05 ± 0.41	0.131 ± 0.006	0.226 ± 0.012	0.282 ± 0.014	1094 ± 47	1901 ± 88	6130 ± 198
Temporal Coding	97.92 ± 0.16	92.12 ± 0.30	79.89 ± 0.38	0.126 ± 0.005	0.184 ± 0.010	0.201 ± 0.012	1186 ± 53	2101 ± 92	6909 ± 205
Threshold Adjustment	96.85 ± 0.24	87.10 ± 0.42	73.71 ± 0.49	0.141 ± 0.008	0.265 ± 0.015	0.352 ± 0.018	979 ± 45	1951 ± 87	6135 ± 190
Hybrid Approach	97.73 ± 0.17	93.34 ± 0.28	83.76 ± 0.35	0.136 ± 0.006	0.221 ± 0.011	0.282 ± 0.013	940 ± 39	1750 ± 80	5853 ± 175

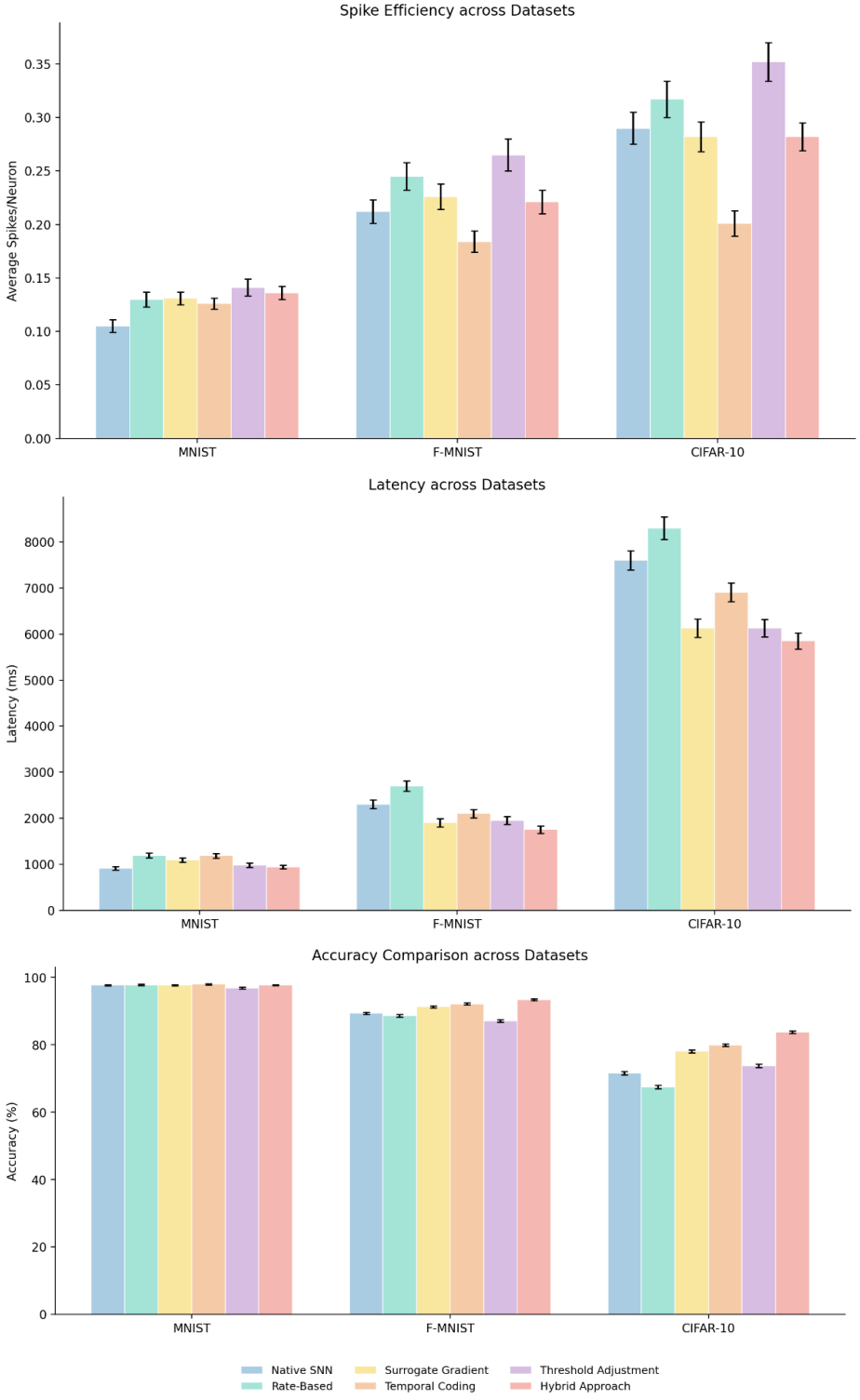


Figure 3.2: Comparative evaluation of ANN-to-SNN conversion methods on MNIST, Fashion-MNIST, and CIFAR-10. Each method is assessed across three metrics: classification accuracy (bottom), spike efficiency measured as average spikes per neuron over 25 time steps (top), and inference latency computed as total processing time across all time steps (middle). The results highlight trade-offs between accuracy, energy efficiency, and computational cost for each method and dataset.

CIFAR-10, the most complex dataset, further emphasised method robustness. The Hybrid Approach outperformed all others, achieving 83.76% accuracy, the lowest latency (5852.61 ms), and low spike counts (0.282). Temporal Coding followed with 79.89% accuracy and the sparsest firing (0.201 average spikes per neuron). Surrogate Gradient achieved 78.05% accuracy with moderate spike and latency metrics. Native SNNs and Rate-Based conversions exhibited reduced performance, with Native SNNs achieving 71.54% accuracy and Rate-Based only 67.42%. Threshold Adjustment provided reasonable accuracy (73.71%) but incurred the highest spike activity (0.352). Notably, Native SNNs consistently produced low spike counts across all datasets, affirming their energy efficiency. However, their accuracy declined markedly as dataset complexity increased, underscoring the advantage of ANN-to-SNN conversion strategies for maintaining robust performance.

Overall, these results reveal that dataset complexity profoundly influences conversion performance. As complexity rises, differences among methods become more pronounced. Hybrid Approaches demonstrate superior capability in managing diverse features while maintaining high accuracy and low latency, making them suitable for applications prioritising predictive performance, such as embedded AI or real-time vision systems. In contrast, for deployment scenarios where energy efficiency is critical, such as neuromorphic hardware implementations, Temporal Coding can be preferable due to its reduced spiking activity, which is a primary contributor to dynamic energy consumption in event-driven systems. Specifically, Temporal Coding decreases the average spikes per neuron by approximately 18.6% on Fashion-MNIST and 28.7% on CIFAR-10 relative to the surrogate-gradient SNN baseline (Table 3.3). Assuming energy scales approximately with synaptic event count, these reductions suggest a corresponding decrease in compute energy, although the precise savings depend on the hardware architecture. Figure 3.2 presents the comparative bar plots of the results, and Table 3.3 summarises the detailed performance metrics. Overall, the choice of ANN-to-SNN conversion strategy should be guided by dataset complexity and application constraints: hybrid approaches are best suited for high-accuracy and low-latency scenarios, while temporal coding is more appropriate for energy-constrained neuromorphic deployments.

3.5 Deploying SNN models to SpiNNaker

So far, we have systematically analysed ANN-to-SNN conversion approaches in simulation. However, simulation-based evaluation does not fully capture the temporal dynamics, communication overhead, and energy characteristics of real neuromorphic hardware. To validate the practical efficiency and deployability of the proposed models, the next step is to implement them on dedicated neuromorphic hardware. In this work, we deploy the SNN model on a 48-chip SpiNNaker board, which provides massively parallel, event-driven computation designed specifically for large-scale spiking neural networks.

3.5.1 48-Chip SpiNNaker

The SpiNNaker (Spiking Neural Network Architecture) project is an innovative initiative spearheaded by the University of Manchester in collaboration with academic and industrial partners [291]. It is designed to simulate large-scale Spiking Neural Networks (SNNs) in real-time by leveraging a biologically inspired high-performance computing architecture. One of its key strengths lies in its fault tolerance and energy efficiency, which resemble the characteristics of the human brain. The SpiNNaker system achieves this by implementing a highly parallel computing structure, enabling real-time neural computations with significantly lower power consumption compared to conventional architectures. SpiNNaker board consists of interconnected SpiNNaker chips and boards, where each custom Application-Specific Integrated Circuit (ASIC) chip contains 18 low-power ARM processors similar to those found in mobile devices. Despite their low power consumption (each chip with 1W power budget), these processors are fully programmable, allowing them to execute a variety of neural and synaptic models. Neuron spikes generated during simulations are transmitted as short data packets across a bespoke Network-on-Chip (NoC). This network facilitates efficient inter-chip communication, ensuring seamless data transfer between processors, both within a chip and across multiple chips in the system. Each processor can handle multiple neurons simultaneously, depending on the computational complexity of the neural model. This scalability allows SpiNNaker to simulate hundreds of

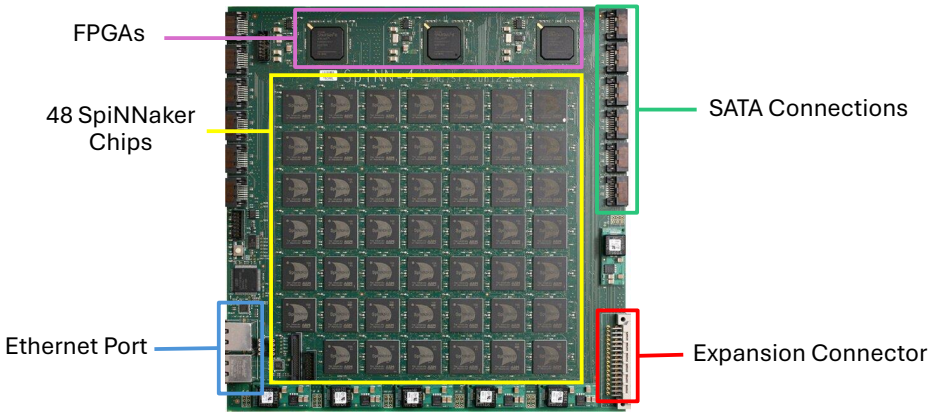


Figure 3.3: A 48-chip SpiNNaker board.

millions of neurons when fully deployed. PyNN [87], a high-level modelling language widely used in computational neuroscience, manages the definition and configuration of spiking neural networks (SNNs) on SpiNNaker. Given the vast scale of simulations, manually defining individual neurons and their connections becomes impractical. Instead, PyNN enables hierarchical network descriptions, allowing users to design and implement complex neural architectures efficiently.

To enable the use of SNNs on SpiNNaker, a dedicated SpiNNaker configuration software stack is utilised. It processes high-level network descriptions from PyNN, subdivides them into manageable sections, and allocates them to the available processors according to the topology and computational potential of the SpiNNaker system. For instance, a simulation involving 10,000 neurons can be segmented into chunks, with each segment managed by a designated processor, thus ensuring efficient allocation of resources and effective data flow management. SpiNNaker offers two primary methods for accessing simulation results: post-simulation analysis and real-time data streaming. In post-simulation analysis, PyNN allows the recording of key parameters, such as neuron membrane potentials over time. Once the simulation concludes, these recorded values can be extracted, analysed, and visualised to gain insights into neural activity. On the other hand, real-time data streaming enables live data retrieval during simulations, allowing continuous monitoring of spikes and neural activity as they occur. This capability is particularly beneficial in robotic

applications, where real-time feedback loops must be dynamically adjusted for effective performance.

3.5.2 SNN Models Deployment

A systematic evaluation of ANN-to-SNN conversion methods is detailed in Section 3.4. Our findings demonstrate that with appropriate conversion and fine-tuning techniques, the accuracy of Spiking Neural Networks (SNNs) can closely approximate that of their original Artificial Neural Network (ANN) counterparts. However, the central motivation for adopting SNNs extends beyond accuracy preservation; it lies in exploiting their core advantages, namely, energy efficiency, event-driven computation, and suitability for neuromorphic hardware.

To realise these benefits, we deployed the converted SNNs onto SpiNNaker, a neuromorphic computing platform designed to emulate large-scale spiking neural systems. Unlike conventional CPUs or GPUs, SpiNNaker executes event-based computations in a massively parallel and asynchronous manner, aligning with the operating principles of biological neural networks. This deployment enables a realistic assessment of SNN performance under practical hardware constraints, focusing particularly on power efficiency and processing latency. Simulation parameters were tuned to reflect dataset-specific demands. For MNIST and Fashion-MNIST, simulations ran for 100 milliseconds per input sample, using a batch size of 64. A fixed simulation time step of 0.1 ms was applied. Inputs were encoded using a Poisson spike generator, where the instantaneous spike probability was modulated by the ANN activation at each pixel location. A baseline maximum input rate of 1000 Hz was applied to ensure sufficient encoding bandwidth. Synaptic dynamics were modelled using first-order exponential kernels, with excitatory and inhibitory synaptic time constants (τ_{synE} and τ_{synI}) set to 0.1 ms and 0.05 ms, respectively. Although separate inhibitory neuron populations were not implemented, inhibitory behaviour was incorporated by adjusting the membrane decay constants to control the rate at which neuron activity diminished over time, effectively suppressing sustained firing in a manner analogous to inhibitory input. For CIFAR-10, the simulation duration

was extended to 500 ms per sample to accommodate the dataset's greater feature complexity. The maximum input spike rate was increased to 4000 Hz to ensure sufficient activation in deeper convolutional layers. The excitatory synaptic time constant was increased to 0.1 ms to allow more sustained current integration, while the inhibitory constant remained fixed at 0.05 ms.

To preserve consistency across models, weight normalisation was disabled, and exponential weight scaling was applied during conversion to ensure dynamic range compatibility with spiking thresholds. Power consumption was continuously monitored using on-board energy tracking tools. Furthermore, both the ANN inference time and the SNN simulation runtime were recorded to benchmark computational efficiency. This hardware-based evaluation serves as a critical validation step, bridging the gap between theoretical conversion accuracy and practical performance, and highlights the viability of SNN deployment in real-time, energy-sensitive applications such as edge computing and autonomous systems.

The key configuration parameters used for deploying the converted SNN models on SpiNNaker are summarized in Table 3.4.

Table 3.4: Configuration Parameters for SNN Deployment on SpiNNaker

Parameter	MNIST / FMNIST	CIFAR-10
Simulation Duration	100 ms	500 ms
Batch Size	64	64
Time Step (dt)	0.1 ms	0.1 ms
Input Spike Rate	1000 Hz	4000 Hz
Excitatory STC (τ_{synE})	0.1	0.1
Inhibitory STC (τ_{synI})	0.05	0.05
Exponential Weight Scaling	Enabled	Enabled

Power Profiling

To evaluate the power efficiency of SNN execution on SpiNNaker, we performed empirical measurements of electrical current and voltage under two conditions: (i) idle, when the SpiNNaker board was powered but not executing any models, and (ii) active, during the execution of converted SNN models for MNIST, Fashion-MNIST, and CIFAR-10. Power consumption was calculated using the standard relation

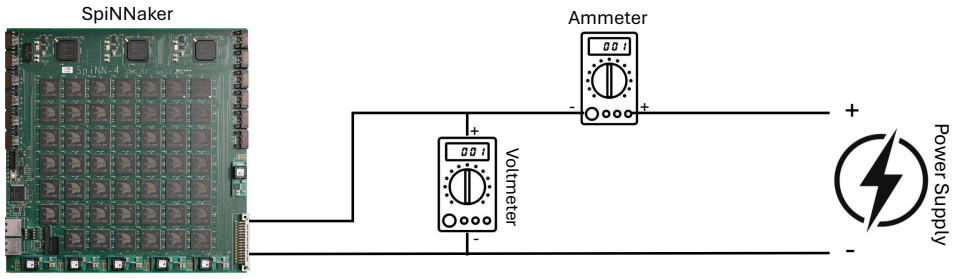


Figure 3.4: Measurement setup for evaluating the power consumption of the SpiNNaker board. A voltmeter is connected in parallel to measure the supply voltage across the board, while an ammeter is placed in series to measure the total current drawn

$P = V \times I$, based on voltage (V) and current (I) readings obtained via digital multimeters. The measurement setup is illustrated in Figure 3.4, where the ammeter was connected in series with the power supply to measure current draw, and the voltmeter was connected in parallel to measure supply voltage across the SpiNNaker board.

In the idle state, the SpiNNaker board consumed 27.10W (2.261A, 11.98V), consistent with previously reported ranges of 25−30W [198]. During active execution, each SNN model incurred a modest increase in power consumption, reflecting computational overhead associated with spiking activity and synaptic processing. Execution of the MNIST model increased power to 30.45W (2.559A, 11.90V), corresponding to a 3.35W overhead. The Fashion-MNIST model showed a similar consumption of 30.36W (2.548A, 11.92V), while CIFAR-10 yielded the highest recorded value at 30.53W (2.547A, 11.92V). These results demonstrate that although the increase in power consumption is relatively small, it scales consistently with task complexity, particularly evident when transitioning from simpler (MNIST) to more complex (CIFAR-10) datasets. This suggests that while SpiNNaker enables energy-efficient SNN inference, variations in power consumption consistently track dataset complexity, with more demanding inputs producing higher measured power draw across experiments, indicating a systematic relationship between computational load and energy usage.

One of the key observations from the power profiling results is that power consumption does not scale linearly with dataset complexity alone. While datasets

Table 3.5: Power consumption of the SpiNNaker board during SNN deployment. ΔP represents the change in power relative to the idle state.

Scenario	Current (A)	Voltage (V)	Power (W)	ΔP (W)
Idle State	2.261	11.98	27.10	-
MNIST	2.559	11.90	30.45	+3.35
FMNIST	2.548	11.92	30.36	+3.26
CIFAR-10	2.561	11.92	30.53	+3.43

such as CIFAR-10 are inherently more complex due to their colour channels and diverse object classes, the corresponding SNN models deployed to process them are also architecturally deeper and contain a greater number of parameters. Therefore, attributing increased power usage solely to dataset complexity conflates it with the influence of model complexity. The observed variation in power consumption is better understood as a combined effect of both factors. More critically, power usage in SNNs is primarily governed by spike activity rather than static model characteristics. Due to the event-driven communication paradigm of SNNs, the dominant dynamic energy cost is associated with spike transmission and synaptic event processing; however, baseline energy is still consumed for neuron state updates and circuit operation even in the absence of spiking activity. As demonstrated in our profiling, even the execution of more complex models for CIFAR-10 resulted in only a marginal increase in power consumption (approximately 3.43 W above the idle state). This behaviour is consistent with prior studies, such as Rast et al. (2020), which highlight the role of spike sparsity in limiting energy expenditure, and Diehl et al. (2015), who reported comparable power overheads (2–5 W) for SNN workloads on SpiNNaker. These findings highlight that, while the overall SpiNNaker board exhibits substantial baseline (idle) power consumption due to system-level components, the incremental power attributable to chip-level neural computation remains comparatively low and scales weakly with model and dataset complexity. This indicates that energy efficiency is realised primarily at the neuromorphic processing level rather than at the full-board system level. While future work could benefit from normalising power consumption with respect to model size or spike rate to enable more rigorous cross-dataset comparisons, the current results demonstrate that SpiNNaker effectively

leverages the temporal sparsity of SNNs to maintain low power usage during inference, even under increasing computational loads [287].

In the following section, we extend this evaluation to a dynamic setting using an event-based camera and SpiNNaker for attention-guided multi-object tracking.

3.6 Event-Driven Dynamic Attention for Multi-Object Tracking on SpiNNaker

We introduce a real-time, event-driven multi-object tracking system that leverages the SpiNNaker neuromorphic platform in combination with a Dynamic Vision Sensor (DVS). At the core of our methodology lies an innovative dynamic attention mechanism, inspired by grid cells in the hippocampus. Grid cells are specialized neurons that form a regular, hexagonal lattice representation of space within the mammalian brain, enabling efficient navigation and spatial memory. Analogously, our approach employs recurrent spiking neural networks (SNNs) that mimic this stable spatial encoding, allowing the system to robustly maintain attention on multiple moving objects even in the presence of distractors. State estimation and motion prediction are performed through Kalman filtering, while morphological open and close operations enhance object detection accuracy. This integrated approach enables precise tracking of objects, including scenarios where they decelerate significantly or stop completely. Our design thus presents a scalable and energy-efficient solution tailored for real-time robotic applications, validated through experiments in dynamic scenarios such as swarm robot evasion.

Multi-object tracking (MOT) represents a significant challenge in computer vision, as it involves preserving the identity and trajectory of several objects across video frames. This challenge becomes increasingly complex when objects move at varying speeds or interact dynamically in real time [236]. Traditional frame-based tracking methods typically require substantial computational resources, particularly under conditions of rapid movement leading to significant pixel displacement between frames [411]. By contrast, the high temporal resolution provided by event-based vision systems helps address these challenges effectively, enabling more robust and accurate tracking in cases where objects decelerate significantly or nearly halt [263, 269]. Additionally, event-driven cameras offer the advantage of interactive, real-time target selection, providing adaptability that is difficult to achieve with conventional frame-based approaches.

Table 3.6: Summary of Existing Research on Object Tracking, Attention Mechanisms, and Neuromorphic Hardware

Reference	MOT	Attention	NH
Indiveri et al. (2000) [178]	No	Yes	Dedicated NH
Lichtsteiner et al. (2008) [230]	No	No	No
Frisch et al. (2015) [131]	Yes	Yes	No
Mueggler et al. (2016) [269]	Yes	No	No
Zhu et al. (2018) [419]	Yes	No	No
Mitrokhin et al. (2018) [263]	Yes	No	No
Pereira et al. (2018) [297]	Yes	Yes	No
Renner et al. (2019) [311]	No	Yes	Loihi
Wang et al. (2023) [385]	Yes	No	No
Afshar et al. (2020) [2]	Yes	No	No
Monforte et al. (2023) [266]	Yes	No	No
Ralph et al. (2022) [306]	Yes	Yes	No
Gava et al. (2022) [139]	No	Yes	SpiNNaker
This Work (2025) [7]	Yes	Yes	SpiNNaker

Event-based vision sensors excel in capturing fast dynamic events; however, the real-time processing of their outputs necessitates specialized hardware capable of handling asynchronous and data-driven signals (see section 1.2). Neuromorphic hardware such as the SpiNNaker chip [347], which mimics the parallel processing capabilities of biological neural networks, addresses this requirement effectively [137] (see subsection 3.5.1 for further details). Designed specifically for asynchronous spike management, this type of hardware provides an optimal platform for deploying complex spiking neural network algorithms in applications like event-based MOT.

Unlike many existing MOT solutions, our focus is on addressing the challenge of handling objects at varying speeds, and we demonstrate that interactive, real-time selection of tracked objects is feasible. Similar principles have been applied in neuromorphic engineering for instance, in the design of a dedicated spiking neuromorphic chip for attention [178] and Renner et al. [311] employed the Intel Loihi chip for single object detection. Here, we showcase MOT on the versatile SpiNNaker platform, which can also support tasks in motor control, cognition, and vision. Our system is built around a recurrent spiking neural network (SNN) that

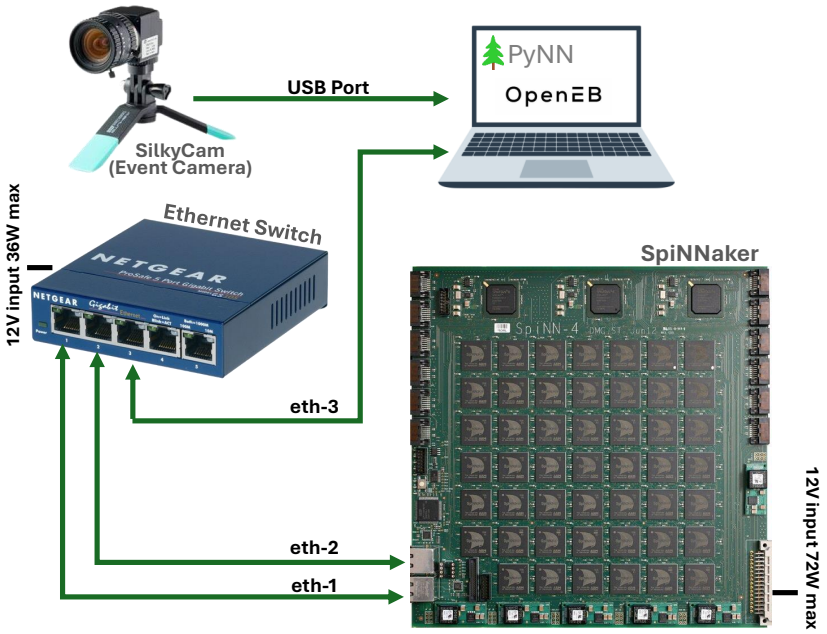


Figure 3.5: Hardware setup for event-based tracking using SpiNNaker.

utilises attractor dynamics to maintain selective attention on the tracked object while suppressing irrelevant distractors [131, 297]. By integrating these components, our approach achieves high tracking accuracy for targets exhibiting various motion profiles, reduces the overall event load, and operates at low power. Table 3.6 offers a comparison of different studies focused on event-based object tracking, attention mechanisms, and the application of neuromorphic hardware (NH).

3.6.1 Hardware Setup

Our configuration combines a CenturyArk SilkyCam VGA event camera [14] with a SpiNNaker 48-chip board [134], as depicted in Figure 3.6. The CenturyArks SilkyCam uses a sensor designed by Prophesee (table 2.8) and it captures asynchronous brightness variations at a 640×480 resolution, streaming events in real time via USB to a laptop. This laptop primarily functions as an interface to execute PyNN scripts that configure and control neural simulations on the SpiNNaker board.

As illustrated in Figure 3.6, the SilkyCam is linked to the SpiNNaker board via an interface machine. This setup enables the system to (1) acquire event data from the camera, (2) preprocess these events, and (3) configure and control spiking neural

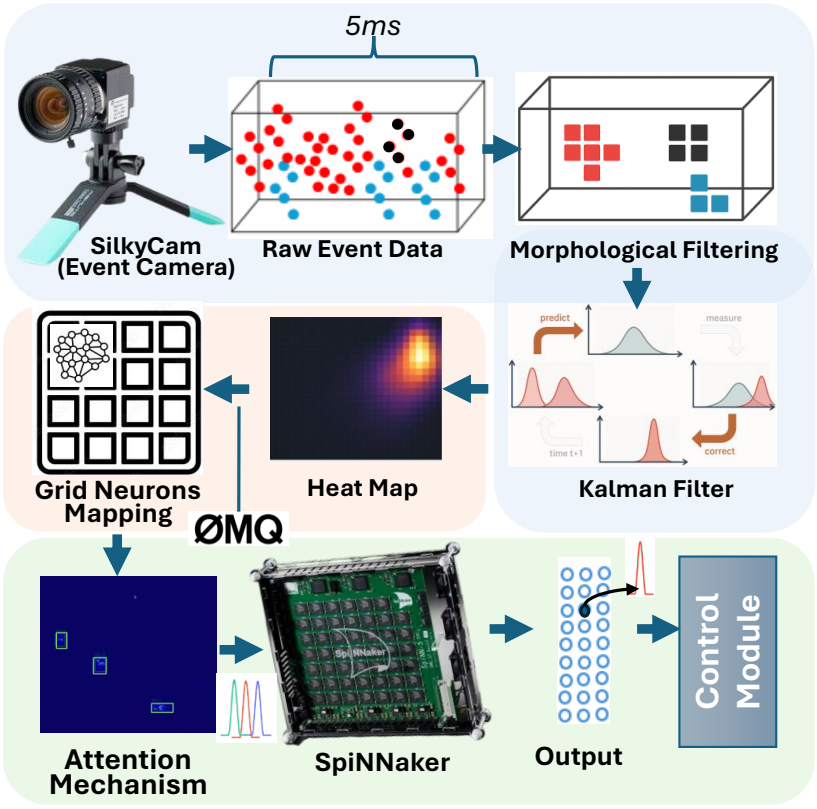


Figure 3.6: Event-based multi-object tracking pipeline implemented on SpiNNaker. The blue region denotes input preprocessing, including raw event acquisition from the SilkyCam, morphological filtering, heat map generation, and Kalman filtering. The yellow region represents spatial mapping, where filtered events are projected onto grid neurons and organised into spatial representations via ZeroMQ communication. The green region corresponds to neuromorphic decision-making, comprising the attention mechanism, spiking computation on SpiNNaker, and the final control module for real-time output generation.

network simulations using PyNN [87]. The PyNN scripts define network parameters, neuron models, and synaptic connections while simulating the SpiNNaker in real time. During preprocessing, morphological filtering is applied to reduce isolated noise events and enhance spatially coherent motion regions. Events are accumulated over a short 5 ms window to form a binary event surface. A 3×3 morphological opening operation (erosion followed by dilation) removes sparse noise, while a subsequent closing operation (dilation followed by erosion) fills small gaps within dense event clusters. This improves spatial consistency and reduces unnecessary spiking activity before neuromorphic processing. Concurrently, the board sends back spike data or other relevant outputs (e.g., tracking information) for logging and visualisation. This hardware arrangement supports real-time experimentation and interactive object selection, offering a robust platform for further development and testing of neuromorphic vision applications like multi-object tracking.

3.6.2 Brain-Inspired Attractor Dynamics for Multi-Object Tracking

Our multi-object tracking algorithm is biologically inspired by the spatial coding principles of hippocampal grid cells, which form robust representations of continuous space [343]. While grid-cell activity is primarily associated with self-location, these mechanisms provide a useful computational analogue for maintaining stable spatial representations under dynamic conditions, which we leverage to handle objects undergoing variable motion [202]. Initially, raw events are consolidated into a brief accumulation map (5ms), normalized, and then merged with a decaying memory map to create a heatmap of recent activity. By applying thresholding along with basic morphological operations (open/close) on this heatmap, we obtain clean, high-contrast regions corresponding to object locations [43]. A connected components analysis is then performed to extract bounding boxes and centroids for these regions. To track multiple targets, the system deploys several "trackers," each using a Kalman filter [225] to smooth short-term motion and predict future positions. Data association is achieved by measuring the Euclidean distance between new

detections and each tracker's current centroid. If a detection falls within a specified distance threshold, the corresponding tracker is updated; otherwise, the tracker's "miss count" increases, and if mismatches persist, that tracker is removed. The distance threshold was determined empirically by analysing centroid displacement statistics across consecutive frames on a validation sequence. The value was tuned to balance false associations and missed detections, ensuring robust performance under moderate motion variability. Sensitivity analysis confirmed that small variations in the threshold did not significantly affect tracking stability. When an unmatched detection occurs and there is an available ID slot, a new tracker is created to lock onto the target. Figure 3.6 illustrates an overview of our event-driven multi-object tracking pipeline.

For real-time dissemination of spatial information, each track's centroid is discretised onto a 32×24 grid, selected as a compromise between spatial resolution and communication efficiency. This resolution preserves sufficient localisation accuracy for downstream control while minimising memory usage and event traffic, which is critical for low-latency processing on neuromorphic hardware. These coordinates are then published via ZeroMQ, a high-performance, asynchronous messaging library designed for efficient and reliable real-time data exchange [402]. ZeroMQ enables communication between the tracking module running the neural simulation and external systems, facilitating seamless integration with visualisation interfaces, robotic controllers, or other downstream applications. The publish-subscribe messaging pattern implemented through ZeroMQ ensures low-latency updates, allowing subscribers to receive continuous, real-time tracking information without polling delays or performance bottlenecks. Additionally, a selective attention mechanism allows users to interactively toggle the tracking of individual or multiple objects by pressing digit keys. This mimics cortical spotlight attention, enabling the system to monitor multiple targets concurrently while prioritizing specific objects as required [403]. The combination of short accumulation intervals, memory-like decay, morphological filtering, Kalman-based smoothing, and interactive selection collectively produces stable attractor dynamics. These dynamics effectively manage diverse motion profiles,

including objects undergoing rapid movements or decelerations. Algorithm 2 outlines the complete multi-object tracking process.

3.6.3 Implementation of Neural Simulation on SpiNNaker

Building on this foundation, we explicitly model the continuous attractor dynamics that underpin the tracking layer. In computational neuroscience, an attractor network is characterised by recurrent connectivity that gives rise to stable activity patterns that persist even after the driving input is removed. To emulate this, the 32×24 grid of 768 leaky-integrate-and-fire neurons is wired with a local excitatory and long-range inhibitory topology (Mexican-hat profile). Each incoming centroid, injected as a brief spike packet, shifts the location of the activity bump to the new object position. The recurrent weights then maintain this bump in a quasi-stable equilibrium, providing temporal continuity whenever the event stream momentarily vanishes (for instance, when an object slows down or pauses). Incoming tracking data, parsed robustly via JSON, is used to compute neuron IDs using equation 3.8.

$$\text{neuron\_id} = \min\left(\left\lfloor \frac{y}{\Delta y} \right\rfloor,\, N_y - 1\right) \times N_x \quad + \min\left(\left\lfloor \frac{x}{\Delta x} \right\rfloor,\, N_x - 1\right) \tag{3.8}$$

where x and y are the object's coordinates, Δx and Δy represent the grid cell sizes, and Nx and Ny are the grid dimensions. The resulting neuron IDs and associated tracking errors are then logged to CSV files for later analysis. A dedicated spike injector population, connected one-to-one with the tracker neurons, ensures that each detected event triggers an appropriate spike, simulating the neural response to spatial stimuli. The simulation leverages the IF-curr-exp neuron model [87] on the SpiNNaker platform [137], interfacing with live spike connections that both send and receive events. The total number of spikes is used to estimate energy consumption. This integrated approach not only validates the attractor dynamics in a hardware-accelerated environment but also provides insight into the energy efficiency of neural computations for multi-object tracking.

Algorithm 2 Multi-Object Tracking with Event Accumulation and One-to-One Data Association

Init: Setup event camera, 5 ms accumulation map `accum`, decaying memory map `memory`, grid (32×24), a tracker set $\mathcal{T}$ (Kalman filters), and ZeroMQ.
for `each event` (x, y, p, t) **do**
    $\texttt{accum}[y, x] \leftarrow \texttt{accum}[y, x] + 1$
end for
$\texttt{memory} \leftarrow (1 - \lambda) \cdot \texttt{memory} + \lambda \cdot \text{normalize}(\texttt{accum})$ ▷ decaying integration
Generate heatmap via thresholding + morphological operations
Extract detections $\mathcal{D}$ (connected components) $\rightarrow$ bounding boxes and centroids
Initialise assignment set $\mathcal{A} \leftarrow \emptyset$
Initialise unmatched detections $\mathcal{D}_u \leftarrow \mathcal{D}$ and unmatched trackers $\mathcal{T}_u \leftarrow \mathcal{T}$
for `each tracker` $t \in \mathcal{T}$ **do**
    for `each detection` $d \in \mathcal{D}$ **do**
        $C[t, d] \leftarrow \sqrt{(x_t - x_d)^2 + (y_t - y_d)^2}$ ▷ cost matrix
    end for
end for
while $\mathcal{T}_u \neq \emptyset$ **and** $\mathcal{D}_u \neq \emptyset$ **do**
    $(t^\star, d^\star) \leftarrow \arg\min_{t \in \mathcal{T}_u,\, d \in \mathcal{D}_u} C[t, d]$
    if $C[t^\star, d^\star] <$ THRESHOLD **then**
        $\mathcal{A} \leftarrow \mathcal{A} \cup \{(t^\star, d^\star)\}$
        $\mathcal{T}_u \leftarrow \mathcal{T}_u \setminus \{t^\star\}$ ▷ enforce one-to-one
        $\mathcal{D}_u \leftarrow \mathcal{D}_u \setminus \{d^\star\}$
    else
        break ▷ remaining pairs are too far
    end if
end while
for `each assigned pair` $(t, d) \in \mathcal{A}$ **do**
    Update tracker t with detection d (Kalman update)
    $\texttt{miss}[t] \leftarrow 0$
end for
for `each unassigned tracker` $t \in \mathcal{T}_u$ **do**
    $\texttt{miss}[t] \leftarrow \texttt{miss}[t] + 1$ ▷ miss count is per tracker
end for
for `each unassigned detection` $d \in \mathcal{D}_u$ **do**
    if tracker slot available **then**
        Create new tracker for d and initialise $\texttt{miss} \leftarrow 0$
    end if
end for
Remove trackers with $\texttt{miss}[t] >$ MAX
Map surviving tracker centroids to grid cells and publish via ZeroMQ
Apply selective attention mechanism (on tracks or ROIs)
Overlay heatmap and tracker state for visualisation

3.6.4 Use Case: Swarm Robots Evasion

To demonstrate the real-world potential of our multi-object tracking framework, we developed a use case centered on swarm robot evasion. In this scenario, a group of robots navigates an environment while actively avoiding collisions. The neuron spikes produced by the SpiNNaker platform serve as indicators of obstacles that the robots must avoid, with these spikes spatially corresponding to the object locations detected by the event camera. Each robot continuously monitors its adjacent cells and, upon detecting a spike, executes an evasion maneuver by selecting the neighbouring cell that minimizes the risk of collision. This approach enables the swarm to dynamically adapt in real time, ensuring rapid responses to swiftly changing target positions. Performance is assessed using metrics such as response time and collision avoidance rate during obstacle evasion.

3.6.5 Experimental Setup

Experiments were conducted using synthetic objects rendered on an LCD display to ensure controlled and repeatable motion patterns. The CenturyArk SilkyCam VGA event camera was positioned directly in front of the screen to capture asynchronous brightness changes generated by object movement. The captured events were transmitted to the interface machine, where spatial coordinates were extracted and forwarded to the SpiNNaker platform. These coordinates were mapped onto neurons arranged in a spatial grid. In the swarm robotic evasion scenario, each moving object was interpreted as a dynamic obstacle, and the corresponding neuronal activation updated its position within the grid for neuromorphic decision-making.

3.6.6 Results and Discussion

We assess our multi-object tracking system using a variety of performance metrics. By harnessing the strengths of event-based vision sensors and neuromorphic hardware, the system exhibits notable enhancements in tracking accuracy, latency, and energy efficiency.

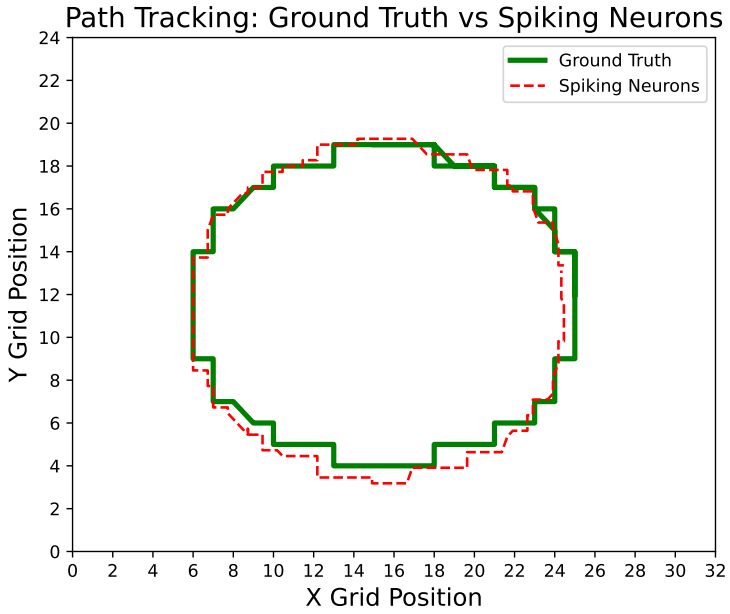


Figure 3.7: Path tracking comparison between ground truth and spiking neurons.

Path Tracking Performance

In the initial experiments, we evaluated path-tracking accuracy by comparing the estimated object centroids produced by the neuromorphic tracking pipeline against ground-truth trajectories obtained from manually annotated event streams. Objects were moved across the field of view at varying speeds and directions, and tracking error was computed as the Euclidean distance between predicted and true positions in grid coordinates at each time step. As illustrated in Figure 3.7, the spiking neurons track object motion with minimal error, remaining below 1.15 grid units across all tested conditions. This performance exceeds that reported in prior event-based tracking approaches, where errors typically grow under dynamic motion [412, 228]. These results demonstrate that the integration of event-driven sensing with neuromorphic processing enables robust real-time tracking.

Raster Plots Analysis

Figure 3.8 illustrates the system's temporal behaviour during the tracking process. In Figure 3.8(a), the initial state is depicted, where neurons fire in a random pattern. Figure 3.8(b) displays neuronal activity during the tracking of three objects, with neurons firing selectively in response to object detection and remaining inhibited when

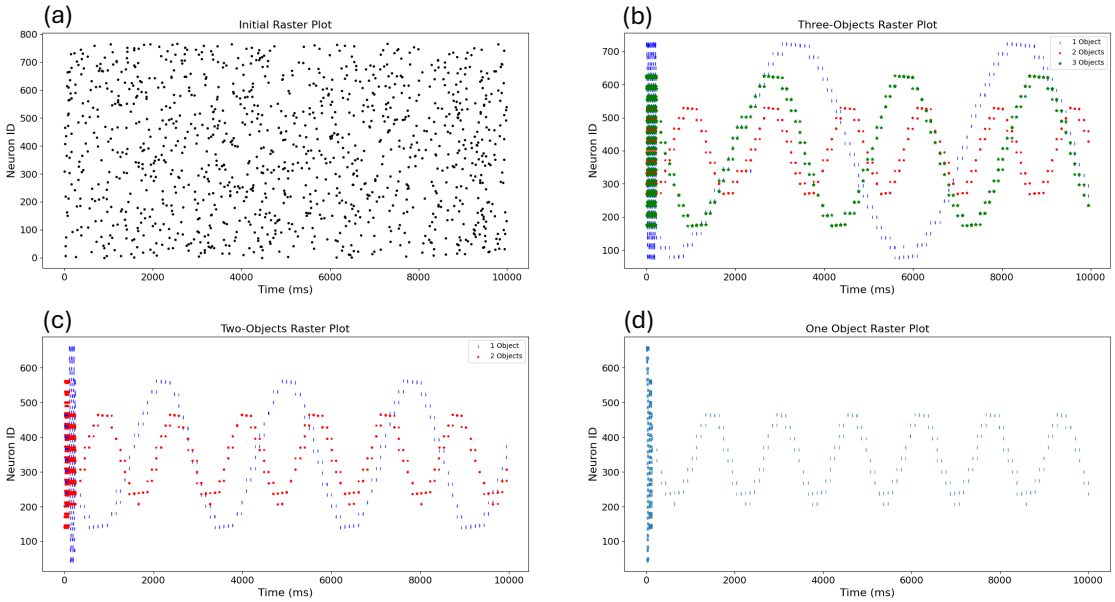


Figure 3.8: Raster plot showing the neuron activity during multi-object tracking. Panels (a) show the initial state of neurons when no object is being tracked. (b) displays neuron activity for 3 objects being tracked, while Panels (c) and (d) show attention-based tracking for two and one object, respectively.

objects are absent. Figures 3.8(c) and (d) demonstrate attention-based tracking for two and one object, respectively. This selective tracking mechanism enables dynamic control, allowing the system to concentrate on specific objects within the field of view and showcasing the versatility of the attention mechanism.

Performance of the Multi-Object Tracking Algorithm

Figure 3.9 demonstrates the effectiveness of the multi-object tracking algorithm running on the SpiNNaker platform. In our experiments, three objects were simulated to revolve around a central point at varying speeds, with an event camera capturing the entire simulation. Row 1 of Figure 3.9 displays a series of frames from the simulation, accompanied by the raw events generated by the objects' movement (row 2). The event-based tracking system processes these events in real time, as evidenced by the Tracking panel, which illustrates all objects being tracked concurrently (row 3).

In the Selective Tracking scenario (row 4 of Figure 3.9), the system demonstrates its ability to focus on a specific object by selecting it for tracking. This feature

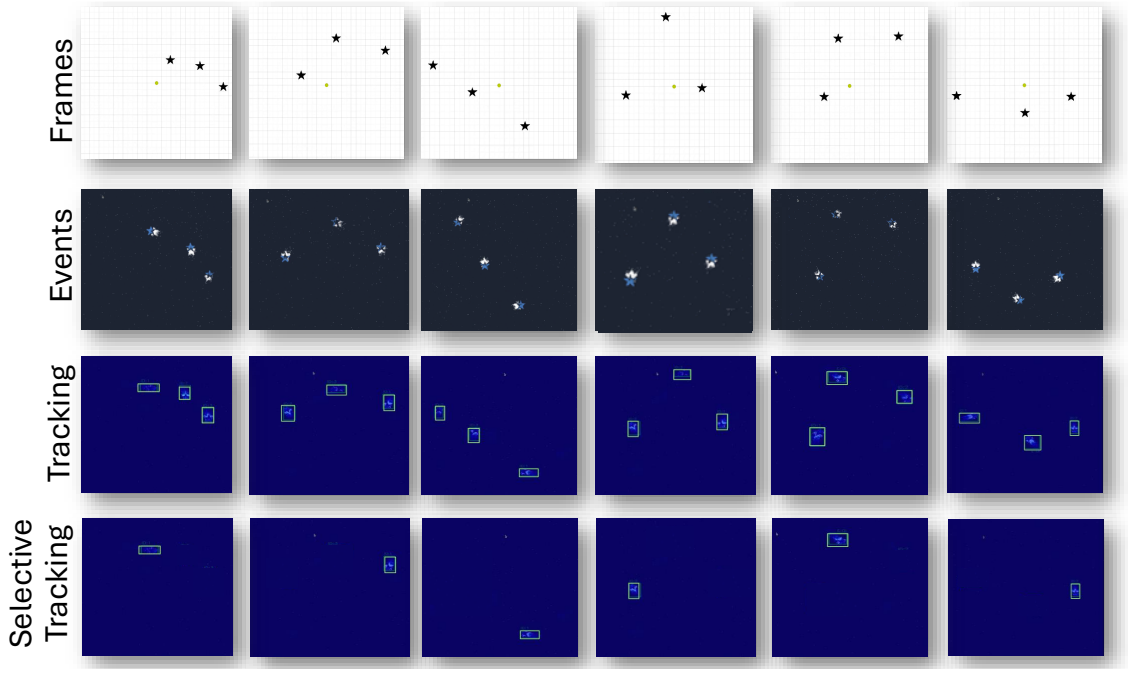


Figure 3.9: Visualisation of the tracking process. The top panel shows event frames generated by the objects, followed by the full tracking of all objects. The bottom panel illustrates attention-based selective tracking, where only the selected object is tracked.

enables dynamic selection and deselection of targets based on their ID, ensuring that only the relevant object is monitored at any given moment. The capacity to filter out distractions and concentrate on a single object further enhances the system's flexibility and performance in real-world applications.

Energy Consumption Analysis

We evaluated the energy and power consumption of our multi-object tracking system under scenarios with 1, 2, and 3 objects moving at the same speed. Energy usage was estimated by counting the number of spikes generated, with each spike incurring an energy cost of roughly 8 nJ per event, as noted by Stromatias et al. [346] for the SpiNNaker 48-chip board. The difference in power between the idle state and active simulation is about 3.25 W, which reflects the additional energy required during the multi-object tracking simulation. The data presented in Table 3.7 reveal that energy consumption increases proportionally with the number of objects tracked, due to the higher number of neuron spikes in multi-object scenarios. The average power is calculated by dividing the total energy by the 10-second simulation duration.

Table 3.7: Energy and Power Consumption for Multi-Object Tracking with SpiNNaker.

Objects	Energy (uJ)	Avg. Power (uW)
1	16.512	1.6512
2	33.776	3.3776
3	53.936	5.3936

Performance for Swarm Robots Evasion Use Case

This use case was evaluated in a controlled experimental environment in which multiple mobile robots were simulated within a two-dimensional arena containing dynamic obstacles generated from real event-camera tracking data. Object positions were extracted in real time by the event-based tracking pipeline and encoded as spiking activity on the SpiNNaker platform, with each active neuron representing an occupied spatial grid cell. These spike signals were transmitted to the swarm controller, where each robot continuously monitored neighbouring cells to trigger evasive manoeuvres when obstacles entered a predefined safety radius.

The experimental scenarios included zero to three moving obstacles with varying trajectories and speeds, as illustrated in Figure 3.9. For each configuration, robots were allowed to navigate for fixed-duration trials while collision events and response delays were recorded. Performance was quantified using (i) average response time, defined as the interval between spike generation and the onset of evasive motion, and (ii) collision avoidance rate, defined as the proportion of time during which robots maintained a minimum safe distance from obstacles and each other. Across all trials, the system achieved an average response time of 28.5 ± 5.2 ms and a collision avoidance rate of 94.3%, demonstrating effective real-time adaptation driven by neuromorphic perception.

Figure 3.10 further illustrates these results. Panel (a) shows the robots navigating an obstacle-free environment, while panels (b), (c), and (d) display scenarios with increasing numbers of obstacles. These images underscore the robots' ability to dynamically modify their paths based on real-time information. Overall, the low response time and high collision avoidance rate confirm the robustness and practicality

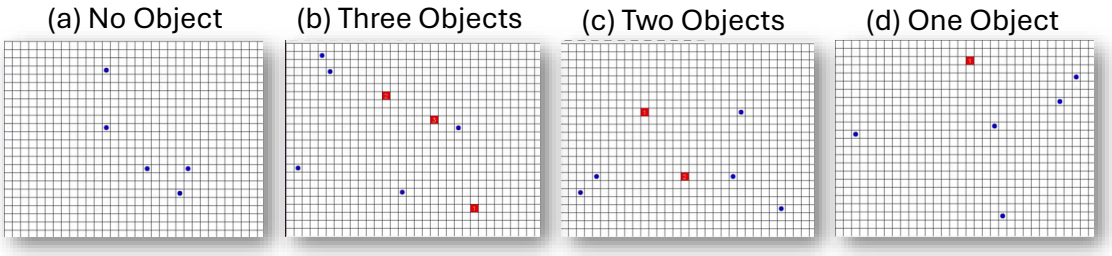


Figure 3.10: Swarm robot evasion performance in different obstacle scenarios. Panel (a) shows no objects, while Panels (b), (c), and (d) display evasion with three, two, and one objects, respectively.

of the multi-object tracking system for real-time robotic control in complex, dynamic settings.

3.7 Summary

This chapter explored the deployment of SNN models on neuromorphic hardware, focusing on learning mechanisms, attention-based processing, and ANN-to-SNN conversion methods. It provided a comparative analysis of different conversion strategies, emphasizing their impact on accuracy, energy efficiency, and computational cost. The discussion also covered the integration of SNNs with SpiNNaker hardware and event-based vision systems, demonstrating their potential for real-time applications. Furthermore, the chapter addressed real-world implementation challenges and offered practical insights into optimising neuromorphic computing systems. Overall, it showcased how advanced SNN architectures and dynamic attention mechanisms can drive the development of energy-efficient, high-performance solutions. While this chapter explored the deployment of spiking neural networks (SNNs) on neuromorphic hardware, efficient processing of event-based data is crucial for real-world applications. Event-based vision, a key advancement in neuromorphic computing, offers improved efficiency in dynamic environments. The next chapter examines the processing of event data, explores various formats, and discusses its applications in gesture recognition and industrial monitoring.

Chapter 4

Applications of Event-Based Vision

4.1 Introduction

This chapter discusses the applications of event-based vision, focusing on its role in interactive robotics and real-time data processing. The chapter first covers methods for processing event-based data, addressing format diversity and standardization challenges. It then explains real-time gesture recognition for robotic guidance and event-camera-based rotational frequency analysis for machinery fault diagnosis.

Novelty & Impact

Novelty: Introduces an efficient data pipeline for processing event-based vision data, along with novel applications in gesture recognition and vibration-based machinery fault diagnosis.

Impact: Enhances real-time robotic perception capabilities, demonstrating significant improvements over traditional vision systems in dynamic and challenging environments.

4.2 Processing Event-based Data

With time, event cameras can be found in various fields such as robotics, autonomous vehicles, and augmented reality [136]. However, the proliferation of event-based vision datasets presents challenges related to their diverse formats and characteristics.

Researchers often encounter obstacles when processing and analysing these datasets efficiently. These challenges impede the comparability and reproducibility of research findings in different studies. To address these issues, several libraries and tools have been developed to process event data, including AERmanager [140], aedat [10], tonic [222], and spikingJelly [118]. Tonic and SpikingJelly are widely used tools for processing spiking neural network data and conducting simulations within the neuromorphic vision community. However, both tools have notable limitations when applied to event-based vision datasets. Tonic, for instance, primarily provide support for AEDAT format and lacks the capability to handle newer event vision sensor data formats, such as EVT2 and EVT3, which are crucial for emerging dynamic vision sensors such as IMX636 and EVc3a by Prophesee and CenturyArks. This restricts its applicability when working with advanced sensors that use proprietary formats. SpikingJelly, while effective for spiking neural network simulations, is not optimised for real-time event-based vision tasks. Its higher computational overhead results in longer processing times, particularly for large-scale datasets. These limitations underscore the need for a more versatile and efficient data processing pipeline.

In response to this gap, our research introduced a comprehensive pipeline, completed in early 2024, designed to support a wide range of event data formats, including those from emerging sensor technologies. By providing native compatibility with newer formats such as EVT2 and EVT3, our pipeline improves accessibility and scalability within the event-based vision research community. Since early 2024, complementary community efforts have emerged, including the Faery library, a Rust-and-Python framework initiated at the Telluride Neuromorphic Workshop 2024 for efficient event data streaming, parsing, and conversion [188].

4.2.1 Various Data Formats

Event-based vision technology employs multiple data formats to represent, store, and process asynchronous event streams efficiently. Among them, EVT2 and EVT3 formats leverage delta encoding [354] to optimise storage and transmission, differing in encoding methods, file structures, and software compatibility, particularly for

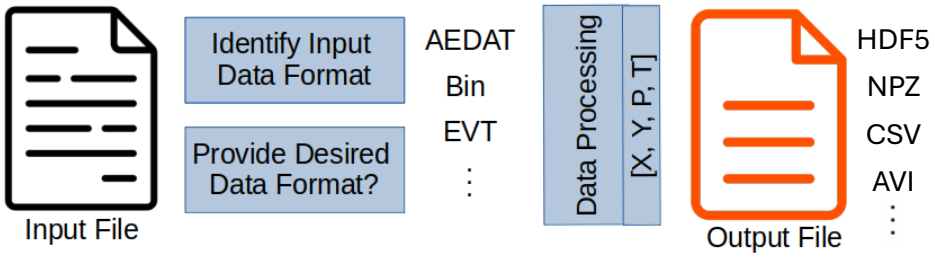


Figure 4.1: Schematic Overview of Data Format Conversion and Processing Pipeline.

Prophesee's vision sensors [305]. Similarly, the AEDAT series (AEDAT 2, 3.1, and 4) plays a crucial role in neuromorphic vision, evolving from capturing basic address-event data in AEDAT 2 to supporting complex metadata and diverse sensor types in AEDAT 3.1, culminating in AEDAT 4's scalability for modern neuromorphic systems. The Bin data format is a compact binary storage method, structuring raw event data (coordinates, timestamps, and polarity) for efficient retrieval and processing. In contrast, the CSV format offers a simple, human-readable structure where each event is represented with x, y coordinates, polarity (ON/OFF), and timestamp, enabling straightforward manipulation in spreadsheet applications or programming environments. The HDF5 format provides hierarchical organization, metadata support, compression, and parallel I/O, making it well-suited for handling large event-based datasets in scientific computing and machine learning workflows. The NPZ format (NumPy Zip) efficiently stores event data arrays, compressing large datasets while maintaining portability and compatibility with Python-based tools. Lastly, the AVI format, though primarily a multimedia container, is used in event-based vision applications to store video data with synchronized playback, typically employing lossy compression to balance quality and file size.

4.2.2 Event Data Processing Methodology

Several methodologies were employed for processing data stored in AEDAT, binary (bin), and EVT formats. Specifically, we explain the algorithms designed to efficiently handle and extract information from these diverse data structures. Our approach encompasses data parsing techniques tailored to each format's unique specifications,

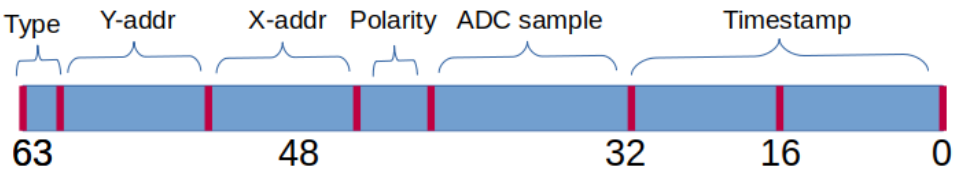


Figure 4.2: Bit-level layout of the 64 bit AEDAT event packet, highlighting the Type flag, Y address, X address, polarity, ADC sample, and 16 bit timestamp fields.

ensuring accurate extraction and conversion into standardised event formats for downstream analysis and interpretation.

Figure 4.1 outlines a streamlined data processing pipeline. It begins with an input file, which is then passed through a format identification stage where the input data format is determined. Following this, there's a decision point where the desired data format (like AEDAT, Bin, EVT, etc.) is selected. After the format is provided, the data undergoes processing, resulting in an output file with a specified format (X, Y, P, T) indicative of the data's spatial (X, Y) and temporal (P, T) attributes.

Algorithm 3 Load AEDAT v3 File

```
1:  procedure LOAD_AEDAT_V3(file_name)
2:      Initialize txyp ← {'t' : [],' x' : [],' y' : [],' p' : []}
3:      Open file_name and skip ASCII header
4:      while reading data from file do
5:          Read and unpack event header
6:          if event type is Polarity event then
7:              while reading event data do
8:                  Unpack event data into x, y, t, p
9:                  Append x, y, t, p to respective lists in txyp
10:             end while
11:         end if
12:     end while
13:     Convert lists in txyp to numpy arrays
14:     return txyp
15: end procedure
```

The AEDAT data format follows a structured 64-bit layout that encodes temporal details, spatial coordinates, and type identifiers for each event. The first bit differentiates between DVS and APS event types, followed by 15 bits allocated to the y-coordinate and another 15 bits to the x-coordinate. A 16-bit field serves multiple functions, either storing the ADC sample for APS events or specifying the read

type for DVS events, such as reset, signal, or IMU reads. The format concludes with a 16-bit timestamp field. Algorithm 3 is designed to process data from an AEDAT v3 file, which contains event streams generated by neuromorphic sensors. It begins by initializing a dictionary named *txyp*, with keys corresponding to time (t), x-coordinate (x), y-coordinate (y), and polarity (p). After opening the file and skipping the ASCII header, the algorithm iterates through the dataset. For each event, it unpacks the event header and verifies whether it is a polarity event. If so, it proceeds to extract the relevant data x-coordinate, y-coordinate, timestamp, and polarity before appending them to the respective lists within the *txyp* dictionary. Once the file has been fully processed, these lists are converted into NumPy arrays, which are then returned. Figure 4.2 illustrates the bit-level layout of the 64-bit AEDAT event packet.

Algorithm 4 Process Binary File

1: **procedure** LOAD_ATIS_BIN($file\_name$)
2: **Input:** Path of the ATIS binary file $file\_name$
3: **Output:** A dictionary with keys $\{'t', 'x', 'y', 'p'\}$ and values as numpy arrays
4: ▷ Each event in the binary file consists of X address, Y address, Polarity, and Timestamp
5: Open $file\_name$ in binary mode
6: Read raw data from the binary file
7: Extract X address, Y address, Polarity, and Timestamp from the raw data
8: **return** $\{'t' : t, 'x' : x, 'y' : y, 'p' : p\}$
9: **end procedure**

Algorithm 4 is designed to handle ATIS (Asynchronous Time-based Image Sensor) binary files, which are commonly used in neuromorphic event-based cameras. It takes an ATIS binary file path as input and outputs a dictionary where the keys represent timestamps (t), x-coordinates (x), y-coordinates (y), and polarities (p). Each event in the binary file contains data related to X address, Y address, Polarity, and Timestamp. The algorithm begins by opening the specified file in binary mode and reading the raw data. It then extracts the relevant components X address, Y address, Polarity, and Timestamp from the raw data stream. After extracting these values, the algorithm structures them into a dictionary, ensuring that each key corresponds to its respective NumPy array containing the processed data. This

process allows for efficient parsing of ATIS binary files, facilitating further analysis and event-based camera data processing.

Algorithm 5 EVT3.0 Data Processing

```
Initialize input and output files for CD events
if trigger output file provided then
    Initialize output file for trigger events
end if
Skip input file header if present
while input file has data do
    Read a batch of data from input file into buffer
    Initialize state variables for decoding
    for each raw event in the buffer do
        Determine the type of the raw event
        Process the raw event based on its type
        if address X event then
            Extract x, y, polarity, and timestamp
            Convert event to XYPT format
        else if vector event (12 or 8 bits) then
            Determine validity of events in the vector
            Extract x, y, polarity, and timestamp information
            Convert valid events to XYPT format
        end if
    end for
    Write processed CD events to CD output file
    if trigger output file provided then
        Write processed trigger events to trigger output file
    end if
end while
```

The EVT3 format represents timestamp information using a 24-bit structure, where the 'Time High' and 'Time Low' fields, each consisting of 12 bits, are combined to reconstruct the full 24-bit timestamp. This structure provides precise temporal resolution, which is essential in event-based vision systems where accurate timing is crucial for interpreting dynamic visual information. Algorithm 5 describes the process for handling EVT3.0 data, focusing on efficient parsing and conversion of raw EVT3 events into a standardised CD event format. Additionally, the algorithm supports the optional processing of trigger events, offering a flexible approach for applications that require both types of event data. The process begins with initializing input and output files for CD events, while a separate output file is created if trigger events

need to be processed. Any headers in the input file are skipped to directly access the event data.

The EVT3.0 data is processed in batches to optimise memory usage, particularly when dealing with large datasets. Each batch is loaded into a buffer, where individual raw events are decoded to determine their type. The algorithm differentiates between Address X events spatial events containing x, y coordinates, polarity, and timestamp and vector events, which encode multiple events using either 12-bit or 8-bit structures. For Address X events, the x and y coordinates, polarity (indicating intensity changes in light), and timestamp (constructed from 'Time High' and 'Time Low') are extracted and formatted into the XYPT structure to ensure compatibility with standard event data analysis tools. Vector events are handled by verifying the validity of each event before extracting the relevant x, y, polarity, and timestamp details. This step ensures that only meaningful events are processed, reducing unnecessary noise and improving the overall accuracy of data interpretation. Once all events in a batch have been processed, the resulting CD events are stored in the output file. If trigger events are included, they are written separately to maintain clarity between different event types. The use of batch processing, along with efficient parsing techniques, allows the algorithm to manage large-scale datasets effectively while maintaining processing speed and accuracy.

Table 4.1: Comparison with existing solutions

	Npz	Aedat 2	Aedat 3.1	Aedat 4	Bin	Csv	EVT 2	EVT 3
Aermanager	✓	x	✓	✓	✓	x	x	x
Aedat	x	x	x	✓	x	x	x	x
Tonic	✓	✓	✓	✓	✓	✓	x	x
SpikingJelly	✓	✓	✓	x	✓	✓	x	x
This Work	✓	✓	✓	x	✓	✓	✓	✓

One of the main strengths of the proposed pipeline is its ability to support multiple event-data formats, including newer formats like EVT2 and EVT3, which are not widely compatible with existing frameworks such as Tonic and AERmanager. While Tonic is primarily tailored for spiking neural networks, the proposed pipeline is designed to efficiently handle both event-based vision data and conventional neuro-

Table 4.2: Data handling capabilities of proposed pipeline

	Aedat2	Aedat3.1	Bin	Npz	Csv	EVT2	EVT3
Aedat 2	-	x	✓	✓	✓	✓	✓
Aedat 3.1	x	-	✓	✓	✓	✓	✓
Bin	✓	✓	-	✓	✓	✓	✓
Npz	✓	✓	✓	-	✓	✓	✓
Csv	✓	✓	✓	✓	-	✓	✓
EVT 2	✓	✓	✓	✓	✓	-	x
EVT 3	✓	✓	✓	✓	✓	x	-

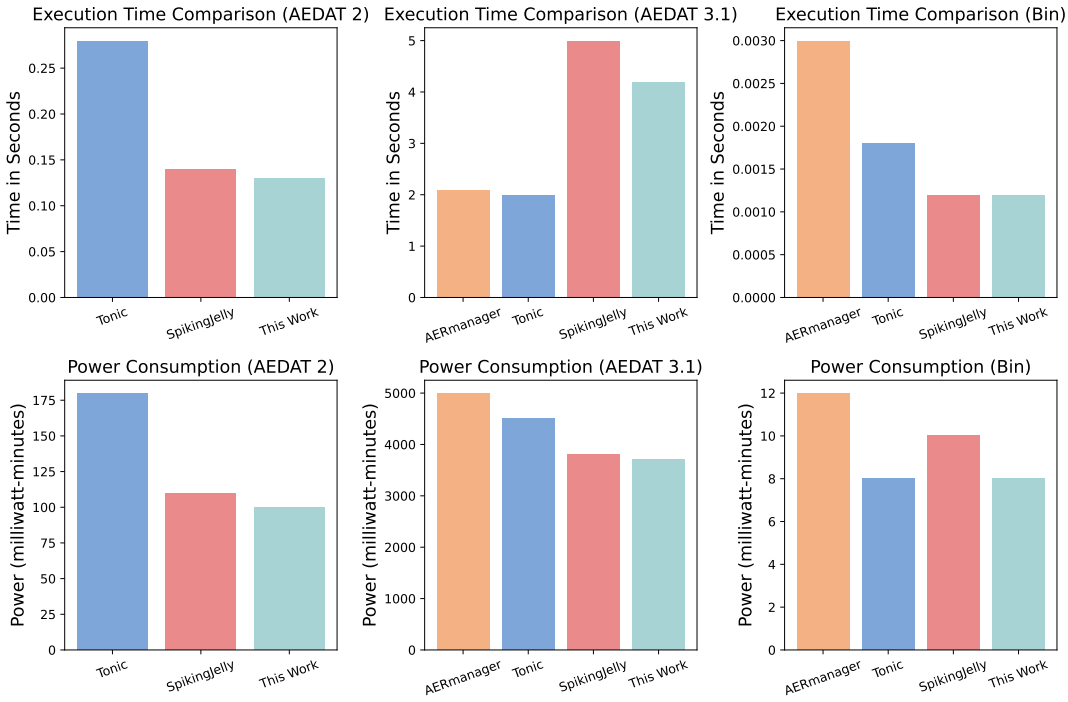


Figure 4.3: The figure shows the execution time and estimated power consumption for processing DVS data across various libraries/frameworks. Each framework's performance is benchmarked using consistent parameters on identical hardware and system conditions to ensure a fair comparison. The results show that the proposed pipeline outperforms the existing libraries/frameworks on several occasions.

morphic datasets, making it applicable to a broader range of use cases. Additionally, the pipeline incorporates a more efficient encoding and decoding algorithm for EVT data, enabling faster execution times compared to AERmanager. Table 4.1 presents a comparison between the proposed pipeline and existing libraries/frameworks, highlighting that none of the existing solutions currently support EVT2 or EVT3 data formats. Table 4.2 shows the data-handling capabilities of our proposed pipeline.

Experiments were conducted using a standardised benchmarking setup to assess execution time and power consumption. The tests were performed on a system featuring an Intel i7-9700K CPU, 16GB RAM, and an Nvidia RTX 2080 GPU. The software environment included Python 3.8, along with data processing libraries such as NumPy and Tonic. Power consumption was measured using Intel Power Gadget, while execution times were recorded with Python's built-in time module. For each dataset format, identical samples were processed and results were compared with those obtained from AERmanager and SpikingJelly. Each test was repeated three times, with the average value recorded to ensure reliability. Figure 4.3 illustrates the execution time and power consumption comparison across different frameworks, including AERmanager [298], Tonic [222], SpikingJelly [118], and the proposed pipeline, for three data formats: AEDAT 2, AEDAT 3.1, and Bin. In AEDAT 2 processing, the proposed pipeline demonstrates the shortest execution time and lowest power consumption. For AEDAT 3.1, it also achieves the fastest execution time, though the power efficiency advantage is less significant. In the Bin format, the proposed pipeline performs competitively in terms of execution time while ranking second in power efficiency.

4.2.3 Selected Open-Source Datasets

We employed the proposed data processing pipeline to convert several publicly available, open-source datasets into multiple formats. The datasets selected for this process are summarized in Table 4.3, which presents a comparison based on data format, size, and licensing. The pipeline was applied uniformly across these datasets, resulting in converted versions consistent with the formats described in the preceding

Table 4.3: Selected event-based datasets that were converted into multiple formats using the proposed data processing pipeline.

Index	Dataset	Recognition Type	Format	# Samples	Size	License	Ref
1	NMNIST	Digit	Binary	70,000	1.2 GB	CC 4.0	[286]
2	N-Caltech101	Object	Binary	9,146	4.0 GB	CC 4.0	[194]
3	Bullying10k	Action	Binary	10,000	47.5 GB	CC 4.0	[99]
4	MNIST-DVS	Digit	AEDAT	60,000	3.72 GB	CC 4.0	[332]
5	CIFAR10-DVS	Image	AEDAT	10,000	8.4 GB	CC 4.0	[224]
6	DVS128Gesture	Gesture	AEDAT	1,342	2.9 GB	Open-source	[119]

section. As all datasets are distributed under the CC BY 4.0 license, the transformed versions may be publicly released to facilitate and support future research efforts.

The N-MNIST dataset consists of 60,000 training and 10,000 testing samples, capturing event-based representations of handwritten digits at the pixel level. Each sample is provided in a binary format, encoding timestamps and pixel locations, making it widely used in neuromorphic vision research [286]. The N-Caltech101 dataset is an event-based adaptation of Caltech101, containing images from 40 object categories. It is commonly used to evaluate event-based object recognition models, with each sample encoding a visual scene in binary format [194]. The Bullying10K dataset is a neuromorphic dataset designed for privacy-preserving bullying detection. It contains over 10,000 labelled instances, providing valuable ground truth for algorithmic development while emphasizing privacy concerns [99]. The MNIST-DVS dataset extends the MNIST dataset for event-based vision, capturing handwritten digits with Dynamic Vision Sensor (DVS) cameras. Unlike N-MNIST, which moves the camera while keeping the digit stationary, MNIST-DVS moves the digit on-screen while keeping the camera fixed. This difference in capture methodology influences event dynamics and dataset applicability in neuromorphic research [331]. The CIFAR10-DVS dataset is an event-based adaptation of CIFAR10, where static RGB images are converted into event representations that capture pixel changes over time. This dataset enables the evaluation of event-based image classification algorithms and is widely used in neuromorphic computing research [224]. Finally, the

DVS128 Gesture dataset, captured using an IBM DVS128 event camera, consists of gesture sequences performed by different individuals. Stored in AEDAT 3.1 format, the dataset encodes x and y coordinates, timestamps, and polarity, supporting event-based gesture recognition tasks [119]. More details on the event data processing pipeline can be found on [10].

In the following two sections (Section 4.3 and Section 4.4), the proposed event data processing techniques are applied to distinct real-world applications. The first application involves real-time gesture recognition aimed at enhancing human-robot interaction, while the second focuses on real-time machinery fault diagnosis using event-based camera data.

4.3 Event Camera-Based Real-Time Gesture Recognition for Improved Robotic Guidance

Gesture recognition offers a natural and intuitive means for humans to interact with robots, holding both theoretical significance and practical applications in Human-Robot Interaction (HRI). As gesture recognition technology progresses, its role in HRI continues to expand. However, collecting gesture data using conventional RGB cameras, which typically operate at 30-60 frames per second (fps), presents challenges, particularly in capturing fast gestures. Motion blur often degrades recognition accuracy [362], making reliable gesture detection difficult. A common approach to mitigate motion blur is increasing the frame rate of standard cameras, but this generates excessive static and redundant information across continuous frames while also capturing unnecessary background details. Additionally, frame-based cameras struggle in extreme lighting conditions, such as overly bright or dark environments, which are common in HRI scenarios where robots must recognize human gestures for effective interaction. To address these limitations, event cameras provide a more efficient and adaptable solution. The event cameras offer a microsecond time resolution, ensuring the smooth capture of gesture motion without being constrained by exposure time or frame rate [344]. Additionally, they perform well in both well-lit

Figure 4.4: Block diagram of the proposed system. Starting with data collection to real-time gesture recognition with an event camera mounted on a robot.

and dimly-lit environments due to their high dynamic range capabilities (see section 1.2 for more information regarding event cameras). Since their introduction, event cameras have found applications in various scenarios within the fields of computer vision and robotics [148][189]. This work primarily explores a real-time gesture recognition method based on event camera technology.

In this study, we developed a real-time hand gesture recognition system. The process began with collecting dynamic gesture data using the CenturyArk SilkyCam VGA event camera. The raw event data, captured in EVT3.0 format, consisted of asynchronous streams containing pixel coordinates, timestamps, and polarity values. To prepare this data for model training, we first converted the continuous event stream into event frames by aggregating events over fixed temporal windows (10 ms). Each window was transformed into a two-dimensional frame representing positive and negative events as separate channels. Following this, we carefully annotated the gesture instances by manually identifying their start and end points, ensuring precise ground-truth labels. The labelled sequences were then converted into the HDF5 format to facilitate efficient storage and batch processing during training. We subsequently trained and fine-tuned a classification model using a ConvRNN-based architecture [46][305], which is well-suited for spatiotemporal data. Finally, we integrated the event camera with a humanoid robot to perform real-time experiments. The event camera demonstrated robust performance under diverse lighting conditions, highlighting its effectiveness compared to standard RGB cameras, which often suffer from motion blur or low-light failures. Figure 4.3 presents the block diagram of the proposed system.

While extensive research has explored RGB and RGB-D-based gesture recognition [301], Convolutional Neural Networks (CNNs) have emerged as a dominant approach due to their ability to extract hierarchical features and capture complex hand

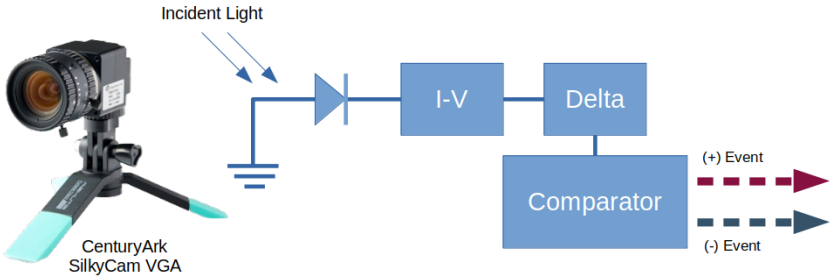


Figure 4.5: CenturyArk SilkyCam VGA event camera alongside a simplified pixel circuit that converts incident light into current, performs delta conversion, and uses a comparator to generate positive and negative polarity events.

gesture patterns [40][203][338]. Deep learning advancements have enabled high-accuracy gesture recognition in applications like sign language interpretation, human-computer interaction, and virtual reality [279]. Event-based approaches have also been explored, notably in a 2017 study by Amir et al. [21], where a TrueNorth neurosynaptic processor was used to process Dynamic Vision Sensor (DVS) data for energy-efficient, real-time gesture recognition. However, limitations persist in robotics applications, particularly due to the lack of high-resolution hand gesture datasets and the restricted availability of neuromorphic processors like SpiNNaker [254], Loihi [373], and TrueNorth [15]. Additionally, no existing system has utilised an event camera for real-time robot control, highlighting a significant research gap.

To address these challenges, a high-resolution (640x480) hand gesture dataset is introduced and we leverage it to train a ConvRNN model, achieving strong recognition performance. By integrating an event camera into a robotic platform, we enable real-time gesture recognition for precise robot control. The SilkyCam VGA event camera, developed by CenturyArk, was used for data acquisition, supported by Prophesee's Metavision software stack [63]. Figure 4.5 illustrates the event camera and its circuit diagram, where light-induced voltage changes trigger event generation upon exceeding a defined threshold. This approach advances the field by providing a commercially viable, high-resolution event-based gesture recognition system for robotic applications.

As explained in section 4.2, the standard notation used to represent an event is:

$$e = [x, y, t, p]$$

Here, event e signifies that the pixel situated at coordinates [x, y] within the pixel array of the event camera emitted an event in response to an illumination change at time t. The polarity attribute is encoded as p=[0, 1], with p=1 denoting an ON event and p=0 representing an OFF event. It is noteworthy that these events are transmitted at a temporal resolution of 1 µs [300], and the data rate of events depends on the rate of illumination changes occurring in the scene. This event camera saves data in EVT2.0/3.0 format.

4.3.1 EB-HandGesture Dataset

Currently, most event-based gesture datasets are either derived from standard camera datasets or created with low-resolution event cameras. Datasets such as N-MNIST, N-Caltech101 [286], CIFAR10-DVS [224], MNIST-DVS [331], and N-ImageNet [201] are generated by capturing event streams while moving an event camera around monitors displaying images from well-known datasets like MNIST [121], Caltech101 [194], and ImageNet [92]. Yihan Lin et al. [235] introduced ES-ImageNet, an event-stream version of ImageNet, by applying the Omnidirectional Discrete Gradient (ODG) algorithm to convert the original dataset. A range of event-based datasets are publicly available, as outlined in recent surveys [136]. This study focuses on datasets relevant to recognition tasks. While datasets such as HARDVS [387], Daily Action [239], and Bullying10k [99] capture human actions, others like IBM-DVS128 Gesture [21], ASL-DVS [44], and Nav-DVS [252] are specifically designed for hand gesture recognition. Common event cameras used for dataset collection include DAVIS128 (128x128), DAVIS240 (240x180), DAVIS346 (346x260) [270], and ATIS (302x245) [76]. The resolution of these cameras plays a crucial role in dataset quality, as higher-resolution sensors provide greater detail, improving object tracking, motion analysis, and scene reconstruction. Even with short accumulation times, high-resolution event cameras capture sufficient information for gesture recognition.

With ongoing advancements, newer high-resolution event cameras are now available [300], posing challenges for researchers relying on older, low-resolution datasets. For example, CenturyArk's SilkyCam VGA, a modern event camera with a resolution

Table 4.4: Comparison of Datasets

Dataset	Year	Type	Data	Sensor	Reso-. lution	Sec./ Inst.	Samp.	Ref.
CIFAR10-DVS	2017	Reproduced	Images	DAVIS128	128x128	1.2s	10k	[224]
MNIST-DVS	2013	Reproduced	Digit Images	DAVIS128	128x128	2-3s	30k	[331]
N-MNIST	2015	Reproduced	Digit Images	ATIS	28x28	0.3s	70k	[286]
N-CALTECH101	2015	Reproduced	Images	ATIS	302x245	0.3s	8.7k	[286]
ES-ImageNet	2021	Converted	Images	-	224x224	-	1.3M	[235]
N-ImageNet	2021	Reproduced	Images	Samsung Gen3	480x640	-	1.7M	[201]
HARDVS	2022	Event-Based	Action	DAVIS346	346x260	5s	100k	[387]
Daily Action	2021	Event-Based	Action	DAVIS346	346x260	5s	1.4k	[239]
Bullying 10K	2023	Event-Based	Action	DAVIS346	346x260	2-20s	10k	[99]
ASL-DVS	2019	Event-Based	Hand Action	DAVIS240	240x180	0.1s	100k	[44]
Nav-DVS	2020	Event-Based	Hand Action	ATIS	302x245	-	1.3k	[252]
DVS128 Gesture	2017	Event-Based	Hand Action	DAVIS128	128x128	6s	1.3k	[21]
EB-HandGesture	2024	Event-Based	Hand Action	SilkyCam Gen3	640x480	0.5s	9k	This work

of 640x480, only supports input resolutions divisible by this format (e.g., 320x240, 160x120). However, one of the widely used event-based hand gesture datasets, IBM-DVS128 Gesture, was recorded with the iniVation DVS128 camera at a resolution of 128x128. This resolution disparity means that even a well-trained model on IBM-DVS128 Gesture would be incompatible with SilkyCam VGA for real-time inference, highlighting the need for updated, high-resolution datasets to match the capabilities of modern event cameras.

We present the EB-HandGesture dataset, the first high-resolution hand gesture dataset captured using the CenturyArk SilkyCam Gen3.0 (640x480). This camera is equipped with a Prophesee event-based vision sensor, offering a temporal resolution of $1\mu s$. Data collection was conducted using the Prophesee Metavision SDK and OpenEB framework [305]. The dataset includes ground-truth annotations, providing gesture labels along with their corresponding start and stop times, which were

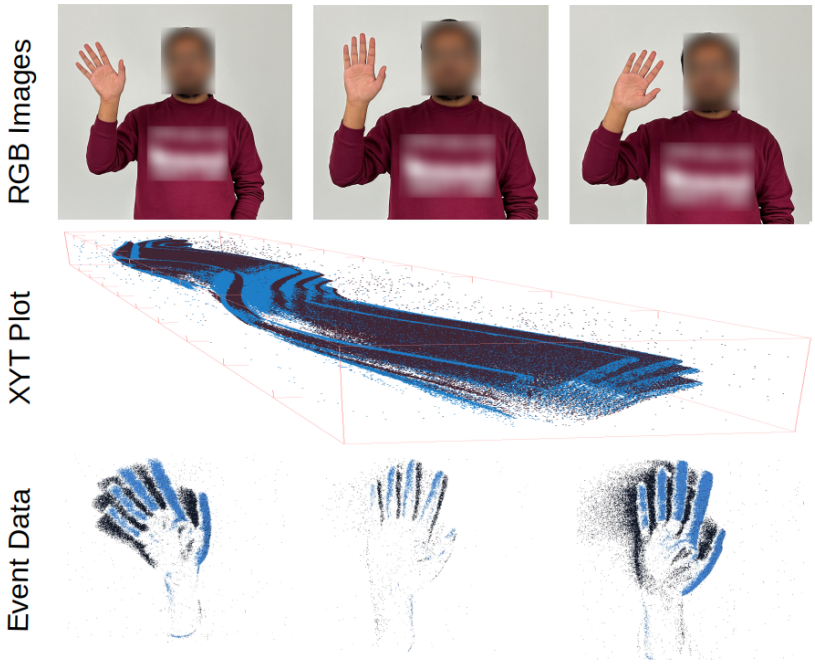


Figure 4.6: The system discussed in this chapter processes data displayed in the final row, illustrating frame-based and event-based camera outputs. The top section shows RGB images of a hand gesture (wave), the middle section depicts positive (blue) and negative (black) DVS events over time, and the bottom section presents the DVS event data corresponding to the executed gesture.

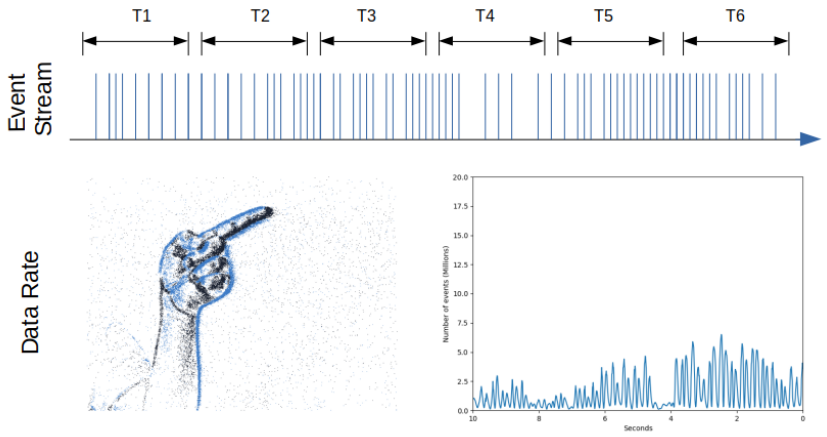


Figure 4.7: (top) event stream sequencing. (bottom) illustration of performed gestures and the number of events captured during time.

recorded using a dedicated labeling system designed for event data. The EB-HandGesture dataset consists of 9000 gesture instances across six different hand gestures, contributed by five participants. Each gesture hand waves, pointing, rock, scissors, claps, and arm roll was performed at three different speeds (slow, normal, fast) and under two lighting conditions (normal and low). Each instance lasts 0.5 seconds, resulting in 1500 instances per gesture (see appendix C.1 for dataset sample). Table 4.4 provides a summary of the dataset alongside other event-based recognition datasets. Figure 4.6 illustrates a comparison between standard RGB images and event data, along with the XYT plot of a performed gesture. Figure 4.7 presents the sequencing of the event stream (top) and the corresponding data rate (bottom) for the recorded gestures.

4.3.2 Model Training

We trained the Convolutional Recurrent Neural Network (ConvRNN) Classifier, which serves as the backbone of our hand gesture recognition system [285]. We provide insights into the dataset preparation, model architecture, and training methodology.

Data Preparation

The dataset was divided into training (70%), validation (20%), and testing (10%) subsets. Given its distinct characteristics, the EB-HandGesture dataset serves as

a valuable resource for evaluating neuromorphic classification models, particularly in hand gesture recognition for robotic control applications. For data preprocessing and model training, we utilised the Prophesee OpenEB open-source framework [285], which is seamlessly integrated with the Metavision SDK. Data was captured using the SilkyCam Gen3.0 event camera in EVT3.0 format. Further details regarding this format are available in the Prophesee documentation [305]. The collected data was subsequently converted into AVI video format for annotation and into HDF5 format to prepare it for training.

Table 4.5: ConvRNN Classifier Architecture (Input: $1 \times 128 \times 128$). All Conv2D layers use 3×3 kernels with padding 1 unless stated otherwise.

Layer Type	**Ch.**	**Output**	**Operation**
Input Tensor	1–1	128×128	Event frame
Conv2D + ReLU	1–16	128×128	Feature extraction
Conv2D + ReLU	16–32	64×64	Down-sampling
Conv2D + ReLU	32–32	64×64	Refinement
Conv2D + ReLU	32–64	32×32	Channel expansion
Conv2D + ReLU	64–64	32×32	Encoding
ConvRNN	64–128	16×16	Spatio-temporal modelling
ConvRNN	128–256	8×8	Deep temporal abstraction
Conv2D (1×1)	256–6	8×8	Logits projection
Global Avg Pool	6–6	1×1	Spatial aggregation
Softmax	6–6	1×1	Probability normalisation

Model Architecture and Training

The proposed ConvRNN classifier is designed to process sequential data and has shown strong performance in our experiments. The architecture details are presented in Table 4.5. The ConvRNN classifier consists of three key components. First, the input data, which consists of a single-channel event stream, is processed through a series of convolutional layers. The number of these layers is determined by the input size, ensuring effective feature extraction. Next, the extracted features are passed through two ConvRNN layers, which incorporate recurrent connections to capture temporal dependencies within the data. Finally, the classification head consists of additional convolutional layers followed by Rectified Linear Unit (ReLU) activation functions, producing class probabilities as the model's output. Each event stream

is stored as an HDF5 file, with an accompanying .npy file containing ground truth labels. During training, we used a batch size of 32, set an initial learning rate of 0.0001, and optimised the model using the Adam optimiser. The implementation was carried out in Python 3.8, utilizing the computational capabilities of an RTX 2080 GPU.

Evaluation Matrics

To assess the performance of our ConvRNN classifier, we conducted comparative experiments using alternative architectures, including MobileNetV2 [98] and SqueezeNet [177]. Additionally, we applied the SpikeBased-BP algorithm to train a classifier on our dataset. These comparisons provide insight into the effectiveness of our proposed model relative to existing architectures. Model performance was primarily evaluated based on classification accuracy, which measures how well the predicted labels match the ground truth. However, relying solely on accuracy may not fully capture the model's ability to distinguish between different classes. To address this, we also employed the Precision-Recall (PR) curve [395], which provides a more comprehensive analysis of the model's predictive performance under varying conditions. This approach ensures a more nuanced evaluation of classification effectiveness beyond standard accuracy metrics.

4.3.3 Results and Discussion

Figure 4.8(a) (Left) illustrates the training and validation accuracy across epochs, with accuracy plotted on the y-axis and epochs on the x-axis. The reported curves represent the mean performance over multiple independent training runs with different random initialisations. Both accuracies follow a consistent upward trend, reaching an average of 96.9% for training and 96.2% for validation. The right side of the figure shows the mean per-class accuracy across epochs, where all gesture classes surpass 93% accuracy by epoch 50, indicating stable and consistent learning across categories. Figure 4.8(b) presents the confusion matrix and error matrix, computed from the averaged evaluation results, providing deeper insight into classification performance.

Most misclassifications occur between visually similar gestures, particularly those involving fine finger motions. Figure 4.8(c) displays the Precision–Recall (PR) curves for each gesture class based on the aggregated predictions. The wave, clap, and scissor classes exhibit high precision and recall, with only a minor decline in precision at higher recall values, indicating minimal false positives and robust class separability. In contrast, performance variability is most pronounced for the Scissors class, which exhibits a clearer precision decline as recall increases. The Rock and Point classes demonstrate comparable behaviour, suggesting that classification difficulty is not isolated to a single static gesture. The reduced precision at higher recall levels likely arises from overlapping spatio-temporal event patterns among gestures, particularly where finger configurations produce similar edge dynamics in the event-based representation. These overlaps result in more frequent misclassifications and highlight the need for further refinement in feature extraction or data augmentation specific to the rock gesture. Notably, the clap class demonstrates perfect precision across all recall levels. This consistent performance may be attributed to the highly distinct motion and temporal profile of the clapping gesture, rapid bilateral movement towards the centre of the frame, making it less likely to be confused with other gestures. However, the presence of such a clear margin also raises concerns about potential dataset imbalance or overfitting. Figure 4.8(d) presents the Receiver Operating Characteristic (ROC) curves, plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) across different threshold settings. The Area Under the Curve (AUC) is 1.00 for all models, indicating perfect classification with no overlap between positive and negative distributions.

Table 4.6 presents a comparison of our EB-HandGesture dataset with existing event-based action and gesture datasets, along with their respective state-of-the-art model performances. Overall, our dataset achieved higher testing accuracy compared to other datasets and their models. Given that DVS128Gesture is the closest dataset to EB-HandGesture, we trained it using our proposed ConvRNN model for a direct comparison. Additionally, we used the SpikeBased-BP algorithm to train the EB-HandGesture dataset as part of the evaluation. For the ConvRNN classifier, the DVS128Gesture dataset achieved a maximum accuracy of 87.42%,

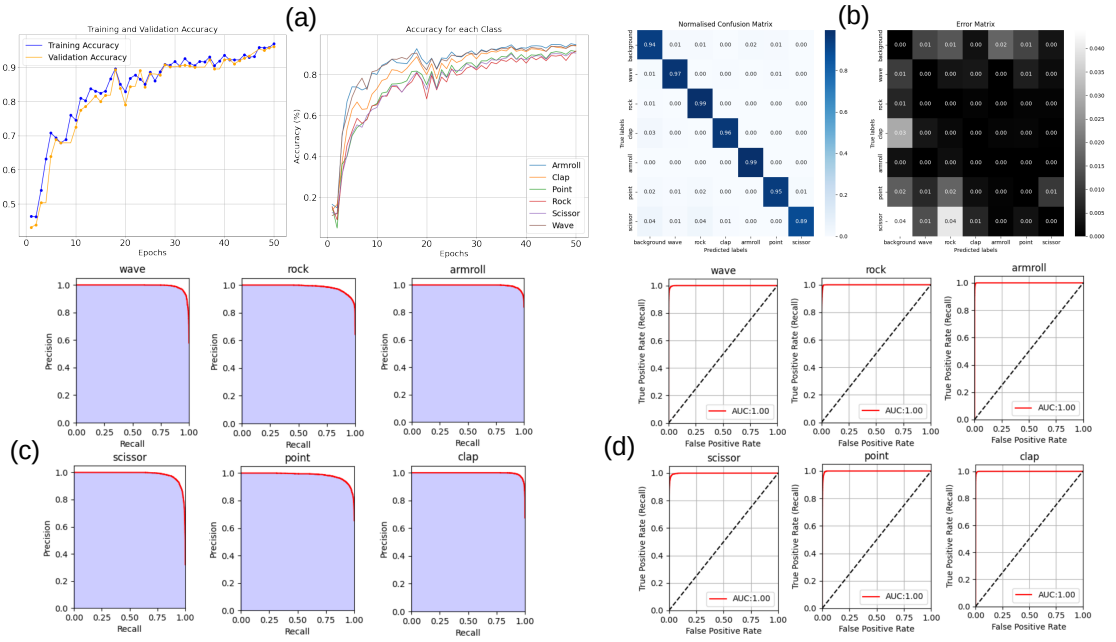


Figure 4.8: (a) left: training and validation accuracy (y-axis) for each epoch (x-axis). right: Validation accuracy of each class. (b) Confusion and Error Matrix. (c) Precision (y-axis), Recall (x-axis) curve. (d) Receiver Operating Characteristic (ROC) Curves for Gesture Recognition Model.

whereas the EB-HandGesture dataset attained 95.58%, demonstrating a significant improvement. Using the SpikeBased-BP algorithm, accuracy for DVS128Gesture was 95.5%, while for EB-HandGesture, it was 85.6%. We also compared our model with MobileNet (70.44%) and SqueezeNet (75.65%), highlighting the advantages of our approach. To further benchmark our dataset, we created a custom RGB-based gesture dataset and trained it using an LSTM model. While LSTM achieved a high accuracy, its real-world performance was limited due to the inherent constraints of RGB cameras, particularly in low-light conditions. Figure 4.9(a) illustrates the superior low-light performance of event cameras, where standard cameras fail to detect any activity. The robustness of our model in such environments enhances human-robot interactions, surpassing current state-of-the-art models in recognizing gestures under challenging conditions.

Table 4.6: Comparison of Various Models and Datasets

Ref.	Dataset	Algo./Model	Accuracy(%)
[45]	ASL-DVS	G-CNN	87.5
[45]		RG-CNN	90.1
[267]		MobileNet	86.7
[388]	HAR-DVS	ESTF	57.53
[165]		ResNet18	56.09
[241]		ResNet50	57.99
[240]	Daily Action	Motion SNN	90.3
[238]		HMAX-SNN	76.9
[123]	Bullying10k	ResNet50	74.01
[370]		ResNet18	72.5
[122]		X3D	65.6
[334]	DVS128Gesture	Deep-SNN	93.6
[399]		ConvRNN-SNN	90.28
[193]		SpikeBased-BP	93.5
This Work		ConvRNN	87.42
This Work	EB-HandGesture	MobileNet	70.44
This Work		SqueezeNet	75.65
This Work		ConvRNN	**95.77**
This Work		SpikeBased-BP	85.6
This Work	Custom RGB Dataset	LSTM	98.9

Robot Control

To showcase the practical application of our gesture recognition model, we implemented intuitive robot control using the ARI humanoid robot and an event camera. The event camera was mounted on the ARI robot, allowing seamless integration of the gesture classifier with the robot's control system. This setup enabled real-time interaction, where the robot's movements were controlled based on predicted hand gestures. The experimental process involved initializing the robot, activating the event camera, and starting the classification pipeline. For real-time classification, the event stream was segmented into 1-second intervals, with each recognized gesture mapped to a specific robot movement. The model and inference pipeline are flexible and can be adapted to associate gestures with different actions based on application requirements. Figure 4.8(b) illustrates the ARI robot recognizing hand gestures. For example, when detecting a wave gesture, ARI responds by waving; for point, it indicates a direction; for rock, it presents a paper response. The clap and arm roll

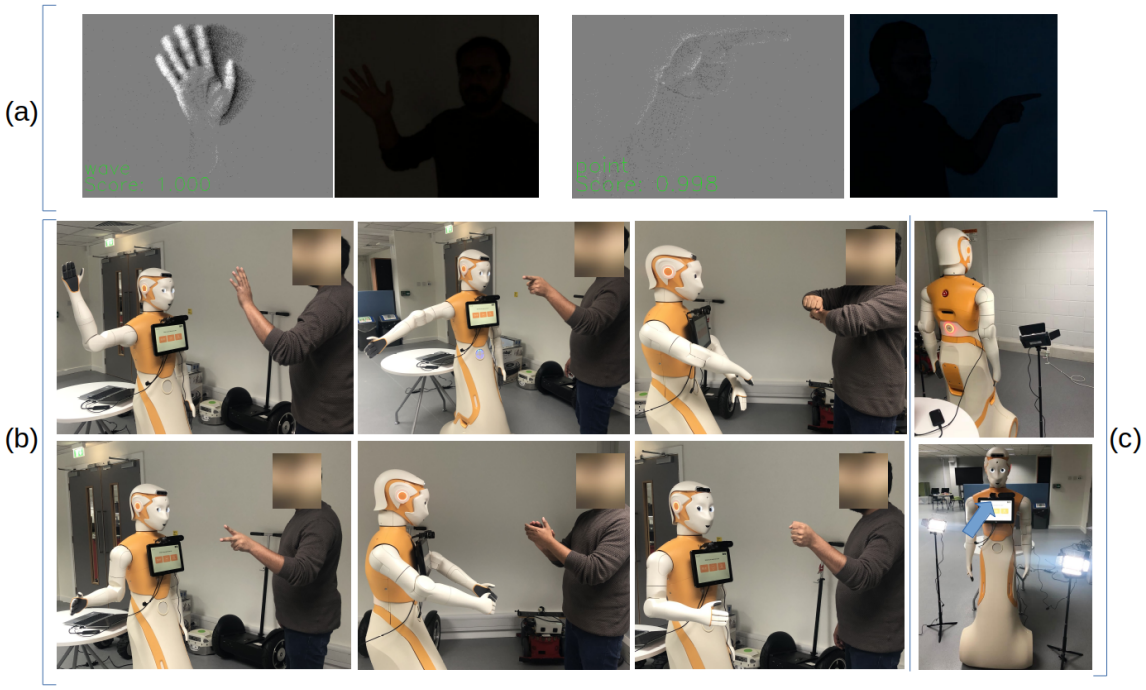


Figure 4.9: (a) Testing our model in low light conditions where the standard camera was not able to detect. (b) real-time experiments with an event camera and ARI robot for different hand gestures. (c) data collection setup. The arrow is pointing toward the mounted event camera.

gestures are mirrored as closely as possible by the robot. Figure 4.8(c) provides an overview of the data collection setup, demonstrating the integration of event-based gesture recognition in robotic applications.

4.3.4 Potential Applications

The integration of hand gesture recognition with an event camera for robot control holds significant potential across various domains, particularly in healthcare and collaborative work environments. In healthcare, especially in elder care, this system enables elderly individuals to control robots using simple hand gestures. This functionality is particularly beneficial for those with limited mobility or communication abilities, as the event camera effectively captures gestures without the motion blur issues commonly associated with standard cameras. This ensures precise and reliable operation, which is crucial in healthcare settings where both speed and accuracy are essential. In collaborative workspaces, such as warehouses, this technology enhances

human-robot interaction by allowing workers to direct robots through gesture-based commands. This is especially advantageous in noisy environments or situations where verbal communication is impractical. The event-based approach ensures consistent performance across different lighting conditions, improving operational efficiency and fostering seamless collaboration between humans and robots. Overall, this gesture-controlled robotic system powered by event cameras represents a meaningful advancement in intuitive human–robot interaction, offering improved reliability and efficiency under controlled conditions where the robot remains stationary during gesture recognition.

In conclusion, this study represents a notable advancement in event-based vision and human-robot interaction, leveraging high-resolution event cameras for hand gesture recognition. The consistent improvements in accuracy, culminating in a testing accuracy of 95.77%, demonstrate the strong performance of our ConvRNN model. To validate its real-world applicability, we seamlessly integrated the model with an ARI robot, enabling precise and intuitive robot control through hand gestures. The successful deployment of our system under diverse environmental conditions demonstrates its robustness, adaptability, and potential to advance collaborative human-robot interaction. To further illustrate the real-world applicability of event-based cameras, the subsequent application focuses on their use in real-time machinery fault diagnosis.

4.4 Event-Based Rotational Motion Analysis for Real-Time Machinery Condition Monitoring

In industrial operations, maintaining machinery reliability is essential not only for ensuring productivity but also for workplace safety [265]. Mechanical failures can lead to substantial disruptions, ranging from downtime and financial losses to severe incidents such as fires or explosions [256]. To mitigate these risks, companies allocate significant resources to maintenance programs aimed at keeping equipment in optimal condition. With advancements in technology, the approach to machinery maintenance

has evolved, with predictive maintenance becoming increasingly prevalent. This method allows failures to be anticipated before they occur, minimizing downtime and preventing costly repairs. Among various predictive maintenance techniques, vibration analysis is widely used [350]. By monitoring the vibration patterns of machinery, abnormalities can be detected early, indicating potential faults.

Traditional vibration monitoring relies on direct-contact sensors such as accelerometers [18]. However, these sensors must be physically attached to the equipment, which is not always feasible, and may also interfere with machine performance [68]. Alternative technologies like laser vibrometers [313] and eddy current sensors [105] present their own challenges, including high costs, material constraints, and limited installation flexibility [96]. More recently, frame-based cameras have been introduced for contactless vibration measurement, but these systems struggle with large data volumes, varying light conditions, and background interference, making real-time processing computationally demanding [310, 116, 324]. Recent advancements have sought to bridge the gap between frame-based techniques and event-based vision, leveraging the asynchronous, high-temporal resolution capabilities of event cameras [214, 280, 418, 378]. Researchers have explored various approaches, from event-based tracking [214, 280] and feature detection [378, 268] to optical flow estimation using learning-based methods [420, 405, 36]. Motion compensation frameworks and luminance gradient-based techniques have also been proposed to enhance event alignment [56, 264]. Additionally, an event-based KLT tracker has been developed for the asynchronous tracking of frame features using intensity measurements [141, 142]. More recently, Woong-jae et al. introduced an event filter-based phase correlation template matching (EF-PCTM) method for detecting micro-vibrations with event cameras [184]. Although promising, this system is computationally intensive and does not explicitly quantify vibration measurement accuracy. In parallel, Bane et al. demonstrated the feasibility of qualitative vibration frequency analysis and motion magnification using event cameras, highlighting both the potential of high-temporal-resolution sensing and the practical challenges associated with subtle motion reconstruction [30]. Similarly, Lv et al. proposed event-based vibration frequency detection methods based on marker tracking and event counting, achieving high-

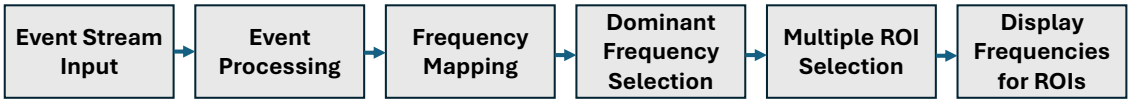


Figure 4.10: Block diagram of the proposed (EBFM) vibration monitoring system. It starts with event stream input from event camera then through event processing and frequency mapping, leading to the selection of dominant frequencies within multiple regions of interest (ROIs)

precision frequency estimation across a wide range of structural vibration scenarios [244].

Despite these advancements, most existing studies focus on edge tracking rather than precise vibration analysis, limiting their applicability to multi-machine industrial environments. To address this gap, we propose the Event-Based Frequency Mapping (EBFM) system, which introduces a novel approach to real-time vibration monitoring using event-based cameras. The EBFM system processes event streams to generate dynamic frequency maps, enabling the detection and monitoring of subtle vibrational anomalies. By computing dominant frequencies within user-defined regions of interest (ROIs), the system allows for targeted analysis of machine vibrations. A key feature of EBFM is its graphical user interface (GUI), which provides real-time visualisation of frequency mappings, allowing users to assess and interpret machine vibrations. Figure 4.10 presents the block diagram of the proposed EBFM vibration monitoring system, illustrating its capability to enhance the precision and efficiency of industrial fault diagnosis through event-based frequency mapping.

4.4.1 Event-Based Frequency Mapping (EBFM)

The first step in EBFM involves capturing an event stream from an event-based camera, where each event corresponds to a change in pixel intensity at a specific location and time. At the core of the algorithm is Frequency Mapping, which generates a spatial frequency representation from the timestamped event data. In this map, each pixel encodes the dominant frequency component of the events occurring at that spatial location. Specifically, the value at each pixel corresponds to the Fourier coefficient of the frequency with the highest magnitude, effectively the

dominant frequency, derived from a transform tailored to the sparse and asynchronous characteristics of event data. This adaptation of the Fourier Transform allows the system to highlight regions exhibiting periodic motion or vibration, crucial for analysing dynamic mechanical behaviour in real time:

$$F(x, y, \omega) = \int E(x, y, t) \cdot e^{-i\omega t} \, dt \tag{4.1}$$

where $F(x, y, \omega)$ denotes the frequency component at a given spatial position (x, y) and angular frequency ω. The term $E(x, y, t)$ represents the signal intensity at time t, derived from the event data. For each defined Region of Interest (ROI), the algorithm calculates the dominant frequency, which is the frequency that most significantly characterizes the vibration within that region. This is done by identifying the frequency with the highest amplitude on the frequency map:

$$f_{\text{dominant}} = \arg\max_{\omega, x, y} |F(x, y, \omega)| \tag{4.2}$$

where f_{dominant} is the dominant frequency, and $|F(x, y, \omega)|$ denotes the magnitude of the frequency component at spatial position (x, y) and angular frequency ω. Users can interactively select and modify ROIs in the GUI, which allows them to focus on different machines or parts of a single machine visible to the camera. Algorithm 6 presents the flow of the EBFM algorithm.

4.4.2 Experiment Setup and Evaluation

Two experiments were designed and performed to assess the performance of the EBFM-based vibration monitoring system. The first experiment focused on measuring the vibration of a rotating disk with varying rotations per minute (RPM), highlighting the system's accuracy in vibration monitoring. The second experiment involved monitoring fault conditions in the rotation motor by adding weights to the motor cylinder. Both experiments utilised the CenturyArk SilkyCam Gen3 event camera [199]. The specifications of the event camera are provided in Table 4.7.

Algorithm 6 Event-Based Frequency Mapping Algorithm

procedure FREQUENCY MAPPING($events, width, height$)
  Initialize frequency maps $F(x, y, \omega) \leftarrow 0$ for each pixel
  Define parameters $min\_freq, max\_freq, diff\_thresh\_us$
  while events available **do**
    Extract (x, y, t, p) from $events$
    Update frequency maps using:

$$F(x, y, \omega) \leftarrow F(x, y, \omega) + e^{-i\omega t}$$

  end while
  for each ROI defined **do**
    Calculate dominant frequency:

$$f_{\text{dominant}} = \arg\max_{\omega, x, y} |F(x, y, \omega)|$$

    Update GUI to display f_{dominant}
  end for
  return frequency maps, ROI data
end procedure

Model	PPS3MVCD
Resolution	640 × 480 pixels
Pixel size	15 μm × 15 μm
Max latency	200 μs
Dynamic range	> 120 dB
Current consumption	200-300 mA

Table 4.7: Specifications of the CenturyArks SilkyCam Gen3 event camera.

Experiment 1

The Buehler MetaServ 250M Grinder Polisher [58] was used for this experiment. This high-performance equipment is specifically designed for precise sample preparation in metallography. It features a 14-pole motor with variable speed control, allowing for adjustments between 100 and 400 RPM, and operates on 115VAC, single-phase power. The aim of this experiment was to measure the frequency of the rotating plate and compare it to the actual frequency. Tests were conducted under varying light conditions low, normal, and bright at 100, 200, 300, and 400 RPM. These experiments helped assess the accuracy of the system.

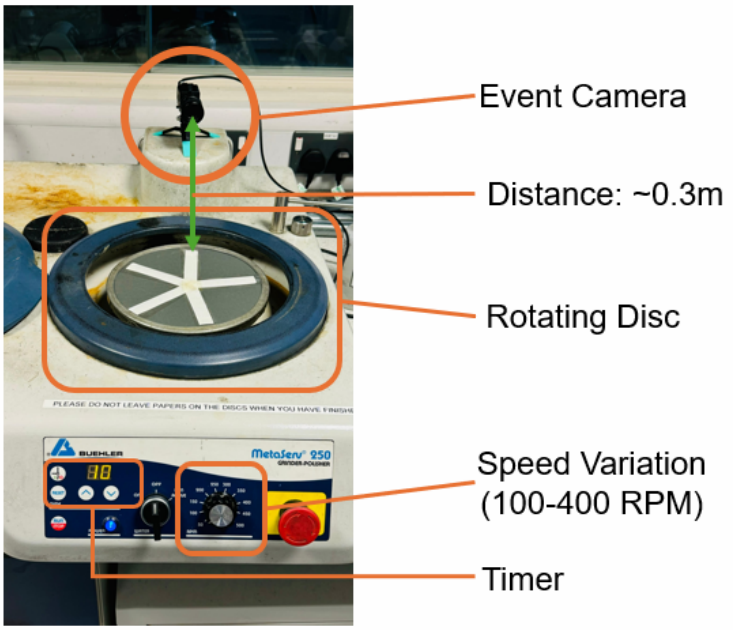


Figure 4.11: Experiment 1 setup. Event camera is placed at the stable surface and $0.3m$ from the rotating disc. Disc speed can be varied from 100rpm to 400rpm.

To calculate the frequency for a given speed level and number of poles in an electric motor, the formula used is:

$$\text{Frequency (Hz)} = \frac{\text{RPM} \times \text{No. of Poles}}{120} \tag{4.3}$$

This formula connects the motor's rotational speed (RPM) with the electrical frequency (Hz) and the number of poles in the motor. The experimental setup is shown in Figure 4.11. The disc was rotated for one minute at each speed setting, and the average frequency was then compared to the calculated value. The event camera was positioned approximately 0.3 meters from the rotating disc. Figure 4.12 compares three lighting conditions (normal, low, and bright illumination), each visualised in three modalities. Column (a) shows the RGB image of the rotating disc. Column (b) presents the accumulated event stream over a fixed temporal window of 15 ms, illustrating motion-induced edge activity. Column (c) displays the output of the EBFM frequency analysis system, where the colour intensity encodes the dominant rotational frequency magnitude estimated at each spatial location. Figure 4.13 displays a comparative analysis of the calculated frequencies (from equation 4.3)

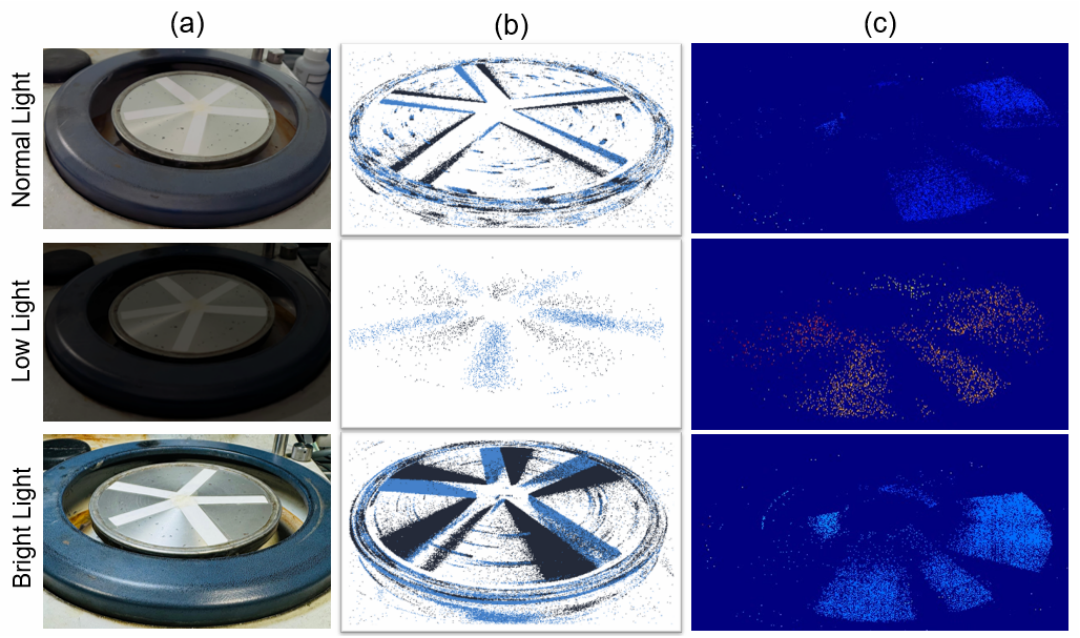


Figure 4.12: Frequency estimation in three different lighting conditions. (a) RGB images of disc, (b) Event-based representation obtained by accumulating events over a fixed temporal window of 15 ms, where each point corresponds to a polarity event detected within that interval. (c) Output of the EBFM system. The colour encodes the estimated dominant rotational frequency at each spatial location.

against those measured under various lighting conditions. The differences between the calculated and measured frequencies are minimal, demonstrating the precision of the EBFM algorithm. However, as the speed increases, a slight variance between the frequencies measured in low light and the calculated values is observed, suggesting some sensitivity to lighting conditions. In contrast, measurements taken under bright light closely match the calculated frequencies, indicating strong performance in well-lit conditions. The consistently low error rates, as shown in Table 4.8, further validate the reliability of the EBFM algorithm for rotational frequency monitoring under various lighting conditions.

Table 4.8: Measured and Ground-Truth Frequencies under Various Lighting Conditions

Speed	Ground Truth	**Normal Light**		**Low Light**		**Bright Light**	
(RPM)	Freq.(Hz)	**Freq.(Hz)**	**% Error**	**Freq.(Hz)**	**% Error**	**Freq.(Hz)**	**% Error**
100	11.67	11.98	2.66	11.91	2.06	12.02	2.99
200	23.33	22.78	2.36	22.32	4.33	23.76	1.84
300	35	33.96	2.97	32.92	5.94	35.61	1.74
400	46.67	44.56	4.52	44.22	5.24	47.12	0.96

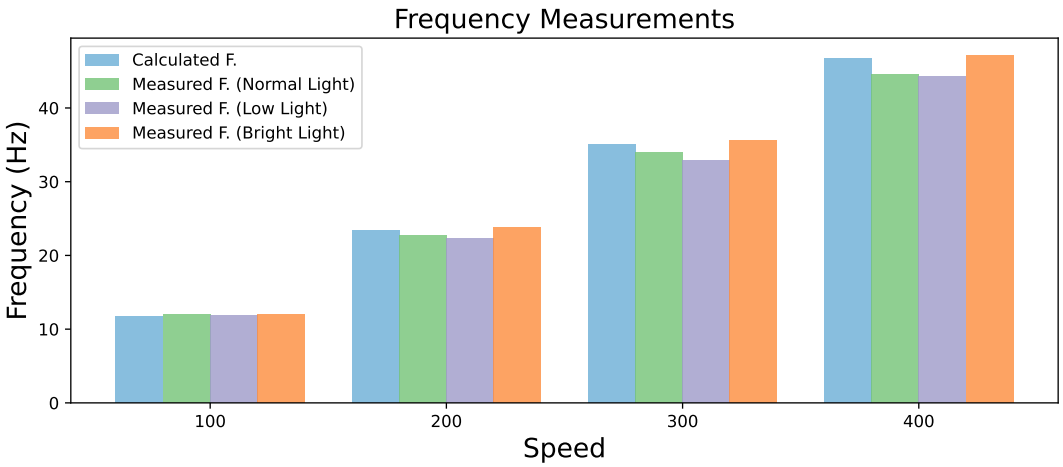


Figure 4.13: Calculated Frequency vs Measured Frequency in different lighting conditions.

Experiment 2

Following the validation of our EBFM rotational frequency monitoring system, we conducted experiments to examine the effects of anomalies on the frequency of a rotating motor. For this, we employed the Capco Testing Equipment Ball Mill machine, which features a single-tier mechanism with three rollers capable of supporting two rows of jars as weights. The machine allows for adjustable roller speeds ranging from 0 to 420 RPM, controlled via a speed regulator. Additionally, the maximum load capacity for the roller is 2.5 kg. The technical specifications of the Ball Mill machine are shown in Table 4.9.

Machine	Capco Ball Mill
Motor	180 w
Weight	47 kg
Roll Speed	0-420 RPM
Load Capacity	2.5 kg
Dimensions	278 x 376 x 988 mm

Table 4.9: Specifications of the Capco Test Equipment Ball Mill Machine.

Figure 4.14 illustrates the experimental setup, where the event camera is positioned approximately 0.3 meters from the machine under bright lighting. To reduce the impact of floor vibrations, the camera is mounted on a damper. Additionally, the Ball Mill's roller remains within the event camera's field of view.

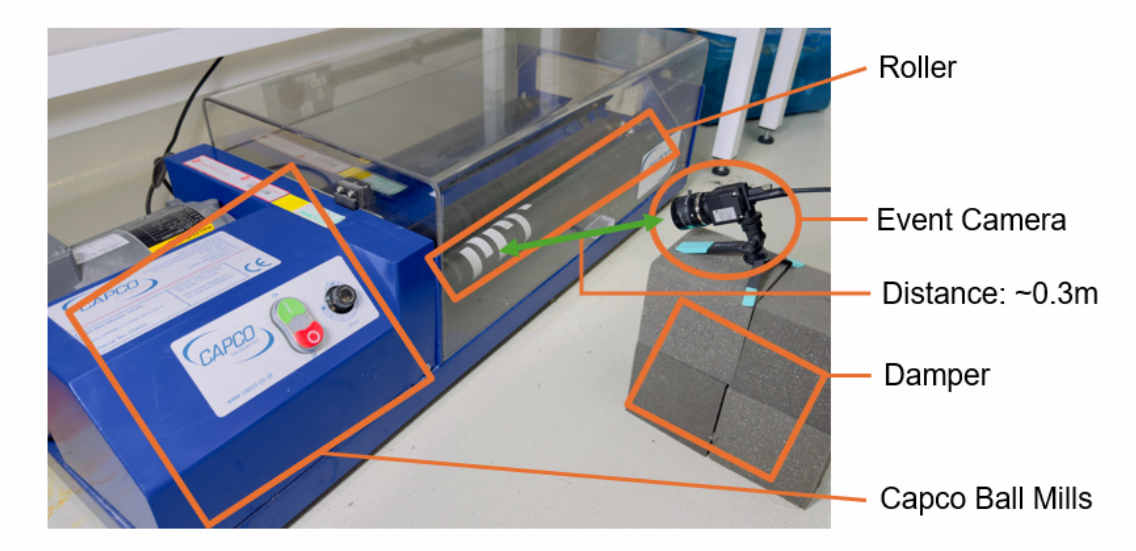


Figure 4.14: Experiment 2 setup. Capco Ball Mill machine is used for this experiment. The roller of a machine can rotate from 0-420 RPM. The distance between roller and event camera is around 0.3 meters.

In this experiment, the machine was operated at three different speed levels (140 RPM, 280 RPM, and 420 RPM). Initially, we measured the frequency of the rotating roller under normal conditions and then with an additional load. The underlying hypothesis is that exceeding the machine's maximum load capacity (as specified in Table 4.9) would introduce an anomaly, leading to a change in the roller's frequency. To evaluate this, we added a 3 kg weight to the roller, ensuring that a heavier load was avoided to prevent potential damage to the machine. Figure 4.15 presents a comparison of the roller with and without the load, including the RGB image, event representation, and EBFM output for both scenarios.

For each condition, data was collected over a one-minute period, and the average frequency was recorded. As shown in Table 4.10, the addition of a load consistently led to a slight decrease in frequency across all speed settings: from 5.26 Hz to 5.13 Hz at 140 RPM, 8.38 Hz to 7.88 Hz at 280 RPM, and 11.85 Hz to 11.06 Hz at 420 RPM. This reduction indicates that the load influences the variance in rotational frequency of the system, supporting our hypothesis that the machine's frequency changes when an additional load is applied.

Figure 4.16 shows the frequency variations at three different rotational speeds 140 RPM, 280 RPM, and 420 RPM under two conditions: without a load and with a load.

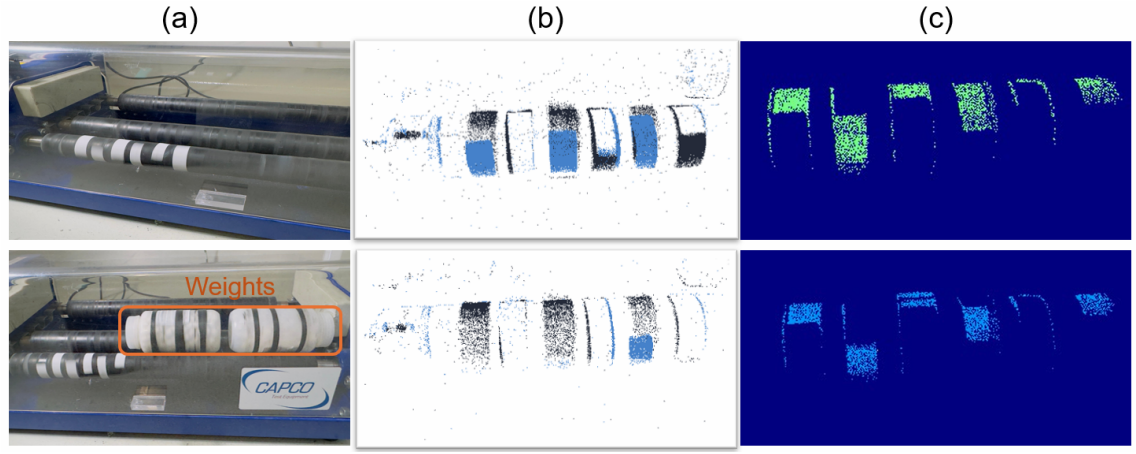


Figure 4.15: compares the rotating roller under two loading conditions: the top row shows the unloaded roller operating at nominal speed, while the bottom row shows the same roller after additional weights are attached. Column (a) presents the RGB images of the roller in both scenarios. Column (b) shows the accumulated event stream over a fixed temporal window of 15 ms, highlighting motion-induced edge activity. Column (c) depicts the output of the EBFM rotational frequency analysis, where colour intensity encodes the dominant motion frequency magnitude estimated at each spatial location.

Table 4.10: Frequency Measurements with and without load

	Frequency (Hz)		
	140rpm	**280rpm**	**420rpm**
No Load	5.26	8.38	11.85
With Load	5.13	7.88	11.06

The vertical axis represents frequency differences in Hertz (Hz), while the horizontal axis displays a sequence of 300 measurements. At 140 RPM, the frequency differences remain relatively stable with only minor fluctuations around zero, indicating minimal impact from the applied load on the rotational speed. However, at 280 RPM, the frequency variations become more pronounced, suggesting that higher speeds may introduce greater sensitivity to load-induced changes. This trend becomes even more significant at 420 RPM, where the frequency differences increase further, particularly under loaded conditions.

In conclusion, our proposed Event-Based Frequency Mapping (EBFM) rotational frequency monitoring system effectively detects abnormal vibrations in machinery, showcasing its potential for real-time health monitoring and automated interventions, such as machine shutdowns or operator alerts. By leveraging event cameras, the system captures high-resolution rotational frequency data without physical contact,

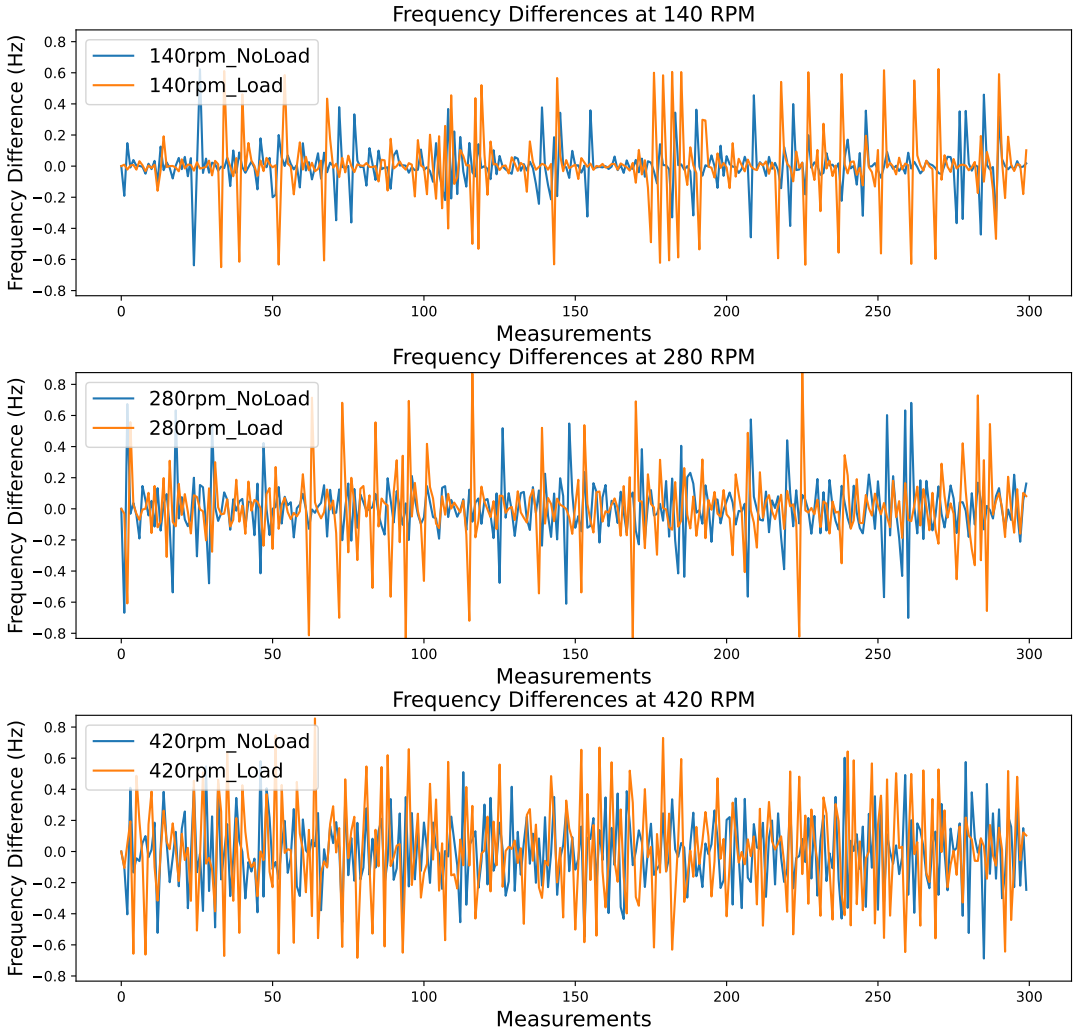


Figure 4.16: Variability of Frequency Differences Under No-Load and Load Conditions Across Three Rotational Speeds (140 RPM, 280 RPM, 420 RPM).

ensuring precise fault detection across various load conditions and rotational speeds. Experimental evaluations confirm its robustness in diverse environments, demonstrating its adaptability for industrial and robotic applications, where robots can identify faults in machines and take appropriate corrective actions.

4.5 Summary

This chapter explored the growing applications of event-based vision in robotics and industrial automation. It discussed various event data formats and proposed a standardised processing pipeline for improved data handling. The chapter also presented an event-camera-based gesture recognition system for intuitive human-robot interaction and introduced an event-based frequency mapping (EBFM) system for real-time machinery fault detection. These advancements demonstrate the potential of event-based vision in enhancing efficiency, accuracy, and adaptability in dynamic environments. Advancements in event-based vision have demonstrated significant potential in enhancing human-robot interaction (HRI) by enabling more efficient perception and gesture recognition. However, to further improve collaborative performance, it is essential to understand the cognitive load experienced by the user during HRI. The following chapter examines methods for estimating cognitive load during interaction, exploring physiological, behavioral, and performance-based indicators to inform the development of adaptive robotic systems.

Chapter 5

Cognitive Load Estimation During Human-Robot Interaction (HRI)

5.1 Introduction

As robots take on increasingly collaborative roles in dynamic, real-world environments, the ability to autonomously recognise and adapt to human cognitive states becomes essential for effective interaction. This chapter introduces a novel, robot-centric framework for cognitive load (CL) estimation during human–robot interaction (HRI), grounded in a multimodal approach that leverages only the robot's embedded sensing and interaction capabilities. Departing from traditional methods that rely heavily on external instrumentation, the framework enables a robot to autonomously administer cognitive tasks and monitor user responses using onboard sensors such as event and RGB cameras, microphones, and interaction interfaces. Through the integration of behavioural cues, including pose dynamics, gaze patterns, facial expressions, and task-related performance metrics, this study lays the foundation for a fully self-contained adaptive robotic system capable of monitoring CL in real time. The chapter outlines the design, data collection, and machine learning methods employed to enable robust CL estimation, offering key insights into the development of cognitively aware robotic platforms suitable for everyday settings.

Novelty & Impact

Novelty: Develops an innovative multimodal cognitive load estimation framework integrating event-based pose estimation, physiological signals, and behavioural indicators.

Impact: Significantly advances human–robot interaction by enabling robots to dynamically adapt to human cognitive states, improving collaborative performance and user experience.

5.2 Cognitive Load in HRI

As robots increasingly enter critical roles in healthcare, education, and manufacturing, their capability to interpret and respond to human cognitive states becomes essential. One significant cognitive state, cognitive load (CL), measures the mental effort required to process information and complete tasks [353, 290]. Unmanaged cognitive overload can negatively impact human performance, decision-making quality, and collaborative interactions [360, 124]. Within human-robot interaction (HRI), failure to detect excessive cognitive load can lead to increased errors, reduced user trust, and impaired collaboration [226, 389]. Several techniques currently exist to measure cognitive load in real-time scenarios, such as physiological signals (heart rate variability, electroencephalography, and skin conductance) [233], behavioural indicators (facial action units, eye gaze, and body posture) [253], and subjective measures (self-reported mental effort or task performance metrics) [27, 349]. Although these methods provide valuable insights, they commonly depend on intrusive sensors, specialised equipment, or controlled environments, making them impractical for widespread deployment in naturalistic interaction contexts. Recent advancements have begun exploring robots' autonomous sensing capabilities using built-in hardware; however, existing studies often focus on isolated modalities or continue to depend partly on external sensing systems [393, 3]. Thus, a notable gap remains: no previous research has comprehensively examined whether robots can independently deliver

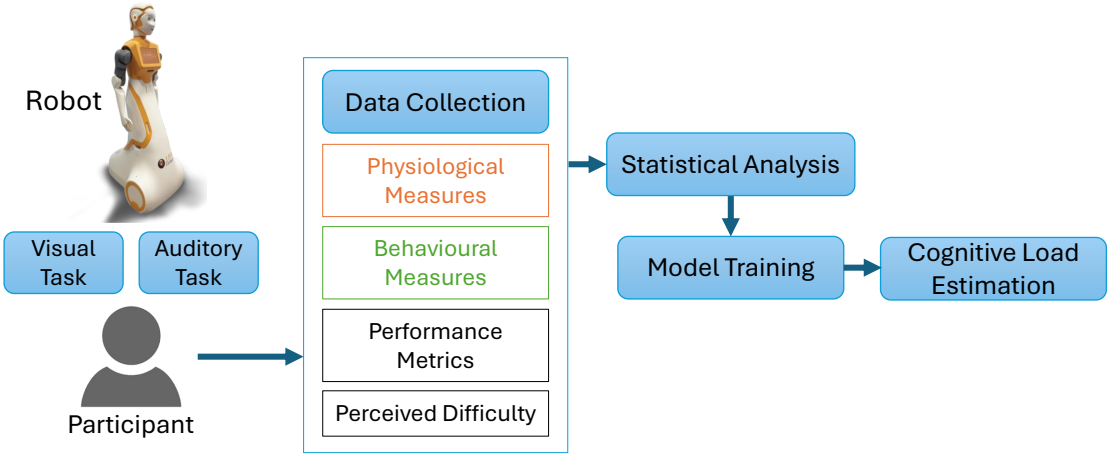


Figure 5.1: Block diagram illustrating the proposed cognitive load estimation framework.

tasks and simultaneously estimate user cognitive load exclusively through embedded sensing modalities.

To address this gap, we propose a fully robot-centric framework that enables a humanoid robot to autonomously administer cognitive tasks and simultaneously monitor user cognitive load using only its internal sensing resources. Specifically, the robot leverages its onboard event camera—selected for robustness to rapid movements and varying lighting conditions—to measure body pose dynamics, and its RGB camera for capturing facial expressions and gaze patterns. Additionally, auditory and visual stimuli are autonomously delivered through the robot's integrated touchscreen and text-to-speech system. A wearable physiological sensor (Shimmer) serves solely as a ground truth reference to validate the robot-derived estimations, not as input for the robot's classification model.

Figure 5.1 illustrates the proposed framework, detailing autonomous data acquisition, feature extraction, and machine learning-based classification of cognitive load states. The experimental evaluation involved 32 participants performing the Stroop and N-back tasks, both recognised for effectively modulating cognitive load.

The main contributions of this chapter are:

1. The introduction and validation of a fully autonomous, robot-centric cognitive load estimation framework, removing the need for external perception or task-delivery devices.

2. Comprehensive statistical analyses comparing robot-derived behavioural signals against physiological ground truth, establishing reliability and robustness of onboard sensing modalities across visual and auditory task conditions.

3. Development and evaluation of a supervised machine learning classifier, trained exclusively on robot-captured behavioural indicators, achieving high accuracy in discriminating between high and low cognitive load.

By demonstrating that socially interactive robots can independently monitor cognitive load using their embedded sensing and interactive resources, this study advances the development of genuinely adaptive, cognitively aware robotic systems suitable for real-world applications.

5.3 Related Work

Cognitive load (CL) is a fundamental concept in cognitive science, initially introduced to understand the mental processes involved in learning and problem-solving [258]. Over the years, the scope of CL has expanded to various domains, including human-robot interaction (HRI), where it plays a crucial role in enhancing collaboration efficiency and user experience [191][112]. This section provides a comprehensive overview of the evolution of CL, its measurement methodologies, and the existing gaps in the context of HRI. The concept of cognitive load was first introduced by John Sweller in the late 1980s within the framework of Cognitive Load Theory (CLT) [352]. Sweller aimed to explain the cognitive processes involved in learning and instructional design, emphasizing the limitations of working memory. CLT posits that working memory has a finite capacity, and effective instructional methods must manage cognitive load to optimise learning outcomes. Cognitive load is typically categorized into three types: intrinsic, extraneous, and germane. Intrinsic load pertains to the inherent difficulty of the material, extraneous load relates to the way information is presented, and germane load involves the mental resources dedicated to processing and understanding the material [353]. The rest of the section details the

various methodologies previously employed to evaluate cognitive load in experimental contexts.

5.3.1 Performance-Based Methods

Performance-based methods assess cognitive load by analysing task performance metrics such as response time, error rates, and task completion efficiency [157]. These measures operate on the premise that higher cognitive load negatively impacts performance, resulting in slower responses and increased errors. For example, studies have demonstrated that as task difficulty escalates, leading to elevated cognitive load, individuals exhibit longer response times and a higher propensity for mistakes [281]. While performance-based metrics provide objective indicators of cognitive load, their applicability is inherently task-dependent. Metrics like response time and error rates vary significantly across different tasks and environments, limiting their generalizability. Moreover, these measures often fail to account for individual differences in cognitive capacity and strategies, which can influence performance independently of cognitive load [288]. Therefore, relying solely on performance-based methods can result in an incomplete and context-specific assessment of cognitive load, highlighting the need for more versatile measurement approaches in HRI.

5.3.2 Physiological and Behavioural Methods

Physiological and behavioural measures provide objective indicators of cognitive load by monitoring changes in the body's physiological responses and observable behaviours [3][307]. Common physiological indicators include heart rate variability (HRV), galvanic skin response (GSR), electroencephalogram (EEG) signals, and pupil dilation [271]. For instance, elevated HRV and increased GSR have been linked to higher cognitive load [155]. EEG measurements, particularly theta wave activity in the frontal region, also correlate with cognitive load levels [406]. Pupil dilation, as a measure of autonomic nervous system activity, has been consistently associated with increased cognitive effort [375][358][112]. behavioural indicators encompass a range of observable actions such as body movement amplitude, velocity,

and frequency [12]. These measures are advantageous due to their non-intrusive nature and ease of integration into HRI systems. For example, increased body movement velocity may indicate heightened cognitive engagement or stress, while reduced movement amplitude could signify cognitive overload [330]. Despite their utility, physiological measures often require specialized and sometimes cumbersome sensors, which can hinder natural interaction and limit their practicality in dynamic HRI environments [24]. Behavioural measures, while more accessible, may not fully capture the complexity of cognitive load without complementary physiological data. Additionally, variations in individual baseline physiological and behavioural responses necessitate personalized calibration for accurate CL estimation. Therefore, combining physiological and behavioural measures with other data sources is essential for a comprehensive and reliable assessment of cognitive load in HRI.

5.3.3 Questionnaire-Based Methods

Subjective questionnaires remain the most widely used method for measuring cognitive load due to their straightforward implementation and ease of administration [207]. Instruments such as the NASA Task Load Index (NASA-TLX) and the Cognitive Load Component Survey are prevalent tools that capture individuals' perceived mental effort [89]. The NASA-TLX, for instance, evaluates six dimensions: mental demand, physical demand, temporal demand, effort, performance, and frustration, providing a comprehensive overview of the user's subjective experience. Despite their popularity, subjective measures have notable limitations. They rely heavily on self-reporting, which can be susceptible to biases and inaccuracies, especially when administered post-task. This retrospective assessment fails to capture the dynamic fluctuations of cognitive load during real-time interactions, making it less suitable for applications like HRI where immediate adaptation is essential [288]. Consequently, while subjective questionnaires offer valuable insights, they are often complemented with more objective measurement techniques to obtain a holistic understanding of cognitive load.

5.3.4 Integrated and Multi-Modal Approaches

Recognizing the limitations of single or dual measurement methods, recent advancements emphasize the integration of multiple data sources to provide a more comprehensive and accurate assessment of cognitive load. Multi-modal approaches combine physiological, behavioural, and performance-based measures to capture the multifaceted nature of cognitive load. For example, integrating heart rate data with body movement patterns and task performance metrics can offer a holistic view of the user's cognitive state. Some studies have ventured into combining gaze tracking, with traditional physiological measures to enhance the accuracy of CL estimation [42][380][129]. Some of these integrated frameworks leverage machine learning techniques to process and analyse diverse data streams, enabling more nuanced and reliable cognitive load assessments [233][69][3][197]. However, despite the potential of multi-modal approaches, most existing studies still rely on a limited number of data streams, often focusing on one or two types of indicators. This fragmentation restricts the ability to capture the full spectrum of cognitive load dynamics, particularly in real-time and dynamic HRI scenarios.

The existing solutions highlight that while substantial progress has been made in cognitive load estimation, current methodologies remain fragmented and limited in scope. Subjective questionnaires, performance-based metrics, speech-based indicators, and physiological and behavioural measures each contribute valuable insights but also present inherent limitations. The reliance on single or dual data sources restricts the ability to capture the dynamic and multifaceted nature of cognitive load. To address these limitations, our proposed framework integrates a comprehensive set of indicators, including physiological signals (heart rate, skin temperature, and resistance), behavioural measures (body amplitude, velocity, and frequency), facial action units (FAUs), gaze tracking, and pose dynamics. This multi-modal approach enhances cognitive load estimation accuracy while ensuring real-time adaptability, making it particularly well-suited for dynamic HRI scenarios. By leveraging state-of-the-art sensing technologies and advanced machine learning techniques, our framework provides a more nuanced and reliable assessment of cognitive load. Furthermore, unlike

existing solutions that often require specialized hardware (e.g., Tobii glasses for pupil tracking), our framework is designed to be adaptable and applicable to various task modalities, whether visual or auditory. This versatility ensures broad applicability across different HRI contexts. Additionally, by making our dataset publicly available, we aim to foster further research in cognitive load estimation, allowing the research community to validate our findings and contribute to advancements in the field. To further address the gap in research quantifying the effects of cognitive load on human pose, we conducted a pilot study investigating this relationship. This study aims to explore how cognitive load influences pose dynamics, providing additional insights that can enhance cognitive load estimation in HRI.

5.4 System Architecture

The goal of the system is to enable a humanoid robot to independently administer cognitive tasks and estimate user cognitive load using only its onboard sensors. This section outlines the robot platform, data acquisition pipelines, and the validation mechanism.

5.4.1 Robot Platform and Onboard Sensing

We employed the ARI humanoid robot for this study. For human pose estimation (HPE), we utilised an event camera-based model, consistent with our previous pilot study. Physiological measurements were obtained using the Shimmer sensor, while facial action units and gaze tracking were analysed with the OpenFace offline software. Additional information about the robot, sensors, and software is provided below.

ARI Robot

ARI is a social humanoid robot developed by PAL Robotics, equipped with a 10.1-inch touchscreen interface for displaying visual information and receiving user inputs. Its interactive capabilities and human-like design facilitated an engaging environment for studying human-robot interaction under varying cognitive load conditions.

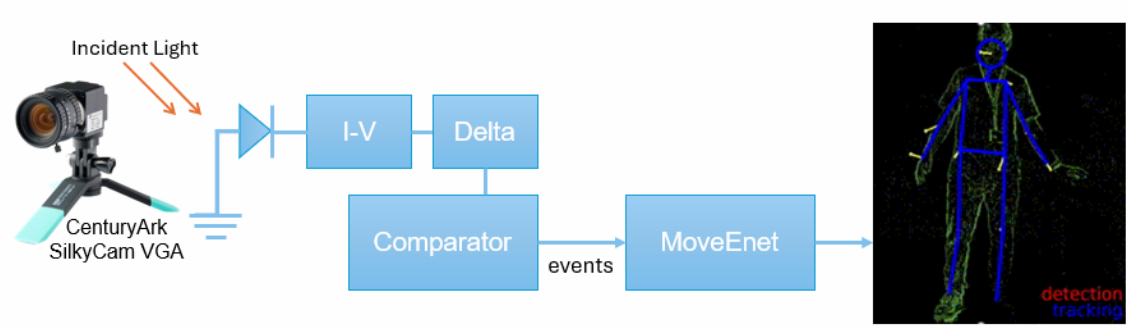
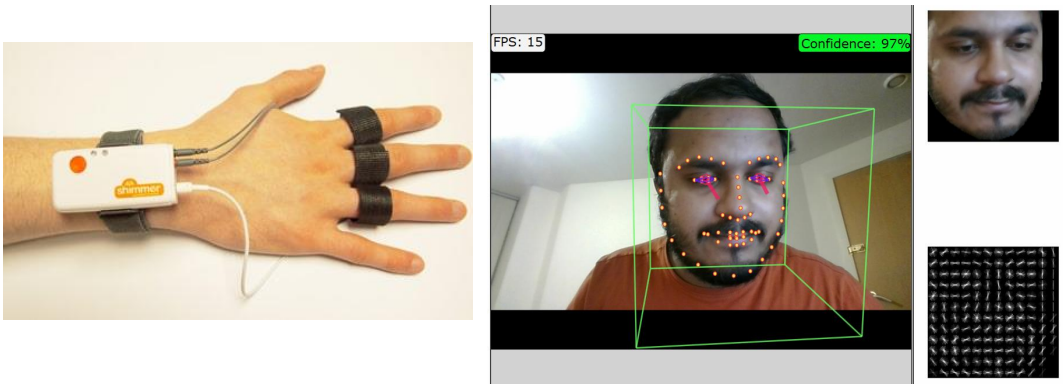


Figure 5.2: In CenturyArk SilkyCam VGA (event camera) each pixel continuously monitors changes in logarithmic light intensity and generates an event whenever the intensity change exceeds a predefined contrast threshold, producing asynchronous brightness-increase or brightness-decrease events rather than frame-based measurements. These events are fed to the MovEnet model to get human pose estimation.

Event Camera for HPE

Human Pose Estimation (HPE) is essential for accurately analysing human position, pose, and movement, particularly in real-time scenarios. While traditional HPE methods using RGB cameras have made significant progress, they often face challenges such as motion blur and low temporal resolution [259]. Event cameras address these limitations by capturing intensity changes in a scene at high temporal rates, operating independently of lighting conditions, and focusing only on motion-relevant data [91]. This results in reduced computational requirements, lower power consumption, and faster processing, making event cameras ideal for real-time applications in robotics. Some hybrid approaches, such as EventCap and EventHPE [422], combine event cameras with traditional frame-based data to leverage the strengths of both technologies but inherent limitations like data redundancy and asynchronous errors. Conversely, exclusive event-based solutions like LiftMono-HPE [327] face challenges with high computational demands, limiting their real-time applicability. Among event-based HPE methods, MoveEnet [150] stands out for its ability to process event streams directly, without relying on frame-based data. This lightweight approach ensures accurate and high-frequency pose estimation in real-time, fully utilizing the advantages of event cameras. For these reasons, MoveEnet was selected for our experiments. Figure 5.2 illustrates the event camera's circuit diagram and an example of pose estimation using MoveEnet [150]. In the pipeline, the raw asynchronous



a) Shimmer Sensor **b) OpenFace**

Figure 5.3: Shimmer sensor is used to measure heart-rate, skin temperature and conductance. OpenFace is used to measure FAUs and gaze tracking.

event stream is first accumulated over short temporal windows and converted into a structured representation compatible with the pose estimation network. MoveEnet then processes this event-derived representation to estimate 2D body keypoints at each time step. From these estimated keypoints, movement dynamics such as joint displacement, velocity, and temporal variation in pose configuration are computed.

Shimmer Sensor

We utilised the Shimmer3 GSR+ Unit, a sensor designed to capture physiological data such as heart rate, skin temperature, and skin conductance. It measures Galvanic Skin Response (GSR), or Electrodermal Activity (EDA), which increases with higher cognitive load [60]. The device monitors skin conductance between two electrodes attached to the fingers and captures Photoplethysmogram (PPG) signals via an optical pulse probe or ear clip, enabling heart rate estimation from blood volume changes. This dual functionality allows simultaneous tracking of electrodermal activity and cardiovascular metrics (Figure 5.3a).

OpenFace

OpenFace is an open-source toolkit designed for facial behaviour analysis, to monitor participants' facial expressions and eye movements [28]. OpenFace is capable of detecting facial landmarks, estimating head poses, recognizing facial action units

(FAUs), and tracking eye gaze. OpenFace employs the Facial Action Coding System (FACS) to identify specific facial muscle movements, providing insights into participants' emotional states and cognitive load during tasks. In Gaze Tracking, OpenFace estimates eye gaze direction by analysing eye position and orientation relative to the head, allowing assessment of visual attention and focus [28]. OpenFace operates in real-time using standard webcams, facilitating seamless integration into various research environments without the need for specialized hardware (Figure 5.3b).

Qualtrics

Qualtrics is an online survey platform renowned for its versatility and user-friendly interface, to collect subjective responses from participants. To assess participants' perceived workload, we employed the NASA Task Load Index (NASA-TLX), a widely recognized subjective workload assessment tool developed by NASA's Ames Research Center. The NASA-TLX evaluates perceived workload across six dimensions [124]. Participants rate each dimension after every task for both of the experiments, providing a measure of cognitive and physical workload after task performance.

5.5 Hypotheses- Main Study

The primary objective of these experiments is to explore how physiological, FAUs, Gaze, and behavioural measures are influenced by variations in cognitive load. The hypotheses are as follows:

H0: Task accuracy will decrease in high cognitive load condition as compared to the low cognitive load condition.

H1: Participants will exhibit higher average heart rates under high cognitive load conditions compared to low cognitive load conditions.

H2: Electrodermal activity (EDA) will increase in the high cognitive load condition compared to the low cognitive load condition.

H3: The frequency of facial action units (e.g., eyebrow raises, frowns), as detected by the RGB camera, will vary between high and low cognitive load conditions.

H4: Gaze patterns will differ across cognitive load levels, specifically: **H4a:** Blinking frequency will vary. **H4b:** Gaze angle changes will be observed.

H5: Movement dynamics, including: **H5a:** Amplitude, **H5b:** Frequency and **H5c:** Velocity, as measured by the event camera, will differ between high and low cognitive load conditions.

H6: Self-reported cognitive load will be higher in the high cognitive load condition than in the low cognitive load condition.

5.6 Pilot Study: Event Camera-Based Human Pose Estimation for Cognitive Load Analysis

To bridge the gap in research quantifying the impact of cognitive load on human pose, we conducted a pilot study to investigate this relationship. The study aimed to determine whether fluctuations in cognitive load affect human posture and movement patterns. Our work is among the first to systematically examine this interaction, providing valuable insights into the connection between cognitive load and body dynamics.

Human Pose Estimation (HPE), which involves detecting and tracking body postures, has proven to be a valuable tool for assessing cognitive load. Research suggests that body movements and postures adapt to varying cognitive demands, making it possible to infer an individual's cognitive state during a task [339, 213]. Specific behavioural cues, such as rubbing the forehead, shifting shoulders, moving legs, or becoming noticeably stiffer, are often associated with different levels of cognitive load [383, 132]. Human pose can be detected using an RGB camera and popular libraries such as OpenCV, MediaPipe [66], and OpenFace [29]. However, traditional RGB cameras operate synchronously at a fixed frame rate, limiting their ability to capture rapid or subtle movements that occur between frames. This restriction reduces their effectiveness in detecting fine-grained changes in human pose.

5.6 Pilot Study: Event Camera-Based Human Pose Estimation for Cognitive Load Analysis

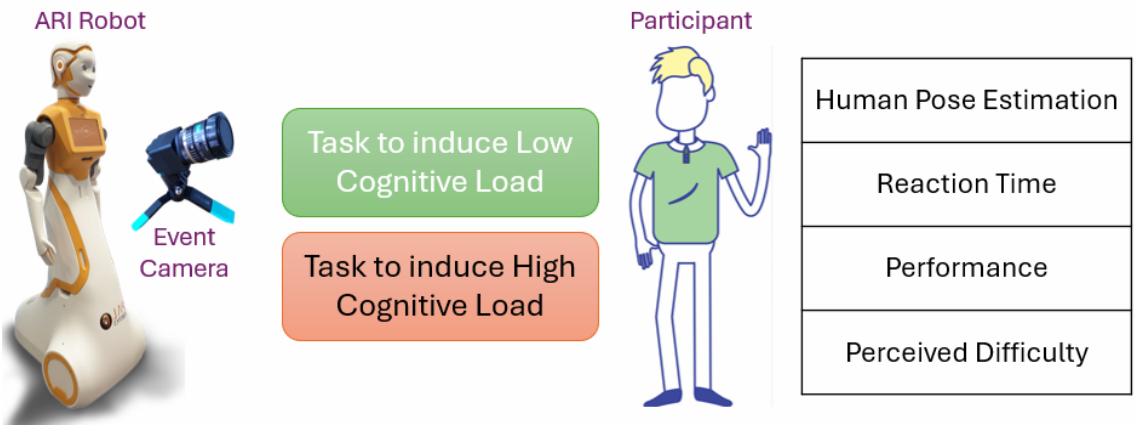


Figure 5.4: Study design to measure behavioural responses to different levels of cognitive load: the human subject is asked to perform a Stroop task with high or low cognitive load. Event-cameras recorded the human behaviour were used to measure pose, task difficulty was assessed using reaction time, accuracy, and perceived difficulty (through a questionnaire).

As discussed in Section 1.2, event cameras overcome these limitations by capturing visual changes asynchronously with high temporal resolution, enabling them to detect fast and subtle movements that standard cameras might miss due to lower frame rates and motion blur [396]. While extensive research has explored cognitive load assessment through physiological metrics such as pupil dilation, eye tracking [156], and heart rate variability [272], there is a significant gap in quantifying how human pose varies under different cognitive load conditions. Despite evidence linking cognitive load to posture and movement changes, few studies have systematically measured and analysed these variations [376, 4]. This study aims to bridge that gap by leveraging event camera technology to quantify and analyse human pose dynamics under varying cognitive loads, offering a new dimension to cognitive load assessment that complements existing behavioural measures.

This pilot study investigates behavioural responses to cognitive load by employing event camera-based human pose estimation (HPE). The central focus is on understanding how varying levels of cognitive load, induced through a Stroop task, influence participants' body posture and movement patterns. To complement the objective measurements, subjective evaluations were also included using a questionnaire designed to capture participants' perceptions of the cognitive demands associated with different tasks. Figure 5.4 presents an overview of this pilot study, illustrating

the cognitive load induction process and outlining how human pose data, reaction time, task performance, and perceived difficulty were systematically measured. The primary aim of this investigation is to determine whether cognitive load leads to observable changes in human movement and posture, and to evaluate the utility of event camera-based HPE as a viable method for capturing these variations. To explore this, cognitive load was experimentally manipulated by comparing high and low load conditions within the Stroop task paradigm. The study tested several hypotheses: (H1) self-reported cognitive load would be higher under the high load condition; (H2) reaction times would be longer in the high load condition; (H3) task accuracy would be lower when cognitive load was high; (H4) the amplitude of bodily movements would differ between the two load conditions; and (H5) the frequency of movements would also vary according to the level of cognitive load.

5.6.1 Pilot Study - Experimental Design

We designed a within-subject experiment in which cognitive load was manipulated at two levels (low vs. high) during task performance. This study was preregistered on aspredicted.org (https://aspredicted.org/2dd7j.pdf) and received ethical approval from the Bielefeld University Ethics Review Board (application no. 2024-126, dated 06.04.2024). The Stroop task [248], a widely used psychological test for assessing cognitive load and interference [154], was selected as the primary task. In this task, participants were presented with colour words (e.g., "red," "blue," "green," "yellow") written in either congruent or incongruent ink colours (e.g., the word "red" printed in blue ink) and were instructed to name the ink colour rather than the word itself. This design effectively measures cognitive load by creating a conflict between the automatic process of reading and the controlled process of identifying the ink colour [248]. The study was implemented using PsychoPy [391] and began with a practice phase to familiarize participants with the Stroop task format and procedures. This phase ensured they understood the task requirements and could perform it accurately. PsychoPy then randomly assigned participants to either the low cognitive load (LL) or high cognitive load (HL) condition first.

5.6 Pilot Study: Event Camera-Based Human Pose Estimation for Cognitive Load Analysis

In the low cognitive load condition, participants identified the ink colours of displayed words and pressed the corresponding keys on a keyboard, with each key labelled with a colour sticker to facilitate responses. In the high cognitive load condition, participants performed the same Stroop task but with an additional working memory challenge—they had to memorize a six-digit string and recall it at the end of the task. This secondary task was designed to further engage working memory, increasing cognitive load. After completing each task, participants rated their experience and perceived difficulty using the NASA Task Load Index (TLX), adapted to a seven-point Likert scale.

Table 5.1: Pilot Study - Participant Demographics

Participants	34
Gender	Male:25, Female:9
Age	23-55 (Mean: 32.03, SD: 6.86)
English Level	B2: 2, C1: 16, C2: 14, Native: 2
Highest Education	Bachelors: 1, Masters: 22, PhD: 11

Participants were recruited during the CapoCaccia Neuromorphic Workshop (CCNW) 2024. Participation was entirely voluntary, and individuals could withdraw at any time without any consequences. Each participant was guided to a designated room where the experiment was set up, standing in front of a projector screen. They were provided with an information sheet and asked to sign a consent form before beginning the experiment. To maintain simplicity and avoid the novelty effect, no robot was used in this setup. All instructions were displayed on the screen, and participants used a keyboard positioned on their preferred side (left or right) to input responses (a: red, s: green, d: yellow, f: blue). The entire procedure took approximately 15 minutes per participant. An event camera was placed in front of each participant to record body pose throughout the experiment. Personal data was handled with strict confidentiality and in full compliance with GDPR, with only movement tracking data being retained. Figure 5.5 illustrates the experimental setup.

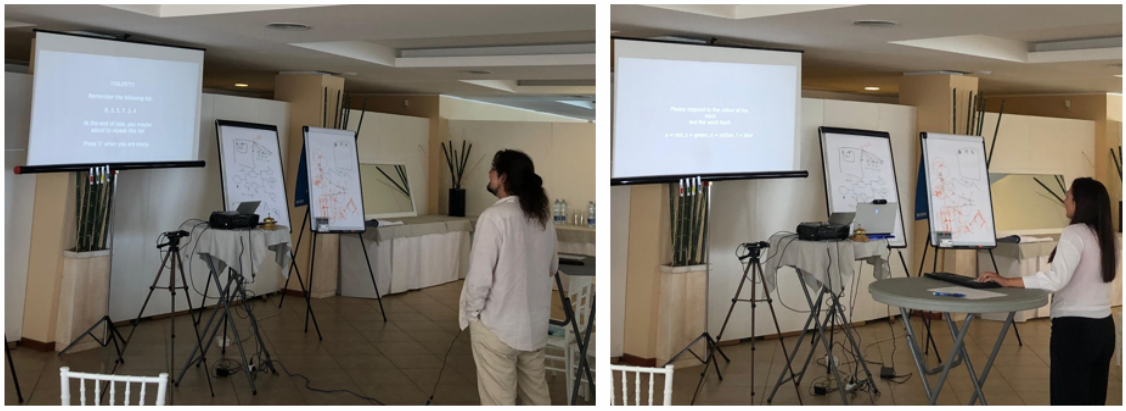


Figure 5.5: Participants performing the study. The task, along with the instructions, is displayed on the projector screen.

5.6.2 Pilot Study - Results and Discussion

We collected body pose data for each timestamp, along with participants' performance and reaction times for the Stroop task under both conditions (LL and HL). Additionally, we recorded performance and reaction time data for congruent and incongruent trials within the Stroop tasks. Participant demographics, including age, gender, English proficiency, and education level, were obtained from the information forms. The completed questionnaires provided insights into participants' experiences and perceived task difficulty for each condition. A total of N=34 participants took part in the study, with 25 identifying as male and 9 as female. The average participant age was 31.56 years. The sample included 1 Master's student, 22 Ph.D. students, and 11 individuals who had completed their Ph.D. In terms of English proficiency, 2 participants were at the B2 level, 16 at the C1 level, 14 at the C2 level, and 2 were native speakers. Table 5.2 provides a summary of these results.

Analysis of the NASA TLX questionnaire data confirmed that our tasks successfully induced distinguishable levels of cognitive load in participants. The NASA TLX scores for mental demand, physical demand, temporal demand, performance, effort, and frustration were averaged to generate an overall score for each condition.

To compare the overall scores between high and low cognitive load conditions, we applied the Wilcoxon signed-rank test, a non-parametric method suitable for paired samples. Since this test does not assume a normal distribution of differences,

Metric	Value
Avg. Reaction Time of Males (Low CL, High CL)	1.450 sec
Avg. Reaction Time of Females (Low CL, High CL)	1.471 sec
Avg. Reaction Time (Low CL)	1.344 sec
Avg. Reaction Time (High CL)	1.367 sec
Percentage Correct (Low CL)	93.202%
Percentage Correct (High CL)	92.630%
Avg. Reaction Time (Low CL, Congruent)	1.341 sec
Avg. Reaction Time (Low CL, Incongruent)	1.347 sec
Avg. Reaction Time (High CL, Congruent)	1.366 sec
Avg. Reaction Time (High CL, Incongruent)	1.367 sec

Table 5.2: Summary of Cognitive Load (CL) Experiment Results

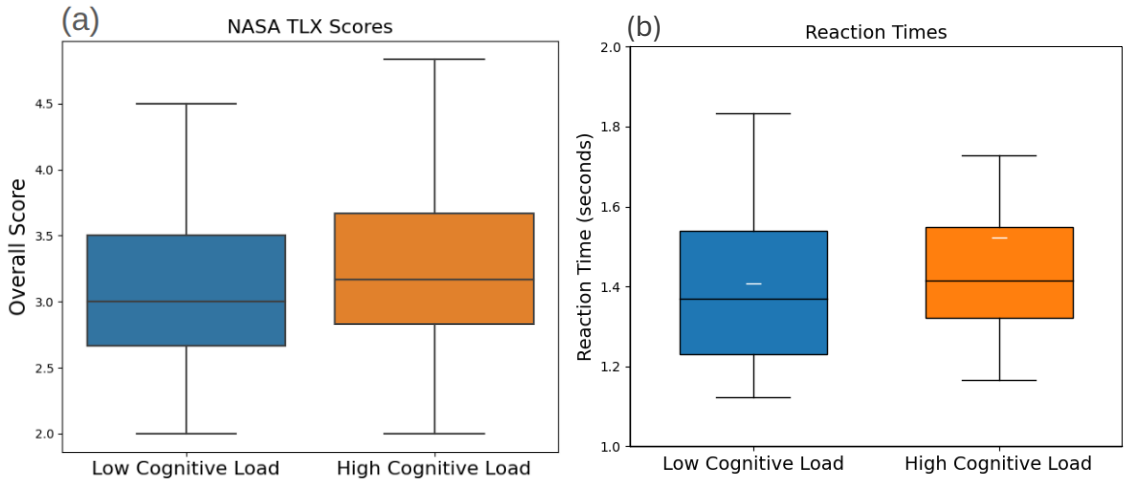


Figure 5.6: NASA TLX score and Reaction time under low and high cognitive load conditions.

it is particularly appropriate for ordinal data collected from the Likert scale. The Wilcoxon signed-rank test revealed a statistically significant difference between the high (M=3.22, SD=0.66) and low (M=3.02, SD=0.64) cognitive load conditions (W = 48.5, $p < .001$, $r = 0.47$). Figure 5.6(a) presents a box plot illustrating the NASA TLX scores for both conditions. These results support hypothesis H1, indicating that cognitive load significantly influences participants' perceived workload and confirming the effectiveness of our task design in inducing varying levels of cognitive load.

Analysis of reaction times revealed that participants took slightly longer to respond under high cognitive load (M=1.37, SD=0.26) compared to low cognitive load (M=1.34, SD=0.25). This pattern was consistent across both congruent and incongruent stimuli. Previous research by Aditya et al. [186] has shown that

5.6 Pilot Study: Event Camera-Based Human Pose Estimation for Cognitive Load Analysis

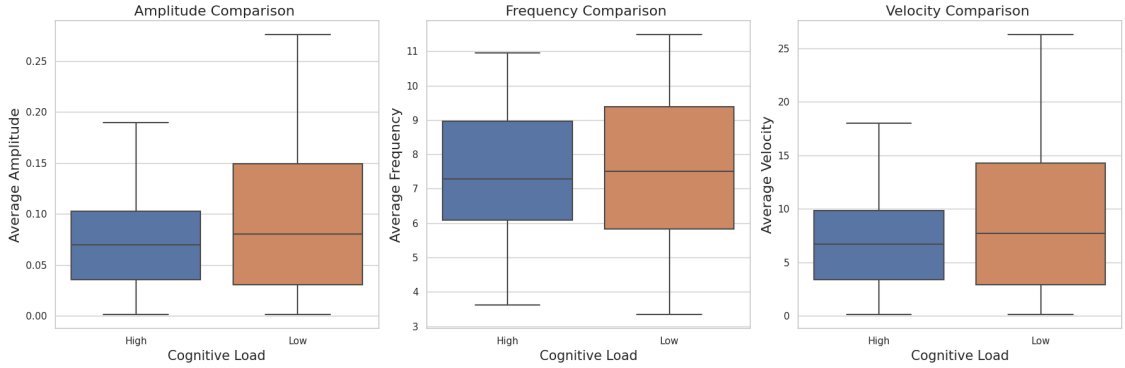


Figure 5.7: The average amplitude, frequency, and velocity of movements under high and low cognitive load conditions. The Wilcoxon signed-rank tests confirmed that these differences are significant.

reaction times can vary by gender. In our study, the average reaction time for male participants across both conditions (LL-HL) was 1.45 seconds, while for female participants, it was 1.47 seconds. To compare reaction times between cognitive load conditions, we conducted a Wilcoxon signed-rank test (W=206, p=.005, r=0.30), which indicated a statistically significant difference. These findings support hypothesis H2, demonstrating that increased cognitive load leads to a measurable and significant delay in response times. Figure 5.6(b) presents a box plot illustrating reaction times under both conditions. This result suggests that higher cognitive load negatively impacts performance by slowing participants' response times.

In terms of the correctness of answers, the percentage of correct responses was slightly higher in the low cognitive load condition (93.20%) compared to the high cognitive load condition (92.63%). However, the Wilcoxon signed-rank test for correctness resulted in W=82.0, p=0.238, and r=0.654, indicating no significant effect of cognitive load on response accuracy. This finding does not support hypothesis H3. The results suggest that while increased cognitive load leads to a significant slowdown in reaction times, it does not have a substantial impact on response accuracy.

The Wilcoxon signed-rank tests on human pose data collected with an event camera indicated differences in movement amplitude, frequency, and velocity between high and low cognitive load conditions. Specifically, average amplitude (W=251, p = .005, r=0.24) and velocity (W=250, p = .005, r=0.24) were higher under

Measure	W	p-value	r
Reaction Time	206	.005*	0.30
Correctness	82	0.237	0.25
Amplitude	251	.005*	0.24
Frequency	248	.077	0.29
Velocity	250	.005*	0.24

Table 5.3: Wilcoxon Signed-Rank Test Results. * represents a significant p-value.

low cognitive load compared to high cognitive load. Frequency results were not significant in this case (W=248, p = .077, r=0.29). These results suggest that movement characteristics are influenced by cognitive load, supporting hypotheses H4 and H5. Figure 5 presents the box plots of these findings, while Table 5.3 summarizes the Wilcoxon signed-rank test results.

The pilot findings demonstrated that pose dynamics derived through onboard event-based sensing are effective and reliable behavioural indicators of cognitive load. Consequently, these pose-based features were integrated into the feature extraction and classification pipeline of the main study. A detailed description of the pilot study, including comprehensive statistical methods and results, can be found in [12].

5.7 Study Design

We designed a within-subject experiment in which the level of cognitive load during task performance was manipulated (i.e., low vs. high cognitive load). This experiment was preregistered on aspredicted.org (https://aspredicted.org/g7sg-fs23.pdf) and received ethical approval from Sheffield Hallam University (ID: ER70783970) and Bielefeld University (ID. 2024-244) Ethics Review Board. Two experiments were conducted: one involving visual stimuli and the other involving auditory stimuli. The order of stimuli was counterbalanced to mitigate carryover effects. A 2–3-minute resting period was introduced after each task in both experiments.

5.7.1 Visual Task

Similar to the pilot study described in Section 5.6.1, the visual stimuli involved a Stroop task [249]. In this version, instead of pressing a button on a keyboard,

participants were instructed to verbally name the ink colour rather than read the word itself. This task introduces cognitive conflict, as the automatic process of reading the word interferes with the controlled process of identifying the ink colour.

In the low cognitive load (LL) condition, 80% of the trials presented congruent stimuli. In contrast, the high cognitive load (HL) condition reversed this ratio, with 20% congruent and 80% incongruent stimuli, thereby increasing conflict and inducing greater cognitive effort. Each condition consisted of 25 trials, and the order of presentation was randomised for each participant to control for potential order effects. Participants responded to the Stroop stimuli by verbally naming the ink colour of the word displayed on the robot's screen.

5.7.2 Auditory Task

The auditory stimuli were based on the N-back test, a well-established cognitive task used to evaluate working memory. Participants were required to track a sequence of auditory stimuli and determine when the current stimulus matched one presented "N" steps earlier. Cognitive load was systematically manipulated by increasing the value of "N," which heightened memory demands and required greater cognitive effort for stimulus matching [282][257].

In the low cognitive load (1-back) condition, participants listened to a sequence of 50 digits spoken by the robot, with a one-second interval between digits. They were instructed to respond with "yes" whenever the current digit matched the one presented immediately before. For example, in the sequence "3, 7, 7," participants responded "yes" upon hearing the second "7." In the high cognitive load (2-back) condition, participants had to identify when the current digit matched the one presented two steps earlier. For instance, in the sequence "5, 2, 5," a "yes" response was required upon hearing the second "5." Each condition began with two practice sessions to familiarize participants with the task and provide feedback for better understanding. The main experimental phase included 50 trials per condition, during which no feedback was given to ensure unbiased performance data. To capture participants' verbal responses accurately, we used OpenAI Whisper [151], a state-

of-the-art speech-to-text tool known for its high transcription accuracy and noise resilience. Whisper enabled real-time transcription, ensuring precise data collection while minimizing errors associated with manual transcription. Additionally, all transcriptions were manually verified against the recorded audio of the entire task.

5.7.3 Data Collection and Ground Truth Labelling

To support classification of cognitive load, we implemented a multimodal data collection protocol that included physiological, behavioural, performance-based, and subjective measures. Physiological data were acquired using a Shimmer3 wearable device, which recorded heart rate (HR), electrodermal activity (EDA), and skin temperature. These signals were not used as input for classification but served exclusively as ground truth labels to validate the features derived from the robot's onboard sensors. Behavioural data were collected using the robot's embedded sensors. Pose dynamics were estimated from event camera input, while facial action units (FAUs), gaze direction, and head orientation were extracted from RGB video using the OpenFace toolkit. These features capture subtle changes in user behaviour under varying cognitive demands.

Performance metrics included task accuracy and reaction time. Accuracy was computed as the proportion of correct responses per trial, and reaction time was defined as the interval between stimulus presentation and participant response. These measures were used to assess behavioural outcomes under different load conditions. Subjective workload was assessed using the NASA Task Load Index (NASA-TLX). After each task, participants rated their perceived workload across six dimensions, providing an additional reference point for cognitive state classification. An overview of the data collection framework is shown in Figure 5.8, which extends the pipeline presented in Figure 5.1. Together, these modalities provide a robust foundation for evaluating the effectiveness of the robot sensing model in distinguishing between high and low cognitive load states.

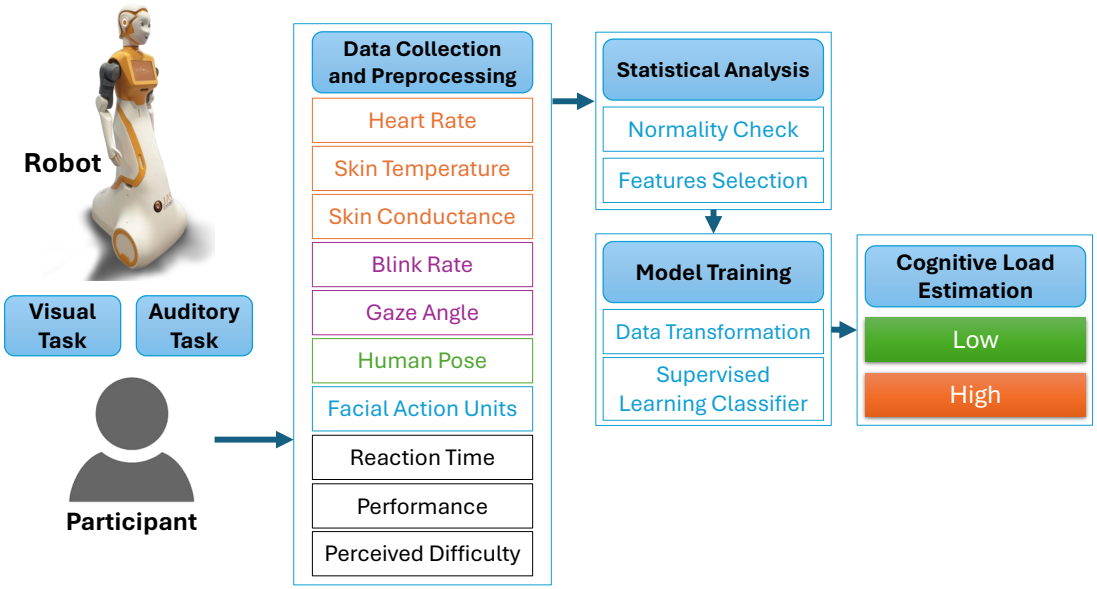


Figure 5.8: Extended block diagram illustrating the detailed workflow of cognitive load estimation, including data collection, preprocessing, statistical analysis, model training, and classification into low and high cognitive load states.

5.7.4 Participants Recruitment and Demographics

An open call for participation was distributed via mailing lists and social media. Participation was voluntary, with the option to withdraw at any time. On the experiment day, participants received an information sheet and consent form. After signing, a wrist sensor was attached, and they were positioned in front of the ARI robot. To mitigate novelty effects and establish a cognitive baseline, they first completed a reverse digit span task, followed by a short break. The system randomly assigned participants to start with either the visual or auditory experiment and determined task difficulty (high or low cognitive load). After each task, they completed the NASA-TLX questionnaire and took a brief break. The entire session lasted approximately 20 minutes per participant. All data were handled confidentially and in compliance with GDPR. Demographic information, including age, gender, English proficiency, and education level, was collected via an information form. A total of $N = 32$ (details of the G-power calculation are available in the supplementary material and on the project website.) participants completed the study (ages 19–37,

$M = 25.12$, $SD = 4.81$). A detailed description of the demographic information of participants can be found in Table 5.4.

Table 5.4: Participant Demographics

Participants	32
Gender	Male:19, Female:13
Age	19-37 (Mean: 25.12, SD: 4.81)
English Level	B1: 3, B2: 8, C1: 11, C2: 6, Native: 4
Highest Education	High School: 4, Bachelors: 9, Masters: 15, PhD: 2

5.8 Statistical Analysis

To systematically assess the differences between high cognitive load (HCL) and low cognitive load (LCL) conditions, we conducted comprehensive statistical analyses. Paired differences between the two conditions were evaluated for normality using the Shapiro-Wilk test [162]. Data satisfying the normality assumption ($p > 0.05$) were analysed using paired t-tests, while non-normally distributed pairs were assessed using the Wilcoxon signed-rank test. This ensured rigorous and accurate hypothesis testing appropriate to our within-subject experimental design.

5.8.1 Visual Task Results

In the visual task, cognitive load had a clear impact on participant performance and behavioural responses. Task accuracy was notably higher under LCL (M = 96.68%, SD = 3.56%) than HCL (M = 91.71%, SD = 3.90%; $p < .001, r = .97$), while reaction times increased under HCL (M = 1.35s, SD = 0.31s) relative to LCL (M = 1.21s, SD = 0.22s; $p = .009, d = .12$). Participants also reported greater perceived workload in the HCL condition, as reflected by NASA-TLX scores ($p = .001, d = .67$), reinforcing the alignment between subjective and objective load indicators. Analysis of behavioural features revealed that movement amplitude ($p = .010, r = .75$) and velocity ($p = .006, r = .77$) decreased under higher cognitive demands, indicating a more constrained posture likely linked to increased mental effort. While pose frequency did not differ significantly ($p = .091, r = .67$), gaze angle varied between

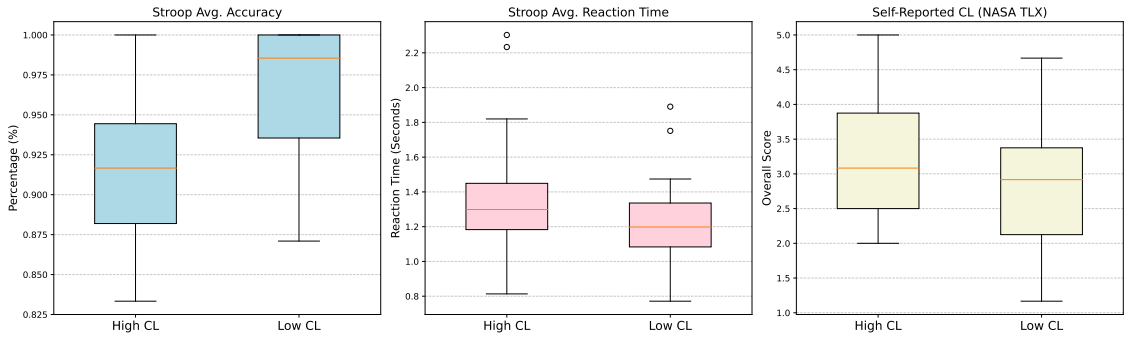


Figure 5.9: Visual Task - Stroop task performance metrics: average accuracy (%), average reaction time (seconds), and self-reported cognitive load under high and low cognitive load conditions.

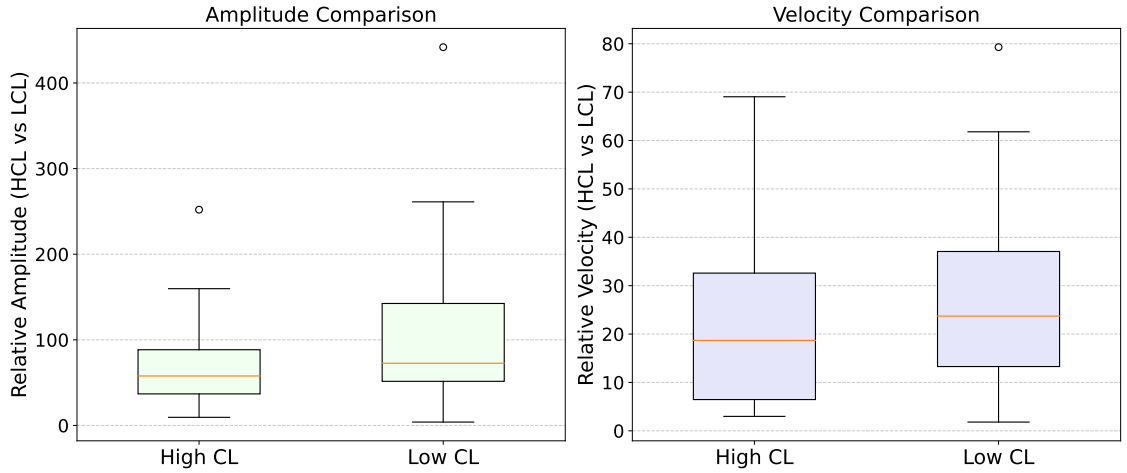


Figure 5.10: Visual Task - Box plot showing amplitude and velocity of behavioural measures under high and low cognitive load conditions.

conditions ($p = .035, r = .71$), suggesting visual attention shifts under load. In contrast, blink frequency ($p = .123, d = .28$) and facial action units ($p = .802, d = .09$) showed minimal differences, suggesting limited responsiveness of these markers in the context of visual tasks. Physiological ground truth metrics exhibited marginal, non-significant changes; heart rate ($p = 0.488, d = .12$), electrodermal activity (EDA; $p = .942, d = .01$), and skin temperature (LCL: M=30.33°C, SD=2.15°C; HCL: M=30.36°C, SD=2.13°C) all remained consistent, highlighting the limited sensitivity of short-duration tasks for physiological signals.

Table 5.5: Visual stimuli statistical test results for hypotheses, including test types, statistics, and p-values. Significant p-values are highlighted in bold. * shows the ground truth results. (EC) refers to the parameters obtained from the event camera.

Hyp.	Measure	Test Used	Statistic	P-Value
H0a	Stroop Avg. Accuracy	Wilcoxon	14	**<.001**
H0b	Stroop Avg. Reaction Time	Wilcoxon	117	**.009**
H1	Heart Rate*	t-test	0.702	0.487
H2	Electrodermal Activity*	t-test	0.073	0.941
H3	Facial Action Units	t-test	0.252	0.802
H4a	Avg. Blink Frequency	t-test	1.587	0.122
H4b	Avg. Gaze Angle	Wilcoxon	152	**.035**
H5a	Avg. Pose Amplitude (EC)	Wilcoxon	129	**.010**
H5b	Avg. Pose Frequency (EC)	Wilcoxon	173	0.090
H5c	Avg. Pose Velocity (EC)	Wilcoxon	121	**.006**
H6	Self-Reported (NASA TLX)	t-test	3.573	**.001**

5.8.2 Auditory Task Results

Results from auditory tasks similarly indicated a robust impact of cognitive load on participants' performance and behaviour. Task accuracy decreased significantly under HCL (M=88.75%, SD=11.40%) compared to LCL (M=99.38%, SD=3.10%; $p < .001, r = .95$), with participants reporting significantly higher subjective cognitive load via NASA-TLX ($p < .001, d = 1.15$), reinforcing the effectiveness of the cognitive manipulation. Behavioural measures revealed significant reductions in movement amplitude and velocity under HCL conditions ($p = .002, r = .79$), confirming the sensitivity of pose dynamics across modalities. Facial action units (FAUs) also showed significant differences between conditions ($p = .012, d = .46$), underscoring their sensitivity to cognitive strain in auditory contexts. However, gaze-related measures, including gaze angle ($p = 0.548, r = .36$) and blink frequency ($p = 0.472, d = .12$), did not differ significantly, reflecting modality-specific characteristics of auditory tasks. Ground truth physiological data presented mixed outcomes; heart rate significantly increased under HCL ($p = .032, r = .71$), but electrodermal activity

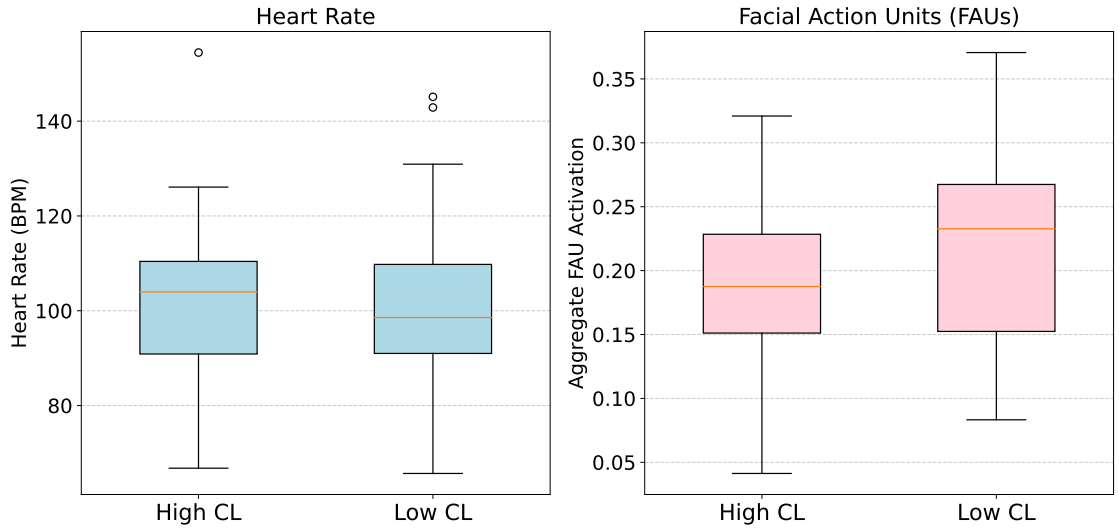


Figure 5.11: Box Plots for Heart Rate and Facial Action Units (FAUs) under high and low cognitive load conditions.

(EDA; $p = 0.573, d = .16$) remained stable across conditions. Skin temperature exhibited minor, non-significant differences (LCL: M=30.42°C, SD=2.10°C; HCL: M=30.27°C, SD=2.20°C), consistent with findings from visual tasks.

Detailed statistical outcomes for all measured indicators are summarised comprehensively in Tables 5.5 and 5.6, with additional visualisations provided in supplementary materials available on the project website.

5.9 Training Classifier

To classify cognitive load using features derived solely from the robot's onboard sensors, we trained a supervised model to distinguish between high and low cognitive load states. Physiological signals from the Shimmer device were used only to label data instances for supervised training and evaluation, not as classifier inputs. All feature vectors were constructed using facial, gaze, pose, and performance indicators described earlier.

Before training, we applied a dimensionality reduction process to improve class separability and computational efficiency. While the original feature set spanned 22 variables, many showed interdependence and overlapping distributions. Kernel Linear Discriminant Analysis (Kernel LDA) was employed to project the data into a

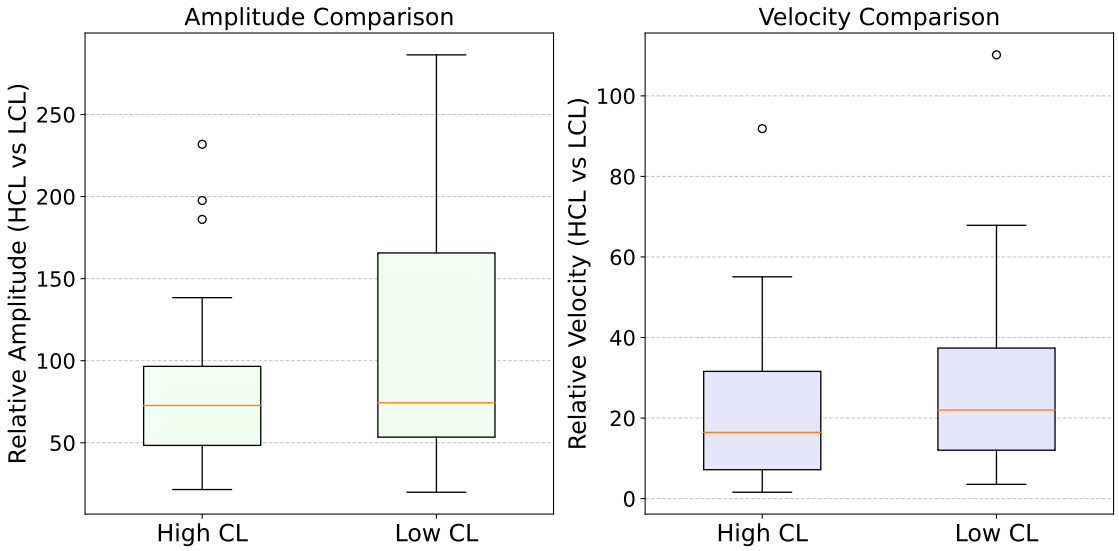


Figure 5.12: Box plot showing amplitude and velocity of behavioural measures under high and low cognitive load conditions.

non-linear discriminative space. Compared to PCA and linear LDA, Kernel LDA achieved the highest Silhouette Score (0.7088), which measures how well samples are clustered by comparing within-class cohesion to between-class separation, thereby confirming improved class compactness and discriminability. This transformation is compatible with real-time applications, although classification in this study was performed offline (more information about feature transformation can be found in the supplementary material).

We benchmarked six classifiers on the transformed feature set: Random Forest (RF), AdaBoost (AB), Logistic Regression (LR), Decision Tree (DT), Naive Bayes (NB), and Support Vector Machine (SVM). The dataset (128 labelled instances) was split into 80% training and 20% testing sets, ensuring participant independence across folds to avoid data leakage. Performance was evaluated using accuracy, precision, recall, and F1-score metrics. Ensemble models showed the highest predictive accuracy, with Random Forest achieving 97.4% and AdaBoost 94.9%. Logistic Regression performed comparably to AdaBoost, suggesting that the feature space was close to linearly separable after transformation. Decision Tree and Naive Bayes achieved moderate accuracy (89.7%). In contrast, SVM underperformed at 73.4%. This lower performance may be attributed to several factors. First, the relatively small dataset

Table 5.6: Statistical test results for various hypotheses, including test types, statistics, and p-values. Significant p-values are highlighted in bold. * shows the ground truth results. (EC) refers to the parameters obtained from the event camera.

Hyp.	Measure	Test Used	Statistic	P-Value
H0	N-Back Avg. Accuracy	Wilcoxon	6	**<.001**
H1	Avg. Heart Rate*	Wilcoxon	150	**.032**
H2	Avg. Electrodermal Activity*	t-test	0.569	.573
H3	Facial Action Units	t-test	2.635	**.012**
H4a	Avg. Blink Frequency	t-test	0.728	.471
H4b	Avg. Gaze Angle	Wilcoxon	231	.548
H5a	Avg. Amplitude (EC)	Wilcoxon	106	**.002**
H5b	Avg. Frequency (EC)	Wilcoxon	261	.963
H5c	Avg. Velocity (EC)	Wilcoxon	106	**.002**
H6	Self-Reported (NASA TLX)	t-test	6.438	**<.001**

size (128 instances) can make SVM models sensitive to hyperparameter selection and kernel configuration. Second, the transformed feature space appears to favour either linear decision boundaries (as indicated by Logistic Regression performance) or ensemble-based partitioning strategies, whereas the SVM may have struggled to identify an optimal margin under the chosen kernel and regularisation settings. Additionally, ensemble methods such as Random Forest are inherently more robust to feature correlations and small-sample variability, which may explain their superior generalisation performance in this setting. Full results are summarised in Table 5.7. Random Forest was selected as the final model for its high accuracy, robustness to overfitting, and interpretability through feature importance rankings. Its relatively low computational overhead also makes it suitable for future deployment in real-time onboard classification on the robot platform.

5.10 Discussion and Future Work

This study demonstrates that a socially interactive robot can successfully estimate human cognitive load using exclusively its onboard sensing capabilities, eliminating

the dependence on external perception systems or wearable devices during inference. By autonomously delivering cognitive tasks and simultaneously capturing relevant behavioural indicators through embedded cameras and audio interfaces, our robot-centric framework confirms the viability of fully embedded cognitive state monitoring. The high accuracy achieved through ensemble learning models further supports the practical feasibility of robot-exclusive sensing in real-world human-robot interactions.

The behavioural indicators examined exhibited varying sensitivity depending on the task conditions, highlighting important modality-specific considerations. Task performance measures such as accuracy and reaction time consistently declined under increased cognitive load, reinforcing their robustness as objective behavioural metrics. Pose dynamics derived from the robot's event camera emerged as particularly reliable, consistently showing reduced movement amplitude and velocity under high cognitive load conditions across both visual and auditory tasks 5.12. These pose-based findings align closely with our earlier pilot study, reinforcing the utility of event-camera-based posture analysis for cognitive load assessment. In contrast, gaze and facial action unit (FAU) metrics showed task-dependent patterns, underscoring their sensitivity to specific interaction contexts. Gaze angle differences were significant primarily in visual tasks, reflecting direct visual engagement, whereas FAUs displayed clearer differentiation in auditory contexts, possibly due to increased expressive responses during verbal interactions. These modality-specific insights suggest future systems may benefit from adaptive, task-aware feature selection mechanisms.

Physiological data, utilised solely as ground truth for validation, provided valuable reference points without compromising the deployability or real-world applicability of the robot's embedded sensing system. This separation ensures that robot-derived behavioural features alone are sufficient for accurate cognitive load classification, enhancing the scalability and practicality of our approach for diverse application areas such as education, healthcare, and collaborative workspaces. Although the current classifier was trained and validated offline, the developed computational framework readily supports real-time deployment, with both feature extraction and classification algorithms optimised for onboard integration. Future research will thus prioritise transitioning this cognitive monitoring system into a real-time, closed-loop

Table 5.7: Performance Comparison of Classifiers Based on Accuracy and F1-Score Across Cognitive Load Levels.

Classifiers	AB	DT	LR	NB	SVM	RF
Accuracy	0.94	0.89	0.94	0.89	0.73	**0.97**

Cognitive Load Level	F1-Score					
	AB	DT	LR	NB	SVM	RF
Low	0.95	0.89	0.95	0.90	0.74	**0.97**
High	0.94	0.90	0.94	0.89	0.72	**0.97**

adaptive framework. Such adaptation will enable robots not only to detect cognitive states but also dynamically adjust their interaction strategies, such as modulating instructional complexity or pacing interactions based on user cognitive demands.

Additionally, future studies should focus on expanding dataset diversity by including a broader participant demographic and employing more complex, ecologically valid tasks that mirror real-world demands. Transitioning toward continuous, spontaneous interactions instead of discrete, stimulus-driven tasks will further enhance the robustness and generalizability of the cognitive load detection framework. Ultimately, this line of research promises significant advancements towards genuinely adaptive, cognitively aware robotic systems capable of seamless integration into everyday human environments.

5.11 Summary

This chapter presented a multimodal approach to estimating cognitive load during human–robot interaction, integrating physiological, behavioural, and task-based indicators. A pilot study using event camera-based human pose estimation explored how posture and movement respond to varying cognitive demands. Building on this, the main study introduced a robot-centric framework that used embedded sensors:

[0]Muhammad Aitsam lead the project, designed the experiments, prepared the experimental setup, and collected the datasets (Pilot Study: Capocaccia Workshop, Italy; Main Study: Sheffield Hallam University). Dimitri Lacroix contributed to the experimental design and statistical analysis. Gaurvi Goyal supported the human pose estimation model setup and data collection for the pilot study. The studies were supervised by Chiara Bartolozzi and Alessandro Di Nuovo, who also contributed to the study design and manuscript preparation. Ethical approval was obtained from Bielefeld University and Sheffield Hallam University.

event and RGB cameras, microphones, and display interfaces to administer tasks and capture responses without external instrumentation. Features from gaze, facial expressions, speech, pose, and task performance were extracted and used to train machine learning models for cognitive load classification. The study demonstrated the viability of onboard, multimodal sensing for cognitive state estimation, laying the foundation for adaptive robotic systems capable of responding to human cognitive states in real time.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis explored the integration of neuromorphic computing and event-based vision in interactive robotics, focusing on spiking neural networks (SNNs), event-driven sensory processing, and their deployment on specialized neuromorphic hardware. A systematic review of neuromorphic computing and vision provided a comprehensive understanding of their role in enhancing robotic perception, cognition, and adaptability. Through an in-depth analysis of SNN learning mechanisms, ANN-to-SNN conversion methods, and real-time inference strategies, the research demonstrated the feasibility of deploying neuromorphic models on SpiNNaker and similar platforms. One of the fundamental contributions of this work is the implementation of event-based vision techniques for robotic perception. Unlike conventional frame-based vision, event-driven sensing enables robots to process dynamic environments with minimal latency and power consumption. This was demonstrated through practical applications such as real-time gesture recognition and machinery fault detection using event-based frequency mapping (EBFM). Additionally, cognitive load estimation was explored to optimise human-robot interaction (HRI), utilizing multimodal approaches that combine event-camera-based human pose estimation with physiological and behavioural indicators. These findings highlight the potential of neuromorphic computing to enhance energy efficiency, real-time adaptability,

and robustness in autonomous robotic systems. The research also addressed key challenges associated with neuromorphic computing, including scalability, hardware constraints, and the need for efficient data processing pipelines. By optimising SNN inference on neuromorphic hardware, improving event-based vision frameworks, and proposing novel cognitive load estimation methods, this thesis contributes to the advancement of neuromorphic robotics. Overall, the work presented in this thesis paves the way for biologically inspired, event-driven robotic systems that can operate efficiently in real-world interactive environments.

Novelty & Impact

This dissertation can be used as a self-consistent neuromorphic cognition loop that begins with asynchronous sensory events and ends with behaviour that adapts to a human partner in real time. A unified algorithmic pipeline should be able to converts raw event streams into sparse spike trains, routes them through timing-precise spiking networks hosted entirely on-chip, and fuses the resulting perceptual codes with physiological and behavioural cues to infer the user's cognitive state. The same spike domain is preserved from sensor to actuator, eliminating frame reconstruction, clock-driven batching and off-board post-processing. The loop therefore operates at microsecond granularity while remaining within sub-watt power budgets. In three exemplar tasks, gesture-based guidance, vibration fault diagnosis and cognitive-state-aware interaction, the approach delivers state-of-the-art accuracy under lighting, motion and latency constraints that disable conventional deep-learning systems. By publishing open code, event datasets and hardware deployment scripts, the work offers a reproducible scaffold that other researchers can extend module by module, providing a practical blueprint for neuromorphic robots that must perceive, decide and collaborate under the energy and timing limits of real-world settings.

6.2 Future Direction

While this thesis has contributed significantly to the integration of neuromorphic computing and event-based vision in interactive robotics, several directions remain open for future exploration to enhance the efficiency, adaptability, and real-world deployment of these technologies.

6.2.1 Enhancing Neuromorphic Computing for Interactive Robotics

In this thesis, we systematically reviewed and deployed spiking neural networks (SNNs) for interactive robotics, focusing on real-time processing and energy efficiency. However, one limitation is the reliance on a limited number of neuromorphic chips, primarily SpiNNaker. Future research should explore the use of alternative neuromorphic hardware platforms, such as Intel's Loihi 2, BrainScaleS, and TrueNorth, to benchmark and optimise SNN models for different robotic applications. Additionally, integrating neuromorphic chips directly with robotic platforms, rather than relying on external processing, will reduce latency and improve real-time adaptability. Furthermore, the work presented here primarily focused on neuromorphic computing applications in a limited range of robotic functionalities, such as motor control and decision-making. Future studies should expand on this by investigating how neuromorphic processing can be applied to more complex cognitive functions, such as long-term memory formation and autonomous decision-making in unstructured environments. Conducting experiments with humanoid robots, rather than wheeled robots, will further advance social neuromorphic robotics by enabling more natural human-robot interactions.

6.2.2 Advancing ANN-to-SNN Conversion and Training Techniques

One of the contributions of this thesis was the systematic evaluation of ANN-to-SNN conversion methods. However, the results showed that converted SNNs often suffer

from lower accuracy compared to their ANN counterparts. Future work should focus on improving conversion algorithms by preserving temporal dynamics and optimising spike-based activation functions. Another important direction is to refine neuromorphic learning algorithms to enable on-chip adaptation in real-time robotic tasks. While this thesis examined various SNN learning mechanisms, such as reinforcement learning-based SNN training remains relatively an underexplored area that could significantly enhance robotic adaptability. Future research should investigate reinforcement learning techniques tailored for SNNs, enabling neuromorphic robots to learn from experience without external supervision.

6.2.3 Improving Event-Based Vision for Robotics

This thesis demonstrated the advantages of event-based vision in gesture recognition and industrial machines fault monitoring. However, several challenges remain in making event-based vision more robust and adaptable for real-world applications. One key limitation identified was the need for improved event-data processing pipelines to handle varying lighting conditions and dynamic environments. Future research should focus on integrating advanced deep learning models with event-based vision to enhance feature extraction and classification accuracy. Additionally, the proposed event-based gesture recognition system showed promising results but was tested in controlled conditions. Future work should extend this system to operate in unconstrained environments where variations in hand size, background noise, and occlusion present significant challenges. Further, event-based visual SLAM (Simultaneous Localization and Mapping) was identified as an area with great potential but was not fully explored in this thesis. Future work should investigate how event-based vision can be integrated with neuromorphic computing for real-time SLAM applications in robotics.

6.2.4 Expanding the Applications of Event-Based Frequency Mapping (EBFM)

The event-based frequency mapping (EBFM) technique proposed in this thesis demonstrated its effectiveness for real-time machinery fault detection. However, one limitation was its sensitivity to environmental conditions such as flickering lighting and vibrating surfaces. Future research should focus on making EBFM more robust to such disturbances by incorporating machine learning techniques for adaptive thresholding and feature extraction. While EBFM was applied to industrial monitoring, its potential in other domains, such as biomedical signal analysis and structural health monitoring, remains unexplored. Future work should investigate how event-based frequency mapping can be adapted for use in medical diagnostics, such as detecting neurological disorders through tremor analysis.

6.2.5 Optimisation and Benchmarking of Event-Data Processing Pipelines

The event-data processing pipeline developed in this thesis has demonstrated strong performance in handling various event-data formats. However, the field of event-based vision is rapidly evolving, and several new frameworks, such as Prophesee's Metavision SDK 5 and iniVation's DVXplorer SDK, have introduced advanced event-processing techniques. Future research should conduct a comparative analysis of our pipeline against these newer solutions to benchmark key performance metrics such as accuracy, execution speed, and power efficiency.

6.2.6 Refining the Experimental Design for Cognitive Load Estimation

This thesis introduced a novel approach to cognitive load estimation using event-camera-based human pose analysis, but the study was conducted under controlled conditions with a limited set of cognitive tasks. The Stroop task used in the experiments primarily involved simple key presses, limiting its applicability in capturing

complex motor-based cognitive load indicators. Future research should extend cognitive load estimation to tasks requiring intricate hand movements, such as the Rey-Osterrieth complex figure test or Luria's fist-edge-palm task, to enhance the granularity of cognitive state assessment. Additionally, increasing task duration and complexity will provide deeper insights into cognitive load variations over time and their impact on human-robot collaboration. To improve real-world applicability, future work should explore naturalistic human-robot interactions and incorporate a between-subject experimental design to capture inter-individual differences in cognitive responses. Moreover, the current framework lacks real-time adaptive feedback, which is crucial for dynamic interaction. Developing neuromorphic models that enable robots to adjust their behaviour based on detected cognitive load variations will facilitate more personalized and context-aware interactions, particularly in assistive robotics, education, and healthcare applications.

The future directions outlined above aim to address the limitations identified in this thesis and expand its contributions to neuromorphic computing and event-based vision in robotics. By improving neuromorphic learning mechanisms, optimising event-based vision pipelines, refining cognitive load estimation methods, and enhancing privacy-preserving neuromorphic AI, future research can drive the development of more intelligent, efficient, and human-like robotic systems. These advancements will help bridge the gap between biological and artificial intelligence, enabling next-generation robotics with enhanced perception, cognition, and adaptability.

Appendix A

A.1 Pattern Recognition Applications

Figure A.1 illustrates two distinct processes in neuromorphic and cognitive robotics. On the left, it outlines the workflow for facial expression recognition using a Spiking Neural Network (SNN) [250]. First, a raw facial image (labeled "a") is transformed by a Laplacian-of-Gaussian (LoG) filter to emphasize edges and contours before generating a Poisson-based spike train (labeled "b"). The spike train moves through a convolution layer to extract relevant features, which feed into an excitatory layer (labeled "c") that propagates essential signals forward. Simultaneously, an inhibitory layer (labeled "d") refines and regulates these signals, enhancing recognition accuracy by suppressing redundant or competing neural responses.

On the right, the figure presents a three-phase cycle of instructed learning [355]. In the teaching phase (labeled "a"), the learner observes a demonstration of the target action, effectively visualizing and understanding what needs to be done. The turn-taking phase (labeled "b") involves interpreting nonverbal cues from the teacher—such as gestures or motion directions—to solidify the conceptual understanding of the task. Lastly, in the trial phase (labeled "c"), the learner attempts to perform or confirm the action independently, allowing for practice and potential feedback from the instructor. This iterative process helps the learner refine its behaviour and skills through repeated observation, engagement, and execution.

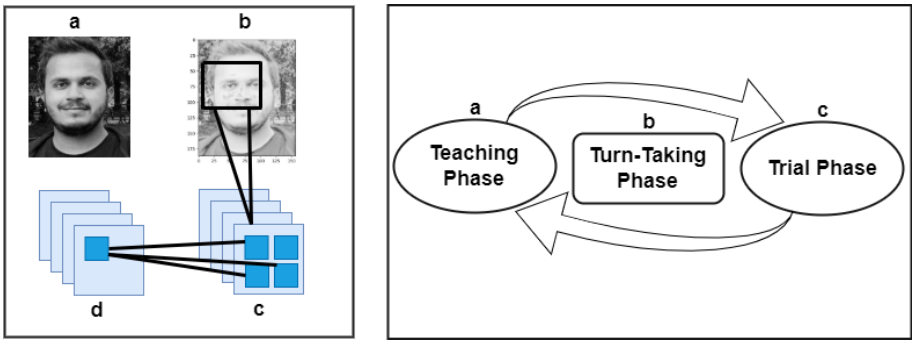


Figure A.1: *Left figure:* represents the SNN workflow for facial expression recognition. a) is a raw image b) image with LoG filter and Poisson spike train creation with convolution layer. c) Excitatory layer. d) Inhibitory layer [250]. *Right Figure:* a) Teaching phase where learner visualize the target action. b) In turn-taking phase, learner extract the nonverbal information. c) In Trial phase. learner confirms the target action [355].

A.2 Motor Control Applications

Figure A.2 shows a high-level overview of the motion generation approach for controlling pointing movements in a robotic arm. The motion generation layer (on the left) produces circular activity patterns, which serve as the fundamental "activation patterns" required for different primitives. A selective disinhibition mechanism then determines which of these primitives is excited enough to execute. Moving toward the center, the motor control layer consists of two key elements: an arm base primitive, which governs basic pointing motions, and arm correction primitives, which fine-tune the trajectory when the robot aligns its arm with a designated target. Finally, the system uses the relative distance between the robot's base point and the target location to activate the appropriate correction primitive. The resulting commands, shown on the right, are transmitted to the robot's actuators to produce precise, real-time pointing actions [364].

Figure A.3 depicts the end-to-end communication flow between a Brain-Computer Interface (BCI) and the Hexapod robot's locomotion system. First, EEG signals are collected via the Emotive Epoc headset and passed to a signal acquisition module, which provides a stable data stream for further processing. Next, the iQSA (improved Quaternion-based Signal Analysis) module analyzes the EEG signals to interpret intended movements or commands. Those results are then sent over Bluetooth to the Hexapod robot's bioinspired locomotion module, which executes the actions.

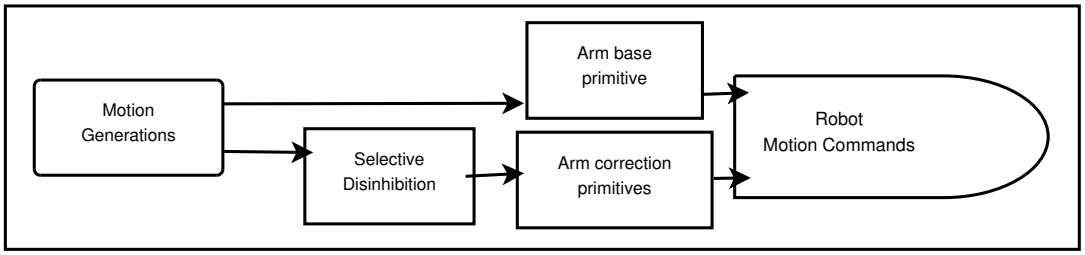


Figure A.2: Brief block diagram of the motion generation approach. It contains three major components: first, the motion generation layer produces circular activity that creates activation patterns for primitives. Second, the motor control layer has arm base primitive and arm correction primitives for pointing motion and to point to target, respectively. Third, the target layer takes the relative distance between target and base point for selective excitation to activate the correction primitives [364].

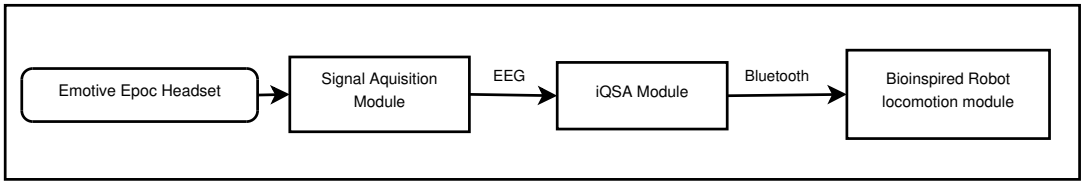


Figure A.3: Communication architecture between the Brain-Computer Interface (BCI) and the Hexapod robot. The EEG signals are acquired through the Emotive Epos headset. This is then transferred to the iQSA module to determine robot movements. Finally, commands are given to the robot locomotion module via Bluetooth [35].

This architecture enables real-time, brain-controlled navigation of the robot by continuously translating EEG-based user intentions into locomotion commands [35].

Figure A.4 illustrates a high-level architecture integrating both perceptual and action systems. On the left, the perceptual system acquires environmental information through devices such as a camera and laser scanner, which feed into an environmental feature extraction module. Simultaneously, an eye-tracker monitors the operator's gaze, and a self-organized spiking neural network (SNN) processes these inputs to extract key perceptual cues. On the right, the action system relies on behavioural feature extraction (informed by signals from a mattress sensor) and a motor control module, which together define how the robot will move or respond. A central SNN-based spatio-temporal modeling module serves as the system's decision core, translating perceptual data into commands for motor control. This setup completes the perception-action cycle, ensuring that environmental and operator inputs continuously inform the robot's behaviour while the robot's actions, in turn, shape the ongoing perception of its surroundings [283].

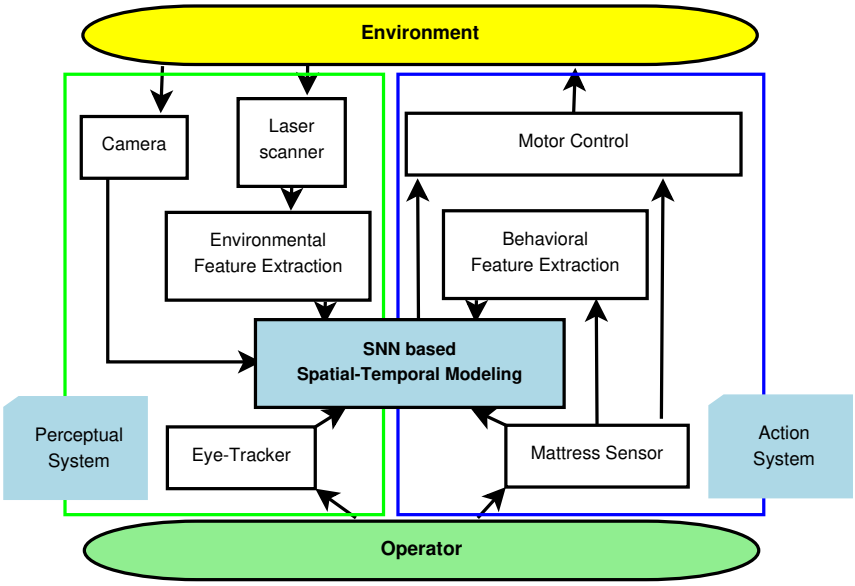


Figure A.4: The summarised system architecture of the system. It has two major parts: 1) the perceptual system, where the information of environmental map find a use to detect the space where a robot can move around. Moreover, the self-organized neural network is utilized to extract perceptual information. 2) the action system, behavioural features in teleoperating are extracted and commands are given to motor control. Based on the perception-action cycle, SNN is used for spatio-temporal modelling [283].

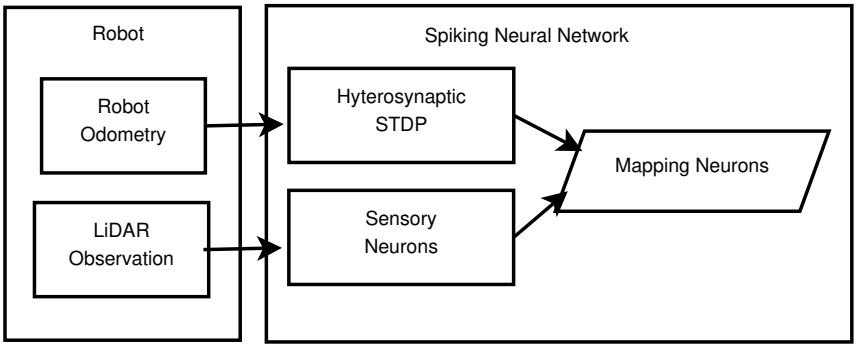


Figure A.5: The network for real-time mapping with Loihi chip. Here robot and LiDAR are providing inputs to the SNN for mapping [356]. *b)* The basic block diagram of the prosthetic control through a Brain-Computer Interface (BCI). As shown, the brain provides the EEG signals to the proposed FeNeuCube framework which, in turn, gives instructions to the controller. Finally, the controller forwards the control command to prosthetic hand [211].

The Figure A.5 illustrates a network for real-time mapping using the Loihi neuromorphic chip. It integrates sensory inputs from a robot's odometry and LiDAR observations, which are fed into a Spiking Neural Network (SNN). Within the SNN, heterosynaptic Spike-Timing Dependent Plasticity (STDP) and sensory neurons process the incoming data, ultimately activating the mapping neurons responsible for constructing a spatial map of the environment. In this setup, EEG signals from the brain are interpreted by the FeNeuCube framework to generate instructions for a controller, which then sends control commands to a prosthetic hand, enabling intuitive neural control [211].

Appendix B

B.1 Object Tracking with SpiNNaker

This appendix describes the implementation of a real-time object tracking system using the SpiNNaker neuromorphic platform, interfacing with a C++ vision module that streams event-based data over ZeroMQ.

Implementation Overview

The Python script `multi_target_tracking.py` is responsible for receiving object positions, encoding them into neuron spikes, and sending them to SpiNNaker for processing. Key components include:

- **Grid and Environment Configuration:**
  - `width = 640`, `height = 480` (pixels)
  - `grid_spacing = 20` $\Rightarrow$ 32 columns $\times$ 24 rows = 768 neurons
- **Neural Model Parameters:** The `IF_curr_exp` neuron model is used with the following configuration:

```
cell_params = {
  'cm': 0.25, 'tau_m': 20.0, 'tau_refrac': 2.0,
  'tau_syn_E': 5.0, 'v_rest': -65.0,
  'v_thresh': -50.0, 'v_reset': -70.0
}
```

- **Simulation Parameters:**
  - `run_time = 10000` ms
  - `weight_to_spike = 2.0`
- **Live Spiking Interface:** Real-time spiking is handled using `SpynnakerLiveSpikesConnec` with callbacks set for spike injection.
- **Data Source (C++ Attention Mechanism):** Events are received via ZMQ from a C++ module using the Metavision SDK. Key parameters:
  - `SPATIAL_BIN_SIZE = 2`, `TEMPORAL_BIN_SIZE = 10000`
  - `MAX_OBJECTS = 2`, `DISTANCE_THRESHOLD = 50.0`
  - Basic Kalman prediction with a fixed offset
- **Logging and Analysis:**
  - Each target logs to a CSV file (e.g., `target_0.csv`)
  - Tracking error is computed using Manhattan distance between predicted and actual neuron positions
  - Post-simulation analysis includes average error and stability (standard deviation in grid indices)

Performance Summary

The average tracking error and per-target spatial stability are printed after the simulation concludes. These metrics are useful for evaluating the reliability of neural population-based tracking in a dynamic environment.

Appendix C

C.1 EB-handGesture Dataset Overview

The **EB-handGesture** dataset is a custom event-based dataset curated to evaluate hand gesture recognition using neuromorphic vision. It captures temporal dynamics of hand movements using event-driven data rather than conventional video frames.

Dataset Specifications

- **Sensor:** CenturyArk SilkyCam VGA Event Camera (resolution: 640×480)
- **Gestures:** 6 distinct hand gestures:
  - `wave`, `point`, `rock`, `scissor`, `clap`, `armroll`
- **Subjects:** 5 individuals performed all gestures
- **Lighting Conditions:** Both *low-light* and *normal-light* environments
- **Gesture Duration:** Approximately 0.5 seconds per instance
- **Total Instances:** 9000 event sequences

Event Stream Format

Each gesture instance is recorded as a stream of asynchronous events in the form (x, y, t, p):

- x, y — spatial pixel coordinates (range: 0–639, 0–479)

- t — timestamp in microseconds
- p — polarity of the event (ON/OFF)

Visualization

Figure C.1 illustrates example visualizations of event-based hand gestures, generated by accumulating polarity events over short time windows.

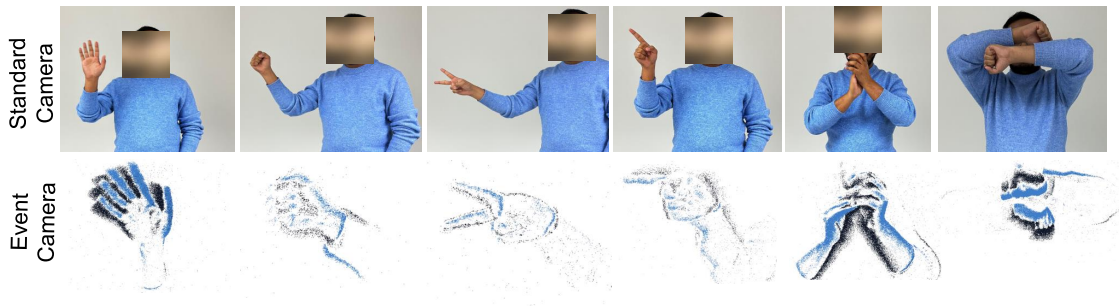


Figure C.1: Sample gestures from the EB-handGesture dataset visualized via event accumulation.

This dataset serves as a benchmark for assessing gesture classification models under neuromorphic sensing conditions, including low-light environments and variable subject performance.

Recognition Results

Figure C.2 presents the performance of the gesture recognition system evaluated on the EB-handGesture dataset. The plot illustrates recognition accuracy across different gesture classes and possibly under varying lighting or model conditions.

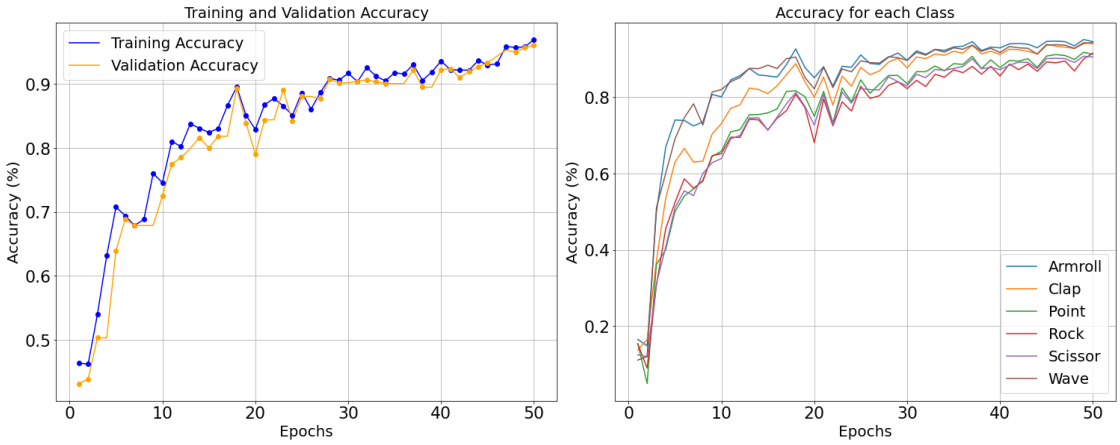


Figure C.2: Recognition accuracy of the proposed model across different hand gestures.

Appendix D

D.1 Stroop Task Visual Stimuli and Metrics

Visual Stimuli

The Stroop task utilized in this study followed a classic interference paradigm [249]. Participants viewed colour words (e.g., *red*, *blue*, *green*, *yellow*) on the robot's display screen, where the ink colour could either match (congruent) or mismatch (incongruent) the word. The task was programmed using the PsychoPy framework [391] and included:

- **Low Cognitive Load (LL):** 80% congruent trials
- **High Cognitive Load (HL):** 80% incongruent trials
- **Response Modality:** Verbal response by naming the ink colour
- **Trial Count:** 25 per condition, randomized order

This setup induced varying cognitive demands while preserving natural interaction via verbal input.

Quantitative Comparison

Figure D.1 presents a quantitative comparison of movement-based metrics during the Stroop task across the high and low cognitive load conditions.

The bar charts show that:

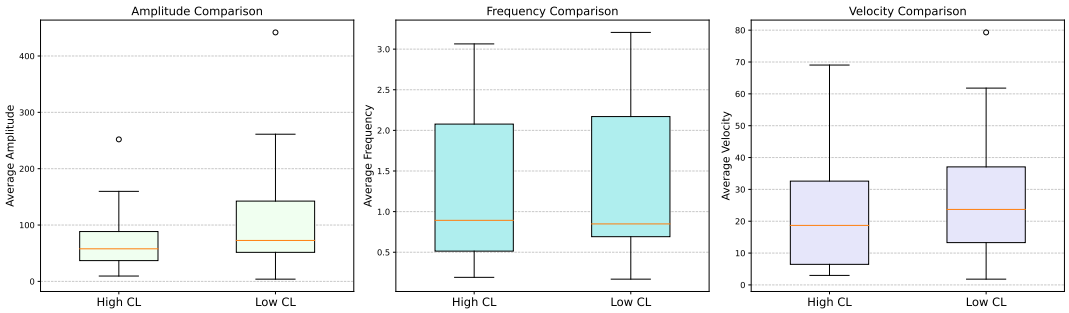


Figure D.1: Comparison of average amplitude, frequency, and velocity metrics between high (HL) and low (LL) cognitive load conditions during the Stroop task.

- **Amplitude:** Higher under high cognitive load, suggesting exaggerated movement or variability.
- **Frequency:** Increased in the HL condition, possibly reflecting restlessness or hesitation.
- **Velocity:** Notably elevated during HL trials, indicative of greater physical or cognitive engagement.

D.2 Auditory Stimuli: N-back Task

This appendix expands on the auditory-based cognitive task described in the Chapter 5, focusing on the implementation and evaluation of the N-back test using spoken digits.

Auditory Stimuli

The N-back task is a cognitive load paradigm widely used to evaluate working memory performance [282, 257]. In our experiment, participants listened to digit sequences and responded verbally when a digit matched one presented "N" steps earlier:

- **Low Load (1-back):** Respond "yes" if the current digit matches the one immediately before.

- **High Load (2-back):** Respond "yes" if the current digit matches the one two steps earlier.

Each condition consisted of:

- **50 trials** with 1-second intervals between digits
- **2 practice sessions** per condition (with feedback)
- **No feedback during the main phase** to ensure unbiased results

Verbal responses were captured using the **OpenAI Whisper** speech-to-text engine [151], ensuring high transcription accuracy in real-time. Manual verification was performed for all trials against recorded audio logs.

Task Accuracy and Cognitive Load Ratings

Figure D.2 presents the auditory task performance and corresponding subjective workload assessments:

- **Top:** Percentage accuracy in low vs. high cognitive load conditions.
- **Bottom:** Self-reported workload based on NASA-TLX ratings.

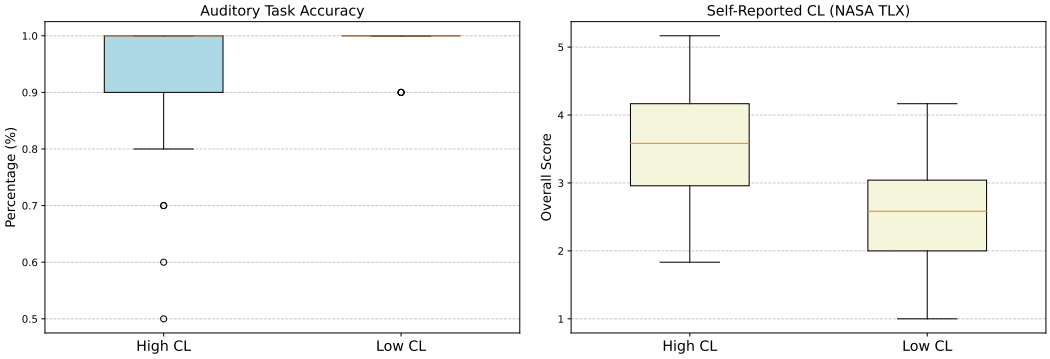


Figure D.2: Top: Auditory task accuracy under low and high load conditions. Bottom: Subjective cognitive load ratings (NASA TLX).

The results demonstrate decreased accuracy and higher self-reported load in the 2-back condition, validating the effectiveness of cognitive load manipulation.

Behavioural Movement Metrics

Figure D.3 illustrates behavioural metrics—amplitude, frequency, and velocity—recorded during the auditory task.

- **Amplitude:** Higher in high-load conditions, potentially due to increased expressiveness or effort.
- **Frequency:** Elevated during 2-back trials, indicating increased gestural/motor activity.
- **Velocity:** Also higher under high load, reflecting more dynamic physical responses.

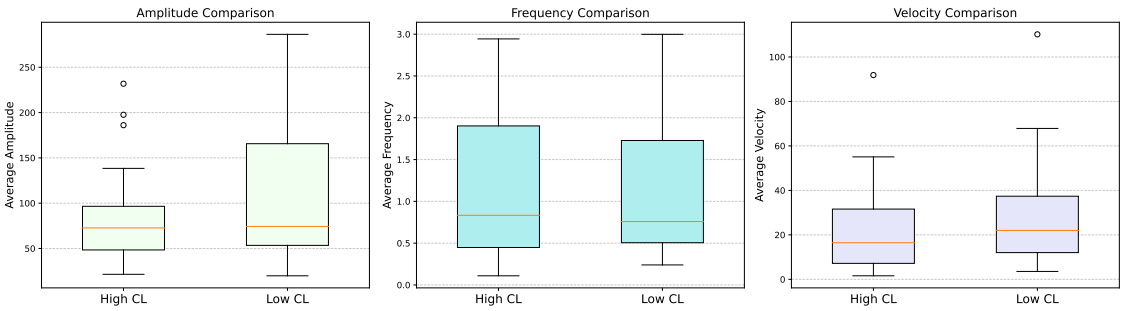


Figure D.3: Comparison of amplitude, frequency, and velocity during low and high load auditory tasks.

These multimodal indicators support the interpretation that greater cognitive demands in the 2-back condition are reflected not only in performance accuracy but also in behavioural cues.

References

[1] Abunahla, H., Halawani, Y., Alazzam, A., and Mohammad, B. (2020). Neuromem: Analog graphene-based resistive memory for artificial neural networks. *Scientific Reports 2020 10:1*, 10:1–11.

[2] Afshar, S., Nicholson, A. P., van Schaik, A., and Cohen, G. (2020). Event-based object detection and tracking for space situational awareness. *IEEE Sensors Journal*, 20(24):15117–15130.

[3] Ahmad, M. I., Keller, I., Robb, D. A., and Lohan, K. S. (2023a). A framework to estimate cognitive load using physiological data. *Personal and Ubiquitous Computing*, 27:2027–2041.

[4] Ahmad, M. I., Keller, I., Robb, D. A., and Lohan, K. S. (2023b). A framework to estimate cognitive load using physiological data. *Personal and Ubiquitous Computing*, 27:2027–2041.

[5] Aitsam, M., Chardiwall, S., and Di Nuovo, A. (2025a). Differentially private spiking neural networks: Enhancing privacy and robustness in social robotics. In Jahankhani, H. and Issac, B., editors, *Cybersecurity and Human Capabilities Through Symbiotic Artificial Intelligence*, pages 199–211, Cham. Springer Nature Switzerland.

[6] Aitsam, M., Davies, S., and Di Nuovo, A. (2022). Neuromorphic computing for interactive robotics: A systematic review. *IEEE Access*, 10:122261–122279.

[7] Aitsam, M., Davies, S., and Di Nuovo, A. (2025b). Event-driven dynamic attention for multi-object tracking on neuromorphic hardware. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR) Workshops*, pages 5055–5062.

[8] Aitsam, M., Davies, S., and Nuovo, A. D. (2024a). Event camera-based real-time gesture recognition for improved robotic guidance. pages 1–8.

[9] Aitsam, M., Goyal, G., Bartolozzi, C., and Di Nuovo, A. (2025c). Vibration vision: Real-time machinery fault diagnosis with event cameras. In Del Bue, A., Canton, C., Pont-Tuset, J., and Tommasi, T., editors, *Computer Vision – ECCV 2024 Workshops*, pages 293–306, Cham. Springer Nature Switzerland.

[10] Aitsam, M., Jimenez Rodriguez, A., and Di Nuovo, A. (2024b). Efficient data processing pipeline for event-based vision datasets: techniques and insights. *Engineering Research Express*, 6(4):045238.

[11] Aitsam, M., Lacroix, D., Goyal, G., Bartolozzi, C., and Di Nuovo, A. (2025d). Measuring cognitive load through event camera based human-pose estimation. In *Human-Friendly Robotics 2024*, pages 229–239, Cham. Springer Nature Switzerland.

[12] Aitsam, M., Lacroix, D., Goyal, G., Bartolozzi, C., and Di Nuovo, A. (2025e). Measuring cognitive load through event camera based human-pose estimation. In *17th International Workshop on Human-Friendly Robotics*. Acceptance Proceedings to be published here on 31/1/2025 - https://link.springer.com/book/9783031816871.

[13] Aitsam, M. and Nuovo, A. D. (2023). Energy efficient personalized hand-gesture recognition with neuromorphic computing. *HRI workshop Concatenate*.

[14] Aitsam, M., Rodriguez, A. J., and Nuovo, A. D. (2024c). Efficient data processing pipeline for event-based vision datasets: techniques and insights. *Engineering Research Express*, 6:045238.

[15] Akopyan, F., Sawada, J., Cassidy, A., Alvarez-Icaza, R., Arthur, J., Merolla, P., Imam, N., Nakamura, Y., Datta, P., Nam, G. J., Taba, B., Beakes, M., Brezzo, B., Kuang, J. B., Manohar, R., Risk, W. P., Jackson, B., and Modha, D. S. (2015). Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34:1537–1557.

[16] Al-Qaderi, M. K. and Rad, A. B. (2018a). A brain-inspired multi-modal perceptual system for social robots: An experimental realization. *IEEE Access*, 6:35402–35424.

[17] Al-Qaderi, M. K. and Rad, A. B. (2018b). A multi-modal person recognition system for social robots. *Applied Sciences (Switzerland)*, 8.

[18] Albarbar, A. and Teay, S. H. (2017). Mems accelerometers: Testing and practical approach for smart sensing and machinery diagnostics. pages 19–40.

[19] Aliyev, I. and Adegbija, T. (2024). Fine-tuning surrogate gradient learning for optimal hardware performance in spiking neural networks.

[20] Amberkar, A., Awasarmol, P., Deshmukh, G., and Dave, P. (2018). Speech recognition using recurrent neural networks. *Proceedings of the 2018 International Conference on Current Trends towards Converging Technologies, ICCTCT 2018*.

[21] Amir, A., Taba, B., Berg, D., Melano, T., Mckinstry, J., Nolfo, C. D., Nayak, T., Andreopoulos, A., Garreau, G., Mendoza, M., Kusnitz, J., Debole, M., Esser, S., Delbruck, T., Flickner, M., and Modha, D. (2017). A low power, fully event-based gesture recognition system. volume 2017-January, pages 7388–7397. Institute of Electrical and Electronics Engineers Inc.

[22] Amirova, A., Rakhymbayeva, N., Yadollahi, E., Sandygulova, A., and Johal, W. (2021). 10 years of human-nao interaction research: A scoping review. *Frontiers in Robotics and AI*, 8:744526.

[23] Ander Arriandiaga, A. G. and Monforte, C. B. M. (2020). *Where and When: Event-Based Spatiotemporal Trajectory Prediction from the iCub's Point-Of-View*. IEEE.

[24] Anders, C., Moontaha, S., Real, S., and Arnrich, B. (2024). Unobtrusive measurement of cognitive load and physiological signals in uncontrolled environments. *Scientific Data 2024 11:1*, 11:1–16.

[25] Andersen, K. F., Pham, H. X., Ugurlu, H. I., and Kayacan, E. (2022). Event-based navigation for autonomous drone racing with sparse gated recurrent network.

[26] Andrew, A. M. (2003). Spiking neuron models: Single neurons, populations, plasticity. *Kybernetes*, 32.

[27] Ayres, P. (2019). Subjective measures of cognitive load. *Cognitive Load Measurement and Application*, pages 9–28.

[28] Baltrusaitis, T., Robinson, P., and Morency, L. P. (2016). Openface: An open source facial behavior analysis toolkit. *2016 IEEE Winter Conference on Applications of Computer Vision, WACV 2016*.

[29] Baltrusaitis, T., Zadeh, A., Lim, Y. C., and Morency, L. P. (2018). Openface 2.0: Facial behavior analysis toolkit. *IEEE International Conference on Automatic Face & Gesture Recognition*, pages 59–66.

[30] Bane, D., Gupta, A., and Suri, M. (2024). Non-invasive qualitative vibration analysis using event camera.

[31] Barbier, T., Teulière, C., and Triesch, J. (2022). Spike timing-based unsupervised learning of orientation, disparity, and motion representations in a spiking neural network.

[32] Barranco, F., Fermuller, C., and Ros, E. (2018). Real-time clustering and multi-target tracking using event-based sensors.

[33] Barrios-Avilés, J., Iakymchuk, T., Samaniego, J., Medus, L. D., and Rosado-Muñoz, A. (2018). Movement detection with event-based cameras: Comparison with frame-based cameras in robot object tracking using powerlink communication. *Electronics (Switzerland)*, 7.

[34] Batres-Mendoza, P., Guerra-Hernandez, E. I., Espinal, A., Perez-Careta, E., and Rostro-Gonzalez, H. (2021). Biologically-inspired legged robot locomotion controlled with a bci by means of cognitive monitoring. *IEEE Access*, 9:35766–35777.

[35] Bayro-Corrochano, E., Solis-Gamboa, S., Altamirano-Escobedo, G., Lechuga-Gutierres, L., and Lisarraga-Rodriguez, J. (2021). Quaternion spiking and quaternion quantum neural networks: Theory and applications. *International journal of neural systems*, 31.

[36] Benosman, R., Ieng, S. H., Clercq, C., Bartolozzi, C., and Srinivasan, M. (2012). Asynchronous frameless event-based optical flow. *Neural Networks*, 27:32–37.

[37] Beohar, D. and Rasool, A. (2021). Handwritten digit recognition of mnist dataset using deep learning state-of-the-art artificial neural network (ann) and convolutional neural network (cnn). In *2021 International Conference on Emerging Smart Computing and Informatics (ESCI)*, pages 542–548.

[38] Berberian, N., Ross, M., and Chartier, S. (2021). Embodied working memory during ongoing input streams. *PLoS ONE*, 16.

[39] Bertrand, J., Yiğit, A., and Durand, S. (2020). Embedded event-based visual odometry. In *2020 6th International Conference on Event-Based Control, Communication, and Signal Processing (EBCCSP)*, pages 1–8.

[40] Bhatt, D., Patel, C., Talsania, H., Patel, J., Vaghela, R., Pandya, S., Modi, K., and Ghayvat, H. (2021). Cnn variants for computer vision: History, architecture, application, challenges and future scope.

[41] Bhattacharya, B. S. and Serrano-Gotarredona, T. (2021). On- and off-centre pathways in a retino-geniculate spiking neural network on spinnaker. volume 2021-May, pages 461–464. IEEE Computer Society.

[42] Bhatti, A., Angkan, P., Behinaein, B., Mahmud, Z., Rodenburg, D., Braund, H., Mclellan, P. J., Ruberto, A., Harrison, G., Wilson, D., Szulewski, A., Howes, D., Etemad, A., Member, S., and Hungler, P. (2024). Clare: Cognitive load assessment in realtime with multimodal data.

[43] Bhutada, S., Yashwanth, N., Dheeraj, P., and Shekar, K. (2022). Opening and closing in morphological image processing. *Journal of Advanced Research and Reviews*, 2022:687–695.

[44] Bi, Y., Chadha, A., Abbas, A., Bourtsoulatze, E., and Andreopoulos, Y. (2019a). Graph-based object classification for neuromorphic vision sensing.

[45] Bi, Y., Chadha, A., Abbas, A., Bourtsoulatze, E., and Andreopoulos, Y. (2019b). Graph-based object classification for neuromorphic vision sensing.

[46] Bianchi, F. M., Maiorino, E., Kampffmeyer, M. C., Rizzi, A., and Jenssen, R. (2017). *Recurrent neural network architectures*, volume 0, pages 23–29. Springer.

[47] Bing, Z., Meschede, C., Röhrbein, F., Huang, K., and Knoll, A. C. (2019a). A survey of robotics control based on learning-inspired spiking neural networks. *Frontiers in Neurorobotics*, 12.

[48] Bing, Z., Meschede, C., Röhrbein, F., Huang, K., and Knoll, A. C. (2019b). A survey of robotics control based on learning-inspired spiking neural networks. *Frontiers in Neurorobotics*, 12.

[49] Blum, H., Dietmüller, A., Milde, M., Conradt, J., Indiveri, G., and Sandamirskaya, Y. (2017). A neuromorphic controller for a robotic vehicle equipped with a dynamic vision sensor. In *Robotics: Science and Systems*, volume 13. MIT Press Journals.

[50] Bogacz, R., Brown, M., and Giraud-Carrier, C. (2000). Frequency-based error backpropagation in a cortical network. *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium.*

[51] Bohté, S., Kok, J., and Poutré, H. L. (2000). Spikeprop: backpropagation for networks of spiking neurons. *undefined.*

[52] Boretti, C., Bich, P., Pareschi, F., Prono, L., Rovatti, R., Setti, G., Det, and Cemse, 2023). Pedro: an event-based dataset for person detection in robotics. Technical report.

[53] Borthakur, A. and Cleland, T. A. (2019). A spike time-dependent online learning algorithm derived from biological olfaction. *Frontiers in Neuroscience*, 13:656.

[54] Bower, J. M., Beeman, D., Hucka, M., Bower, B., Genesis, H. ., and Beeman, D. (2003). The genesis simulation system biomodels view project cerebellar function view project the genesis simulation system.

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

[56] Brosch, T., Tschechne, S., and Neumann, H. (2015). On event-based optical flow detection. *Frontiers in Neuroscience*, 9.

[57] Bryner, S., Gallego, G., Rebecq, H., and Scaramuzza, D. (2019). Event-based, direct camera tracking from a photometric 3d map using nonlinear optimization. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2019-May, pages 325–331. Institute of Electrical and Electronics Engineers Inc.

[58] Buehler (2013). Metaserv 250 single and twin grinder polishers vector vector lc 250 power heads solutions for materials preparation, testing and analysis.

[59] Bugueno-Cordova, I., Campusano, M., Guaman-Rivera, R., and Verschae, R. (2024). A color event-based camera emulator for robot vision. In *Communications in Computer and Information Science*, volume 2077 CCIS, pages 375–390. Springer Science and Business Media Deutschland GmbH.

[60] Burns, A., Greene, B. R., McGrath, M. J., J.Shea, T., Kuris, B., Ayer, S. M., Stroiescu, F., and Cionca, V. (2010). Shimmer™ – a wireless sensor platform for noninvasive biomedical research. *IEEE Sensors Journal*, 10:1527–1534.

[61] Cao, H., Chen, G., Li, Z., Hu, Y., and Knoll, A. (2022). Neurograsp: Multimodal neural network with euler region regression for neuromorphic vision-based grasp pose estimation. *IEEE Transactions on Instrumentation and Measurement*, 71.

[62] Cassidy, A. S., Merolla, P., Arthur, J. V., Esser, S. K., Jackson, B., Alvarez-Icaza, R., Datta, P., Sawada, J., Wong, T. M., Feldman, V., Amir, A., Rubin, D. B. D., Akopyan, F., McQuinn, E., Risk, W. P., and Modha, D. S. (2013). Cognitive computing building block: A versatile and efficient digital neuron model for neurosynaptic cores. *Proceedings of the International Joint Conference on Neural Networks*.

[63] CenturyArk (2020). Silkyevcam (vga) - centuryarks co., ltd.

[64] Chan, V. Y. S., Jin, C. T., and van Schaik, A. (2012). Neuromorphic audio-visual sensor fusion on a sound-localizing robot. *Frontiers in Neuroscience*, 6:16013.

[65] Chaney, K., Cladera, F., Wang, Z., Bisulco, A., Hsieh, M. A., Korpela, C., Kumar, V., Taylor, C. J., and Daniilidis, K. (2023). M3ed: Multi-robot, multi-sensor, multi-environment event dataset. Technical report.

[66] Chaudhary, I., Singh, N. T., Chaudhary, M., and Yadav, K. (2023). Real-time yoga pose detection using opencv and mediapipe. *2023 4th International Conference for Emerging Technology, INCET 2023*.

[67] Chaudhuri, A., Liu, M., and Chakrabarty, K. (2019). Fault-tolerant neuromorphic computing systems. *Proceedings - International Test Conference*, 2019-November.

[68] Chaudhury, S., journal of . . . , M. S. I., and undefined 2014 (2014). Vibration monitoring of rotating machines using mems accelerometer. *academia.eduSB Chaudhury, M Sengupta, K MukherjeeInternational journal of scientific engineering and research, 2014 • academia.edu*.

[69] Chen, F., Zhou, J., Wang, Y., Yu, K., Arshad, S. Z., Khawaji, A., and Conway, D. (2016). Robust multimodal cognitive load measurement.

[70] Chen, G., Cao, H., Conradt, J., Tang, H., Rohrbein, F., and Knoll, A. (2020). Event-based neuromorphic vision for autonomous driving: A paradigm shift for bio-inspired visual sensing and perception. *IEEE Signal Processing Magazine*, 37:34–49.

[71] Chen, G., Xu, Z., Li, Z., Tang, H., Qu, S., Ren, K., and Knoll, A. (2021). A novel illumination-robust hand gesture recognition system with event-based neuromorphic vision sensor. *IEEE Transactions on Automation Science and Engineering*, 18:508–520.

[72] Chen, H. M. and Hu, T. D. Y. (2019). *Slasher: Stadium Racer Car for Event Camera End-to-End Learning Autonomous Driving Experiments*. IEEE.

[73] Chen, P., Guan, W., and Lu, P. (2022a). Esvio: Event-based stereo visual inertial odometry.

[74] Chen, X., Yajima, T., Inoue, I. H., and Iizuka, T. (2022b). An ultra-compact leaky integrate-and-fire neuron with long and tunable time constant utilizing pseudo resistors for spiking neural networks. *Japanese Journal of Applied Physics*, 61:SC1051.

[75] Chrisley, R., Müller, V. C., Sandamirskaya, Y., and Vincze, M. (2016). Cognitive robot architectures. *Cognitive Robot Architectures CEUR WS*, 1855:1.

[76] Clady, X., Maro, J.-M., barre, S. E. B., and Benosman, R. (2016). A motion-based feature for event-based pattern recognition.

[77] Cyr, A., Avarguès-Weber, A., and Thériault, F. (2017). Sameness/difference spiking neural circuit as a relational concept precursor model: A bio-inspired robotic implementation. *Biologically Inspired Cognitive Architectures*, 21:59–66.

[78] Cyr, A., Boukadoum, M., and Poirier, P. (2009). Ai-simcog: A simulator for spiking neurons and multiple animats behaviours. *Neural Computing and Applications*, 18:431–446.

[79] Cyr, A., Morand-Ferron, J., and Theriault, F. (2021). Dual exploration strategies using artificial spiking neural networks in a robotic learning task. *Adaptive Behavior*, 29:567–578.

[80] Cyr, A. and Theriault, F. (2019). Spatial concept learning: A spiking neural network implementation in virtual and physical robots. *Computational Intelligence and Neuroscience*, 2019.

[81] Cyr, A., Thériault, F., and Chartier, S. (2020). Revisiting the xor problem: a neurorobotic implementation. *Neural Computing and Applications*, 32:9965–9973.

[82] Damasio, A. R. (1989). The brain binds entities and events by multiregional activation from convergence zones. *Neural Computation*, 1:123–132.

[83] Dananjaya, P. A., Gopalakrishnan, R., and Lew, W. S. (2021). Rram-based neuromorphic computing systems. *Emerging Non-volatile Memory Technologies*, pages 383–414.

[84] Davies, M., Srinivasa, N., Lin, T.-H., Chinya, G., Cao, Y., and Choday, H. (2018). Loihi: A neuromorphic manycore processor with on-chip learning.

[85] Davies. S, D. N. A. A. M. (2024). Event camera-based real-time gesture recognition for improved robotic guidance anonymous authors. Technical report.

[86] Davila-Chacon, J., Liu, J., and Wermter, S. (2019). Enhanced robot speech recognition using biomimetic binaural sound source localization. *IEEE Transactions on Neural Networks and Learning Systems*, 30:138–150.

[87] Davison, A. P., Brüderle, D., Eppler, J., Kremkow, J., Muller, E., Pecevski, D., Perrinet, L., and Yger, P. (2009). Pynn: A common interface for neuronal network simulators. *Frontiers in Neuroinformatics*, 2:388.

[88] Dayan, P. and Abbott, L. (2001). Theoretical neuroscience: Computational and mathematical modeling of neural systems.

[89] de Jong, T. (2010). Cognitive load theory, educational research, and instructional design: Some food for thought. *Instructional Science*, 38:105–134.

[90] Delamater, A. R. and Lattal, K. M. (2013). The study of associative learning: Mapping from psychological to neural levels of analysis. *Neurobiology of learning and memory*, 108:1.

[91] Delbrück, T. (2008). Frame-free dynamic digital vision.

[92] Deng, J., Dong, W., Socher, R., Li, L.-J., and Fei-Fei, L. (2009). *Computer Vision and Pattern Recognition, 2009, CVPR 2009, IEEE Conference on : dates: 20-25 June 2009.* IEEE.

[93] Deng, L., Wu, Y., Hu, X., Liang, L., Ding, Y., Li, G., Zhao, G., Li, P., and Xie, Y. (2020). Rethinking the performance comparison between snns and anns. *Neural Networks*, 121:294–307.

[94] Dentler, J., Kannan, S., Olivares-Mendez, M. A., and Voos, H. (2017). Implementation and validation of an event-based real-time nonlinear model predictive control framework with ros interface for single and multi-robot systems. In *1st Annual IEEE Conference on Control Technology and Applications, CCTA 2017*, volume 2017-January, pages 1000–1006. Institute of Electrical and Electronics Engineers Inc.

[95] Devarajan, S. (2003). Object identification for robotic applications using expert systems.

[96] Devillez, A. and Dudzinski, D. (2007). Tool vibration detection with eddy current sensors in machining process and computation of stability lobes using fuzzy classifiers. *Mechanical Systems and Signal Processing*, 21:441–456.

[97] Dios, J. R. M.-D., Eguiluz, A. G., Rodriguez-Gomez, J. P., Tapia, R., and Ollero, A. (2020). Towards uas surveillance using event cameras. Technical report.

[98] Dong, K., Zhou, C., Ruan, Y., and Li, Y. (2020). Mobilenetv2 model for image classification. pages 476–480. Institute of Electrical and Electronics Engineers Inc.

[99] Dong, Y., Li, Y., Zhao, D., Shen, G., and Zeng, Y. (2023). Bullying10k: A neuromorphic dataset towards privacy-preserving bullying recognition.

[100] Doon, R., Kumar Rawat, T., and Gautam, S. (2018). Cifar-10 classification using deep convolutional neural network. In *2018 IEEE Punecon*, pages 1–5.

[101] Dorafshanian, M., Aitsam, M., Mejri, M., and Di Nuovo, A. (2024). Beyond data collection: Safeguarding user privacy in social robotics. In *2024 IEEE International Conference on Industrial Technology (ICIT)*, pages 1–6.

[102] Doya, K. and Taniguchi, T. (2019). Toward evolutionary and developmental intelligence. *Current Opinion in Behavioral Sciences*, 29:91–96.

[103] Dumesnil, E., Beaulieu, P. O., and Boukadoum, M. (2017). Single snn architecture for classical and operant conditioning using reinforcement learning. *International Journal of Cognitive Informatics and Natural Intelligence*, 11:1–24.

[104] Dupeyroux, J., Stroobants, S., and Croon, G. C. D. (2022). A toolbox for neuromorphic perception in robotics. In *Proceedings - 2022 8th International Conference on Event-Based Control, Communication, and Signal Processing, EBCCSP 2022*. Institute of Electrical and Electronics Engineers Inc.

[105] Dutta, C., Sagar, S. P., Kumar, A., Bhushan, R., Kadu, S., and Das, T. K. (2023). An adaptive sampling protocol for real-time defect assessment using eddy current sensor and machine learning algorithm. *IEEE Transactions on Industry Applications*, 59:5682–5690.

[106] Dutta, S., Kumar, V., Shukla, A., Mohapatra, N. R., and Ganguly, U. (2017). Leaky integrate and fire neuron by charge-discharge dynamics in floating-body mosfet. *Scientific Reports*, 7.

[107] D'Angelo, G., Voto, S., Iacono, M., Glover, A., Niebur, E., and Bartolozzi, C. (2025). Event-driven figure-ground organisation model for the humanoid robot icub. *Nature Communications 2025 16:1*, 16:1874–.

[108] Ebmer, G., Loch, A., Vu, M. N., Mecca, R., Haessig, G., Hartl-Nesic, C., Vincze, M., and Kugi, A. (2024). Real-time 6-dof pose estimation by an event-based camera using active led markers. Technical report.

[109] Eguiluz, A. G., Rodriguez-Gomez, J. P., Dios, J. R. M.-D., and Ollero, A. (2020). Asynchronous event-based line tracking for time-to-contact maneuvers in uas. Technical report.

[110] Eguiluz, A. G., Rodriguez-Gomez, J. P., Paneque, J. L., Grau, P., Dios, J. R. M.-D., and Ollero, A. (2019). Towards flapping wing robot visual perception: Opportunities and challenges. Technical report.

[111] Eguiluz, A. G., Rodriguez-Gomez, J. P., Tapia, R., Maldonado, F. J., Acosta, J. A., Dios, J. R. M.-D., and Ollero, A. (2021). Why fly blind? event-based visual guidance for ornithopter robot flight. Technical report.

[112] Einhäuser, W. (2017). *The Pupil as Marker of Cognitive Processes*, pages 141–169. Springer Singapore, Singapore.

[113] Eshraghian, J. K., Ward, M., Neftci, E. O., Wang, X., Lenz, G., Dwivedi, G., Bennamoun, M., Jeong, D. S., and Lu, W. D. (2023). Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*, 111(9):1016–1054.

[114] Evangelos Ntouros, G. C. K. S. N. A. (2023). *An Event-Based Tracking Control Framework for Multirotor Aerial Vehicles Using a Dynamic Vision Sensor and Neuromorphic Hardware*. IEEE.

[115] Falotico, E., Vannucci, L., Ambrosano, A., Albanese, U., Ulbrich, S., Tieck, J. C. V., Hinkel, G., Kaiser, J., Peric, I., Denninger, O., Cauli, N., Kirtay, M., Roennau, A., Klinker, G., Arnim, A. V., Guyot, L., Peppicelli, D., Mactinaz-Cañada, P., Ros, E., Maier, P., Weber, S., Huber, M., Plecher, D., Röhrbein, F., Deser, S., Roitberg, A., Smagt, P. V. D., Dillman, R., Levi, P., Laschi, C., Knoll, A. C., and Gewaltig, M. O. (2017). Connecting artificial brains to robots in a comprehensive simulation framework: The neurorobotics platform. *Frontiers in Neurorobotics*, 11.

[116] Fan, L., Zhang, F., Fan, H., and Zhang, C. (2019). Brief review of image denoising techniques. *Visual Computing for Industry, Biomedicine, and Art*, 2:1–12.

[117] Fanello, S. R., Ciliberto, C., Noceti, N., Metta, G., and Odone, F. (2017). Visual recognition for humanoid robots. *Robotics and Autonomous Systems*, 91:151–168.

[118] Fang, W., Chen, Y., Ding, J., Yu, Z., Masquelier, T., Chen, D., Huang, L., Zhou, H., Li, G., and Tian, Y. (2023). Spikingjelly: An open-source machine learning infrastructure platform for spike-based intelligence. *Science Advances*, 9(40):eadi1480.

[119] Fang, W., Yu, Z., Chen, Y., Masquelier, T., Huang, T., and Tian, Y. (2021). Incorporating learnable membrane time constant to enhance learning of spiking neural networks.

[120] Faris, O., Muthusamy, R., Renda, F., Hussain, I., Gan, D., Seneviratne, L., and Zweiri, Y. (2023). Proprioception and exteroception of a soft robotic finger using neuromorphic vision-based sensing. *Soft Robotics*, 10:467–481.

[121] Fatahi, M. (2014). Mnist handwritten digits description and using neuromorphic hardware view project.

[122] Feichtenhofer, C. (2020). X3d: Expanding architectures for efficient video recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 200–210.

[123] Feichtenhofer, C., Fan, H., Malik, J., and He, K. (2019). Slowfast networks for video recognition.

[124] Filho, P. C., da Silva, L., Pombeiro, A., Costa, N., Carneiro, P., and Arezes, P. (2024). Assessing mental workload in industrial environments: A review of applied studies. *Studies in Systems, Decision and Control*, 492:677–689.

[125] Fischer, T. and Milford, M. (2022). How many events do you need? event-based visual place recognition using sparse but varying pixels. *IEEE Robotics and Automation Letters*, 7(4):12275–12282.

[126] Fischl, K. D., Cellon, A. B., Stewart, T. C., Horiuchi, T. K., and Andreou, A. G. (2019). Socio-emotional robot with distributed multi-platform neuromorphic processing: (invited presentation). *2019 53rd Annual Conference on Information Sciences and Systems, CISS 2019*.

[127] Florian, R. V. (2007). Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural computation*, 19:1468–1502.

[128] Folowosele, F., Vogelstein, R. J., and Etienne-Cummings, R. (2011). Towards a cortical prosthesis: Implementing a spike-based hmax model of visual object recognition in silico. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 1(4):516–525.

[129] Foltyn, A., Deuschel, J., Lang-Richter, N. R., Holzer, N., and Oppelt, M. P. (2024). Evaluating the robustness of multimodal task load estimation models. *Frontiers in Computer Science*, 6:1371181.

[130] Forrai, B., Miki, T., Gehrig, D., Hutter, M., and Scaramuzza, D. (2023). Event-based agile object catching with a quadrupedal robot. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2023-May, pages 12177–12183. Institute of Electrical and Electronics Engineers Inc.

[131] Frisch, S., Dshemuchadse, M., Görner, M., Goschke, T., and Scherbaum, S. (2015). Unraveling the sub-processes of selective attention: insights from dynamic modeling and continuous behavior. *Cognitive Processing*, 16:377–388.

[132] Frosina, P., Logue, M., Book, A., Huizinga, T., Amos, S., and Stark, S. (2018). The effect of cognitive load on nonverbal behavior in the cognitive interview for suspects. *Personality and Individual Differences*, 130:51–58.

[133] Furber, S. (2016). Large-scale neuromorphic computing systems. *Journal of Neural Engineering*, 13.

[134] Furber, S. and Bogdan, P. (2020a). Spinnaker - a spiking neural network architecture. *SpiNNaker - A Spiking Neural Network Architecture*, pages 1–320.

[135] Furber, S. and Bogdan, P. (2020b). Spinnaker: A spiking neural network architecture. *SpiNNaker: A Spiking Neural Network Architecture*.

[136] Gallego, G., Delbruck, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A., Conradt, J., Daniilidis, K., and Scaramuzza, D. (2019). Event-based vision: A survey.

[137] Galluppi, F., Davies, S., Rast, A., Sharp, T., Plana, L. A., and Furber, S. (2012). A hierachical configuration system for a massively parallel neural hardware platform. *CF '12 - Proceedings of the ACM Computing Frontiers Conference*, pages 183–192.

[138] García, D. H., Adams, S., Rast, A., Wennekers, T., Furber, S., and Cangelosi, A. (2018). Visual attention and object naming in humanoid robots using a bio-inspired spiking neural network. *Robotics and Autonomous Systems*, 104:56–71.

[139] Gava, L., Monforte, M., Iacono, M., Bartolozzi, C., and Glover, A. (2022). Puck: Parallel surface and convolution-kernel tracking for event-based cameras.

[140] Gehrig, D., Loquercio, A., Derpanis, K., and Scaramuzza, D. (2019). End-to-end learning of representations for asynchronous event-based data. *Proceedings of the IEEE International Conference on Computer Vision*, 2019-October:5632–5642.

[141] Gehrig, D., Rebecq, H., Gallego, G., and Scaramuzza, D. (2018). Asynchronous, photometric feature tracking using events and frames. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11216 LNCS:766–781.

[142] Gehrig, D., Rebecq, H., Gallego, G., and Scaramuzza, D. (2020). Eklt: Asynchronous photometric feature tracking using events and frames. *International Journal of Computer Vision*, 128:601–618.

[143] Gehrig, M., Aarents, W., Gehrig, D., and Scaramuzza, D. (2023). Dsec: A stereo event camera dataset for driving scenarios. Technical report.

[144] Gerstner, W., Ritz, R., and van Hemmen, J. L. (1993). Why spikes? hebbian learning and retrieval of time-resolved excitation patterns. *Biological Cybernetics 1993 69:5*, 69:503–515.

[145] Gewaltig, M.-O. and Diesmann, M. (2007). Nest (neural simulation tool). *Scholarpedia*, 2(4):1430.

[146] Glover, A. and Bartolozzi, C. (2017). Robust visual tracking with a freely-moving event camera. In *IEEE International Conference on Intelligent Robots and Systems*, volume 2017-September, pages 3769–3776. Institute of Electrical and Electronics Engineers Inc.

[147] Glover, A., Vasco, V., Iacono, M., and Bartolozzi, C. (2018). The event-driven software library for yarp—with algorithms and icub applications. *Frontiers in Robotics and AI*, 4.

[148] Golibrzuch, K., Schwabe, S., Zhong, T., Papendorf, K., and Wodtke, A. M. (2022). Application of an event-based camera for real-time velocity resolved kinetics. *Journal of Physical Chemistry A*, 126:2142–2148.

[149] Goodman, D. F. M. and Brette, R. (2009). Focused review the brian simulator. 3.

[150] Goyal, G., Pietro, F. D., Carissimi, N., Glover, A., and Bartolozzi, C. (2023). Moveenet: Online high-frequency human pose estimation with an event camera. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2023-June:4024–4033.

[151] Graham, C. and Roll, N. (2024). Evaluating openai's whisper asr: Performance analysis across diverse accents and speaker traits. *JASA Express Letters*, 4.

[152] Guo, M., Huang, J., and Chen, S. (2017). Live demonstration: A 768 × 640 pixels 200meps dynamic vision sensor. In *Proceedings - IEEE International Symposium on Circuits and Systems*. Institute of Electrical and Electronics Engineers Inc.

[153] Guo, Q., Yu, Z., Fu, J., Lu, Y., Zweiri, Y., and Gan, D. (2024). Forceevt: A closer look at robotic gripper force measurement with event-based vision transformer.

[154] Gwizdka, J. (2010). Using stroop task to assess cognitive load. pages 219–222.

[155] Haapalainen, E., Kim, S., Forlizzi, J. F., and Dey, A. K. (2010a). Psychophysiological measures for assessing cognitive load. *UbiComp'10 - Proceedings of the 2010 ACM Conference on Ubiquitous Computing*, pages 301–310.

[156] Haapalainen, E., Kim, S., Forlizzi, J. F., and Dey, A. K. (2010b). Psychophysiological measures for assessing cognitive load. pages 301–310.

[157] Haji, F. A., Rojas, D., Childs, R., de Ribaupierre, S., and Dubrowski, A. (2015). Measuring cognitive load: performance, mental effort and simulation task complexity. *Medical education*, 49:815–827.

[158] Hajizada, E., Berggold, P., Iacono, M., Glover, A., and Sandamirskaya, Y. (2022). Interactive continual learning for robots: a neuromorphic approach. In *ACM International Conference Proceeding Series*. Association for Computing Machinery.

[159] Halvagal, M. S. and Zenke, F. (2023). The combination of hebbian and predictive plasticity learns invariant object representations in deep sensory networks. *Nature Neuroscience 2023 26:11*, 26:1906–1915.

[160] Hampo, M., Fan, D., Jenkins, T., Demange, A., Westberg, S., Bihl, T., and Taha, T. (2020). Associative memory in spiking neural network form implemented on neuromorphic hardware. Association for Computing Machinery.

[161] Han, J., Wang, Z., Shen, J., and Tang, H. (2023). Symmetric-threshold relu for fast and nearly lossless ann-snn conversion. *Machine Intelligence Research*, 20:435–446.

[162] Hanusz, Z., Tarasinska, J., and Zielinski, W. (2016). Shapiro–wilk test with known mean. *REVSTAT-Statistical Journal*, 14:89–100–89–100.

[163] Hao, Z., Ding, J., Bu, T., Huang, T., and Yu, Z. (2023). Bridging the gap between anns and snns by calibrating offset spikes. *11th International Conference on Learning Representations, ICLR 2023*.

[164] He, B., Wang, Z., Zhou, Y., Chen, J., Singh, C. D., Li, H., Gao, Y., Shen, S., Wang, K., Cao, Y., Xu, C., Aloimonos, Y., Gao, F., and Fermuller, C. (2024). Microsaccade-inspired event camera for robotics.

[165] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December:770–778.

[166] Hebb, D. (2005). The organization of behavior : A neuropsychological theory. *The Organization of Behavior*.

[167] Herculano-Houzel, S. (2009). The human brain in numbers: A linearly scaled-up primate brain. *Frontiers in Human Neuroscience*, 3:31.

[168] Hermann, A., Sun, J., Xue, Z., Ruehl, S. W., Oberlaender, J., Roennau, A., Zoellner, J. M., and Dillmann, R. (2013). Hardware and software architecture of the bimanual mobile manipulation robot hollie and its actuated upper body. *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics: Mechatronics for Human Wellbeing, AIM 2013*, pages 286–292.

[169] Hien, D. S., Luong, N. T., Minh, L. H., Phuc, T. T., Trung, P. T., Dong, B. A., Thao, H. L. T., Thanh, N. V. L., Tuan, T. T. A., Trung, H. H., Nhan, N. T. T., and Nga, D. V. (2009). Development of quantum device simulator nemo-vn1. *Journal of Physics: Conference Series*, 187.

[170] Hines, M. L. and Carnevale, N. T. (2020). The neuron simulation environment.

[171] Hodgkin, A. L. and Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117:500.

[172] Huang, X., Halwani, M., Muthusamy, R., Ayyad, A., Swart, D., Seneviratne, L., Gan, D., and Zweiri, Y. (2022). Real-time grasping strategies using event camera. *Journal of Intelligent Manufacturing*, 33:593–615.

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

[174] Huang, X., Muthusamy, R., Hassan, E., Niu, Z., Seneviratne, L., Gan, D., and Zweiri, Y. (2020b). Neuromorphic vision based contact-level classification in robotic grasping applications. *Sensors (Switzerland)*, 20:1–15.

[175] Huang, X., Wu, W., and Qiao, H. (2021). Computational modeling of emotion-motivated decisions for continuous control of mobile robots. *IEEE Transactions on Cognitive and Developmental Systems*, 13:31–44.

[176] Iakymchuk, T., Rosado-Muñoz, A., Guerrero-Martínez, J. F., Bataller-Mompeán, M., and Francés-Víllora, J. V. (2015). Simplified spiking neural network architecture and stdp learning algorithm applied to image classification. *Eurasip Journal on Image and Video Processing*, 2015.

[177] Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. (2016). Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size.

[178] Indiveri, G. (2000). Modeling selective attention using a neuromorphic analog vlsi device. *Neural computation*, 12:2857–2880.

[179] iniVation (2024). Dvxplorer — inivation 2024-03-12 documentation.

[180] Izhikevich, E. (2004a). Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, 15(5):1063–1070.

[181] Izhikevich, E. M. (2003). Simple model of spiking neurons. *IEEE TRANSACTIONS ON NEURAL NETWORKS*, 14.

[182] Izhikevich, E. M. (2004b). Which model to use for cortical spiking neurons? *IEEE Transactions on Neural Networks*, 15:1063–1070.

[183] Izhikevich, E. M. and Desai, N. S. (2002). Relating stdp to bcm.

[184] jae Na, W., Sun, K. H., Jeon, B. C., Lee, J., and ho Shin, Y. (2023). Event-based micro vibration measurement using phase correlation template matching with event filter optimization. *Measurement: Journal of the International Measurement Confederation*, 215.

[185] Jaiem, L., Crestani, D., Lapierre, L., and Druon, S. (2021). Energy consumption of control schemes for the pioneer 3dx mobile robot: Models and evaluation. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 102:1–15.

[186] Jain, A., Bansal, R., Kumar, A., and Singh, K. (2015). A comparative study of visual and auditory reaction times on the basis of gender and physical activity levels of medical first year students. *International Journal of Applied and Basic Medical Research*, 5:124.

[187] Jason Yik, Korneel Van den Berghe, D. d. B. Y. B. M. F. P. H. W. K. M. A. K. D. K. N. P.-N. A. P. P. S. P.-S. V. S. G. T. S. W. B. Z. S. H. A. G. V. J. B. L. A. M. A. K. M. G. L. T. S. Z. A. M. A. B. A. A. G. A. C. B. A. B. P. B. S. B. S. B. G. C. E. C. F. C. G. d. C. A. D. A. D. M. D. Y. D. J. E. T. F. J. F. V. F. S. F. P. M. F. W. G. A. G. H. A. G. G. I. S. J. V. K. L. K. J. C. K. L. K. R. K. D. K. S.-C. L. Y.-H. L. H. M. R. M. J. M. M.-T. C. M. K. M. D. R. M. E. N. T. N. F. O. A. O. P. P. J. P. M. P. C. P. M. A. P. C. P. A. R. Y. S. C. J. S. S. A. v. S. J. S. S. S. C. S. J.-s. S. S. S. S. B. S. M. S. A. S. K. S. M. S. T. C. S. J. T. N. T. G. U. M. V. C. M. V. B. V. A. Y. F. T. Z. C. F. . V. J. (2025). The neurobench framework for benchmarking neuromorphic computing algorithms and systems. *Nature Communications 2025 16:1*, 16:1–24.

[188] Jens Egholm Pedersen, Gregor Lenz, G. C. A. M. (2024). Faery: A stream processing library for neuromorphic event-based data - faery docs.

[189] Jia, S. (2022). Event camera survey and extension application to semantic segmentation. *ACM International Conference Proceeding Series*, pages 115–121.

[190] Jiang, Z., Bing, Z., Huang, K., Chen, G., Cheng, L., and Knoll, A. (2017). Event-based target tracking control for a snake robot using a dynamic vision sensor. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10639 LNCS, pages 111–121. Springer Verlag.

[191] Jo, W., Wang, R., Yang, B., Foti, D., Rastgaar, M., and Min, B.-C. (2024). Cognitive load-based affective workload allocation for multihuman multirobot teams. *IEEE Transactions on Human-Machine Systems*, pages 1–14.

[192] Joubert, D., Marcireau, A., Ralph, N., Jolley, A., van Schaik, A., and Cohen, G. (2021). Event camera simulator improvements via characterized parameters. *Frontiers in Neuroscience*, 15. Publisher Copyright: © Copyright © 2021 Joubert, Marcireau, Ralph, Jolley, van Schaik and Cohen.

[193] Kaiser, J., Mostafa, H., and Neftci, E. (2018). Synaptic plasticity dynamics for deep continuous local learning (decolle). *Frontiers in Neuroscience*, 14.

[194] Kamarudin, N., Makhtar, M., Abdullah, F. S., Mohamad, M., Kamarudin, N. S., Fadzli, S. A., Mohamad, F. S., and Kadir, M. F. A. (2015). Comparison of image classification techniques using caltech 101 dataset texture-based image retrieval view. *Article in Journal of Theoretical and Applied Information Technology*, 10.

[195] Kaminski, W. A. and Wojcik, G. M. (2004). Liquid state machine built of hodgkin-huxley neurons. *Informatica*, 15:39–44.

[196] Kempter, R., Gerstner, W., and Hemmen, J. L. V. (1999). Hebbian learning and spiking neurons. 18.

[197] Khan, M. A., Asadi, H., Zhang, L., Qazani, M. R. C., Oladazimi, S., Loo, C. K., Lim, C. P., and Nahavandi, S. (2024). Application of artificial intelligence in cognitive load analysis using functional near-infrared spectroscopy: A systematic review. *Expert Systems with Applications*, 249:123717.

[198] Khan, M. M., Lester, D. R., Plana, L. A., Rast, A., Jin, X., Painkras, E., and Furber, S. B. (2008). Spinnaker: Mapping neural networks onto a massively-parallel chip multiprocessor. *Proceedings of the International Joint Conference on Neural Networks*, pages 2849–2856.

[199] Kikuchi, H. and Oka, Y. (2020). Silkyevcam event based camera specification product name - silkyevcam, model name - evc3a rev.1.0 approval responsible person.

[200] Kim, J. (2020). New neuromorphic ai nm500 and its adas application. *Lecture Notes in Electrical Engineering*, 554:3–12.

[201] Kim, J., Bae, J., Park, G., Zhang, D., and Kim, Y. M. (2021). N-imagenet: Towards robust, fine-grained object recognition with event cameras.

[202] Kinsky, N. R., Sullivan, D. W., Mau, W., Hasselmo, M. E., and Eichenbaum, H. B. (2018). Hippocampal place fields maintain a coherent and flexible map across long time scales. *Current biology : CB*, 28:3578.

[203] Koller, O., Ney, H., and Bowden, R. (2016). Deep hand: How to train a cnn on 1 million hand images when your data is continuous and weakly labelled.

[204] Kong, D., Fang, Z., Li, H., Hou, K., Coleman, S., and Kerr, D. (2020). Event-vpr: End-to-end weakly supervised network architecture for event-based visual place recognition.

[205] Kreiser, R., Aathmani, D., Qiao, N., Indiveri, G., and Sandamirskaya, Y. (2018). Organizing sequential memory in a neuromorphic device using dynamic neural fields. *Frontiers in Neuroscience*, 12:407706.

[206] Kreiser, R., Cartiglia, M., Martel, J., Conradt, J., and Sandamirskaya, Y. (2021). A neuromorphic approach to path integration: a head-direction spiking neural network with vision-driven reset. Technical report.

[207] Krieglstein, F., Beege, M., Rey, G. D., Sanchez-Stockhammer, C., and Schneider, S. (2023). Development and validation of a theory-based questionnaire to measure different types of cognitive load. *Educational Psychology Review*, 35.

[208] Kudithipudi, D., Schuman, C., Vineyard, C. M., Pandit, T., Merkel, C., Kubendran, R., Aimone, J. B., Orchard, G., Mayr, C., Benosman, R., Hays, J., Young, C., Bartolozzi, C., Majumdar, A., Cardwell, S. G., Payvand, M., Buckley, S., Kulkarni, S., Gonzalez, H. A., Cauwenberghs, G., Thakur, C. S., Subramoney, A., and Furber, S. (2025). Neuromorphic computing at scale. *Nature 2025 637:8047*, 637:801–812.

[209] Kugele, A., Pfeil, T., Pfeiffer, M., and Chicca, E. (2021a). Hybrid snn-ann: Energy-efficient classification and object detection for event-based vision. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 13024 LNCS:297–312.

[210] Kugele, A., Pfeil, T., Pfeiffer, M., and Chicca, E. (2021b). Hybrid snn-ann: Energy-efficient classification and object detection for event-based vision. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 13024 LNCS:297–312.

[211] Kumarasinghe, K., Owen, M., Taylor, D., Kasabov, N., and Kit, C. (2018). Faneurobot: A framework for robot and prosthetics control using the neucube spiking neural network architecture and finite automata theory. pages 4465–4472. Institute of Electrical and Electronics Engineers Inc.

[212] Kyung, M., Weiss, E., and Rangan, V. (2011). A vlsi implementation: Izhikevichs neuron model.

[213] Lagomarsino, M., Lorenzini, M., Momi, E. D., and Ajoudani, A. (2022). An online framework for cognitive load assessment in industrial tasks. *Robotics and Computer-Integrated Manufacturing*, 78:102380.

[214] Lagorce, X., Meyer, C., Ieng, S. H., Filliat, D., and Benosman, R. (2015). Asynchronous event-based multikernel algorithm for high-speed visual features tracking. *IEEE Transactions on Neural Networks and Learning Systems*, 26:1710–1720.

[215] Lawson, K. S. W. (2017). *Representing Motion Information from Event-Based Cameras*. IEEE.

[216] Lecun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521:436–444.

[217] Lee, C., Panda, P., Srinivasan, G., and Roy, K. (2018). Training deep spiking convolutional neural networks with stdp-based unsupervised pre-training followed by supervised fine-tuning. *Frontiers in Neuroscience*, 12:435.

[218] Lee, C., Sarwar, S. S., Panda, P., Srinivasan, G., and Roy, K. (2020). Enabling spike-based backpropagation for training deep neural network architectures. *Frontiers in Neuroscience*, 14:119.

[219] Lee, J. H., Delbruck, T., and Pfeiffer, M. (2016). Training deep spiking neural networks using backpropagation. *Frontiers in neuroscience*, 10.

[220] Lele, A., Fang, Y., Ting, J., and Raychowdhury, A. (2021). An end-to-end spiking neural network platform for edge robotics: From event-cameras to central pattern generation. *IEEE Transactions on Cognitive and Developmental Systems*.

[221] Lele, A., Fang, Y., Ting, J., and Raychowdhury, A. (2022). An end-to-end spiking neural network platform for edge robotics: From event-cameras to central pattern generation. *IEEE Transactions on Cognitive and Developmental Systems*, 14:1092–1103.

[222] Lenz, G., Chaney, K., Shrestha, S. B., Oubari, O., Picaud, S., and Zarrella, G. (2021). Tonic: event-based datasets and transformations. Documentation available under https://tonic.readthedocs.io.

[223] Li, B., Cao, H., Qu, Z., Hu, Y., Wang, Z., and Liang, Z. (2020). Event-based robotic grasping detection with neuromorphic vision sensor and event-stream dataset.

[224] Li, H., Liu, H., Ji, X., Li, G., and Shi, L. (2017). Cifar10-dvs: An event-stream dataset for object classification. *Frontiers in Neuroscience*, 11.

[225] Li, Q., Li, R., Ji, K., and Dai, W. (2016). Kalman filter and its application. *Proceedings - 8th International Conference on Intelligent Networks and Intelligent Systems, ICINIS 2015*, pages 74–77.

[226] Li, S., Zheng, P., Liu, S., Wang, Z., Wang, X. V., Zheng, L., and Wang, L. (2023). Proactive human–robot collaboration: Mutual-cognitive, predictable, and self-organising perspectives. *Robotics and Computer-Integrated Manufacturing*, 81:102510.

[227] Li, W., Piëch, V., and Gilbert, C. D. (2004). Perceptual learning and top-down influences in primary visual cortex. *Nature Neuroscience 2004 7:6*, 7:651–657.

[228] Li, W., Zhao, J., Su, L., Jiang, N., and Hu, Q. (2024). Spiking neural networks for object detection based on integrating neuronal variants and self-attention mechanisms. *Applied Sciences 2024, Vol. 14, Page 9607*, 14:9607.

[229] Li, Y., Deng, S., Dong, X., Gong, R., and Gu, S. (2021). A free lunch from ann: Towards efficient, accurate spiking neural networks calibration. *Proceedings of Machine Learning Research*, 139:6316–6325.

[230] Lichtsteiner, P., Posch, C., and Delbruck, T. (2008a). A 128 $\times$ 128 120 db 15 μs latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits*, 43:566–576.

[231] Lichtsteiner, P., Posch, C., and Delbruck, T. (2008b). A 128 $\times$ 128 120db 15ms latency asynchronous temporal contrast vision sensor. *IEEE J. Solid State Circuits*, 43:566–576.

[232] Lienen, C. and Platzner, M. (2022). Reconros executor: Event-driven programming of fpga-accelerated ros 2 applications.

[233] Lin, C. J. and Lukodono, R. P. (2022). Classification of mental workload in human-robot collaboration using machine learning based on physiological feedback. *Journal of Manufacturing Systems*, 65:673–685.

[234] Lin, S., Zhang, Y., Huang, D., Zhou, B., Luo, X., and Pan, J. (2023). Fast event-based double integral for real-time robotics.

[235] Lin, Y., Ding, W., Qiang, S., Deng, L., and Li, G. (2021). Es-imagenet: A million event-stream classification dataset for spiking neural networks.

[236] Liu, B., Xu, C., Yang, W., Yu, H., and Yu, L. (2023a). Motion robust high-speed light-weighted object detection with event camera. *IEEE Transactions on Instrumentation and Measurement*, 72:1–13.

[237] Liu, D., Parra, A., and Chin, T.-J. (2021a). Spatiotemporal registration for event-based visual odometry.

[238] Liu, Q., Ruan, H., Xing, D., Tang, H., and Pan, G. (2020). Effective aer object classification using segmented probability-maximization learning in spiking neural networks. *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, pages 1308–1315.

[239] Liu, Q., Xing, D., Tang, H., Ma, D., and Pan, G. (2021b). Event-based action recognition using motion information and spiking neural networks.

[240] Liu, Q., Xing, D., Tang, H., Ma, D., and Pan, G. (2021c). Event-based action recognition using motion information and spiking neural networks.

[241] Liu, Z., Wang, L., Wu, W., Qian, C., and Lu, T. (2023b). Tam: Temporal adaptive module for video recognition.

[242] Lobov, S. A., Zharinov, A. I., Makarov, V. A., and Kazantsev, V. B. (2021). Spatial memory in a spiking neural network with robot embodiment. *Sensors*, 21.

[243] Lopez, D. A., Lopez, M. A., Muñoz, D. S., Santa, J. A., Gomez, D. F., Barone, D., Torresen, J., and Salas, J. A. (2022). Controlling the ur3 robotic arm using a leap motion: A comparative study. *Communications in Computer and Information Science*, 1519 CCIS:64–77.

[244] Lv, Y., Zhou, L., Liu, Z., and Zhang, H. (2024). Structural vibration frequency monitoring based on event camera. *Measurement Science and Technology*, 35:085007.

[245] López-Osorio, P., Domínguez-Morales, J. P., and Perez-Peña, F. (2024). A neuromorphic vision and feedback sensor fusion based on spiking neural networks for real-time robot adaption. *Advanced Intelligent Systems*, 6.

[246] López-Randulfe, J., Duswald, T., Bing, Z., and Knoll, A. (2021). Spiking neural network for fourier transform and object detection for automotive radar. *Frontiers in Neurorobotics*, 15:69.

[247] Maass, W., Natschläger, T., and Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14:2531–2560.

[248] MacLeod, C. M. (1992a). The stroop task: The "gold standard" of attentional measures. *Journal of Experimental Psychology: General*, 121:12–14.

[249] MacLeod, C. M. (1992b). The stroop task: The "gold standard" of attentional measures. *Journal of Experimental Psychology: General*, 121:12–14.

[250] Mansouri-Benssassi, E. and Ye, J. (2018). Bio-inspired spiking neural networks for facial expression recognition: generalisation investigation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11324 LNCS:426–437.

[251] Marekˇ, M. and Marekž (2015). Design, construction and control of hexapod walking robot.

[252] Maro, J. M., Ieng, S. H., and Benosman, R. (2020). Event-based gesture recognition with dynamic background suppression using smartphone computational capabilities. *Frontiers in Neuroscience*, 14.

[253] Martin, S. (2014). Measuring cognitive load and cognition: metrics for technology-enhanced learning. *Educational Research and Evaluation*, 20:592–621.

[254] Mayr, C., Hoeppner, S., and Furber, S. (2019). Spinnaker 2: A 10 million core processor system for brain simulation and machine learning.

[255] McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5:115–133.

[256] Meddaoui, A., Hain, M., and Hachmoud, A. (2023). The benefits of predictive maintenance in manufacturing excellence: a case study to establish reliable methods for predicting failures. *International Journal of Advanced Manufacturing Technology*, 128:3685–3690.

[257] Meier, C. (2019). Back-test. *Encyclopedia of Animal Cognition and Behavior*, pages 1–2.

[258] Merriënboer, J. J. V. and Aryres, P. (2005). Research on cognitive load theory and its design implications for e-learning. *Educational Technology Research and Development*, 53:5–13.

[259] Messikommer, N., Gehrig, D., Gehrig, M., and Scaramuzza, D. (2022). Bridging the gap between events and frames through unsupervised domain adaptation.

[260] Metta, G., Fitzpatrick, P., and Natale, L. (2006). Yarp: Yet another robot platform. *International Journal of Advanced Robotic Systems*, 3:043–048.

[261] Milde, M. B., Blum, H., Dietmüller, A., Sumislawska, D., Conradt, J., Indiveri, G., and Sandamirskaya, Y. (2017). Obstacle avoidance and target acquisition for robot navigation using a mixed signal analog/digital neuromorphic processing system. *Frontiers in Neurorobotics*, 11.

[262] Mirus, F., Axenie, C., Stewart, T. C., and Conradt, J. (2018). Neuromorphic sensorimotor adaptation for robotic mobile manipulation: From sensing to behaviour. *Cognitive Systems Research*, 50:52–66.

[263] Mitrokhin, A., Fermuller, C., Parameshwara, C., and Aloimonos, Y. (2018a). Event-based moving object detection and tracking. *IEEE International Conference on Intelligent Robots and Systems*, pages 6895–6902.

[264] Mitrokhin, A., Fermuller, C., Parameshwara, C., and Aloimonos, Y. (2018b). Event-based moving object detection and tracking. *IEEE International Conference on Intelligent Robots and Systems*, pages 6895–6902.

[265] Mobley, R. K. (2002). *An Introduction to Predictive Maintenance*.

[266] Monforte, A. et al. (2023). Fast trajectory end-point prediction with event cameras for reactive robot navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.

[267] Monti, F., Boscaini, D., Masci, J., Rodolà, E., Svoboda, J., and Bronstein, M. M. (2017). Geometric deep learning on graphs and manifolds using mixture model cnns. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-January:5425–5434.

[268] Mueggler, E., Bartolozzi, C., and Scaramuzza, D. (2017a). Fast event-based corner detection. *British Machine Vision Conference 2017, BMVC 2017*.

[269] Mueggler, E., Rebecq, H., Gallego, G., Delbruck, T., and Scaramuzza, D. (2016). The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *International Journal of Robotics Research*, 36:142–149.

[270] Mueggler, E., Rebecq, H., Gallego, G., Delbruck, T., and Scaramuzza, D. (2017b). The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *International Journal of Robotics Research*, 36:142–149.

[271] Mullikin, D. R., Flanagan, R. P., Merkebu, J., Durning, S. J., and Soh, M. (2024a). Physiologic measurements of cognitive load in clinical reasoning. *Diagnosis*, 11:125–131.

[272] Mullikin, D. R., Flanagan, R. P., Merkebu, J., Durning, S. J., and Soh, M. (2024b). Physiologic measurements of cognitive load in clinical reasoning. *Diagnosis*, 11(2):125–131.

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

[274] Nandakumar, S. R., Gallo, M. L., Boybat, I., Rajendran, B., Sebastian, A., and Eleftheriou, E. (2018a). A phase-change memory model for neuromorphic computing. *Journal of Applied Physics*, 124:152135.

[275] Nandakumar, S. R., Kulkarni, S. R., Babu, A. V., and Rajendran, B. (2018b). Building brain-inspired computing systems: Examining the role of nanoscale devices. *IEEE Nanotechnology Magazine*, 12:19–35.

[276] Nasir, W. A. K. (2020). Introduction to a simulation environment-gazebo.

[277] Naveros, F., Luque, N. R., Ros, E., and Arleo, A. (2020). Vor adaptation on a humanoid icub robot using a spiking cerebellar model. *IEEE Transactions on Cybernetics*, 50:4744–4757.

[278] Neckar, A., Fok, S., Benjamin, B. V., Stewart, T. C., Oza, N. N., Voelker, A. R., Eliasmith, C., Manohar, R., and Boahen, K. (2019). Braindrop: A mixed-signal neuromorphic architecture with a dynamical systems-based programming model. *Proceedings of the IEEE*, 107:144–164.

[279] Neverova, N., Wolf, C., Taylor, G. W., and Nebout, F. (2014). Moddrop: adaptive multi-modal gesture recognition.

[280] Ni, Z., Bolopion, A., Agnus, J., Benosman, R., and Régnier, S. (2012). Asynchronous event-based visual shape tracking for stable haptic feedback in microrobotics. *IEEE Transactions on Robotics*, 28:1081–1089.

[281] Nilsson, E. J., Aust, M. L., Engström, J., Svanberg, B., and Lindén, P. (2018). Effects of cognitive load on response time in an unexpected lead vehicle braking scenario and the detection response task (drt). *Transportation Research Part F: Traffic Psychology and Behaviour*, 59:463–474.

[282] Noll-Hussong, M., Bouragui, K. E., and Meule, A. (2017). Reporting and interpreting working memory performance in n-back tasks. *Frontiers in Psychology*, 8:352.

[283] Obo, T., Hase, R., Kobayashi, K., Sueta, K., Nakano, T., and Shin, D. (2020). Cognitive modeling based on perceiving-acting cycle in robotic avatar system for disabled patients. *Proceedings of the International Joint Conference on Neural Networks*.

[284] of Manchester, U. (2015). Lab manual spinnaker interfacing external devices.

[285] OpenEB and Prophesee (2021). Training an eb classification model — metavision sdk docs 4.5.1 documentation.

[286] Orchard, G., Jayawant, A., Cohen, G. K., and Thakor, N. (2015). Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in Neuroscience*, 9.

[287] Ostrau, C., Homburg, J., Klarhorst, C., Thies, M., and Rückert, U. (2020). Benchmarking deep spiking neural networks on neuromorphic hardware. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12397 LNCS:610–621.

[288] Ouwehand, K., van der Kroef, A., Wong, J., and Paas, F. (2021). Measuring cognitive load: Are there more valid alternatives to likert rating scales? *Frontiers in Education*, 6.

[289] Ouzzani, M., Hammady, H., Fedorowicz, Z., and Elmagarmid, A. (2016). Rayyan-a web and mobile app for systematic reviews. *Systematic Reviews*, 5.

[290] Paas, F., Renkl, A., and Sweller, J. (2003). Cognitive load theory and instructional design: Recent developments. *Educational Psychologist*, 38:1–4.

[291] Painkras, E., Plana, L. A., Garside, J., Temple, S., Davidson, S., Pepper, J., Clark, D., Patterson, C., and Furber, S. (2012). Spinnaker: A multi-core system-on-chip for massively-parallel neural net simulation.

[292] Pandey, A. K. and Gelin, R. (2018). A mass-produced sociable humanoid robot: Pepper: the first machine of its kind. *IEEE Robotics and Automation Magazine*, 25:40–48.

[293] Parvizi-Fard, A., Amiri, M., Kumar, D., Iskarous, M. M., and Thakor, N. V. (2021). A functional spiking neuronal network for tactile sensing pathway to process edge orientation. *Scientific Reports 2021 11:1*, 11:1–16.

[294] Patel, H., Iaboni, C., Lobo, D., won Choi, J., and Abichandani, P. (2021). Event camera based real-time detection and tracking of indoor ground robots.

[295] Pellerito, R., Cannici, M., Gehrig, D., Belhadj, J., Dubois-Matra, O., Casasco, M., and Scaramuzza, D. (2023). Deep visual odometry with events and frames.

[296] Pellerito, R., Cannici, M., Gehrig, D., Belhadj, J., Dubois-Matra, O., Casasco, M., and Scaramuzza, D. (2024). Deep visual odometry with events and frames. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8966–8973.

[297] Pereira, U. and Brunel, N. (2018). Attractor dynamics in networks with learning rules inferred from in vivo data. *Neuron*, 99:227–238.e4.

[298] Perez-Carrasco, J. A., Acha, B., Serrano, C., Camunas-Mesa, L., Serrano-Gotarredona, T., and Linares-Barranco, B. (2010). Fast vision through frameless event-based sensing and convolutional processing: application to texture recognition. *IEEE transactions on neural networks*, 21:609–620.

[299] Perez-Cutiño, M. A., Eguíluz, A. G., Dios, J. R. M.-D., and Ollero, A. (2021). Event-based human intrusion detection in uas using deep learning. Technical report.

[300] Perot, E., de Tournemire, P., Nitti, D., Masci, J., and Sironi, A. (2020). Learning to detect objects with a 1 megapixel event camera.

[301] Pisharady, P. K. and Saerbeck, M. (2015). Recent methods and databases in vision-based hand gesture recognition: A review. *Computer Vision and Image Understanding*, 141:152–165.

[302] Pitchai, M., Xiong, X., Thor, M., Billeschou, P., Mailänder, P. L., Leung, B., Kulvicius, T., and Manoonpong, P. (2019). Cpg driven rbf network control with reinforcement learning for gait optimization of a dung beetle-like robot. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11727 LNCS:698–710.

[303] Ponulak, F. (2005). Resume-new supervised learning method for spiking neural networks.

[304] Potjans, W., Morrison, A., and Diesmann, M. (2009). A spiking neural network model of an actor-critic learning agent. *Neural computation*, 21:301–339.

[305] Prophesee (2023). Evt 3.0 format — metavision sdk docs 4.3.0 documentation.

[306] Ralph, N., Joubert, D., Jolley, A., Afshar, S., van Schaik, A., and Cohen, G. (2022). Real-time event-based unsupervised feature consolidation and tracking for space situational awareness. *Frontiers in Neuroscience*, 16:821157.

[307] Ramakrishnan, P., Balasingam, B., and Biondi, F. (2021). Cognitive load estimation for adaptive human–machine system automation. *Learning Control: Applications in Robotics and Complex Dynamical Systems*, pages 35–58.

[308] Rast, A. D., Adams, S. V., Davidson, S., Davies, S., Hopkins, M., Rowley, A., Stokes, A. B., Wennekers, T., Furber, S., and Cangelosi, A. (2018). Behavioral learning in a cognitive neuromorphic robot: An integrative approach. *IEEE Transactions on Neural Networks and Learning Systems*, 29:6132–6144.

[309] Rebecq, H., Horstschaefer, T., Gallego, G., and Scaramuzza, D. (2017). Evo: A geometric approach to event-based 6-dof parallel tracking and mapping in real time. *IEEE Robotics and Automation Letters*, 2:593–600.

[310] Rebecq, H., Ranftl, R., Koltun, V., and Scaramuzza, D. (2019). Events-to-video: Bringing modern computer vision to event cameras. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019-June:3852–3861.

[311] Renner, A., Evanusa, M., and Sandamirskaya, Y. (2019). Event-based attention and tracking on neuromorphic hardware.

[312] Report, S. B. R. A. and Canosa, R. (2011). Obstacle detection and avoidance using turtlebot platform and xbox kinect.

[313] Rodriguez, R. M., Cristalli, C., and Paone, N. (2002). Comparative study between laser vibrometer and accelerometer measurements for mechanical fault detection of electric motors. *Fifth International Conference on Vibration Measurements by Laser Techniques: Advances and Applications*, 4827:521–529.

[314] Rodriguez-Gomez, J. P., de Dios, J. R., Ollero, A., and Gallego, G. (2024). On the benefits of visual stabilization for frame- and event-based perception. *IEEE Robotics and Automation Letters*.

[315] Rodriguez-Gomez, J. P., Eguiluz, A. G., Dios, J. R. M.-D., and Ollero, A. (2020). Asynchronous event-based clustering and tracking for intrusion monitoring in uas. Technical report.

[316] Rodriguez-Gomez, J. P., Eguiluz, A. G., Dios, J. R. M.-D., and Ollero, A. (2021). Auto-tuned event-based perception scheme for intrusion monitoring with uas. *IEEE Access*, 9:44840–44854.

[317] Roy, K., Jaiswal, A., and Panda, P. (2019). Towards spike-based machine intelligence with neuromorphic computing. *Nature 2019 575:7784*, 575:607–617.

[318] Rueckauer, B. and Liu, S.-C. (2018). Conversion of analog to spiking neural networks using sparse temporal coding. In *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5.

[319] Rueckauer, B., Lungu, I.-A., Hu, Y., and Pfeiffer, M. (2016). Theory and tools for the conversion of analog to spiking convolutional neural networks.

[320] Rueckauer, B., Lungu, I. A., Hu, Y., Pfeiffer, M., and Liu, S. C. (2017). Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in neuroscience*, 11.

[321] Rueckert, E., Kappel, D., Tanneberg, D., Pecevski, D., and Peters, J. (2016). Recurrent spiking networks solve planning tasks. *Scientific Reports 2016 6:1*, 6:1–10.

[322] Rzeszut, P., Chciński, J., Brzozowski, I., Zitek, S., Skowroński, W., and Stobiecki, T. (2022). Multi-state mram cells for hardware neuromorphic computing. *Scientific Reports 2022 12:1*, 12:1–11.

[323] Saenz, A. (2009). Facets: Making computers work like brains.

[324] Sammoud, A., Kumar, A., Bayoumi, M., and Elarabi, T. (2017). Real-time streaming challenges in internet of video things (iovt). *Proceedings - IEEE International Symposium on Circuits and Systems*.

[325] Sanyal, S., Manna, R. K., and Roy, K. (2023). Ev-planner: Energy-efficient robot navigation via event-based physics-guided neuromorphic planner.

[326] Saucedo, M. A., Patel, A., Sawlekar, R., Saradagi, A., Kanellakis, C., Agha-Mohammadi, A. A., and Nikolakopoulos, G. (2023). Event camera and lidar based human tracking for adverse lighting conditions in subterranean environments. In *IFAC-PapersOnLine*, volume 56, pages 9257–9262. Elsevier B.V.

[327] Scarpellini, G., Morerio, P., and Bue, A. D. (2021). Lifting monocular events to 3d human poses. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1358–1368.

[328] Schaffer, J. D. (2020). Evolving spiking neural networks for robot sensory-motor decision tasks of varying difficulty. Association for Computing Machinery.

[329] Schemmel, J., Brüderle, D., Grübl, A., Hock, M., Meier, K., and Millner, S. (2010). A wafer-scale neuromorphic hardware system for large-scale neural modeling. *ISCAS 2010 - 2010 IEEE International Symposium on Circuits and Systems: Nano-Bio Circuit Fabrics and Systems*, pages 1947–1950.

[330] Sepp, S., Howard, S. J., Tindall-Ford, S., Agostinho, S., and Paas, F. (2019). Cognitive load theory and human movement: Towards an integrated model of working memory. *Educational Psychology Review*, 31:293–317.

[331] Serrano-Gotarredona, T. and Linares-Barranco, B. (2013). A 128 $\times$ 128 1.5 contrast sensitivity 0.9 fpn 3 s latency 4 mw asynchronous frame-free dynamic vision sensor using transimpedance preamplifiers. *IEEE Journal of Solid-State Circuits*, 48:827–838.

[332] Serrano-Gotarredona, T. and Linares-Barranco, B. (2015). Poker-dvs and mnist-dvs. their history, how they were made, and other details. *Frontiers in Neuroscience*, 9:165839.

[333] Sharma, S., Aubin, S., and Eliasmith, C. (2016). Large-scale cognitive model design using the nengo neural simulator.

[334] Shrestha, S. B. and Orchard, G. (2018). Slayer: Spike layer error reassignment in time. *Advances in Neural Information Processing Systems*, 31.

[335] Siciliano, R. (2012). The hodgkin-huxley model its extensions, analysis and numerics contents.

[336] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Driessche, G. V. D., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of go with deep neural networks and tree search. *Nature 2016 529:7587*, 529:484–489.

[337] Soares, J. M., Navarro, I., and Martinoli, A. (2016). The khepera iv mobile robot: Performance evaluation, sensory data, and software toolbox.

[338] Society, I. C., of Electrical, I., and Engineers, E. (2015). *Computer Vision and Pattern Recognition Workshops (CVPRW), 2015 IEEE Conference on : date, 7-12 June 2015.*

[339] Song, L., Yu, G., Yuan, J., and Liu, Z. (2021). Human pose estimation and its application to action recognition: A survey. *Journal of Visual Communication and Image Representation*, 76:103055.

[340] SpikeFun (2012). Spikefun v0.90 - more complex pyramidal neurons, demo with 3.52 billion synapses.

[341] Srinivasa, N. and Cruz-Albrecht, J. M. (2012). Neuromorphic adaptive plastic scalable electronics: Analog learning systems. *IEEE Pulse*, 1:51 – 56.

[342] Steffen, L., Hauck, B., Kaiser, J., Weinland, J., Ulbrich, S., Reichard, D., Roennau, A., and Dillmann, R. (2025). Creating an obstacle memory through event-based stereo vision and robotic proprioception. Technical report.

[343] Stensola, T. and Moser, E. I. (2016). Grid cells and spatial maps in entorhinal cortex and hippocampus. *Research and Perspectives in Neurosciences*, pages 59–80.

[344] Stoffregen, T. and Kleeman, L. (2019). Event cameras, contrast maximization and reward functions: an analysis.

[345] Strohmer, B., Stagsted, R. K., Manoonpong, P., and Larsen, L. B. (2021). Integrating non-spiking interneurons in spiking neural networks. *Frontiers in Neuroscience*, 15:633945.

[346] Stromatias, E., Galluppi, F., Patterson, C., and Furber, S. (2013). Power analysis of large-scale, real-time neural networks on spinnaker. *Proceedings of the International Joint Conference on Neural Networks*, pages 1570–1577.

[347] Sugiarto, I., Liu, G., Davidson, S., Plana, L. A., and Furber, S. B. (2016). High performance computing on spinnaker neuromorphic platform: A case study for energy efficient image processing.

[348] Sun, H. and Fremont, V. (2023). Object tracking with a fusion of event-based camera and frame-based camera. *Lecture Notes in Networks and Systems*, 543 LNNS:250–264.

[349] Suryani, M., Santoso, H. B., Schrepp, M., Aji, R. F., Hadi, S., Sensuse, D. I., Suryono, R. R., and Kautsarina (2024). Role, methodology, and measurement of cognitive load in computer science and information systems research. *IEEE Access*.

[350] Susto, G. A., Schirru, A., Pampuri, S., McLoone, S., and Beghi, A. (2015). Machine learning for predictive maintenance: A multiple classifier approach. *IEEE Transactions on Industrial Informatics*, 11:812–820.

[351] SVH, S. (2016). Translation of original operating manual assembly and operating manual svh servo-electric 5-finger gripping hand.

[352] Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12:257–285.

[353] Sweller, J., Ayres, P., and Kalyuga, S. (2011). Cognitive load theory.

[354] Tan, H., Zhang, Z., Zou, X., Liao, Q., and Xia, W. (2020). Exploring the potential of fast delta encoding: Marching to a higher compression ratio. In *2020 IEEE International Conference on Cluster Computing (CLUSTER)*, pages 198–208.

[355] Tanaka, R., Woo, J., and Kubota, N. (2019). *Action Acquisition Method for Constructing Cognitive Development System Through Instructed Learning; Action Acquisition Method for Constructing Cognitive Development System Through Instructed Learning*.

[356] Tang, G. and Michmizos, K. P. (2020). Real-time mapping on a neuromorphic processor. Association for Computing Machinery.

[357] Tang, H., Yan, R., and Tan, K. C. (2018). Cognitive navigation by neuro-inspired localization, mapping, and episodic memory. *IEEE Transactions on Cognitive and Developmental Systems*, 10:751–761.

[358] Tao, D., Zhang, X., Cai, J., Tan, H., Zhang, X., and Zhang, T. (2020). Physiological measures of mental workload: Evidence from empirical studies. *Lecture Notes in Electrical Engineering*, 576:217–225.

[359] Tapia, R., Rodriguez-Gomez, J. P., Sanchez-Diaz, J. A., Gañán, F. J., Rodriguez, I. G., Luna-Santamaria, J., de Dios, J. R. M., and Ollero, A. (2023). A comparison between frame-based and event-based cameras for flapping-wing robot perception.

[360] Tarakli, I. and Nuovo, A. D. (2024). User perception of teachable robots: A comparative study of teaching strategies, task complexity and user characteristics. *Lecture Notes in Computer Science*, 14454 LNAI:357–370.

[361] Tian, S., Qu, L., Wang, L., Hu, K., Li, N., and Xu, W. (2021). A neural architecture search based framework for liquid state machine design. *Neurocomputing*, 443:174–182.

[362] Tico, M., Gelfand, N., and Pulli, K. (2010). Motion-blur-free exposure fusion. *Proceedings - International Conference on Image Processing, ICIP*, pages 3321–3324.

[363] Tieck, J. C. V., Kaiser, J., Steffen, L., Schulze, M., Arnim, A. V., Reichard, D., Roennau, A., and Dillmann, R. (2019a). The neurorobotics platform for teaching - embodiment experiments with spiking neural networks and virtual robots. *2019 IEEE International Conference on Cyborg and Bionic Systems, CBS 2019*, pages 291–298.

[364] Tieck, J. C. V., Schnell, T., Kaiser, J., Mauch, F., Roennau, A., and Dillmann, R. (2019b). Generating pointing motions for a humanoid robot by combining motor primitives. *Frontiers in Neurorobotics*, 13.

[365] Tieck, J. C. V., Secker, K., Kaiser, J., Roennau, A., and Dillmann, R. (2021). Soft-grasping with an anthropomorphic robotic hand using spiking neurons. *IEEE Robotics and Automation Letters*, 6:2894–2901.

[366] Tieck, J. C. V., Weber, S., Stewart, T. C., Kaiser, J., Roennau, A., and Dillmann, R. (2020). A spiking network classifies human semg signals and triggers finger reflexes on a robotic hand. *Robotics and Autonomous Systems*, 131.

[367] Tieck, J. C. V., Weber, S., Stewart, T. C., Roennau, A., and Dillmann, R. (2018). Triggering robot hand reflexes with human emg data using spiking neurons. *Advances in Intelligent Systems and Computing*, 867:902–916.

[368] Ting, J., Fang, Y., Lele, A. S., and Raychowdhury, A. (2020). Bio-inspired gait imitation of hexapod robot using event-based vision sensor and spiking neural network.

[369] Toyoizumi, T., Pfister, J. P., Aihara, K., and Gerstner, W. (2005). Generalized bienenstock-cooper-munro rule for spiking neurons that maximizes information transmission. *Proceedings of the National Academy of Sciences of the United States of America*, 102:5239–5244.

[370] Tran, D., Ray, J., Shou, Z., Chang, S.-F., and Paluri, M. (2017). Convnet architecture search for spatiotemporal feature learning.

[371] Treisman, A. (1996). The binding problem. *Current Opinion in Neurobiology*, 6:171–178.

[372] Tsagarakis, N. G., Metta, G., Sandini, G., Vernon, D., Beira, R., Becchi, F., Righetti, L., Santos-Victor, J., Ijspeert, A. J., Carrozza, M. C., and Caldwell, D. G. (2007). Full paper icub: the design and realization of an open humanoid platform for cognitive and neuroscience research. *Advanced Robotics*, 21:1151–1175.

[373] Uludağ, R. B., Çağdaş, S., İşler, Y. S., Şengör, N. S., and Akturk, I. (2023). Bio-realistic neural network implementation on loihi 2 with izhikevich neurons.

[374] Valeiras, D. R., Clady, X., Ieng, S. H., and Benosman, R. (2019). Event-based line fitting and segment detection using a neuromorphic visual sensor. *IEEE Transactions on Neural Networks and Learning Systems*, 30:1218–1230.

[375] Vanneste, P., Raes, A., Morton, J., Bombeke, K., Acker, B. B. V., Larmuseau, C., Depaepe, F., and den Noortgate, W. V. (2021a). Towards measuring cognitive load through multimodal physiological data. *Cognition, Technology and Work*, 23:567–585.

[376] Vanneste, P., Raes, A., Morton, J., Bombeke, K., Acker, B. B. V., Larmuseau, C., Depaepe, F., and den Noortgate, W. V. (2021b). Towards measuring cognitive load through multimodal physiological data. *Cognition, Technology and Work*, 23:567–585.

[377] Varoquaux, G., Ramachandran, P., and Mayavi, P. R. (2008). Making 3d data visualization reusable. *SciPy*.

[378] Vasco, V., Glover, A., and Bartolozzi, C. (2016). Fast event-based harris corner detection exploiting the advantages of event-driven cameras. *IEEE International Conference on Intelligent Robots and Systems*, 2016-November:4144–4149.

[379] Viale, A., Marchisio, A., Martina, M., Masera, G., and Shafique, M. (2021). Carsnn: An efficient spiking neural network for event-based autonomous cars on the loihi neuromorphic research processor.

[380] Villarreal, R. T., Nordstrom, P. A., and Duffy, V. G. (2024). A bibliometric analysis of cognitive load sensing methodologies and its applications. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 14709 LNCS:113–134.

[381] Vitale, A., Renner, A., Nauer, C., Scaramuzza, D., and Sandamirskaya, Y. (2021). Event-driven vision and control for uavs on a neuromorphic chip.

[382] Vora, H., Kathiria, P., Agrawal, S., and Patel, U. (2022). Neuromorphic computing: Review of architecture, issues, applications and research opportunities. *Lecture Notes in Electrical Engineering*, 855:371–383.

[383] Vrij, A., Semin, G. R., and Bull, R. (1996). Insight into behavior displayed during deception. *Human Communication Research*, 22:544–562.

[384] Wang, F., Gupta, S. K., and Hsieh, M. (2016). Csim: A mos switch-level simulator.

[385] Wang, S., Wang, Z., Li, C., Qi, X., and So, H. K.-H. (2025). Spikemot: Event-based multi-object tracking with sparse motion features. *IEEE Access*, 13:214–230.

[386] Wang, X., Lin, X., and Dang, X. (2020). Supervised learning in spiking neural networks: A review of algorithms and evaluations. *Neural Networks*, 125:258–280.

[387] Wang, X., Wu, Z., Jiang, B., Bao, Z., Zhu, L., Li, G., Wang, Y., and Tian, Y. (2022a). Hardvs: Revisiting human activity recognition with dynamic vision sensors.

[388] Wang, X., Wu, Z., Jiang, B., Bao, Z., Zhu, L., Li, G., Wang, Y., and Tian, Y. (2022b). Hardvs: Revisiting human activity recognition with dynamic vision sensors.

[389] Wang, Y., Li, F., Zheng, H., Jiang, L., Mahani, M. F., and Liao, Z. (2023a). Human trust in robots: A survey on trust models and their controls/robotics applications. *IEEE Open Journal of Control Systems*, 3:58–86.

[390] Wang, Y., Liu, H., Zhang, M., Luo, X., and Qu, H. (2024). A universal ann-to-snn framework for achieving high accuracy and low latency deep spiking neural networks. *Neural Networks*, 174:106244.

[391] Wang, Z. (2021). Building experiments with psychopy. *Eye-Tracking with Python and Pylink*, pages 27–63.

[392] Wang, Z., Ojeda, F. C., Bisulco, A., Lee, D., Taylor, C. J., Daniilidis, K., Hsieh, M. A., Lee, D. D., and Isler, V. (2023b). Ev-catcher: High-speed object catching using low-latency event-based neural networks.

[393] Wilbanks, B. A., Aroke, E., and Dudding, K. M. (2021). Using eye tracking for measuring cognitive workload during clinical simulations: Literature review and synthesis. *Computers, informatics, nursing : CIN*, 39:499–507.

[394] William Chamorro, J. S. and Andrade-Cetto, J. (2023). Event-based line slam in real-time. *Lenguas Modernas*, pages 183–209.

[395] Williams, C. K. I. (2020). The effect of class imbalance on precision-recall curves.

[396] Wu, J., Zhang, K., Zhang, Y., Xie, X., and Shi, G. (2019). High-speed object tracking with dynamic vision sensor. *Lecture Notes in Electrical Engineering*, 552:164–174.

[397] Xiao, H. and Chen, X. (2022). *Target Tracking with Frame- and Event-based Cameras Involving Delayed and Irregularly-Sampled Visual Feedback for a Robotic Air-Hockey System*. IEEE.

[398] Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

[399] Xing, Y., Caterina, G. D., and Soraghan, J. (2020). A new spiking convolutional recurrent neural network (scrnn) with applications to event-based hand gesture recognition. *Frontiers in Neuroscience*, 14:590164.

[400] Yan, Y., Chu, H., Jin, Y., Huan, Y., Zou, Z., and Zheng, L. (2022). Backpropagation with sparsity regularization for spiking neural network learning. *Frontiers in Neuroscience*, 0:340.

[401] Yang, H., Yang, S., Zhang, L., Dou, H., Shen, F., and Zhao, J. (2025). Cs-qcfs: Bridging the performance gap in ultra-low latency spiking neural networks. *Neural Networks*, 184:107076.

[402] Yang, Y., Zhang, L., and Wang, X. (2019). Time delay performance analysis of distributed communication platform based on zeromq. *Proceedings - 2019 International Conference on Communications, Information System, and Computer Engineering, CISCE 2019*, pages 319–323.

[403] Yantis, S., Dir, C., and Author, P. S. (2008). The neural basis of selective attention: Cortical sources and targets of attentional modulation. *Current directions in psychological science*, 17:86.

[404] Yao, M., Richter, O., Zhao, G., Qiao, N., Xing, Y., Wang, D., Hu, T., Fang, W., Demirci, T., Marchi, M. D., Deng, L., Yan, T., Nielsen, C., Sheik, S., Wu, C., Tian, Y., Xu, B., and Li, G. (2024). Spike-based dynamic computing with asynchronous sensing-computing neuromorphic chip. *Nature Communications 2024 15:1*, 15:1–18.

[405] Ye, C., Mitrokhin, A., Fermuller, C., Yorke, J. A., and Aloimonos, Y. (2020). Unsupervised learning of dense optical flow, depth and egomotion with event-based sensors. *IEEE International Conference on Intelligent Robots and Systems*, pages 5831–5838.

[406] Yedukondalu, J., Sunkara, K., Radhika, V., Kondaveeti, S., Anumothu, M., and Krishna, Y. M. (2025). Cognitive load detection through eeg lead wise feature optimization and ensemble classification. *Scientific Reports 2024 15:1*, 15:1–18.

[407] Youssef, I., Mutlu, M., Bayat, B., Crespi, A., Hauser, S., Conradt, J., Bernardino, A., and Ijspeert, A. (2020). A neuro-inspired computational model for a visually guided robotic lamprey using frame and event based cameras. Technical report.

[408] Yu, M., Xiang, T., P, S., Chu, K. T. N., Amornpaisannon, B., Tavva, Y., Miriyala, V. P. K., and Carlson, T. E. (2023). A ttfs-based energy and utilization efficient neuromorphic cnn accelerator. *Frontiers in Neuroscience*, 17:1121592.

[409] Zahra, O., Navarro-Alarcon, D., and Tolu, S. (2021). A neurorobotic embodiment for exploring the dynamical interactions of a spiking cerebellar model and a robot arm during vision-based manipulation tasks.

[410] Zeng, Y., Zhao, D., Zhao, F., Shen, G., Dong, Y., Lu, E., Zhang, Q., Sun, Y., Liang, Q., Zhao, Y., Zhao, Z., Fang, H., Wang, Y., Li, Y., Liu, X., Du, C., Kong, Q., Ruan, Z., and Bi, W. (2023). Braincog: A spiking neural network based, brain-inspired cognitive intelligence engine for brain-inspired ai and brain simulation. *Patterns*, 4(8):100789.

[411] Zhang, J., Zhou, S., Wang, J., and Huang, D. (2019). Frame-wise motion and appearance for real-time multiple object tracking.

[412] Zhang, W., Li, X., Liu, X., Lu, S., and Tang, H. (2025). Facing challenges: A survey of object tracking. *Digital Signal Processing*, 161:105082.

[413] Zharinov, A. I., Makarov, V. A., Kazantsev, V. B., and Lobov, S. A. (2020). Spatial memory based on an stdp-driven neural network. *Conference Proceedings - 4th Scientific School on Dynamics of Complex Networks and their Application in Intellectual Robotics, DCNAIR 2020*, pages 269–271.

[414] Zhou, T. and Wachs, J. P. (2018). Early turn-taking prediction with spiking neural networks for human robot collaboration. pages 3250–3256. Institute of Electrical and Electronics Engineers Inc.

[415] Zhou, T. and Wachs, J. P. (2019). Spiking neural networks for early prediction in human–robot collaboration. *International Journal of Robotics Research*, 38:1619–1643.

[416] Zhou, Y., Gallego, G., and Shen, S. (2020). Event-based stereo visual odometry.

[417] Zhou, Z., Wu, Z., Boutteau, R., Yang, F., Demonceaux, C., and Ginhac, D. (2022). Rgb-event fusion for moving object detection in autonomous driving.

[418] Zhu, A. Z., Atanasov, N., and Daniilidis, K. (2017). Event-based feature tracking with probabilistic data association. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 4465–4470.

[419] Zhu, A. Z., Chen, Y., and Daniilidis, K. (2018a). Realtime time synchronized event-based stereo. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11210 LNCS:438–452.

[420] Zhu, A. Z., Yuan, L., Chaney, K., and Daniilidis, K. (2018b). Ev-flownet: Self-supervised optical flow estimation for event-based cameras. *Robotics: Science and Systems*.

[421] Zhu, S., Tang, Z., Yang, M., Learned-Miller, E., and Kim, D. (2023). Event camera-based visual odometry for dynamic motion tracking of a legged robot using adaptive time surface.

[422] Zou, S., Guo, C., Zuo, X., Wang, S., Wang, P., Hu, X., Chen, S., Gong, M., and Cheng, L. (2021). Eventhpe: Event-based 3d human pose and shape estimation. *Proceedings of the IEEE International Conference on Computer Vision*, pages 10976–10985.

[423] Zujevs, A., Pudzs, M., Osadcuks, V., Ardavs, A., Galauskis, M., and Grundspenkis, J. (2021). An event-based vision dataset for visual navigation tasks in agricultural environments. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2021-May, pages 3707–3713. Institute of Electrical and Electronics Engineers Inc.