Potentials of Neuromorphic Computing for Automated Driving and Future Transportation Systems

Lorenz Bayerlein*†, Jonas V. Schulte*† and Steven Peters*

Abstract: Neuromorphic computing, inspired by the structure and functionality of the human brain, offers a transformative potential for advancing automated driving systems. This review examines the role of neuromorphic computing in overcoming current limitations in AI-based perception systems, particularly with respect to energy efficiency, real-time processing, and robustness. Using spiking neural networks and event-driven architectures, neuromorphic systems enable more efficient computation than traditional AI models, which require significant computational resources and power. This work explores specific scenarios where neuromorphic computing outperforms traditional methods and highlights how neuromorphic hardware can improve data integration, reduce power consumption, and increase the reliability of automated driving systems. This review concludes that neuromorphic computing is not only a viable alternative, but a superior approach for future advances in automated driving technology, offering a path to more efficient and adaptive systems.

Keywords: AI, automated driving, neuromorphic computing, spiking neural networks, energy efficiency

1 Introduction

The rapid advancements in automation and digital transformation are significantly increasing the complexity of modern vehicle systems. Artificial Intelligence (AI) is an important part of this transformation and a key enabler for automated driving and future transportation systems. As a core component of such automated driving systems, AI must meet performance, latency and energy efficiency requirements. However, current AI benchmarks focus primarily on performance metrics, prioritising model accuracy while overlooking critical issues such as energy efficiency, real-time capabilities and robustness.

When it comes to environmental perception for vehicle automation, Convolutional Neural Networks (CNNs) have traditionally been considered the gold standard for perception tasks. In recent years, the emergence of transformer architectures has introduced a new paradigm with models that can outperform CNNs in terms of model quality, see Srivastava and Sharma [1] on the ImageNet dataset [2]. Despite these improvements,

*All authors are with the Institute of Automotive Engineering Darmstadt (FZD), TU Darmstadt, Otto-Berndt-Str. 2, 64287 Darmstadt, Germany (e-mail: {fistname.lastname}@tu-darmstadt.de).

†These authors contributed equally to this work and share first authorship.

transformer architectures fall short in addressing key concerns related to latency and energy efficiency [3].

As AI becomes more deeply embedded in a wide range of products, the balance between performance and power consumption is gradually being reassessed. In conjunction with the transition to integrated systems, the hardware on which AI is computed is also evolving towards hardware that is specialised for the task at hand. Initially, neural networks were processed using CPUs. Since the late 2000s, the use of GPUs for training CNNs has become more widespread [4]. GPUs allow a high degree of parallelisation of relatively simple computational tasks, making them well suited to computing the operations required for CNNs. The move to GPUs was a first step towards dedicated hardware designed to meet the specific needs of AI applications. GPUs provide a more suitable computational structure for AI than CPUs, and scaling to larger architectures and more complex tasks increases the power requirements of GPUs. For example, OpenAI as an AI deployment company relies on large numbers of GPUs to train its networks [5]. On the other hand, for neural network inference in consumer products, there is a trend towards Application-Specific Integrated Circuits (ASICs) and dedicated AI accelerators, especially in cases where real-time decision making and energy efficiency are critical. Examples include Tesla's Full Self-Driving (FSD) chip and Mobileye's EyeQ chips [6].

Beyond CNNs and transformer architectures, brain-inspired algorithms and neuromorphic hardware offer promising ways to improve both energy efficiency and real-time performance for AI-driven functions of automated vehicles and future transportation systems. This paper explores possible use cases and the potential of neuromorphic computing to overcome the latency and power limitations of traditional AI.

In particular, this paper discusses the potential of neuromorphic systems in key areas such as energy-efficient and robust perception, driver monitoring and assistance, highly integrated smart sensors, and event-based data processing. By exploring these areas, this paper aims to highlight the future role of neuromorphic computing in advancing automotive technology, improving energy efficiency, and meeting the real-time demands of automated driving systems without compromising performance.

We will first review the background and state of the art of Spiking Neural Networks (SNN) and neuromorphic hardware in section 2. From this, we will derive potential applications and benefits of neuromorphic computing in section 3 and identify remaining challenges to achieving them. Section 4 provides a summary of the paper and outlines the goals for improving the maturity level of neuromorphic computing, and offers a perspective on future research directions.

2 Spiking Neural Networks and Neuromorphic Hardware

The field of neuromorphic engineering aims to mimic the structure and functionality of the brain by designing sensors, algorithms, and hardware capable of simulating biological neural networks. These systems promise significant potential in terms of processing speed, energy efficiency, and tasks requiring complex pattern recognition and decision-making. This paragraph explores several key areas within neuromorphic computing. First, we introduce spiking neuron models, discussing different levels of abstraction in approximating

biological neurons and the computational trade-offs involved. Next, we examine the challenges and methods in the training of deep SNNs, including popular approaches like ANN-to-SNN conversion and direct training via surrogate gradients. Then, the article delves into SNN architectures, highlighting their application in tasks like image classification, object detection, and radar perception. Finally, we provide an overview of neuromorphic hardware, emphasising the distinctions between analogue, digital, and mixed-signal implementations, and the unique benefits of integrating such hardware with SNNs to drive advances in computational neuroscience and machine learning.

2.1 Spiking Neuron Models

The processes in biological neurons can be described as follows. A neuron receives electrical signals from other neurons through its dendrites. When the combined input exceeds a certain threshold, the neuron generates an action potential, also known as a spike, which travels down its axon to communicate with other neurons. Axons and dendrites are connected by synapses. These connections represent the strength of the connection between two neurons. To create an artificial neuron, the biological neuron must first be represented in an abstract form. Different levels of abstraction can be used to approximate biological neurons [7]. In the models, the dendrites and axons are typically simplified and represented as synapses.

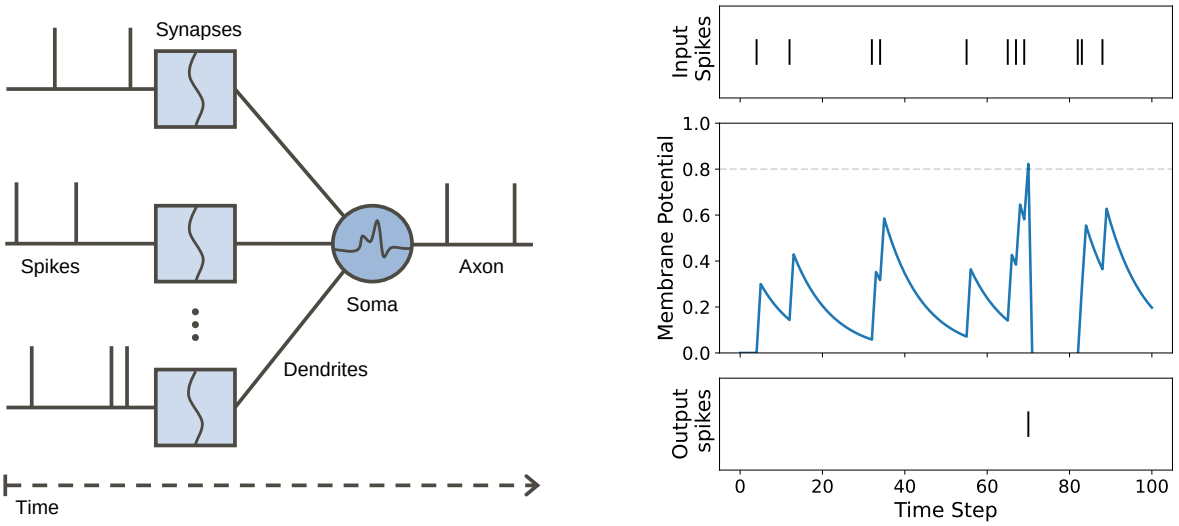
In this context the artificial neuron of an Artificial Neural Network (ANN) can be understood as a broad generalisation of the biological neuron, comprising a neuron model without any temporal dynamics. This generalisation allows us to negate the time dependency in ANNs and describe them as successive layers of matrix and tensor operations. The inputs, outputs, and activation in each layer are represented using floating-point or integer values.

In contrast to the abstract artificial neuron of ANNs, there are neuron models that represent biological neurons in greater detail [8]. These models aim to approximate the internal processes of neurons in order to calculate their behaviour. They require multiple Ordinary Differential Equations (ODEs) to describe the neural dynamics, which makes them computationally expensive. Between these two extremes of a time-independent artificial neuron and a neuron model that mimics sub-neuronal components, there are phenomenological neuron models that represent the dynamics of biological neurons at the behavioural level.

The first phenomenological neuron model was documented by Lapique in 1907 [9]. He attempted to reproduce the observed dynamics of biological neurons using an RC circuit, thereby developing the Leaky-Integrate-and-Fire (LIF) model. It can be described as follows:

$$-\tau \frac{dU(t)}{dt} = U(t) + I_{\mathrm{In}}(t)R. \tag{1}$$

In this model, $U(t)$ is the membrane potential of the neuron cell. The activation $I_{\mathrm{In}}(t)$ is introduced by incoming spikes. R represents the strength of the synaptic connection and scales the activation accordingly. The leakage τ leads to a decay of the membrane potential over time. There are many other phenomenological models besides the LIF neuron model. However, the LIF neuron is the most widely used due to its computational efficiency and ease of use in training [7].



(a) Spiking neuron with multiple synaptic inputs.

(b) Spiking response of the LIF neuron to several input spikes.

Figure 1: Schematic representation of a spiking neuron with multiple synaptic inputs and its membrane potential dynamics. In (a), the diagram depicts a spiking neuron, including key components such as dendrites receiving multiple synaptic inputs, the soma (cell body), and the axon responsible for propagating spikes. In (b), the diagram shows the temporal buildup of the membrane potential in response to multiple input spikes, highlighting the gradual integration of inputs and the reset mechanism following a spike threshold. The response of the LIF neuron was simulated using snnTorch [7].

In spiking networks, the weighted inputs are aggregated over all neurons and then added to the membrane potential as an activation. The membrane potential accumulates electrical charges similar to a capacitor. When the membrane potential exceeds the threshold due to sufficient activation, the neuron emits a spike, which is transmitted across synapses to subsequent neurons, resulting in their activation. This process is illustrated in Figure 1.

2.2 Training of Deep SNNs

Training deep SNNs represents a complex and evolving area of research. Due to their closer approximation to biological processes and their energy efficiency, SNNs offer distinct advantages over traditional ANNs. However, the development of effective training methods for SNNs is a significant challenge, especially given their discrete and event-driven nature. Several training approaches have emerged, each with its own strengths and limitations.

One of the most popular methods is the conversion of pre-trained ANNs into SNNs, also known as ANN-to-SNN conversion or shadow training. This approach uses fully trained ANNs that are then converted to spike-based models, allowing the benefits of well-researched ANN training techniques to be applied to spiking networks. Typically, this process uses rate coding [10]–[12], where the activation of neurons in an ANN is mapped to spike frequencies in an SNN. This method allows the extensive research on

deep learning with ANNs to be leveraged, but it comes with trade-offs. Although ANN-to-SNN conversion can preserve the overall structure and functionality of the original network, its reliance on rate code often leads to inefficiencies in capturing precise temporal dynamics, which can limit performance in tasks requiring fine temporal resolution [13]. In addition, converting high-precision activations into spikes often requires a large number of simulation time steps, which can undermine the energy and latency benefits originally intended with SNNs [14].

Another prominent training approach is the direct training of SNNs via backpropagation [15], [16]. This allows the use of gradient-based optimisation, a key technique in deep learning. Unlike ANN-to-SNN conversion, direct training allows SNNs to take full advantage of their inherent temporal dynamics, making this method suitable for tasks that depend on the precise timing of spikes [17]. Due to the non-differentiable nature of spikes, direct training of SNNs presents unique challenges. To address this, surrogate gradient methods have been explored [18], which approximate the non-differentiable spike function with a smooth gradient during the training process. However, direct training is computationally expensive because the SNN needs to be simulated for multiple time steps in the forward pass. In addition, the optimisation process is highly sensitive to various hyperparameters, including the choice of time steps, firing thresholds, and the surrogate gradient function itself.

An alternative approach to training SNNs is provided by local learning rules, such as Spike-Timing-Dependent Plasticity (STDP) [19]. These rules are inspired by biological neural systems and adjust synaptic weights based on the relative timing of pre- and post-synaptic spikes. STDP provides a biologically plausible mechanism for learning, but it struggles to scale effectively to deep architectures. While local learning rules excel at capturing fine temporal relationships between spikes, they often require additional support from global optimisation techniques to match the performance of other methods [20].

The relationship between the chosen training method and the type of spike coding used is crucial in determining the performance and efficiency of SNNs. For example, ANN-to-SNN conversion typically uses rate coding, where information is encoded in the frequency of spikes over a period of time [21]. This is relatively easy to implement, but may not fully exploit the temporal dynamics that SNNs are capable of. On the other hand, direct training can work with both rate and temporal coding [22]. Temporal coding, where the exact timing of spikes conveys information, is particularly well suited to SNNs and offers the potential for more efficient and powerful representations [23]. Local learning rules are inherently linked to temporal coding, as the timing of spikes directly influences synaptic adaptations [24].

Several challenges arise when training deep SNNs. One major issue is the vanishing or exploding gradient problem, which becomes particularly pronounced when dealing with the temporal dimension of spike trains [25]. In addition, the computational complexity of simulating spiking neurons, especially across multiple layers and time steps, makes training deep networks significantly more resource intensive than their ANN counterparts [26]. Furthermore, the choice of spike coding directly affects both the efficiency and accuracy of the model, creating a delicate balance between energy efficiency and computational power [27].

While techniques from traditional deep learning, such as dropout [28] and weight initialisation [29], have been successfully adapted to improve certain aspects of SNN training,

they are not yet widely or fully integrated into the SNN domain. For example, methods such as batch normalisation [30], which stabilises activations in ANNs, are still being explored for their potential to control spike activity and increase training stability in SNNs [31]. Despite the promise of these techniques, their application to SNNs is still an active area of research and many challenges remain before they can be as effective as they are in the ANN context.

2.3 SNN Architectures

The architecture of a neural network defines its structure and how information is processed. It determines how data flows through the network, what transformations are applied, and what tasks the network can perform based on its inputs and outputs. In the case of SNNs, the basic architectures often mirror those of ANNs. However, SNNs differ in their neuron models and learning rules, which are specifically designed to account for spiking dynamics.

The first implementation of a spiking LeNet architecture was introduced by Cao et al. in 2014 [32]. In 2017, Rueckauer et al. [11] extended this approach by implementing a VGG architecture with spiking neurons. Since then, several studies have explored spiking versions of VGG and ResNet architectures for image processing. Notable works include those by Sengupta et al. [12], Li et al. [33], and Deng & Guo [34], as well as specific architectures such as RMP-SNN [35], SpikingResNet [36], and SEW-ResNet [37]. While all these approaches build upon the same fundamental network architectures, they differ in key aspects such as neuron models, encoding schemes, training methods, and pooling strategies.

The studies mentioned above demonstrate that the implementation of these architectures is well-established within the SNN literature. In 2019, Kim et al. [38] introduced the first SNN-based approach for object detection. Later, in 2022, they further demonstrated the feasibility of image segmentation using spiking neurons [39]. Building on this, Su et al. [40] were the first to train an SNN directly in the spike domain for object detection with EMS-YOLO. More recently, spiking transformer architectures for camera-based perception have also been explored [41], [42].

The first studies on spiking LiDAR perception were conducted by Zhou et al. [43] in 2018. They developed a hybrid LiDAR detector combining artificial and spiking neurons. In subsequent work, they introduced a spiking LiDAR detector based on YOLOv2 [44], [45]. More recently, Ren et al. [46] extended this field by presenting a spiking PointNet approach in 2023. These advances demonstrate the growing interest in applying SNNs to LiDAR-based perception.

Beyond lidar, researchers have also explored the potential of SNNs for radar perception. Notable contributions come from Vogginger [47] and Javier López-Randulfe [48], [49], who have implemented radar perception as a sequence of spiking processing modules. Their approach follows the conventional radar processing chain, with dedicated spiking components for Fourier transformation, azimuth estimation, target detection and classification. This modular design demonstrates how traditional radar signal processing can be adapted to spiking domains.

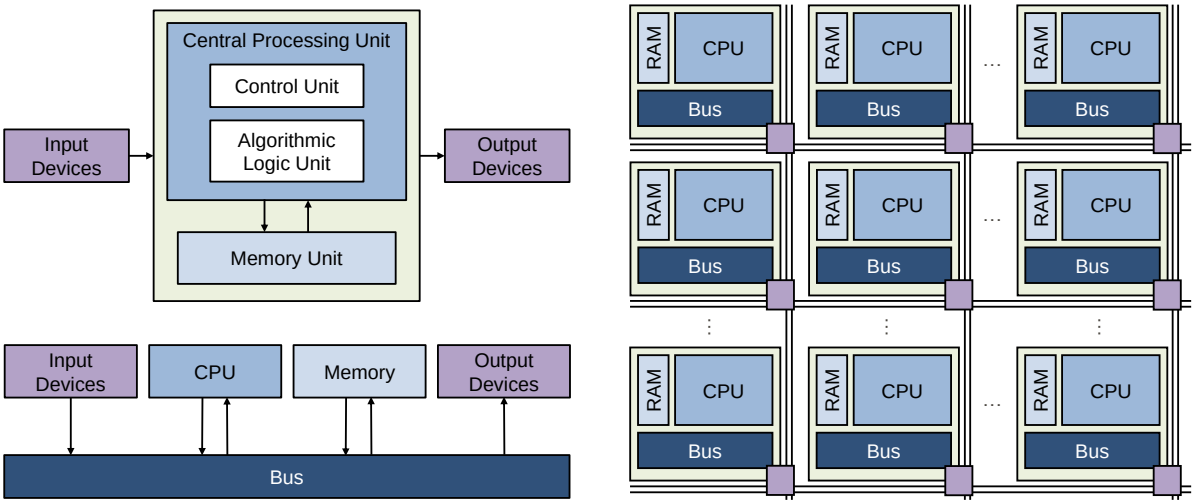
In addition to LiDAR and radar, event-based vision has gained increasing attention in the context of SNNs. Wu et al. [50] and Fang et al. [37], [51] have investigated the processing of Dynamic Vision Sensor (DVS) data, also known as event camera data,

alongside conventional camera-based methods. However, large-scale DVS datasets for object detection did not become available until 2020 with the release of the Prophesee Gen1 and Gen4 datasets [52], [53]. These datasets have since enabled more advanced research into spiking-based event-driven vision systems.

2.4 Neuromorphic Hardware

The traditional von Neumann architecture [54], illustrated in Figure 2a, is characterised by a strict separation between memory and processing units. While this design has been the foundation of computing for decades, it faces growing limitations, particularly in applications requiring parallel processing and low energy consumption. As the demand for more efficient and embedded systems increases, neuromorphic hardware emerges as a promising alternative. Inspired by the brain's architecture, it integrates memory and processing functions in a biologically inspired manner [55].

Neuromorphic hardware seeks to replicate both the structure and function of the brain's neural networks [56]. Instead of processing data sequentially like conventional processors, neuromorphic systems leverage parallel computation, making them highly efficient for real-time processing and adaptive learning [57]. The core components of these systems are neural processing units, which mimic the behaviour of biological neurons and synapses. Neurons in neuromorphic hardware generate and propagate electrical spikes, analogous to action potentials in the brain [58]. Synapses, in turn, regulate the strength of connections between neurons, enabling learning and adaptation through mechanisms similar to synaptic plasticity [59].



(a) Scheme of a von Neumann architecture.

(b) Scheme of a digital neuromorphic chip inspired by [60].

Figure 2: Schematic representation of a von Neumann architecture and a digital neuromorphic chip. The von Neumann architecture in (a) has a separation of the computation unit and the memory unit. In contrast, the neuromorphic architecture in (b) combines memory (RAM) and processor (CPU) at a very low level in a brain-like fashion, allowing a very high degree of parallelism in the processes.

Neuromorphic hardware can be implemented using analogue, digital, or mixed-signal processing techniques, each offering distinct advantages and challenges depending on the application.

Analogue neuromorphic systems aim to closely mimic the continuous, time-dependent behaviour of biological neurons. By emulating ion channels and other sub-neuronal components, they process information in a way that closely resembles biological neural dynamics [61], [62]. However, their reliance on physical variability in analogue circuits can make them less predictable and harder to scale.

Digital neuromorphic systems, in contrast, represent neural activity using discrete signals and are built with standard digital logic circuits. This makes them more compatible with conventional semiconductor manufacturing, allowing for better scalability and integration with existing computing architectures [63]–[65]. A schematic of a digital neuromorphic chip is shown in Figure 2b, where computation is handled by CPUs, and Random-Access Memory (RAM) stores synaptic weights and internal neuron states.

Mixed-signal neuromorphic systems combine elements of both analogue and digital processing to leverage the strengths of each approach. Typically, they use analogue circuits for neuron and synapse emulation while employing digital components for higher-level functions such as data storage and communication [66], [67]. This hybrid approach aims to balance the efficiency and biological fidelity of analogue circuits with the robustness and scalability of digital processing.

3 Use Cases, Potentials and Challenges of Neuromorphic Computing in Automated Driving

Neuromorphic computing holds particular promise for energy-intensive AI processes in automated driving, where the potential for energy savings through SNNs is most significant. In general, the energy consumption of a computational process scales with the number of required operations, which depends on both the task complexity and the execution frequency.

Perception algorithms are among the most computationally demanding processes in automated driving, as they combine high complexity with frequent execution. Their execution rates are comparable to those of safety-critical systems such as Anti-Lock Braking Systems (ABS) and Electronic Stability Control (ESC) [68], [69], often operating in the double-digit Hertz range. The complexity of AI models for perception tasks involves billions of operations per execution [70], leading to considerable energy consumption to maintain low-latency processing. Beyond complex tasks, even seemingly simple processes can become energy-intensive if they remain continuously active. A prime example is wake-word detection in voice control systems, where an audio signal is constantly analysed in the background. While the energy demand per operation is relatively low, the cumulative energy consumption over the vehicle's lifetime can be substantial. This highlights the broad potential for energy savings in both high-performance and always-on AI applications.

A study by Davies et al. [65] examined the energy efficiency of SNNs on the Intel Loihi2 chip compared to traditional implementations on a GPU. Their findings suggest that SNNs can be up to 100 times more energy-efficient, demonstrating the viability of

neuromorphic computing for energy-conscious AI applications.

In addition to their potential for pure energy savings, SNNs on neuromorphic hardware can be directly integrated into intelligent sensor systems, extending their range of applications. By enabling edge processing, this approach reduces the amount of data transmitted within the vehicle, leading to lower bandwidth requirements and improved efficiency. Intelligent sensors, such as automotive radar and cameras, are already widely used in Advanced Driver Assistance Systems (ADAS), including adaptive cruise control and lane keeping assistance.

A particularly promising example of an intelligent sensor is the Dynamic Vision Sensor (DVS), also known as an event camera. Unlike conventional video cameras, which capture frames at a fixed rate, DVS sensors operate asynchronously, detecting only changes in brightness within their field of view. The sensor output consists of a stream of events $E(x, y, p, t)$, where each event represents a change in brightness at a specific location (x, y), time t and polarity p. Each event E_i can be interpreted as a spike occurring at time t_i, with its spatial coordinates and polarity (x_i, y_i, p_i) determining its mapping to an input neuron. By combining DVS with SNNs running on neuromorphic hardware, a highly efficient perception system can be created that processes information natively in spike form, operates with low power consumption, and can be seamlessly integrated into existing vehicle architectures. This makes it an attractive solution for energy-efficient embedded perception in automated driving systems.

In addition to DVS, LiDAR is another suitable technology for integration with SNNs. The time of flight (ToF) signals generated by LiDAR are inherently time-encoded, as the ToF Δt can be directly converted into a spike time for an input neuron in an SNN. The time of flight depends on the distance d to the surface reflecting the laser beam, following the relationship $\Delta t = \frac{2d}{c}$, where c is the speed of light. Each laser beam emitted by a LiDAR is directed at a specific angle, and by combining this beam orientation with the ToF Δt, the precise position of the reflection can be reconstructed. For this approach to be viable, it is essential that the beam pattern remains stable over time or that any temporal variations in the pattern are accounted for. The mapping of spikes to input neurons could follow a similar approach to DVS, ensuring that spatial information is preserved.

Despite its potential, several key challenges must be overcome before this concept can be practically implemented. One of the main obstacles is the training of deep SNNs, which remains significantly more complex than training ANNs. This complexity arises from the additional hyperparameters introduced by temporal dynamics. Factors such as the number of time steps, pooling methods, and the gradient function used in direct training all influence model performance, making optimisation more challenging. Addressing these issues is crucial to enabling the practical application of SNNs in LiDAR-based perception.

ANN-to-SNN conversion has already enabled the implementation of several ANN architectures with spiking neurons. However, a remaining bottleneck is the representation of activation functions [71]. The optimal representation of the Rectified Linear Unit (ReLU) activation function remains an open research question, with ongoing studies exploring different approaches [33]–[35]. An alternative to ANN-to-SNN conversion is the direct training of SNNs, which has led to the development of novel surrogate gradient functions. A notable example is the arctan surrogate [37], which is the default gradient function in snnTorch [7], an open-source Python library designed to facilitate the simulation and training of SNNs within the PyTorch framework.

To fully exploit the potential of the neuromorphic approach, SNNs need to be deployed on dedicated neuromorphic hardware, as this combination enables the highest efficiency and energy savings in automotive applications. While the current state of neuromorphic hardware provides a solid foundation, the market for neuromorphic chips remains relatively limited, with no significant economic demand at present. However, ongoing research and development efforts are expected to drive progress in this area, enabling increasingly complex tasks to be performed on neuromorphic chips in the coming years.

In order to integrate this technology into products, a certain level of standardisation is essential. This includes the development of frameworks, drivers and formats that facilitate the training, exchange and deployment of SNNs on neuromorphic hardware. Currently, several frameworks support the development and deployment of SNNs, some of which provide dedicated hardware drivers, while others operate independently of specific hardware.

A significant step towards standardisation has been achieved with the introduction of the Neuromorphic Intermediate Representation (NIR) [72] in late 2023. NIR enables the transfer of SNN models across a wide range of frameworks, improving interoperability. Several major frameworks have already adopted this format, including Lava [73], Norse [74] and snnTorch [7].

There is currently a lack of dedicated benchmarks and challenges for SNNs. Establishing such benchmarks would allow a direct comparison of different SNN architectures and provide valuable insights into the progress of the field. In many cases, existing ANN benchmarks can serve as a reference for evaluating SNNs, since the underlying tasks and datasets remain the same. However, these benchmarks focus primarily on prediction accuracy, making them inadequate for evaluating key advantages of SNNs, such as event-based processing, temporal dynamics, or energy efficiency. Therefore, dedicated neuromorphic benchmarks are essential to accurately evaluate SNNs and highlight their advantages over traditional AI models. A notable effort in this direction is NeuroBench [75], an initiative that aims to develop a standardised benchmark suite for evaluating SNNs across multiple tasks and hardware platforms.

Similarly, challenges can play a crucial role in fostering innovation and raising the visibility of SNN research. In the computer vision and automated driving communities, challenges have been instrumental in driving progress for at least two decades. Notable examples include the DARPA Grand Challenge and the PASCAL VOC Challenge [76], both of which have significantly shaped their respective fields. The introduction of similar competitions for SNNs could provide strong incentives for researchers to develop and publish their work in a competitive, high-impact environment.

Another step towards practical deployment would be the availability of model zoos. These repositories would provide pre-trained SNN architectures, allowing researchers and developers to apply them directly to neuromorphic hardware without extensive retraining. In the ANN domain, MMDetection [77] and Hugging Face [78] serve as prime examples of such frameworks, demonstrating the benefits of shared, reusable models. Establishing similar resources for SNNs would accelerate adoption and standardisation in the field. The following points summarise key aspects of neuromorphic computing in automated and assisted driving, including its use cases, potential benefits, and remaining challenges.

Use Cases:

- Any perception task currently performed by a CNN could be implemented using an SNN, regardless of the sensor modality.
- This includes tasks such as traffic sign detection, driver attention monitoring, lane detection, and object detection.

Potentials:

- Higher energy efficiency and lower latency in sensor data processing.
- Seamless integration of neuromorphic hardware and software into intelligent sensors.

Challenges:

- Training deep SNNs remains complex, requiring extensive optimisation and additional hyperparameters.
- Limited availability of neuromorphic hardware, restricting widespread adoption.
- Lack of standardised benchmarks, model zoos, and interoperability between frameworks and hardware, hindering broader applicability.

4 Conclusion

Neuromorphic computing offers great potential for the advancement of automated driving and future transportation systems by addressing key challenges in energy efficiency, real-time processing, and adaptability. Conventional AI models such as CNNs and transformers are often used for perception tasks, but their high energy consumption and computational complexity make them difficult to use for the real-time requirements of automated driving. Neuromorphic computing, particularly through SNNs, offers a efficient and scalable alternative by mimicking the event-based processing of the human brain, enabling more adaptive and responsive decision-making in dynamic driving environments.

One of the most compelling advantages of neuromorphic systems is their ability to significantly reduce energy consumption. Hardware such as Intel's Loihi [64], [65] has demonstrated energy savings of up to 100 times compared to traditional GPU-based systems while performing equivalent tasks. Additionally, neuromorphic hardware can be integrated into intelligent sensor systems, allowing for data to be processed directly at the edge, reducing the amount of data transmitted within the vehicle, and lowering overall system complexity.

Beyond energy savings, neuromorphic systems offer key advantages in real-time adaptability. By processing sensory inputs as asynchronous events, SNNs can respond more quickly to changes in the driving environment, making them ideal for tasks such as object detection, sensor fusion, and vehicle control. This real-time capability, combined with the ability to handle noisy and variable data, positions neuromorphic systems as a robust solution for the unpredictable conditions that highly automated vehicles must navigate.

However, significant challenges remain before neuromorphic computing can be fully adopted in commercial automated driving systems. Training deep SNNs, for example, is

complex due to their non-differentiable and event-driven nature. While methods such as ANN-to-SNN conversion and direct training with surrogate gradient methods are promising approaches, they still face scalability issues and may require fine-tuning of time-dependent hyperparameters. Further research is needed to simplify and optimise these training processes for large-scale deployment in real-world applications.

Additionally, the current availability of neuromorphic hardware is mostly limited to research-focused chips, with only a few commercial products on the market. For neuromorphic computing to become mainstream for engineering applications such as automated driving, more specialised hardware and standardised frameworks are needed. Recent developments, such as the introduction of the Neuromorphic Intermediate Representation (NIR) format [72], provide a promising basis for interoperability across different neuromorphic systems, but further progress is needed in this area. Benchmarks and challenges specifically designed for neuromorphic systems would also help accelerate the development and adoption of this technology by providing a clear measure of progress and performance.

Despite these challenges, the long-term potential of neuromorphic computing for automated driving is undeniable. By combining energy efficiency, real-time adaptability, and the ability to handle complex sensory integration, neuromorphic systems could play a pivotal role in the future of automated vehicles. As research continues to refine training methods, hardware development, and standardisation, neuromorphic computing is poised to become a cornerstone technology for the next generation of AI-driven transportation systems. Its ability to operate efficiently in resource-constrained environments and its promise of real-time performance make it an ideal solution for the evolving needs of automated driving, providing a path to more sustainable and robust transportation systems.

ACKNOWLEDGEMENTS

This research is accomplished within the project "AUTOtechagil" (FKZ 01IS22088x). We acknowledge the financial support for the project by the German Federal Ministry of Education and Research (BMBF).

References

[1] S. Srivastava and G. Sharma, "Omnivec: Learning robust representations with cross modal sharing," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2024, pp. 1236–1248.

[2] O. Russakovsky, J. Deng, H. Su, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, pp. 211–252, 2015.

[3] D. Qin, C. Leichner, M. Delakis, *et al.*, "Mobilenetv4-universal models for the mobile ecosystem," *arXiv preprint arXiv:2404.10518*, 2024.

[4] K. Chellapilla, S. Puri, and P. Simard, "High performance convolutional neural networks for document processing," in *Tenth international workshop on frontiers in handwriting recognition*, Suvisoft, 2006.

[5] *Scaling kubernetes to 7,500 nodes*, Accessed: (06.01.2025), Jan. 2025. [Online]. Available: `https://openai.com/index/scaling-kubernetes-to-7500-nodes/`.

[6] M. G. Augusto, J. B. Krug, B. Acar, F. Sivrikaya, and S. Albayrak, "Debunking the myth of high consumption: Power realities in autonomous vehicles," in *2024 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2024, pp. 2869–2875.

[7] J. K. Eshraghian, M. Ward, E. O. Neftci, *et al.*, "Training spiking neural networks using lessons from deep learning," *Proceedings of the IEEE*, 2023.

[8] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology*, vol. 117, no. 4, p. 500, 1952.

[9] L. É. Lapicque, "Recherches quantitatives sur l'excitation electrique des nerfs traitee comme une polarization," *J. physiol*, vol. 9, pp. 620–635, 1907.

[10] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *2015 International joint conference on neural networks (IJCNN)*, ieee, 2015, pp. 1–8.

[11] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers in neuroscience*, vol. 11, p. 682, 2017.

[12] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, "Going deeper in spiking neural networks: Vgg and residual architectures," *Frontiers in neuroscience*, vol. 13, p. 95, 2019.

[13] F. Ponulak and A. Kasiński, "Supervised learning in spiking neural networks with resume: Sequence learning, classification, and spike shifting," *Neural computation*, vol. 22, no. 2, pp. 467–510, 2010.

[14] E. Hunsberger and C. Eliasmith, "Spiking deep networks with lif neurons," *arXiv preprint arXiv:1510.08829*, 2015.

[15] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[16] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019.

[17] G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass, "Long short-term memory and learning-to-learn in networks of spiking neurons," *Advances in neural information processing systems*, vol. 31, 2018.

[18] F. Zenke, W. Gerstner, and S. Ganguli, "The temporal paradox of hebbian learning and homeostatic plasticity," *Current opinion in neurobiology*, vol. 43, pp. 166–176, 2017.

[19] N. Caporale and Y. Dan, "Spike timing–dependent plasticity: A hebbian learning rule," *Annu. Rev. Neurosci.*, vol. 31, no. 1, pp. 25–46, 2008.

[20] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, "Deep learning in spiking neural networks," *Neural networks*, vol. 111, pp. 47–63, 2019.

[21] A. Belatreche, L. Maguire, M. McGinnity, and Q. Wu, "A method for supervised training of spiking neural networks," *Cybernetic Intelligence, Challenges and Advances*, p. 11, 2003.

[22] H. Mostafa, "Supervised learning based on temporal coding in spiking neural networks," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 7, pp. 3227–3235, 2017.

[23] A. Mohemmed, S. Schliebs, S. Matsuda, and N. Kasabov, "Training spiking neural networks to associate spatio-temporal input–output spike patterns," *Neurocomputing*, vol. 107, pp. 3–10, 2013.

[24] A. Kasiński and F. Ponulak, "Comparison of supervised learning methods for spike time coding in spiking neural networks," *International journal of applied mathematics and computer science*, vol. 16, no. 1, pp. 101–113, 2006.

[25] D. Huh and T. J. Sejnowski, "Gradient descent for spiking neural networks," *Advances in neural information processing systems*, vol. 31, 2018.

[26] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neurons: Opportunities and challenges," *Frontiers in neuroscience*, vol. 12, p. 409 662, 2018.

[27] K. Roy, A. Jaiswal, and P. Panda, "Towards spike-based machine intelligence with neuromorphic computing," *Nature*, vol. 575, no. 7784, pp. 607–617, 2019.

[28] T. Syed, V. Kakani, X. Cui, and H. Kim, "Exploring optimized spiking neural network architectures for classification tasks on embedded platforms," *Sensors*, vol. 21, no. 9, p. 3240, 2021.

[29] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[30] S. Ioffe, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[31] Y. Kim and P. Panda, "Revisiting batch normalization for training low-latency deep spiking neural networks from scratch," *Frontiers in neuroscience*, vol. 15, p. 773 954, 2021.

[32] Y. Cao, Y. Chen, and D. Khosla, "Spiking deep convolutional neural networks for energy-efficient object recognition," *International Journal of Computer Vision*, vol. 113, pp. 54–66, 2015.

[33] Y. Li, S. Deng, X. Dong, R. Gong, and S. Gu, "A free lunch from ann: Towards efficient, accurate spiking neural networks calibration," in *International conference on machine learning*, PMLR, 2021, pp. 6316–6325.

[34] S. Deng and S. Gu, "Optimal conversion of conventional artificial neural networks to spiking neural networks," *arXiv preprint arXiv:2103.00476*, 2021.

[35] B. Han, G. Srinivasan, and K. Roy, "Rmp-snn: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 13 558–13 567.

[36] Y. Hu, H. Tang, and G. Pan, "Spiking deep residual networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 8, pp. 5200–5205, 2021.

[37] W. Fang, Z. Yu, Y. Chen, T. Huang, T. Masquelier, and Y. Tian, *Deep residual learning in spiking neural networks*, 2021.

[38] S. Kim, S. Park, B. Na, and S. Yoon, "Spiking-yolo: Spiking neural network for energy-efficient object detection," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, 2020, pp. 11 270–11 277.

[39] Y. Kim, J. Chough, and P. Panda, "Beyond classification: Directly training spiking neural networks for semantic segmentation," *Neuromorphic Computing and Engineering*, vol. 2, no. 4, p. 044 015, 2022.

[40] Q. Su, Y. Chou, Y. Hu, *et al.*, "Deep directly-trained spiking neural networks for object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 6555–6565.

[41] M. Yao, J. Hu, Z. Zhou, *et al.*, "Spike-driven transformer," *Advances in neural information processing systems*, vol. 36, 2024.

[42] M. Yao, J. Hu, T. Hu, *et al.*, "Spike-driven transformer v2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips," *arXiv preprint arXiv:2404.03663*, 2024.

[43] S. Zhou, Y. Chen, X. Li, and A. Sanyal, "Deep scnn-based real-time object detection for self-driving vehicles using lidar temporal data," *IEEE Access*, vol. 8, pp. 76 903–76 912, 2020.

[44] S. Zhou and W. Wang, "Object detection based on lidar temporal pulses using spiking neural networks," *arXiv preprint arXiv:1810.12436*, 2018.

[45] S. Zhou, X. Li, Y. Chen, S. T. Chandrasekaran, and A. Sanyal, "Temporal-coded deep spiking neural network with easy training and robust performance," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, 2021, pp. 11 143–11 151.

[46] D. Ren, Z. Ma, Y. Chen, *et al.*, "Spiking pointnet: Spiking neural networks for point clouds," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[47] B. Vogginger, F. Kreutz, J. López-Randulfe, *et al.*, "Automotive radar processing with spiking neural networks: Concepts and challenges," *Frontiers in neuroscience*, vol. 16, p. 851 774, 2022.

[48] J. López-Randulfe, T. Duswald, Z. Bing, and A. Knoll, "Spiking neural network for fourier transform and object detection for automotive radar," *Frontiers in Neurorobotics*, vol. 15, p. 688 344, 2021.

[49] J. López-Randulfe, N. Reeb, N. Karimi, *et al.*, "Time-coded spiking fourier transform in neuromorphic hardware," *IEEE Transactions on Computers*, vol. 71, no. 11, pp. 2792–2802, 2022.

[50] Y. Wu, L. Deng, G. Li, J. Zhu, Y. Xie, and L. Shi, "Direct training for spiking neural networks: Faster, larger, better," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, 2019, pp. 1311–1318.

[51] W. Fang, Z. Yu, Y. Chen, T. Masquelier, T. Huang, and Y. Tian, "Incorporating learnable membrane time constant to enhance learning of spiking neural networks," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 2661–2671.

[52] P. De Tournemire, D. Nitti, E. Perot, D. Migliore, and A. Sironi, "A large scale event-based detection dataset for automotive," *arXiv preprint arXiv:2001.08499*, 2020.

[53] E. Perot, P. De Tournemire, D. Nitti, J. Masci, and A. Sironi, "Learning to detect objects with a 1 megapixel event camera," *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 639–16 652, 2020.

[54] J. Von Neumann, "First draft of a report on the edvac," *IEEE Annals of the History of Computing*, vol. 15, no. 4, pp. 27–75, 1993.

[55] C. Mead, "Neuromorphic electronic systems," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1629–1636, 1990.

[56] G. Indiveri, B. Linares-Barranco, T. J. Hamilton, *et al.*, "Neuromorphic silicon neuron circuits," *Frontiers in neuroscience*, vol. 5, p. 73, 2011.

[57] S.-C. Liu, B. Rueckauer, E. Ceolini, A. Huber, and T. Delbruck, "Event-driven sensing for efficient perception: Vision and audition algorithms," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 29–37, 2019.

[58] S. Furber, "Large-scale neuromorphic computing systems," *Journal of neural engineering*, vol. 13, no. 5, p. 051 001, 2016.

[59] E. Neftci, S. Das, B. Pedroni, K. Kreutz-Delgado, and G. Cauwenberghs, "Event-driven contrastive divergence for spiking neuromorphic systems," *Frontiers in neuroscience*, vol. 7, p. 272, 2014.

[60] B. Rajendran, A. Sebastian, M. Schmuker, N. Srinivasa, and E. Eleftheriou, "Low-power neuromorphic hardware for signal processing applications: A review of architectural and system-level design approaches," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 97–110, 2019. DOI: 10.1109/MSP.2019.2933719.

[61] E. A. Vittoz, "The design of high-performance analog circuits on digital cmos chips," *IEEE Journal of Solid-State Circuits*, vol. 20, no. 3, pp. 657–665, 1985.

[62] M. Prezioso, F. Merrikh-Bayat, B. D. Hoskins, G. C. Adam, K. K. Likharev, and D. B. Strukov, "Training and operation of an integrated neuromorphic network based on metal-oxide memristors," *Nature*, vol. 521, no. 7550, pp. 61–64, 2015.

[63] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[64] M. Davies, N. Srinivasa, T.-H. Lin, *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *Ieee Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[65] M. Davies, A. Wild, G. Orchard, *et al.*, "Advancing neuromorphic computing with loihi: A survey of results and outlook," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 911–934, 2021.

[66] N. Qiao, H. Mostafa, F. Corradi, *et al.*, "A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128k synapses," *Frontiers in neuroscience*, vol. 9, p. 141, 2015.

[67] J. Pei, L. Deng, S. Song, *et al.*, "Towards artificial general intelligence with hybrid tianjic chip architecture," *Nature*, vol. 572, no. 7767, pp. 106–111, 2019.

[68] *Fahrdynamik-Regelung*, de. Vieweg, 2006. DOI: 10.1007/978-3-8348-9049-8. [Online]. Available: http://dx.doi.org/10.1007/978-3-8348-9049-8.

[69] *Elektronische stabilitäts-programm, bosch*, de, Accessed: (06.01.2025), Jan. 2025. [Online]. Available: https://www.bosch-mobility.com/de/loesungen/fahrsicherheit/elektronisches-stabilitaets-programm/.

[70] A. Wong, M. Famuori, M. J. Shafiee, F. Li, B. Chwyl, and J. Chung, "Yolo nano: A highly compact you only look once convolutional neural network for object detection," in *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS Edition (EMC2-NIPS)*, 2019, pp. 22–25. DOI: 10.1109/EMC2-NIPS53020.2019.00013.

[71] M. Pfeiffer and T. Pfeil, "Deep learning with spiking neurons: Opportunities and challenges," *Frontiers in neuroscience*, vol. 12, p. 409 662, 2018.

[72] J. E. Pedersen, S. Abreu, M. Jobst, *et al.*, "Neuromorphic intermediate representation: A unified instruction set for interoperable brain-inspired computing," *Nature Communications*, vol. 15, no. 1, p. 8122, 2024.

[73] *Lava software framework*, version 0.10.0, Accessed: (18.09.2024), Aug. 2024. [Online]. Available: https://lava-nc.org.

[74] C. Pehle and J. E. Pedersen, *Norse - A deep learning library for spiking neural networks*, version 0.0.7, Documentation: https://norse.ai/docs/, Jan. 2021. DOI: 10.5281/zenodo.4422025. [Online]. Available: https://doi.org/10.5281/zenodo.4422025.

[75] J. Yik, K. V. den Berghe, D. den Blanken, *et al.*, "Neurobench: A framework for benchmarking neuromorphic computing algorithms and systems," 2025. arXiv: 2304.04640 [cs.AI]. [Online]. Available: https://arxiv.org/abs/2304.04640.

[76] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, pp. 303–338, 2010.

[77] K. Chen, J. Wang, J. Pang, *et al.*, "Mmdetection: Open mmlab detection toolbox and benchmark," *arXiv preprint arXiv:1906.07155*, 2019.

[78] *Hugging face inc.* Accessed: (19.09.2024), Aug. 2024. [Online]. Available: https://huggingface.co/.