SPIKE-DRIVEN VISION–LANGUAGE–ACTION MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision–language–action (VLA) models bridge multimodal understanding and robotic control, advancing the dominant paradigm for embodied intelligence. However, most existing models rely on large Transformers, whose latency and energy costs hinder deployment on resource-constrained platforms. Through sparse event-driven computation, spiking neural networks offer a promising paradigm for high-performance and energy-efficient computing. Here, we propose the first Spike-driven VLA framework enabling end-to-end direct training for robotic manipulation, which mainly comprises three core components. First, we develop spiking visual and instruction encoders for multimodal perception, encoding visual observations and language instructions into sparse, reliable spike representations for subsequent cross-modal fusion. Then, we introduce Multi-Winner Spike Fusion for instruction-guided scene understanding, using bidirectional top-k winner-take-all spike routing to suppress background interference and yield fused memory. Finally, we propose a Spike Action Chunking Transformer that incorporates spiking cross-attention over the fused memory and the current robot state, enabling efficient end-to-end generation of continuous action chunks for robotic control. Extensive experiments on LIBERO and Meta-World demonstrate that Spike-driven VLA achieves competitive performance with fewer parameters and lower estimated inference energy than conventional VLA models. This work establishes a foundational framework for neuromorphic VLA modeling, paving the way for future advances in resource-efficient embodied intelligence.

1 INTRODUCTION

Vision–language–action (VLA) models (Brohan et al., 2023; Kim et al., 2025) enable robots to perceive their surroundings, interpret natural-language instructions, and generate corresponding actions within a unified policy, providing a promising foundation for general-purpose robotic control (Black et al., 2025). However, many existing VLAs build on large pretrained vision–language backbones for instruction understanding and multimodal reasoning (Brohan et al., 2023; Kim et al., 2025; Black et al., 2025). Their large model scale and dense computation impose substantial memory and latency overheads, hindering deployment on resource-constrained robotic platforms (Wen et al., 2024; Shukor et al., 2025; Budzianowski et al., 2025). To this end, lightweight VLAs (Wen et al., 2024; Xie et al., 2026; Shukor et al., 2025) improve efficiency through compact architectures and streamlined action decoding. However, they still rely on dense multiply–accumulate (MAC) (You et al., 2020) operations, making repeated policy inference computationally costly during closed-loop control. Therefore, developing compact and efficient VLAs for edge deployment remains an important research direction.

Spiking Neural Networks (SNNs) (Maass, 1997; Gerstner & Kistler, 2002) offer a promising alternative for energy-efficient intelligence through brain-inspired dynamics and spike-driven computation (Chen et al., 2024). Unlike Artificial Neural Networks (ANNs) (He et al., 2016), which primarily rely on dense MAC operations (You et al., 2020), spiking neurons Izhikevich (2003) trigger synaptic computation only when spikes arrive, replacing multiplications with sparse accumulate (AC) operations (Bouvier et al., 2019; Roy et al., 2019). An AC operation typically consumes only 1/5 to 1/10 of the energy required by a MAC operation on a representative 45-nm ASIC (You et al., 2020), while dedicated hardware platforms can further exploit sparse spike activity (Deng et al., 2020). These properties make SNNs a promising computational foundation for efficient VLA deployment on resource-constrained edge platforms.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Existing spiking VLAs Song et al. (2026) primarily rely on ANN-to-SNN conversion Li et al. (2021); You et al. (2024), transferring the representations learned by pretrained ANNs to their spiking counterparts. However, ANN-to-SNN methods (Chen et al., 2026) typically retain the source ANN architecture, restricting task-specific architectural design. They also approximate continuous activations by accumulating spikes (Wang et al., 2026) over multiple timesteps, resulting in substantial temporal overhead Wang et al. (2026). In contrast, direct training methods Wu et al. (2018); Zhou et al. (2023) enable task-specific spiking architectures and end-to-end learning of task-relevant representations in the spike domain. Recent advances in directly trained SNNs span perception tasks, including classification (Cai et al., 2024), detection (Yao et al., 2025), and tracking (Wang et al., 2024; Shan et al., 2025), as well as language modeling (Zhu et al., 2023) and robotic control (Zhang et al., 2025a). Despite these advances, extending directly trained SNNs to end-to-end VLA modeling for robotic manipulation remains challenging.

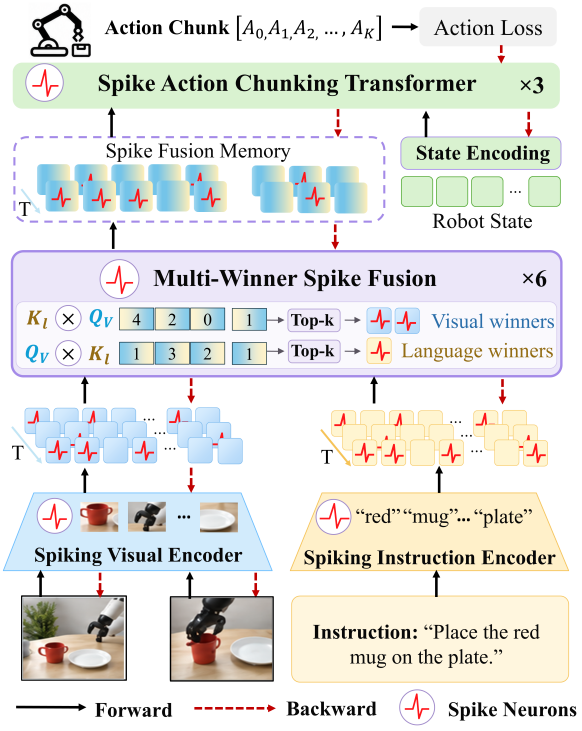

Figure 1: Overview of Spike-driven VLA.

To break the ice, we propose the first Spike-driven VLA framework that supports end-to-end direct training for robotic manipulation. As shown in Fig. 1, Spike-driven VLA comprises three stages: multi-modal spike encoding, cross-modal fusion, and state-conditioned action decoding. First, we propose a spiking vision encoder (SVE) and a spiking instruction encoder (SIE) to capture visual context and instruction semantics in modality-specific spike representations. Then, we propose a Multi-Winner Spike Fusion (MWSF) module that establishes task-relevant cross-modal correspondence, yielding action-ready fusion memory. Finally, based on this memory and the current robot state, we propose Spike Action Chunking Transformer (SpikeACT) to generate continuous action chunks for instruction-guided manipulation. We evaluate Spike-driven VLA on the LIBERO (Liu et al., 2023) and Meta-World (Yu et al., 2020) benchmarks, which span diverse language-conditioned manipulation tasks and simulated environments. Extensive experiments demonstrate that Spike-driven VLA achieves competitive performance against representative ANN-based methods while requiring fewer parameters and computational operations. Further evaluation on LIBERO-Plus (Fei et al., 2026) demonstrates its strong robustness under diverse distribution shifts. These results demonstrate that our Spike-driven VLA offers a new solution for efficient VLA deployment on resource-constrained edge platforms. The main contributions are as follows:

- We develop an SVE and a BERT-based SIE for multimodal perception. These encoders capture hierarchical visual features and contextual instruction semantics, respectively, and preserve them in sparse spike representations.
- We propose MWSF to fuse visual and instruction features into a memory for scene understanding. Its bidirectional top-k winner-take-all spike routing suppresses background interference and reduces cross-modal computation through sparse token aggregation.
- We develop the SpikeACT module for multimodal decision-making and continuous control. It combines fused memory with the current robot state to jointly predict action chunks, enabling efficient action generation through sparse spike-driven computation.
- Building on these components, we propose a Spike-driven VLA framework that supports end-to-end direct training for robotic manipulation. Experiments on LIBERO and Meta-World demonstrate competitive performance with fewer parameters and operations, offering a new path toward efficient VLA deployment on resource-constrained edge platforms.

2 Related Work

2.1 Spiking Neural Networks

Benefiting from sparse event-driven computation, SNNs offer a promising paradigm for energy-efficient computing on resource-constrained edge devices. However, early SNNs suffer from limited performance due to immature training methods and shallow architectures. Advances in spatio-temporal backpropagation (Fang et al., 2021) have substantially facilitated the training of high-performance SNNs. Building on these advances, Spikformer (Zhou et al., 2023) established the spiking self-attention paradigm, while SDT-V1/V2 (Yao et al., 2023; 2024) reformulated self-attention with the Hadamard product. Subsequent studies have extended Spiking Transformers to diverse vision tasks, such as classification (Zheng et al., 2021), detection (Yao et al., 2025), and tracking (Shan et al., 2025), demonstrating their potential as general spiking backbones. Beyond vision, SNNs have also made substantial progress in language modeling. SpikeGPT (Zhu et al., 2023) demonstrated generative language modeling with directly trained SNNs. SpikeBERT (Liang et al., 2026; Zhou et al., 2026) advanced language understanding through knowledge distillation and direct training, respectively. Collectively, advances in spiking perception and language modeling, together with recent progress in spiking robotic control (Zhang et al., 2025a), enable the end-to-end direct training of a fully Spike-driven VLA framework.

2.2 Vision–Language–Action Models

VLA models have emerged as an important paradigm for embodied intelligence. RT-2 (Brohan et al., 2023) transfers knowledge acquired through vision-language pretraining to robotic control, while PaLM-E (Driess et al., 2023) extends multimodal language modeling to embodied decision-making. Built upon the Open X-Embodiment dataset, RT-X (O'Neill et al., 2024) further advances cross-robot policy learning. More recent models, including Octo (Ghosh et al., 2024), OpenVLA (Kim et al., 2025), and π_0 (Black et al., 2025), improve task generalization by scaling robot data and multimodal pretraining. Despite their strong performance, these models rely heavily on large pre-trained backbones and computationally intensive action-generation pipelines, imposing substantial memory, computational, and latency overhead for deployment on resource-constrained robotic platforms. Existing efforts improve VLA efficiency through action chunking and lightweight model design. Action chunking predicts multiple future actions in a single inference pass, spreading the computational cost across successive control steps. Meanwhile, TinyVLA (Wen et al., 2024), TurboVLA (Xie et al., 2026), and SmolVLA (Shukor et al., 2025) reduce inference costs through compact architectures and efficient backbones. However, these methods improve efficiency primarily through architectural compression, while leaving the underlying dense computation paradigm unchanged. Therefore, how to make full use of the event-driven nature of SNNs offers a promising path toward efficient VLA models.

3 Preliminaries

3.1 Spiking Neuron Model

Spiking neurons replicate the behavior of biological neurons that integrate input spikes into their membrane potential u and fire a spike only when u exceeds a threshold. For layer l at timestep t, the synaptic input is given by $I^l[t] = W^l s^{l-1}[t]$, where W^l denotes the synaptic weight matrix and $s^{l-1}[t] \in \{0, 1\}$ represents the binary spike activity from the preceding layer. Since only active presynaptic neurons contribute to $I^l[t]$, synaptic computation can be implemented through sparse accumulation of the corresponding weights. We adopt the discrete Leaky Integrate-and-Fire (LIF) model (Maass, 1997; Gerstner & Kistler, 2002) denoted by $(\mathcal{SN}(\cdot))$, formulated as:

$$\tilde{u}^l[t] = \lambda u^l[t-1] + I^l[t], \quad s^l[t] = H\big(\tilde{u}^l[t] - V_{\text{th}}\big), \quad u^l[t] = \tilde{u}^l[t] - V_{\text{th}} s^l[t], \tag{1}$$

where $\tilde{u}^l[t]$ and $u^l[t]$ denote the pre- and post-reset membrane potentials, respectively; $H(\cdot)$ is the Heaviside step function; V_{th} is the firing threshold; and $\lambda \in [0, 1]$ controls membrane leakage. Since $H(\cdot)$ is non-differentiable at the threshold and has zero derivatives elsewhere, we approximate its derivative with a surrogate gradient during backpropagation to enable end-to-end training. Common choices Wu et al. (2018) include derivatives of sigmoid, fast sigmoid, and arctangent functions.

3.2 SPIKE SELF-ATTENTION

Building on Vanilla Self-attention (VSA) Vaswani et al. (2017) in ANNs, Spikformer (Zhou et al., 2023) introduced Spiking Self-attention (SSA). For an input matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$, queries $\mathbf{Q}$, keys $\mathbf{K}$, and values $\mathbf{V}$ are first computed via learnable weight matrices and then converted into spike trains by binary spiking neurons for subsequent processing. The $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ in SSA can be described as:

$$\mathbf{Q} = \mathcal{SN}(\mathrm{BN}(\mathrm{Linear}(\mathbf{X}))), \quad \mathbf{K} = \mathcal{SN}(\mathrm{BN}(\mathrm{Linear}(\mathbf{X}))), \quad \mathbf{V} = \mathcal{SN}(\mathrm{BN}(\mathrm{Linear}(\mathbf{X}))), \quad (2)$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{T\times n\times d}$, $\mathrm{BN}(\cdot)$ denotes batch normalization and $\mathrm{Linear}(\cdot)$ refers to a linear operation. Unlike VSA, SSA omits the Softmax operation while retaining scaling to control the magnitude of the attention output. The attention output $\mathbf{Attn}$ is obtained by performing a matrix multiplication of the spiking $\mathbf{Q}$ and $\mathbf{K}$, scaled by factor s, and then converted into spike trains:

$$\mathbf{Score} = s \cdot \mathbf{Q} \times \mathbf{K^T}, \quad \mathbf{Attn} = \mathcal{SN}(\mathbf{Score} \times \mathbf{V}). \quad (3)$$

Therefore, SSA provides an energy-efficient self-attention computation paradigm without softmax normalization. It allows the ordering of the Q, K, and V matrices to be flexibly adjusted as needed, enabling the Spiking Transformer to capture global dependencies efficiently.

4 SPIKE-DRIVEN VISION–LANGUAGE–ACTION MODELS

4.1 OVERALL ARCHITECTURE

Spike-driven VLA consists of three functional stages: (i) spiking visual and instruction encoders (SVE and SIE) for environment perception, detailed in Section 4.2; (ii) a Multi-Winner Spike Fusion module for instruction-guided visual-language interaction, detailed in Section 4.3; and (iii) a SpikeACT module for chunked control generation, detailed in Section 4.4. At control step n, the current visual observation $\mathcal{O}_n$ and language instruction $\mathcal{L}$ are encoded and integrated through the MWSF module:

$$\mathbf{Z}_{v,n} = E_v(\mathcal{O}_n), \quad \mathbf{Z}_\ell = E_\ell(\mathrm{Tok}(\mathcal{L})), \quad \mathcal{M}_n = \mathcal{F}(\mathbf{Z}_{v,n}, \mathbf{Z}_\ell). \quad (4)$$

Here, E_v and E_ℓ denote SVE and SIE, respectively. They produce spike representations $\mathbf{Z}_{v,n} \in \mathbb{R}^{N_v\times d}$ and $\mathbf{Z}_\ell \in \mathbb{R}^{N_\ell\times d}$, where N_v and N_ℓ are the visual and instruction token counts, and d is their shared feature dimension. $\mathrm{Tok}(\mathcal{L})$ contains N_ℓ token indices. $\mathcal{F}$ performs competitive visual-language interaction and produces the task-conditioned memory $\mathcal{M}_n \in \mathbb{R}^{(N_v+N_\ell)\times d}$. The memory retains continuous membrane-potential residuals. The robot state $\mathcal{S}_n \in \mathbb{R}^{d_s}$ is then encoded and jointly processed with the fused memory:

$$\mathbf{Z}_{s,n} = E_s(\mathcal{S}_n), \quad \widehat{\mathcal{A}}_n = \mathrm{SpikeACT}\left(\mathbf{P}_a, [\mathcal{M}_n; \mathbf{Z}_{s,n}]\right) \in \mathbb{R}^{K\times d_a}. \quad (5)$$

Specifically, the State Encoder E_s maps the current robot state into N_s state tokens $\mathbf{Z}_{s,n} \in \mathbb{R}^{N_s\times d}$, which are concatenated with the fused memory. Conditioned on $[\mathcal{M}_n; \mathbf{Z}_{s,n}]$, SpikeACT updates K learnable action tokens $\mathbf{P}_a \in \mathbb{R}^{K\times d}$ through spiking cross-attention.The K decoded action tokens are aggregated across timesteps and projected to produce a continuous action chunk, $\widehat{\mathcal{A}}_n = [\widehat{\mathbf{a}}_n, \ldots, \widehat{\mathbf{a}}_{n+K-1}]$, where each $\widehat{\mathbf{a}}_{n+k} \in \mathbb{R}^{d_a}$. The individual components are detailed below.

4.2 SPIKE-DRIVEN VISUAL AND INSTRUCTION ENCODERS

The perception stage of Spike-driven VLA comprises SVE and SIE, which transform visual observations and language instructions into spike token representations. As shown in Fig. 2(a), SVE encodes the main and wrist observations $\mathcal{O}_n^{\mathrm{Main}}$ and $\mathcal{O}_n^{\mathrm{wrist}}$, which can be described as:

$$\mathbf{Z}_{v,n} = E_v(\mathcal{O}_n^{\mathrm{Main}}, \mathcal{O}_n^{\mathrm{wrist}}) = \mathrm{BN}(\mathrm{Linear}_v\big(\big[\mathbf{Z}^{\mathrm{main}}; \mathbf{Z}^{\mathrm{wrist}}\big]\big)), \quad \mathbf{Z}_{v,n} \in \mathbb{R}^{T\times B\times N_v\times d}. \quad (6)$$

where $\mathbf{Z}^{\mathrm{main}}$ and $\mathbf{Z}^{\mathrm{wrist}}$ denote dual-view features from a shared spiking visual transformer (Yao et al., 2025; Zhou et al., 2024; Wang et al., 2025a). The backbone is pretrained on ImageNet (Deng et al., 2009) and subsequently fine-tuned during Spike-driven VLA training. The first two stages of the backbone contain one and two Conv Mixer blocks, respectively, which extract local features through convolution and downsampling. The final stage uses SSA modules. These updates enrich

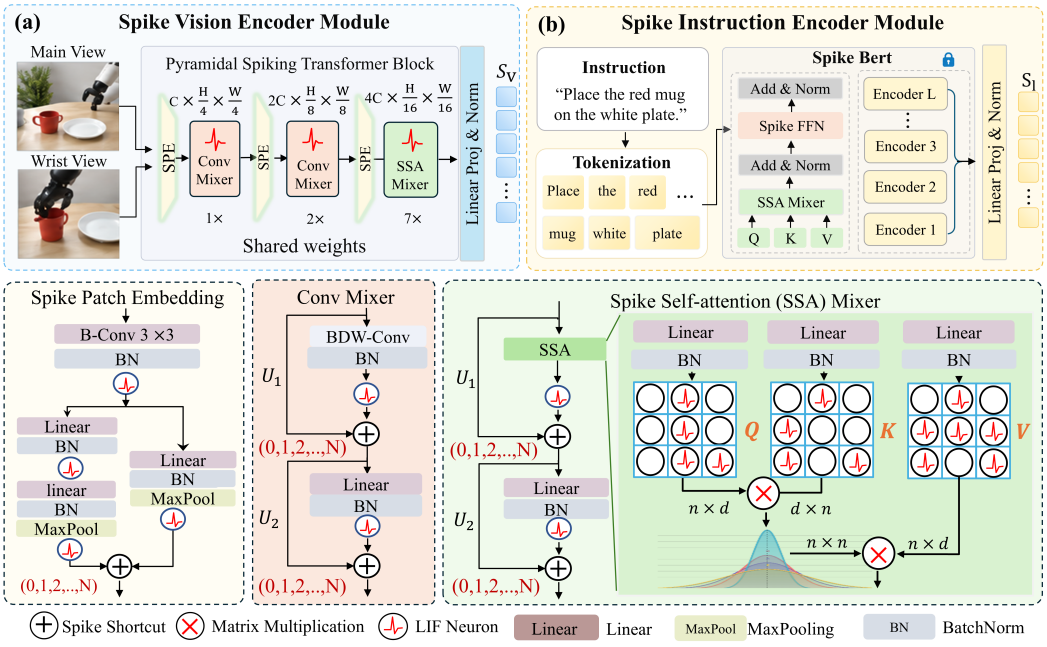

Figure 2: (a) SVE encodes main- and wrist-view observations into visual spike tokens capturing appearance and spatial relations; (b) SIE encodes language instructions into contextual semantic spike tokens that condition visual perception.

local visual features with global context, producing visual tokens that encode object appearance and spatial relationships for subsequent instruction-guided fusion. Detailed architectural configurations are provided in Appendix A. As shown in Fig. 2(b), SIE encodes the instruction to provide semantic guidance for task-relevant visual perception, which can be described as:

$$\mathbf{Z}_\ell = E_\ell\left(\mathrm{Tok}(\mathcal{L})\right) = \mathrm{BN}\left(\mathrm{Linear}_\ell\left(\mathbf{Z}^{\mathrm{text}}\right)\right), \quad \mathbf{Z}_\ell \in \mathbb{R}^{T\times B\times N_\ell\times d}. \tag{7}$$

where $\mathbf{Z}^{\mathrm{text}}$ denotes contextual language features extracted by a spiking BERT (Liang et al., 2026; Zhou et al., 2026) backbone. The backbone is pretrained with masked language modeling (Zhou et al., 2026) and remains frozen during Spike-driven VLA training, providing fixed semantic features for each instruction. Detailed architectural configurations of Spiking BERT are provided in Appendix B. SVE and SIE reduce intermediate activation memory while preserving control-relevant information, providing informative representations for subsequent scene understanding.

4.3 Multi-Winner Spike Fusion

Although SVE and SIE provide effective visual and language representations, precise robotic manipulation requires language instructions to direct visual perception toward task-relevant evidence. In robotic manipulation tasks, visual tokens typically far outnumber language tokens and contain substantial background clutter. Directly applying SSA (Zhou et al., 2023) to global cross-modal interaction allows every visual token with a nonzero affinity to contribute to feature aggregation, causing weak background responses to accumulate and dilute salient evidence. To address this issue, we introduce the MWSF module, which combines a shared cross-modal affinity matrix with bidirectional competitive routing to establish sparse, task-relevant visual-language correspondences. As shown in Fig. 3(a), given $\mathbf{Z}_{v,n}$ and $\mathbf{Z}_\ell$, modality-specific spiking projections generate the features used for cross-modal matching and value aggregation:

$$\begin{aligned}\mathbf{Q}_v &= \mathcal{SN}(\mathrm{BN}(\mathrm{Linear}(\mathcal{SN}(\mathbf{Z}_{v,n})))), \quad \mathbf{V}_v = \mathcal{SN}(\mathrm{BN}(\mathrm{Linear}(\mathcal{SN}(\mathbf{Z}_{v,n})))),\\ \mathbf{K}_\ell &= \mathcal{SN}(\mathrm{BN}(\mathrm{Linear}(\mathcal{SN}(\mathbf{Z}_\ell)))), \quad \mathbf{V}_\ell = \mathcal{SN}(\mathrm{BN}(\mathrm{Linear}(\mathcal{SN}(\mathbf{Z}_\ell)))).\end{aligned} \tag{8}$$

Here, $\mathbf{Q}_v, \mathbf{V}_v \in \mathbb{S}^{T\times B\times N_v\times d}$ and $\mathbf{K}_\ell, \mathbf{V}_\ell \in \mathbb{S}^{T\times B\times N_\ell\times d}$, $\mathbb{S} = \{0,1\}$ denotes the binary spike domain. Each Linear and BN is independently parameterized. The $\mathbf{Q}_v$ and $\mathbf{K}_\ell$ form a shared cross-modal affinity matrix $\mathbf{R}_n = \mathbf{Q}_v\mathbf{K}_\ell^\top \in \mathbb{Z}^{T\times B\times N_v\times N_\ell}$. The visual and language branches perform

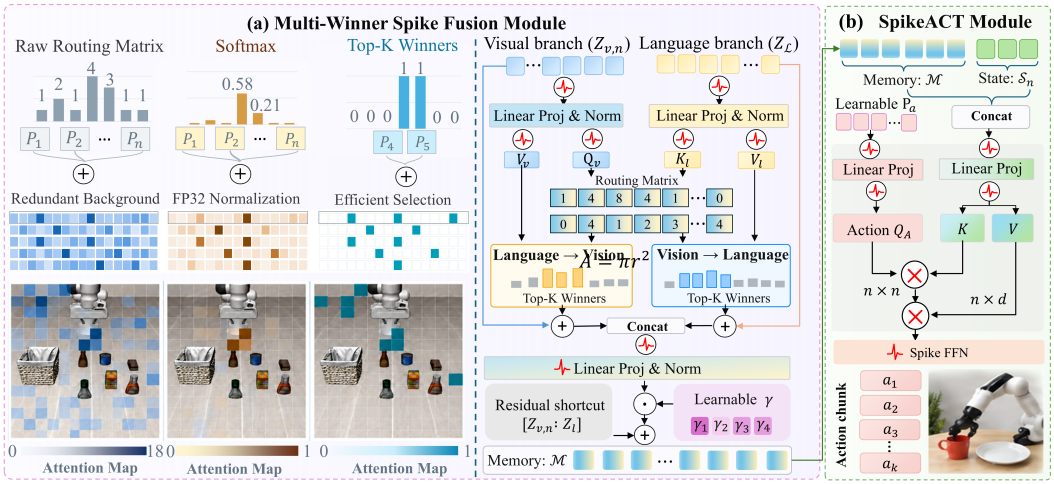

Figure 3: (a) MWSF uses bidirectional Top-k WTA routing to retain instruction-relevant visual evidence, suppress background interference, and reduce computation. (b) SpikeACT uses learnable action tokens to predict continuous action chunks in parallel from fused memory and the robot state.

Top-k multi-winner selection (WTA_k) over $\mathbf{R}_n$ and $\mathbf{R}_n^\top$. The WTA_k operator acts row-wise on $\mathbf{R} \in \mathbb{Z}^{T\times B\times N_q\times N_k}$, where N_q and N_k are the numbers of $\mathbf{Q}_v$ and $\mathbf{K}_\ell$ tokens. For any row $\mathbf{r}$ and its output $\mathbf{a}^{(k)}$, the j-th mask entry is defined as:

$$a_j^{(k)} = \begin{cases} 1, & j \in \mathrm{TopK}_k(\mathbf{r}) \text{ and } r_j > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

Here, $\mathrm{TopK}_k(\mathbf{r})$ returns the indices of up to k highest-scoring valid keys. As shown in Fig.3(a), WTA concentrates routing on instruction-relevant visual regions, suppressing background responses present in raw routing without requiring softmax normalization. The resulting cross-modal contexts are computed as follows:

$$\mathbf{C}_v = \mathbf{Z}_{v,n} + \mathrm{BN}(\mathrm{Linear}(\mathcal{SN}(\mathrm{WTA}_{k_v}(\mathbf{R}_n)\mathbf{V}_\ell))), \quad \mathbf{C}_v \in \mathbb{R}^{T\times B\times N_v\times d}. \tag{10}$$

$$\mathbf{C}_\ell = \mathbf{Z}_\ell + \mathrm{BN}\big(\mathrm{Linear}\big(\mathcal{SN}\big(\mathrm{WTA}_{k_\ell}(\mathbf{R}_n^\top)\mathbf{V}_v\big)\big)\big), \quad \mathbf{C}_\ell \in \mathbb{R}^{T\times B\times N_\ell\times d}. \tag{11}$$

where $\mathbf{C}_v$ denotes the instruction-guided visual representation, while $\mathbf{C}_\ell$ denotes the visually grounded instruction representation. The two branches employ k_v and k_ℓ as direction-specific routing budgets to accommodate the imbalance between visual and language sequence lengths. Finally, we concatenate $\mathbf{C}_v$ and $\mathbf{C}_\ell$ along the token dimension to obtain the scene memory $\mathcal{M}_n$:

$$\mathcal{M}_n = [\mathbf{C}_v; \mathbf{C}_\ell] + \boldsymbol{\gamma} \odot \mathrm{BN}(\mathrm{Linear}(\mathcal{SN}([\mathbf{C}_v; \mathbf{C}_\ell]))), \quad \mathcal{M}_n \in \mathbb{R}^{T\times B\times (N_v+N_\ell)\times d}. \tag{12}$$

The residual connection preserves the cross-modal contexts obtained through bidirectional routing and provides a direct pathway for information and gradient propagation across stacked MWSF blocks, preventing sparse routing updates from overwriting the original representations. The learnable channel-wise scale $\boldsymbol{\gamma} \in \mathbb{R}^{1\times d}$ adaptively controls the magnitude of the residual update. This scaling does not compromise spike-driven computation because the learnable projection operates on the spike representation produced by $\mathcal{SN}$. Moreover, $\boldsymbol{\gamma}$ can be folded into the affine scale and bias of $\mathrm{BN}(\cdot)$ during inference, introducing no additional non-spiking operation or runtime overhead. The resulting $\mathcal{M}_n \in \mathbb{R}^{T\times B\times (N_v+N_\ell)\times d}$ remains continuous and serves as the task-conditioned memory for subsequent spiking action decoding.

4.4 Spike Action Chunking Transformer

As described in the previous section, $\mathcal{M}_n$ encodes scene context aligned with the current instruction. Mapping this memory to executable continuous control commands further requires conditioning on the robot's current state. To this end, we develop SpikeACT module as shown in Fig.3(b), which employs both the scene memory and robot state and predicts a K-step action chunk in parallel.

Specifically, the state encoder produces tokens $\mathbf{Z}_{s,n} \in \mathbb{R}^{T\times B\times N_s\times d}$. These tokens are concatenated with $\mathcal{M}_n$ to form the state-augmented memory $\widetilde{\mathcal{M}}_n$:

$$\widetilde{\mathcal{M}}_n = [\mathcal{M}_n; \mathbf{Z}_{s,n}]_{\text{token}}, \quad \widetilde{\mathcal{M}}_n \in \mathbb{R}^{T\times B\times N_m\times d}, \tag{13}$$

where $N_m = N_v + N_\ell + N_s$. We use K learnable action $\mathbf{P}_a \in \mathbb{R}^{K\times d}$, each corresponding to one future position in the action chunk. Independent spiking projections then generate the query features from $\mathbf{P}_a$ and the key and value features from $\widetilde{\mathcal{M}}_n$:

$$\mathbf{Q}_a = \mathcal{SN}(\mathrm{BN}(\mathrm{Linear}(\mathcal{SN}(\mathbf{P}_a)))), \quad \mathbf{K}_{\widetilde{\mathcal{M}}_\mathbf{n}}, \mathbf{V}_{\widetilde{\mathcal{M}}_\mathbf{n}} = \mathcal{SN}\Big(\mathrm{BN}\Big(\mathrm{Linear}\Big(\mathcal{SN}(\widetilde{\mathcal{M}}_n)\Big)\Big)\Big), \tag{14}$$

where $\mathbf{Q}_a \in \mathbb{R}^{T\times B\times K\times d}$ and $\mathbf{K}, \mathbf{V} \in \mathbb{R}^{T\times B\times N_m\times d}$. Each occurrence of Linear and BN has independent parameters. The affinity matrix $\mathbf{R}_{a,n} = \mathbf{Q}_a\mathbf{K}^\top \in \mathbb{Z}^{T\times B\times K\times N_m}$ associates each future action position with the state-augmented memory. $\mathrm{WTA}_k(\mathbf{R}_{a,n})$ converts these affinities into a sparse binary routing matrix, allowing each action query to retrieve a position-specific subset of visual, language, and robot-state evidence. This retrieval is learned end-to-end through action supervision. The routed features are temporally read out to produce the continuous action chunk:

$$\widehat{\mathcal{A}}_n = \tanh(\mathrm{Avg}_T[\mathrm{Linear}_a(\mathcal{SN}(\mathrm{WTA}_k(\mathbf{R}_{a,n})\mathbf{V}))]) \in \mathbb{R}^{B\times K\times d_a}. \tag{15}$$

Here, Linear_a maps the routed spike features to the action dimension d_a, while Avg_T performs temporal readout over the spiking timesteps. The final $\tanh$ bounds the continuous control commands. Consequently, $\widehat{\mathcal{A}}_n = [\widehat{\mathbf{a}}_n, \ldots, \widehat{\mathbf{a}}_{n+K-1}]$ predicts all K future actions in parallel.

5 Experiments

5.1 Experiment Setup

Meta-World: Meta-World (Yu et al., 2020) comprises 50 diverse tabletop manipulation tasks performed by a Sawyer robot. We use the MT50 dataset released with SmolVLA (Shukor et al., 2025), containing 2,500 demonstrations with RGB observations, robot states, language instructions, and continuous actions. We jointly train a single policy on all tasks and evaluate it over 50 rollouts per task, reporting the average success rate based on task completion.

LIBERO: LIBERO (Liu et al., 2023) consists of four suites: Spatial, Object, Goal, and Long. Each suite contains ten tasks specified by language instructions. We use the modified no-op RLDS datasets released with OpenVLA (Kim et al., 2025) and jointly train a single policy on all four suites. The policy predicts continuous 7-DoF actions in chunks of 12. We train the model for 80K steps, including 10K warm-up steps, with an effective batch size of 256. Following VLA-Adapter (Wang et al., 2025b), we conduct 50 rollouts for each task and report success rates for each suite and across all 2,000 evaluation trials. To further assess zero-shot robustness under distribution shifts, we directly evaluate the same LIBERO-trained policy on LIBERO-Plus (Fei et al., 2026), without any additional training or adaptation. LIBERO-Plus contains 10,030 perturbed task instances spanning seven dimensions: object layout, camera viewpoint, robot initial state, language instruction, lighting, background texture, and sensor noise.

5.2 Main Results

As shown in Table 1, Spike-driven VLA achieves an average success rate of 72.4%, outperforming all compared ANN-based VLA models. It ranks first on Medium and Hard tasks, with success rates of 66.7% and 81.7%, respectively. Notably, it surpasses SmolVLA (2.25B) and SmolVLA (0.45B) (Shukor et al., 2025) by 4.2 and 15.1% while using $1/15$ and $1/3$ of their respective parameter counts. These results demonstrate that a compact spike-driven architecture can outperform larger ANN-based counterparts, with only 0.15B parameters, 10.5 GFLOPs, and an estimated inference energy of 11.6 mJ.

As shown in Table 2, Spike-driven VLA achieves an average success rate of 92.4% on LIBERO, with success rates above 95% or equal to it on Spatial, Object, and Goal tasks and 82.4% on Long tasks. It outperforms OpenVLA (Kim et al., 2025) and SmolVLA (Shukor et al., 2025) by 15.9 and 3.6 percentage points, respectively, while closely matching DreamVLA (92.6%) (Zhang et al., 2025b).

Table 1: Comparison on the Meta-World benchmark (Yu et al., 2020). Baseline success rates are taken from (Shukor et al., 2025). Our method is evaluated over 50 episodes per task.

Method	Spike-driven	Computation Efficiency			Meta-World Success Rate (%)				
		Params (B)↓	FLOPs (G)↓	Energy (mJ)↓	Easy	Medium	Hard	V. Hard	Avg.↑
TinyVLA (Wen et al., 2024)	×	0.42	1259.1	2896.0	77.6	21.5	11.4	15.8	31.6
π_0 (Black et al., 2025)	×	3.2	3069.8	7060.6	71.8	48.2	41.7	30.0	47.9
SmolVLA (0.45B) (Shukor et al., 2025)	×	0.45	598.4	1376.4	82.5	41.8	45.0	60.0	57.3
SmolVLA (2.25B) (Shukor et al., 2025)	×	2.25	4239.5	9751.1	87.1	51.8	70.0	64.0	68.2
Spike-driven VLA (Ours)	✓	0.15	10.5	11.6	85.1	66.7	81.7	56.0	72.4

Table 2: Comparison on LIBERO (Liu et al., 2023) benchmark. We report suite-level and overall average success rates based on 50 rollouts per task.

Method	Spike-driven	Computational Efficiency			LIBERO Success Rate (%)				
		Params (B)↓	FLOPs (G)↓	Energy (mJ)↓	Spa.	Obj.	Goal	Long	Avg.↑
OpenVLA (Kim et al., 2025)	×	7.5	4158.3	9564	84.7	88.4	79.2	53.7	76.5
π_0 (Black et al., 2025)	×	3.2	4371.9	10055	96.8	98.8	95.8	85.2	94.2
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	×	3.4	5014.2	11533	98.8	98.2	98.0	92.4	96.9
CogVLA (Li et al., 2025)	×	8.3	3299.2	7588	98.6	98.8	96.6	95.4	97.4
TurboVLA (Xie et al., 2026)	×	0.22	100.27	230.6	99.2	99.8	97.4	94.2	97.7
SmolVLA (Shukor et al., 2025)	×	2.3	521.1	1199	93.0	94.0	91.0	77.0	88.8
DreamVLA (Zhang et al., 2025b)	×	0.7	995.2	2289	97.5	94.0	89.5	89.5	92.6
Spike-driven VLA (Ours)	✓	0.15	15.68	20.9	95.0	95.4	96.6	82.4	92.4

Compared with π_0 (Black et al., 2025), it uses 95.3% fewer parameters and reduces computational cost. We further evaluate robustness across seven perturbation dimensions on LIBERO-Plus (Fei et al., 2026). As shown in Table 3, with only 0.15B parameters, Spike-driven VLA achieves an overall success rate of 53.2%, outperforming most ANN baselines and remaining competitive with substantially larger VLA models. This performance indicates that Spike-driven VLA retains competitive robustness under distribution shifts. Overall, these results demonstrate Spike-driven VLA as a promising path toward resource-efficient embodied intelligence on edge platforms.

5.3 Ablation Studies

In this section, we conduct ablation studies on the perception and fusion stages of Spike-driven VLA. For perception, we compare spiking encoders with larger, full-precision ANN counterparts to assess the trade-off between task performance and computational efficiency. For fusion, we examine the effect of WTA on manipulation performance when visual tokens outnumber language tokens. The experimental settings and results are presented below.

Table 4: Ablation study of WTA.

Suite	w/o WTA	w/ WTA	Δ
Spatial	90.2	**95.0**	+4.8
Object	**95.6**	95.4	−0.2
Goal	96.2	**96.6**	+0.4
Long	78.6	**82.4**	+3.8
Overall	90.2	**92.4**	+2.2

Perception: As shown in Table 5, with the fusion module and action decoder fixed, replacing BERT with SIE reduces the average success rate by only 0.2% (B0 vs. B1), whereas replacing DINOv3-B with SVE leads to a 3.7% decrease (B1 vs. B3). This larger visual gap may reflect DINOv3-B's greater model capacity and richer visual priors acquired through large-scale pretraining. These results suggest that visual perception accounts for most of the performance gap, while SIE achieves performance comparable to BERT and retains sufficient capacity to distinguish task instructions on LIBERO.

Table 3: Comparison on LIBERO-Plus (Fei et al., 2026) across seven perturbation dimensions.

Method	Spike-driven	Params (B)↓	LIBERO-Plus Success Rate (%)							
			Camera	Robot	Language	Light	Background	Noise	Layout	Avg.↑
OpenVLA (Kim et al., 2025)	×	7.5	1.1	4.1	26.8	4.4	25.3	19.3	31.6	16.1
NORA (Hung et al., 2025)	×	3.8	4.0	41.1	67.0	31.0	50.5	17.6	63.9	39.3
UniVLA (Bu et al., 2025)	×	7.0	4.3	50.3	71.8	59.1	80.0	25.3	34.3	46.4
π_0 (Black et al., 2025)	×	3.2	15.8	6.6	61.0	79.6	78.5	79.4	70.4	55.9
Spike-driven VLA (Ours)	✓	0.15	41.2	46.7	48.7	84.2	61.3	34.5	67.5	54.9

Table 5: Ablation of visual and language encoders on LIBERO. VE and LE denote the visual and language encoders, respectively. DINOv3-B (Siméoni et al., 2025) and BERT (Devlin et al., 2019) serve as the corresponding ANN baselines.

Variant	VE	LE	Params (M)	Energy (mJ)	LIBERO Success Rate (%)				
					Spatial	Object	Goal	Long	Avg.
B0	DINOv3-B	BERT	216.1	179.7	98.6	99.2	93.2	94.0	96.3
B1	DINOv3-B	SIE	216.1	173.0	98.0	98.6	95.4	92.4	96.1
B2	SVE	BERT	148.6	28.8	95.8	98.2	95.6	80.8	92.6
B3	SVE	SIE	148.6	20.9	95.0	95.4	96.6	82.4	92.4

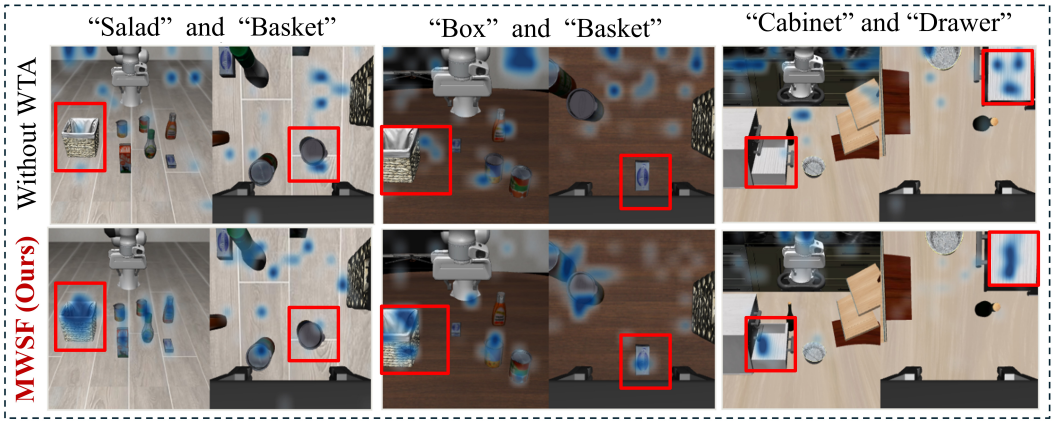

Figure 4: Attention heatmaps showing that our MWSF module can better focus on instruction-relevant visual regions, highlighted by red boxes.

Fusion: As shown in Table 4, WTA increases the overall success rate from 90.2% to 92.4%, with gains of 4.8% and 3.8% on the Spatial and Long suites, respectively. The heatmaps in Fig. 4 further show that MWSF with WTA focuses attention on instruction-relevant visual regions (red boxes) while suppressing background responses. These findings suggest that WTA strengthens instruction-guided visual grounding, contributing to improved performance in long-horizon manipulation.

6 Conclusion

In this work, we present Spike-driven VLA, the first spiking VLA framework to support end-to-end direct training for robotic manipulation. By integrating SVE, SIE, the MWSF module, and SpikeACT, it achieves competitive performance on LIBERO and Meta-World with fewer parameters and operations than representative ANN-based methods. These results demonstrate that directly trained spiking architectures offer a viable approach to resource-efficient embodied intelligence. Future work will extend the framework to broader tasks and diverse robotic platforms to advance the development of neuromorphic embodied agents.

7 USE OF LARGE LANGUAGE MODEL

In this manuscript, we used a Large Language Model (LLM) solely to refine the manuscript and improve the language. Specifically, we used the LLM to polish sentences, correct grammatical errors, and improve the clarity, readability, and academic style of the text. The LLM was not involved in the development of research ideas, methodology design, experimental implementation, result analysis, or generation of scientific claims. All technical contributions, experimental results, and conclusions presented in this paper are solely developed and verified by the authors.

REFERENCES

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.010.

Maxence Bouvier, Alexandre Valentian, Thomas Mesquida, François Rummens, Marina Reyboz, Elisa Vianello, and Edith Beigné. Spiking neural networks hardware implementations and challenges: A survey. *ACM Journal on Emerging Technologies in Computing Systems*, 15(2):1–35, 2019. doi: 10.1145/3304103.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023. URL `https://proceedings.mlr.press/v229/zitkovich23a.html`.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025.

Paweł Budzianowski et al. EdgeVLA: Efficient vision-language-action models. *arXiv preprint arXiv:2507.14049*, 2025.

Siqi Cai, Ran Zhang, Malu Zhang, Jibin Wu, and Haizhou Li. Eeg-based auditory attention detection with spiking graph convolutional network. *IEEE Transactions on Cognitive and Developmental Systems*, 16(5):1698–1706, 2024.

Long Chen, Xiaotian Song, and Yanan Sun. Las: Loss-less ann-snn conversion for fully spike-driven large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 1730–1738, 2026.

Xinyi Chen, Qu Yang, Jibin Wu, Haizhou Li, and Kay Chen Tan. A hybrid neural coding approach for pattern recognition with spiking neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3064–3078, May 2024. doi: 10.1109/TPAMI.2023.3339211.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Lei Deng, Guanrui Wang, Guoqi Li, Shuangchen Li, Ling Liang, Maohua Zhu, Yujie Wu, Zheyu Yang, Zhe Zou, Jing Pei, Zhenzhi Wu, Xing Hu, Yufei Ding, Wei He, Yuan Xie, and Luping Shi. Tianjic: A unified and scalable chip bridging spike-based and continuous neural computation. *IEEE Journal of Solid-State Circuits*, 55(8):2228–2246, 2020. doi: 10.1109/JSSC.2020.2970709.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long*

and Short Papers), pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. PaLM-E: An embodied multimodal language model. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 8469–8488. PMLR, 2023. URL https://proceedings.mlr.press/v202/driess23a.html.

Wei Fang, Zhaofei Yu, Yanqi Chen, Timothée Masquelier, Tiejun Huang, and Yonghong Tian. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2641–2651, 2021. doi: 10.1109/ICCV48922.2021.00266.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. LIBERO-Plus: A progressive robustness benchmark for visual-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 38574–38583, 2026.

Wulfram Gerstner and Werner M. Kistler. *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press, Cambridge, UK, 2002. ISBN 9780521890793. doi: 10.1017/CBO9780511815706.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R. Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.090.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U Tan, Navonil Majumder, Soujanya Poria, et al. Nora: A small open-sourced generalist vision language action model for embodied tasks. *arXiv preprint arXiv:2504.19854*, 2025.

Eugene M Izhikevich. Simple model of spiking neurons. *IEEE Transactions on neural networks*, 14(6):1569–1572, 2003.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Wei Li, Renshan Zhang, Rui Shao, Jie He, and Liqiang Nie. CogVLA: Cognition-aligned vision-language-action model via instruction-driven routing & sparsification. *arXiv preprint arXiv:2508.21046*, 2025. doi: 10.48550/arXiv.2508.21046. URL https://arxiv.org/abs/2508.21046.

Yuhang Li, Shikuang Deng, Xin Dong, Ruihao Gong, and Shi Gu. A free lunch from ann: Towards efficient, accurate spiking neural networks calibration. In *International conference on machine learning*, pp. 6316–6325. PMLR, 2021.

Yu Liang, Zijian Zhou, Wenjie Wei, Shuai Wang, Honglin Cao, Ammar Belatreche, Yu Yang, Malu Zhang, Yang Yang, and Haizhou Li. Spikinglm: Towards fully spiking language model. In *Forty-third International Conference on Machine Learning*, 2026.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Wolfgang Maass. Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997. doi: 10.1016/S0893-6080(97)00011-7.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and RT-X models. In *2024 IEEE International Conference on Robotics and Automation*, pp. 6892–6903. IEEE, 2024. doi: 10.1109/ICRA57147.2024.10611477.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025. doi: 10.48550/arXiv.2504.16054. URL https://arxiv.org/abs/2504.16054.

Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019. doi: 10.1038/s41586-019-1677-2.

Yimeng Shan, Zhenbang Ren, Haodi Wu, Wenjie Wei, Rui-Jie Zhu, Shuai Wang, Dehao Zhang, Yichen Xiao, Jieyuan Zhang, Kexin Shi, et al. Sdtrack: A baseline for event-based tracking via spiking neural networks. *arXiv preprint arXiv:2503.08703*, 2025.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. SmolVLA: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025. doi: 10.48550/arXiv.2506.01844. URL https://arxiv.org/abs/2506.01844.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Ruiqi Song, Dujun Nie, Siyu Teng, Baiyong Ding, Xiaotong Zhang, Dong Li, Chenming Zhang, Yuchen Li, Hangbin Wu, and Long Chen. Spikevla: Vision-language-action models with spiking neural networks. *arXiv preprint arXiv:2606.27807*, 2026.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Hongzhi Wang, Xiubo Liang, Tao Zhang, Yue Gu, and Weidong Geng. Pssd-transformer: powerful sparse spike-driven transformer for image semantic segmentation. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 758–767, 2024.

Jingya Wang, Xin Deng, Wenjie Wei, Dehao Zhang, Shuai Wang, Qian Sun, Jieyuan Zhang, Hanwen Liu, Ning Xie, and Malu Zhang. Training-free ann-to-snn conversion for high-performance spiking transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 2128–2136, 2026.

Shuai Wang, Malu Zhang, Jingya Wang, Dehao Zhang, Yimeng Shan, Jieyuan Zhang, Yichen Xiao, Honglin Cao, Haonan Zhang, Zeyu Ma, et al. Bipolar self-attention for spiking transformers. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a.

Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, Siteng Huang, Yifan Tang, Wenhui Wang, Ru Zhang, Jianyi

Liu, and Donglin Wang. VLA-Adapter: An effective paradigm for tiny-scale vision-language-action model. *arXiv preprint arXiv:2509.09372*, 2025b. doi: 10.48550/arXiv.2509.09372. URL https://arxiv.org/abs/2509.09372.

Junjie Wen et al. TinyVLA: Towards fast, data-efficient vision-language-action models for robotic manipulation. *arXiv preprint arXiv:2409.12514*, 2024.

Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in neuroscience*, 12:331, 2018.

Hengyi Xie, Chenfei Yao, Xianjin Wu, Xuanyang Xi, Yiping Tang, Di Xu, Yingying Zhu, Dingkang Liang, Xiang Bai, and Han Ding. TurboVLA: Real-time vision-language-action model at 32 Hz on an RTX 4090 with <1 GB VRAM. *arXiv preprint arXiv:2607.27205*, 2026. doi: 10.48550/arXiv.2607.27205. URL https://arxiv.org/abs/2607.27205.

Man Yao, Jiakui Hu, Zhaokun Zhou, Li Yuan, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer. *Advances in neural information processing systems*, 36:64043–64058, 2023.

Man Yao, Jiakui Hu, Tianxiang Hu, Yifan Xu, Zhaokun Zhou, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer v2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips. *arXiv preprint arXiv:2404.03663*, 2024.

Man Yao, Xuerui Qiu, Tianxiang Hu, Jiakui Hu, Yuhong Chou, Keyu Tian, Jianxing Liao, Luziwei Leng, Bo Xu, and Guoqi Li. Scaling spike-driven transformer with efficient spike firing approximation training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(4): 2973–2990, 2025. doi: 10.1109/TPAMI.2025.3530246.

Haoran You, Xiaohan Chen, Yongan Zhang, Chaojian Li, Sicheng Li, Zihao Liu, Zhangyang Wang, and Yingyan Lin. Shiftaddnet: A hardware-inspired deep network. *Advances in Neural Information Processing Systems*, 33:2771–2783, 2020.

Kang You, Zekai Xu, Chen Nie, Zhijie Deng, Qinghai Guo, Xiang Wang, and Zhezhi He. Spikezip-tf: Conversion is all you need for transformer-based snn. *arXiv preprint arXiv:2406.03470*, 2024.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020.

Malu Zhang, Shuai Wang, Jibin Wu, Wenjie Wei, Dehao Zhang, Zijian Zhou, Siying Wang, Fan Zhang, and Yang Yang. Toward energy-efficient spike-based deep reinforcement learning with temporal coding. *IEEE Computational Intelligence Magazine*, 20(2):45–57, 2025a.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, Fan Lu, He Wang, Zhizheng Zhang, Li Yi, Wenjun Zeng, and Xin Jin. DreamVLA: A vision-language-action model dreamed with comprehensive world knowledge. *arXiv preprint arXiv:2507.04447*, 2025b. doi: 10.48550/arXiv.2507.04447. URL https://arxiv.org/abs/2507.04447.

Yajing Zheng, Lingxiao Zheng, Zhaofei Yu, Boxin Shi, Yonghong Tian, and Tiejun Huang. High-speed image reconstruction through short-term plasticity for spiking cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6358–6367, 2021.

Chenlin Zhou, Han Zhang, Zhaokun Zhou, Liutao Yu, Liwei Huang, Xiaopeng Fan, Li Yuan, Zhengyu Ma, Huihui Zhou, and Yonghong Tian. Qkformer: Hierarchical spiking transformer using qk attention. *arXiv preprint arXiv:2403.16552*, 2024.

Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, YAN Shuicheng, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. In *The Eleventh International Conference on Learning Representations*, 2023.

Zijian Zhou, Wenjie Wei, Yu Liang, Jialin Li, Ammar Belatreche, Honglin Cao, Shuai Wang, Malu Zhang, Yang Yang, and Haizhou Li. SmoothSpike: Spiking transformer with learnable hadamard transformation. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/63665.

Rui-Jie Zhu, Qihang Zhao, Guoqi Li, and Jason K Eshraghian. Spikegpt: Generative pre-trained language model with spiking neural networks. *arXiv preprint arXiv:2302.13939*, 2023.

A Spiking Visual Encoder

Network architecture. Our spiking visual encoder adopts a three-stage pyramidal architecture. The first two stages contain one and two convolutional mixer blocks, respectively. These blocks extract local features through spatial and channel mixing. The third stage comprises seven spiking self-attention blocks to model global spatial dependencies. Each attention block combines spike-based feature transformations with residual connections. Spike Patch Embedding layers progressively reduce spatial resolution and increase channel width. For an input of size $H \times W$, the three stages produce feature maps of size $C \times H/4 \times W/4$, $2C \times H/8 \times W/8$, and $4C \times H/16 \times W/16$, respectively. Here, C denotes the base channel width. This hierarchy extracts fine-grained local features at high resolutions and captures global spatial dependencies with fewer visual tokens.

ImageNet-1K pretraining. To improve visual generalization, we pretrain SVE on ImageNet-1K before training Spike-driven VLA for robotic grasping. The dataset contains approximately 1.3 million training images and 50,000 validation images across 1,000 classes. A classification head maps visual features to 1,000 class logits. Training uses cross-entropy loss with a label-smoothing coefficient of 0.1. Surrogate gradients enable backpropagation through spiking activations.

Pretraining is implemented in PyTorch with distributed data parallelism on four NVIDIA A800 GPUs. We train for 200 epochs with 224×224 inputs and a batch size of 600. The LAMB optimizer uses a base learning rate of 6×10^{-4} and a weight decay of 0.05. The first 10 epochs are used for learning-rate warmup. The layer-wise learning-rate decay factor is 1.0, assigning the same multiplier to all backbone layers. Mixup and CutMix are disabled. Table 6 summarizes the pretraining configuration.

Table 6: ImageNet-1K pretraining settings for the spiking visual encoder.

Hyperparameter	Setting
Time steps	4
Training epochs	200
Input resolution	224×224
Batch size	600
Optimizer	LAMB
Base learning rate	6×10^{-4}
Warmup epochs	10
Layer-wise LR decay	1.0
Weight decay	0.05
Mixup	0
CutMix	0
Label smoothing	0.1
Hardware	$4\times$ NVIDIA A800

Dual-View Encoding and Joint Fine-Tuning. After pretraining, we remove the classification head and retain the visual backbone. Main-view and wrist-view observations are processed by the same backbone with shared weights. For each view, the final 14×14 feature map is flattened into 196 spatial tokens. These tokens are linearly projected to the common embedding dimension $d = 256$ and normalized. We then concatenate the tokens from both views along the token dimension, yielding $\mathbf{Z}_{v,n} \in \mathbb{R}^{T\times B\times 392\times 256}$ for the LIBERO task. The backbone is jointly fine-tuned with the cross-modal fusion and action-decoding modules for language-conditioned robotic manipulation.

B Spike Instruction Encoder

The SIE E_ℓ encodes a language instruction $\mathcal{L}$ into task-conditioned spiking tokens for visual perception. It first tokenizes $\mathcal{L}$ into $\mathrm{Tok}(\mathcal{L}) \in \mathbb{Z}^{B\times N_\ell}$. A spiking BERT backbone (Liang et al., 2026; Zhou et al., 2026) then extracts contextual representations $\mathbf{Z}^{\mathrm{text}}$, which are projected from d_ℓ to the shared VLA embedding dimension d by a dimensional adapter:

$$\mathbf{Z}_\ell = E_\ell(\mathrm{Tok}(\mathcal{L})) = \mathrm{BN}\big(\mathrm{Linear}_\ell\big(\mathbf{Z}^{\mathrm{text}}\big)\big), \qquad \mathbf{Z}_\ell \in \mathbb{R}^{T\times B\times N_\ell\times d}. \tag{16}$$

We instantiate the backbone with SmoothSpike (Zhou et al., 2026), which comprises 12 encoder layers with 12 attention heads per layer. It uses a hidden dimension of $d_\ell = 768$, an MLP dimension

of 3,072, and $T = 4$ simulation steps. The backbone is pretrained using masked language modeling (Devlin et al., 2019) and remains frozen during Spike-driven VLA training. Consequently, the manipulation loss updates only Linear_ℓ and the subsequent cross-modal fusion and action-decoding modules. In the following, we detail the pretraining data, the SmoothSpike method, and the SIE architecture.

Pre-training corpus. SmoothSpike follows the BERT masked-language-modeling (MLM) paradigm (Devlin et al., 2019). The pre-training corpus is assembled from publicly available English text collections accessed through the Hugging Face Datasets library: TinyStories,[1] BookCorpus,[2] CC-News,[3] OpenWebText,[4] and English Wikipedia.[5] Each source is stored as a Hugging Face `Dataset` (or `DatasetDict`) with a free-text field. When a corpus provides a `text` column, we use it directly; otherwise, we take the first available text-bearing column. For sources without an official validation split, the first $\min(1000, N_{\text{train}})$ raw documents of that source are reserved as its validation contribution. Train splits from all five sources are concatenated into a single training pool, and the corresponding validation contributions form a held-out validation pool. No additional document-level filtering, deduplication, or language identification is applied beyond what is already present in the upstream releases.

We tokenize the corpus using the bert-base-uncased WordPiece tokenizer with a vocabulary of 30,522 tokens. Within each preprocessing batch, tokenized documents are concatenated and divided into non-overlapping blocks of $L = 128$ tokens. Incomplete trailing blocks are discarded. Each block stores token IDs, token-type IDs, attention masks, and special-token masks. The processed corpus contains 137,271,688 training blocks and 1,627,657 validation blocks, corresponding to approximately 17.6 billion training tokens. MLM targets are generated dynamically by Hugging Face's DataCollatorForLanguageModeling with a masking probability of 15%. Special tokens are excluded from masking, and new masking patterns are sampled in each epoch. The pretrained backbone is frozen to provide semantic features $\mathbf{Z}^{\text{text}}$ for SIE during VLA training and inference. Instructions from LIBERO and Meta-World use the same tokenizer and are encoded as variable-length sequences of $N_\ell \leq 128$ tokens.

Table 7: Training hyperparameters of the SmoothSpike backbone used as SIE. The VLA column summarizes its configuration in Spike-driven VLA, where the backbone remains frozen.

Hyperparameter	MLM pre-training	Spike-driven VLA training
Objective	MLM (15% masking)	Frozen encoder
Tokenizer	`bert-base-uncased`	Same
Maximum sequence length	128	$N_\ell \leq 128$
Optimizer	AdamW	–
Peak learning rate	2×10^{-4}	–
Weight decay	0	–
LR schedule	Linear, 5k warm-up	–
Per-device / global batch size	64 / 512	–
Time steps T	4	4
Tokens seen	36.0B	–
Trainable modules	Full SmoothSpike	Linear_ℓ, BN only

Training protocol. SmoothSpike is trained directly as an SNN, without ANN-to-SNN conversion. Each forward pass resets neuronal states before the sequence is consumed. LIF neurons run in multi-step mode with membrane time constant $\tau = 2.0$ and detached reset; the default firing threshold is 1.0, except for the attention-output LIF whose threshold is 0.5. Learnable orthogonal transforms $\widetilde{\mathbf{H}}$ are initialized as randomized Hadamard matrices and projected onto the orthogonal group at every forward pass by a quintic Newton–Schulz iteration with $K = 5$ steps and coefficients $(a, b, c) =$

[1] `https://huggingface.co/datasets/roneneldan/TinyStories`
[2] `https://huggingface.co/datasets/bookcorpus`
[3] `https://huggingface.co/datasets/vblagoje/cc_news`
[4] `https://huggingface.co/datasets/Skylion007/openwebtext`
[5] `https://huggingface.co/datasets/wikimedia/wikipedia`

Table 8: Configuration of the Spike Instruction Encoder (SIE) used in Spike-driven VLA. The backbone is SmoothSpike-12-768 (Zhou et al., 2026). N_ℓ is the instruction length ($N_\ell \leq 128$). $\mathbf{H}_1$ is shared across residual branches; $\mathbf{H}_2$ is per-layer on the value branch; $\mathbf{H}_3$ is a 4-block diagonal transform in the MLP. Query/key LIF neurons are not transformed.

stage	# Tokens	Layer Specification			SIE
Emb.	N_ℓ	Token / position / segment Emb.		Dim	768
		Vocabulary / max position		–	30522 / 512
		Embedding Norm		–	LayerNorm
		Shared orthogonal encoding		–	$\mathbf{H}_1$
		Temporal repeat		T	4
Encoder block	N_ℓ	SSA	RMSNorm + LIF + $\mathbf{H}_1^\top$	τ	2.0
			Q/K Linear + RMSNorm + LIF	Heads	12
			V Linear + $\mathbf{H}_2$ + RMSNorm + LIF	Dim	768
			Spike-driven attn. $(QK^\top \odot M)sV$	s	$1/8$
			LIF + $\mathbf{H}_2^\top$ + W_o + $\mathbf{H}_1$	v_{th}	0.5
			Residual + dropout	p	0.1
		MLP	RMSNorm + LIF + $\mathbf{H}_1^\top$	τ	2.0
			Up-projection	Dim	$768 \to 3072$
			Block-diagonal $\mathbf{H}_3$	# blocks	4
			LIF + $\mathbf{H}_3^\top$ + down-projection + $\mathbf{H}_1$	Dim	$3072 \to 768$
			Residual + dropout	p	0.1
		# Blocks		–	12
Head	N_ℓ	MLM readout (pre-training)		–	mean_T + $\mathbf{H}_1^\top$
		SIE features $\mathbf{Z}^{\text{text}}$		Dim	$T \times N_\ell \times 768$
		VLA adapter $\text{BN}(\text{Linear}_\ell(\cdot))$		Dim	$768 \to d$

$(3.4445, -4.7750, 2.0315)$. $\mathbf{H}_1$ is shared across residual-connected branches, $\mathbf{H}_2$ is applied to the value branch, and $\mathbf{H}_3$ is a block-diagonal MLP transform with four 768×768 blocks. Query/key LIF neurons are left untransformed. After language-model training, all transforms are fused into adjacent linear weights, so SIE inference introduces no extra matmuls and remains spike-driven.

MLM pre-training minimizes the cross-entropy on masked tokens with AdamW (peak learning rate 2×10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, no weight decay). The learning rate warms up linearly for $5{,}000$ steps and then decays linearly to zero. Training uses 8 GPUs, 64 sequences per device, no gradient accumulation (global batch size 512), and maximum length 128. The run consumes approximately 36.0 billion tokens. Hidden-state and attention dropout are both 0.1, and training is performed in full precision. Table 7 summarizes the language-backbone hyperparameters.

Architecture. Table 8 specifies the SIE architecture. Although the table follows the stage-wise convention used for hierarchical spike-driven Transformers, SIE preserves the instruction token count N_ℓ throughout. It adopts a pre-norm spiking BERT encoder, with LIF neurons implementing the spiking operation $\mathcal{SN}(\cdot)$. Token, position, and segment embeddings are summed and normalized, right-multiplied by the shared orthogonal transform $\mathbf{H}_1$, and presented over T timesteps. Each encoder block consists of SSA and a spiking MLP.In SSA, RMSNorm and a projection LIF generate spikes that are mapped back by $\mathbf{H}_1^\top$ before the $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ projections. The $\mathbf{Q}$ and $\mathbf{K}$ branches each apply RMSNorm and $\mathcal{SN}(\cdot)$, whereas the $\mathbf{V}$ branch applies $\mathbf{H}_2$ before RMSNorm and $\mathcal{SN}(\cdot)$. Following the SSA notation introduced above, attention is computed without softmax as:

$$\mathbf{Score} = s \cdot \big((\mathbf{Q} \times \mathbf{K}^\top) \odot \mathbf{M}\big), \quad \mathbf{Attn} = \mathcal{SN}(\mathbf{Score} \times \mathbf{V}), \tag{17}$$

where $\mathbf{M}$ denotes the attention mask and $s = 1/\sqrt{d_h} = 1/8$. The spiking attention output $\mathbf{Attn}$ then passes through $\mathbf{H}_2^\top$, the output projection $\mathbf{W}_o$, and $\mathbf{H}_1$ before the residual addition. The MLP applies RMSNorm and $\mathcal{SN}(\cdot)$, followed by $\mathbf{H}_1^\top$, an up-projection from 768 to 3072 dimensions, and the block-diagonal transform $\mathbf{H}_3$, which comprises four 768×768 blocks. A second $\mathcal{SN}(\cdot)$ operation is followed by $\mathbf{H}_3^\top$, a down-projection to 768 dimensions, and $\mathbf{H}_1$ before the residual

addition. For MLM pre-training, the encoder outputs are averaged over T, mapped back by $\mathbf{H}_1^\top$, and passed to the MLM head. In Spike-driven VLA, the MLM-pretrained SmoothSpike encoder is frozen, and each instruction is processed over $T = 4$ timesteps to match the visual stream. We retain the timestep-resolved encoder features $\mathbf{Z}^{\text{text}} \in \mathbb{R}^{T\times B\times N_\ell\times d_\ell}$ without averaging over time, where B denotes the batch size and$d_\ell = 768$ is the encoder feature dimension. A linear projection followed by batch normalization maps each token feature to the shared feature dimension $d = 256$:

$$\mathbf{Z}_\ell = \mathrm{BN}\big(\mathrm{Linear}(\mathbf{Z}^{\text{text}})\big) \in \mathbb{R}^{T\times B\times N_\ell\times d}. \tag{18}$$

These features preserve the distinctions among task instructions and provide the semantic cues used by MWSF module to select task-relevant visual information.

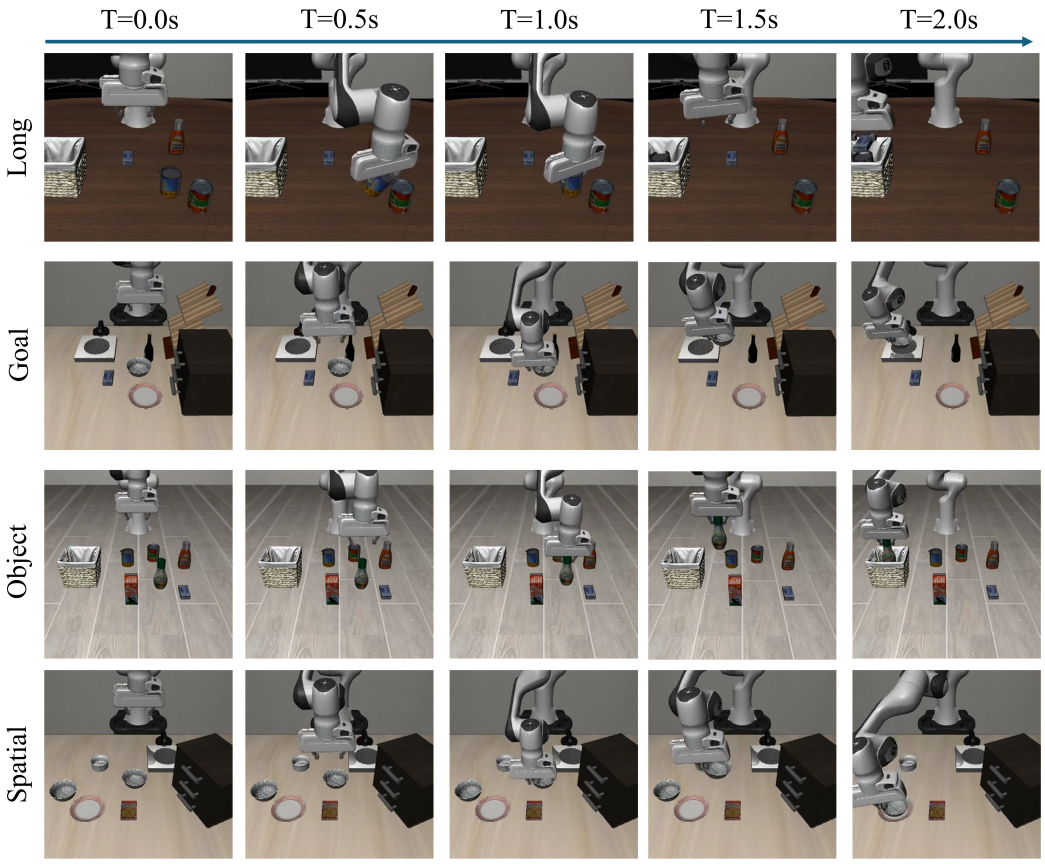

Figure 5: Successful Spike-driven VLA rollouts on LIBERO. Rows (a)–(d) show four tasks. Columns show frames sampled at T= 0, 0.5, 1.0, 1.5, and 2.0.

C Training Details of Spike-driven VLA

Training flow. For each benchmark, Spike-driven VLA is trained as a single policy across its tasks. The SmoothSpike BERT backbone remains frozen, while SVE, the visual and text projections, the MWSF layers, the state encoder, and the SpikeACT decoder are optimized jointly. Before policy training, the text projection is calibrated on benchmark-specific instructions to initialize its weights. For each sample, SVE encodes the RGB observations into N_v visual tokens, while SmoothSpike BERT encodes the instruction into N_ℓ text tokens. Both streams are projected to the shared feature dimension d. The MWSF layers fuse them into $N_v + N_\ell$ tokens. Concatenating N_s encoded state tokens forms a memory of shape $T \times B \times N_m \times d$, where $N_m = N_v + N_\ell + N_s$. SpikeACT uses this memory to predict an action chunk of shape $B \times K \times d_a$, where K is the action horizon and d_a is the action dimension. The neuronal simulation length T is distinct from the action horizon K.

Table 9: Training configuration of Spike-driven VLA on LIBERO.

Item	Configuration
Effective batch size	8 GPUs $\times$ 32 samples/GPU $\times$ 1 accumulation step $= 256$
Training budget	80,000 updates; 20,480,000 sample exposures
Optimizer	AdamW, $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$
Visual backbone LR	1×10^{-5}
Other trainable parameters LR	5×10^{-5}
LR schedule	10,000 steps of linear warm-up, followed by a constant LR
Gradient clipping	Global gradient norm capped at 1
EMA	Decay 0.999; updated after every optimizer step
Policy precision	FP32; highest float32 matrix-multiplication precision; TF32 disabled
$\mathcal{L}_{\mathrm{BC}}$	Masked L1 loss on $B \times 12 \times 7$ action chunks;

For both benchmarks, behavior cloning uses a masked L1 loss between predicted and demonstrated action chunks. The loss is averaged over valid action elements, with equal weight assigned to each action dimension. Text-projection calibration is performed before policy training and serves only as initialization. We use neither staged objective switching nor ANN-to-SNN conversion.

On LIBERO, $K = 12$ and $d_a = 7$, giving action chunks of shape $B \times 12 \times 7$. The text projection is calibrated on 40 LIBERO instructions. Table 9 reports the optimization settings for the LIBERO policy. On Meta-World MT50, the policy predicts 12-step chunks in the benchmark's four-dimensional action space, giving a shape of $B \times 12 \times 4$. Its text projection is calibrated using MT50 instructions.

D Theoretical Energy Estimation

We estimate arithmetic energy per policy forward pass, which produces a 12-step action chunk. The estimate includes SmoothSpike, whose weights are frozen during policy training but whose computation remains part of inference. Direct encoding supplies continuous-valued images to the first convolution, so its operations are counted as MACs. In subsequent spike-driven layers, synaptic accumulations occur only in response to nonzero input spikes.

For a spike-driven convolutional or linear layer ℓ, let M_ℓ denote the dense MAC count per execution, $r_{\ell,t}$ the input firing rate at its t-th execution, and T_ℓ the number of executions in one forward pass. We estimate its synaptic operations as:

$$\widehat{\mathrm{SOP}}_\ell \simeq M_\ell \sum_{t=1}^{T_\ell} r_{\ell,t}. \tag{19}$$

Counting actual executions avoids applying the four simulation steps again to operations performed after temporal aggregation. Spike-triggered accumulations in attention are counted as ACs, while operations on continuous-valued inputs are counted as MACs. Let N_{AC} and N_{MAC} be the respective totals over the complete forward pass. The theoretical computation energy is:

$$\widehat{E}_{\mathrm{SpikeVLA}} = e_{\mathrm{AC}} N_{\mathrm{AC}} + e_{\mathrm{MAC}} N_{\mathrm{MAC}}. \tag{20}$$

Following the 45-nm reference model for 32-bit arithmetic (Zhou et al., 2023; 2024), we use $e_{\mathrm{AC}} = 0.9\,\mathrm{pJ}$ and $e_{\mathrm{MAC}} = 4.6\,\mathrm{pJ}$. Where applicable, batch normalization and SmoothSpike's orthogonal transforms are folded into adjacent weights before counting operations. The estimate excludes memory access and non-synaptic overhead.

E Evaluation on LIBERO and Meta-World

LIBERO. We evaluate a single policy on the Spatial, Object, Goal, and Long suites. Each demonstration provides workspace and wrist RGB observations, an instruction, robot states, and actions. The two camera views are processed by the shared SVE to produce 392 visual tokens, while SmoothSpike encodes the instruction into 21 text tokens. At each trajectory step, we construct a 12-step

Table 10: Success rates (%) on LIBERO-Plus. Parentheses give successful and total rollout counts. Overall rates are computed from pooled rollout counts.

Suite	**Overall**	Background Textures	Robot Initial States	Camera Viewpoints	Language Instructions	Sensor Noise	Object Layout	Light Conditions
Spatial	65.7	72.1 (186/258)	57.7 (202/350)	63.0 (237/376)	40.0 (156/390)	48.4 (170/351)	87.0 (335/385)	99.7 (291/292)
Object	56.0	70.6 (175/248)	42.2 (168/398)	42.9 (170/396)	58.2 (206/354)	32.7 (138/422)	65.8 (265/403)	97.3 (289/297)
Goal	54.0	69.4 (195/281)	50.9 (208/409)	46.8 (191/408)	35.4 (145/410)	43.5 (165/379)	60.9 (259/425)	84.6 (236/279)
Long	37.6	36.0 (104/289)	37.2 (146/393)	14.6 (61/419)	63.2 (242/383)	17.6 (79/449)	54.5 (170/312)	52.9 (145/274)

relative-action target and mask padded actions near the trajectory end. Each suite contains ten tasks, with 50 evaluation trials per task, giving 2,000 trials overall. Evaluation uses EMA weights, FP32 inference, and seed 7; each predicted action chunk is executed open loop for 12 steps. Figure 5 shows four successful rollouts generated by the policy.

Across seven LIBERO-Plus perturbation categories, Spike-driven VLA achieves 53.2% success over 10,030 rollouts (Table 10). Success is highest under lighting (84.2%) and object-layout changes (67.5%), but lowest under sensor noise (34.5%) and camera-viewpoint changes (41.2%). The Long suite is particularly challenging (37.6% overall), dropping to 14.6% under viewpoint changes and 17.6% under sensor noise.

Meta-World MT50. For MT50, we train a single policy across all 50 tasks using 50 expert demonstrations per task, yielding 2,500 trajectories in total. Rendered observations, task descriptions, robot states, and expert actions are aligned at each simulation step. The action sequences are divided into 12-step targets, with padded positions excluded from the training loss. We evaluate the policy over 50 trials per task, yielding 2,500 evaluation trials in total. Figure 6 presents successful rollouts from representative MT50 tasks.

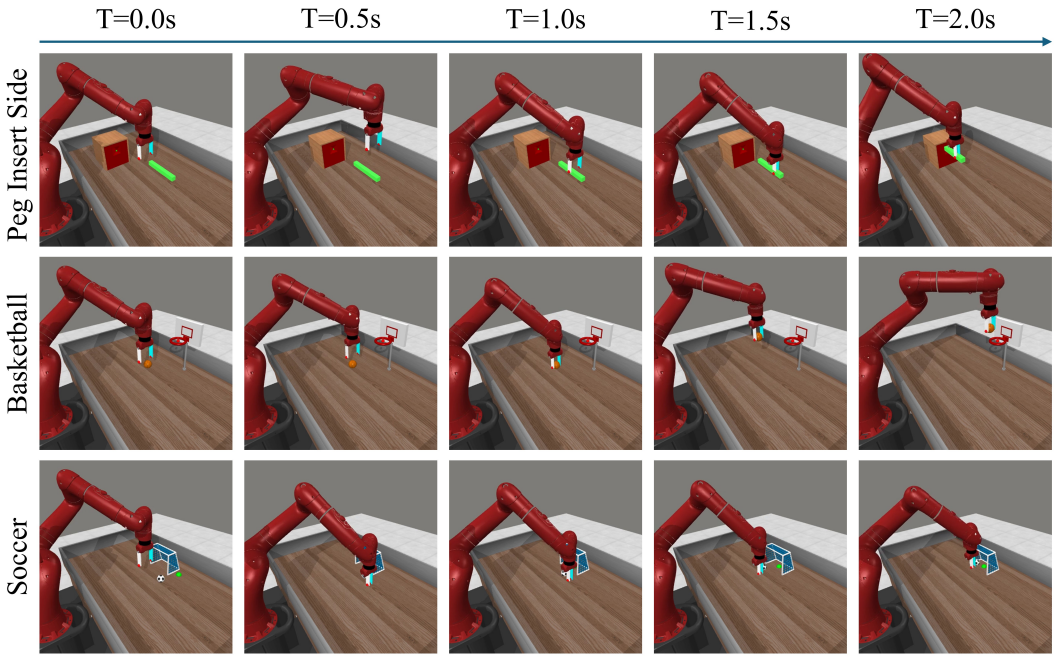

Figure 6: Successful Spike-driven VLA rollouts on Meta-World MT50. Rows (a)–(d) show four tasks. Columns show frames sampled at T= 0, 0.5, 1.0, 1.5, and 2.0.