

# A Real-Time Robotic Diffusion Policy Accelerator Exploiting Self- and Cross-Guided Modal Similarity

Boju Chen, Xiaoyu Feng<sup>†</sup>, Junyan Lin, Chen Tang, Huazhong Yang, Yongpan Liu

*Department of Electronic Engineering, Tsinghua University, Beijing, China*

<sup>†</sup>fengxy14@tsinghua.org.cn

**Abstract**—Diffusion Policy (DP) has demonstrated strong potential in robotic visuomotor control, offering robust generalization and seamless integration of multi-modal data. However, its complex model structure and increasing multi-modal inputs have brought latency and power challenges for edge resource-constrained robotic platforms. To address the above challenges, we identify the potential intra-model and inter-model redundancies in DP. We observe that DP relies on frequent multi-modal inputs such as images and text during execution. However, the demands of fine-grained robotic manipulation result in substantial intra-modal similarity across consecutive image frames, which, combined with inter-modal semantic redundancy between images and language, indicates that much of the input information is repetitive and potentially compressible. Yet prior works have not exploited these characteristics for targeted optimization. We therefore propose a hardware–software co-design accelerator. On the algorithmic side, we introduce self- and cross-guided modal compression, leveraging intra- and inter-modality similarity to reduce redundant computation within the key DP modules. On the hardware side, we design a tailored architecture that supports multiple operators with optimized sparse memory access, lightweight computation engines, and reconfigurable on-chip dataflow, substantially reducing energy cost. Experimental results demonstrate a 26× speedup over a high-performance GPU while consuming only 1.5 W, enabling low-power and real-time robotic control on edge robotic devices.

**Index Terms**—robot, diffusion policy, accelerator architecture

## I. INTRODUCTION

In recent years, the emergence of diffusion-based paradigms for action generation has fundamentally transformed the field of robotic manipulation. Among these, Diffusion Policy (DP) [1] has emerged as a representative framework, notable for probabilistic action modeling, multi-modal conditioning, and robust generalization. In practice, most DP models are guided predominantly by visual and textual inputs [2], where frequent large-scale visual interactions combined with explicit task instructions enable DP to achieve state-of-the-art performance in robotic control tasks.

Accelerating diffusion models has been widely studied in image and video generation, where techniques such as differential methods and mixed-precision quantization have proven effective [3], [4]. However, these approaches are difficult to transfer to robotic action generation. This is due to the unique characteristics of DP: (1) As shown in Fig.1(a), in image and video diffusion, the majority of computation is dominated by denoised data—for example, in Stable Diffusion [5], 96.8% of tokens correspond to denoised images, while only 3.2% correspond to conditional text. In contrast, robotic diffusion

policies primarily process multi-modal condition data, with denoised action outputs constituting a minor fraction—for instance, in 3D Diffuser Actor [6], 97.7% of tokens are conditional image and text tokens, whereas only 2.3% are denoised action tokens. This distribution highlights the ineffectiveness of methods that focus on compressing denoised data. (2) Robotic tasks prioritize end-to-end latency over throughput, with inference typically performed in a single-batch, real-time setting, making some multi-batch parallelization techniques ineffective. (3) Certain approaches, such as distillation-based sampling [7], [8] or mixture-of-experts (MoE) [9], [10], could in principle be applied to robotic tasks. However, they often demand substantial retraining because they require fundamental changes to the training pipeline or network architecture, making rapid deployment infeasible in real-world scenarios.

As shown in Fig.1(b), we illustrate the representative architecture of DP. A single end-to-end inference involves encoding both image inputs and natural-language instructions into condition tokens. These tokens are then fed into the diffusion module, where a sequence of denoising steps progressively transforms noise into precise action tokens as the final output. Some policy variants additionally incorporate other sensory signals or proprioceptive states [11], [12], but their scale is relatively small compared to visual inputs and thus do not substantially affect complexity. In Fig.1(b), we also report the equivalent parameter access ratio in DP, where the main components—constituting the major bottlenecks—are the vision encoder and the conditional diffusion module. The former typically adopts a multi-layer convolutional neural network, while the latter is primarily built on Diffusion Transformer (DiT) blocks.

This work builds on the unique characteristics of robotic manipulation and highlights potential avenues for achieving more efficient policies. We identify two forms of similarity that systematically introduce redundancy in policy execution:

- **Intra-modal similarity:** This refers to strong correlations within a single modality, most evident in the temporal domain. Under the robot’s continuous interaction with environment for rapid responses, the model often processes highly similar observations across adjacent time steps. Given that temporal correlations manifest most prominently near raw sensory inputs, we examined feature correlations across successive frames within the vision encoder, the component closest to the camera. In Fig.1(c), the results reveal that except for a few sparse regions associated with moving objects

<sup>†</sup> Corresponding author

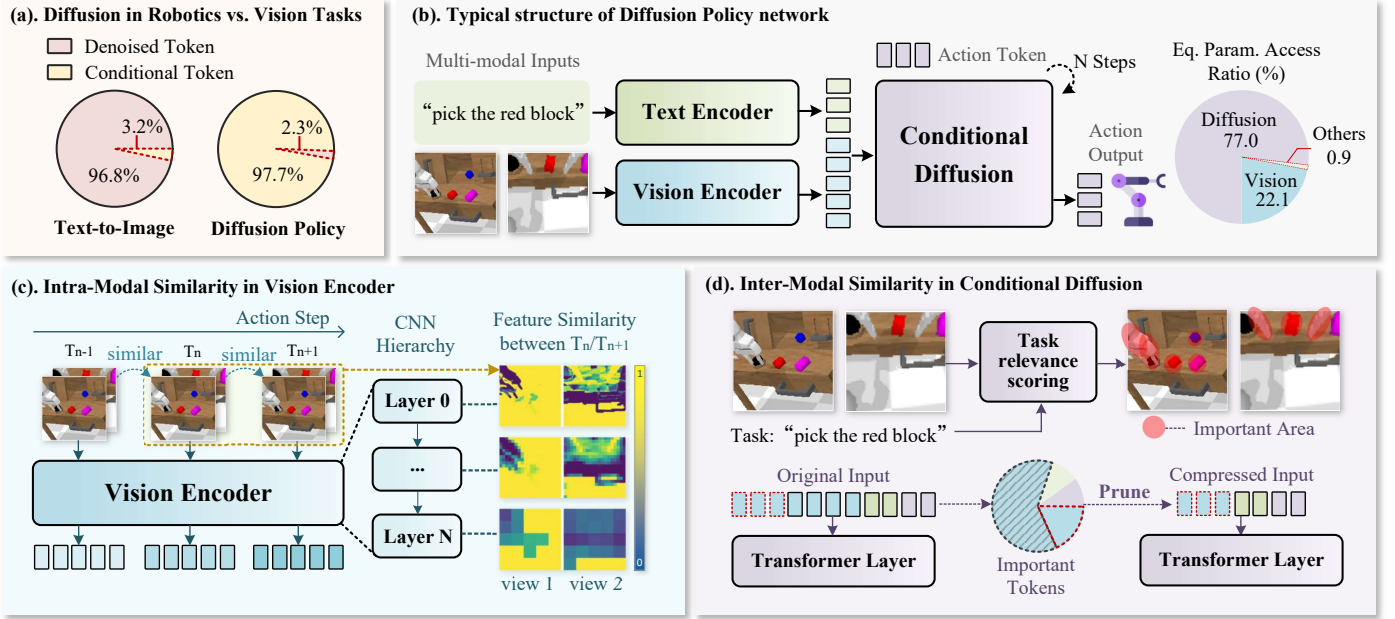


Fig. 1. Overview of Diffusion Policy (DP). (a) Comparison of token ratio in Robotics Diffusion Policy versus Vision Diffusion. (b) A typical architecture of DP with a breakdown of equivalent parameter access ratio in DP. (c) Analysis of intra-modal similarity in the Vision Encoder by examining the correlation of inter-frame features. (d) Analysis of inter-modal similarity in Conditional Diffusion by investigating the semantic alignment between text and image tokens.

or local changes, most features remain nearly identical across consecutive frames, indicating substantial redundancy.

- **Inter-modal similarity:** Alongside intra-modal similarity, inter-modal correlations across modalities such as vision and language are equally non-negligible, reflecting their semantic overlap. Inspired by how humans focus attention on task-relevant cues while ignoring distractions, we assume that, when performing a specific task, only a subset of visual features contribute meaningfully for robot manipulation. As illustrated in Fig.1(d), we observe that only a fraction of visual tokens shows high semantic similarity to the task descriptions, while these aligned tokens consistently provide key information for guiding the robot’s actions.

Motivated by these observations, this work proposes a sparsity-aware, dataflow-reconfigurable accelerator for edge robotic platforms, aiming to achieve both higher response frequency and lower energy consumption. The main contributions of this paper are summarized as follows:

On the algorithmic level: (1) We propose a *self-guided* feature reuse mechanism in the vision encoder that leverages intra-modal similarity with Receptive Field Propagation and sparse-layer substitution, thereby reducing redundant computation. (2) We introduce a *cross-guided* token pruning strategy in the conditional diffusion module, which leverages inter-modal semantic alignment via an adaptive importance evaluator with controllable sparsity, incurring negligible fine-tuning overhead.

On the hardware level: (3) We design a dedicated subsystem that supports sparse data handling for convolution through block-wise data transfer and pixel-wise computation, while also supporting structured operations in attention. (4) We propose a dataflow-reconfigurable architecture allowing flexible module

interconnections and operator fusion specifically optimized for attention, thereby maximizing on-chip buffer utilization and minimizing off-chip memory access to boost inference.

## II. BACKGROUND

### A. Diffusion Policy in Robotic Manipulation

Building on diffusion models’ strong expressiveness and stable likelihood-based training [13], [14], Diffusion Policy (DP) formulates robotic visuomotor control as a conditional denoising diffusion process for robust modeling of complex multi-modal action distributions [1], [15]. Since its initial development, DP has been inherently compatible with diverse modalities, incorporating multi-view/3D vision [2], [6], tactile sensing [11], [16], and proprioceptive feedback [12], with visual perception remaining the dominant source of information. DP can operate either as a standalone policy for tasks such as grasping, assembly, and tool use [17], [18] or as a downstream action expert in dual-system frameworks [19], [20], where an upstream vision–language model (VLM) provides high-level guidance for tasks requiring strong generalization. Given its role in real-time, low-level control with relatively lightweight inference requirements, DP is naturally suited for edge deployment in both standalone and dual-system settings, whereas the upstream VLM, due to its large size and computational cost, is better hosted on a server. However, as multi-modal input scale and dimensionality increase, DP’s inference latency rises, limiting responsiveness and motivating acceleration for fast action generation.

### B. Efficient Diffusion Inference

To address the high inference cost of diffusion models, several acceleration strategies have been developed in image

and video generation. These can be grouped into three categories—each with their core techniques and limitations when applied to DP: (1) **Denoised-data compression** methods such as difference-based reduction [3], low-dimensional data compression [21], [22], efficient quantization [23], and structured pruning [24], [25], aim to compress the heavy denoised outputs but are less effective for DP where multi-modal condition input dominates the overhead. (2) **Parallelization and throughput-oriented** methods employ mixed-precision with dynamic resource allocation [26], [27] or task scheduling for multi-batch sampling [28], but fails to meet DP’s single-batch, real-time latency demands despite its effectiveness in high-throughput settings. (3) **Structural modification** methods reduce computation via step distillation [7], [8], classifier-guided denoising [14], [29], and architectural overhauls like Mixture-of-Experts with dynamic subnetworks [9], [10], but require substantial retraining or redesign, hindering rapid deployment in data-intensive robotics scenarios.

Beyond algorithms, hardware acceleration has been used to improve diffusion inference efficiency. Typical strategies include hardware adaptations for efficient algorithms, such as multi-core array designs for speculative parallelism [30] or difference-based recomputation with on-chip region detection [31]. Other works emphasize microarchitectural and dataflow optimizations, leveraging techniques such as multi-precision CIM units [32], sparse LUT-based computation [33], and data reordering [34], alongside flexibility layer fusion and optimized memory access [35]. However, these approaches do not specifically target efficient compatibility across multiple operators in DP, nor do they fully exploit sparsity present in multi-modal processing data.

### III. ALGORITHMIC OPTIMIZATION

#### A. Self-guided Feature Reuse Mechanism

As noted earlier, consecutive input frames and their features exhibit high temporal similarity, with differences confined to regions affected by shifts in object positions. To exploit this similarity for computational sparsity, we need to predict regions in the current frame features that are similar to the previous frame before each inference step. So we propose a Self-guided Feature Reuse Mechanism (SFR) to restructure the vision encoder, implemented via two dedicated modules. Firstly, we introduce a Receptive Field Propagation (RF Prop) method that models how similarity propagates across layers in the encoder. Specifically, we compare the current and previous image inputs to obtain an initial pixel-wise similarity map, which is then propagated through RF Prop layers that expand receptive fields and refine similarity estimates. As shown in Fig.2, each RF Prop layer consists of a 2D convolution with a customized kernel followed by an average pooling unit. The kernel adopts a Gaussian distribution with layer-dependent scales to emulate the increasing receptive field size in deeper layers, as the Gaussian weights  $\hat{K}_{i,j}$  naturally captures the spatial propagation characteristics of receptive fields (see Eq.1). Experiments show that the predicted similarity maps are slightly conservative compared to the ground truth but sufficiently sparse for effective

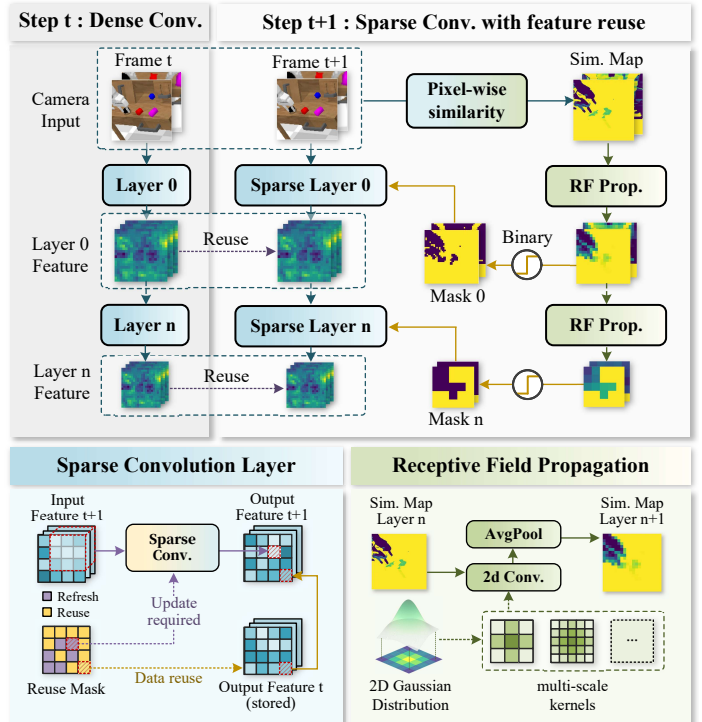


Fig. 2. Architecture of the Self-guided Feature Reuse Mechanism, with details of the sparse layer and Receptive Field Propagation.

feature reuse. Notably, the RF Prop convolution only needs to be applied once per block at each scale, and its input/output dimensions are much smaller than the main feature maps, making its computational overhead negligible.

$$\hat{K}_{i,j} = \text{Norm} \left( \exp \left( - \frac{(i-c)^2 + (j-c)^2}{c^2 \sigma^2} \right) \right), \quad (1)$$

where  $i, j = 0, \dots, k-1$ ;  $c = \frac{k-1}{2}$

After obtaining similarity maps for each layer, we binarize them to generate masks marking pixels for recomputation. Sparse convolution is applied only to these pixels, while other regions reuse previous-frame features, thereby greatly reducing computation. During implementation, to prevent the accumulation of errors from prolonged feature reuse, we perform a fully dense computation of the encoder every fixed number of inference steps, effectively resetting the error.

#### B. Cross-guided Token Pruning Strategy

Critical visual regions differ across tasks, and relying solely on visual features makes it difficult to separate informative areas from redundant background tokens. To address this, we introduce a Cross-guided Token Pruning Strategy (CTP), which leverages task-specific textual prompts to highlight aligned visual regions and prune low-importance tokens. As shown in Fig.3, given  $N_v$  visual tokens  $\mathbf{V} = v_1, v_2, \dots, v_{N_v}$  from the encoder and  $N_t$  textual tokens  $\mathbf{T} = t_1, t_2, \dots, t_{N_t}$ , we employ a lightweight scoring network to estimate each visual token’s task-specific importance. Concretely, we compute cross-attention scores between the two modalities using multi-head attention ( $H$  heads), with each head designed to capture

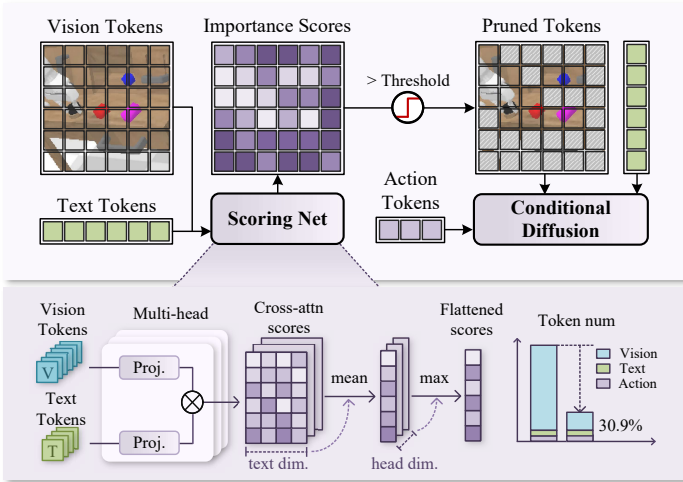


Fig. 3. Visualization of the Cross-guided Token Pruning Strategy, illustrating the pruning process and the scoring net used to evaluate cross-modal scores.

distinct importance cues. We average attention scores over the text token dimension  $N_t$  to obtain each visual token’s task relevance, then take the maximum across heads  $H$  to form a length- $N_v$  importance vector for vision tokens. A learnable thresholding module is applied to select only the most informative tokens, and the resulting compact token set will replace the full set in the conditional diffusion module. Experiments show that CTP reduces the number of visual tokens to as low as 22%, compressing the overall token count to about 31% of its original size.

$$\mathcal{L}_{\text{prune}} = d \cdot \left( \frac{d - 2d^*}{1 - 2d^*} \right), \quad d = \frac{1}{N} \sum_{i=1}^N \text{Sigmoid} \left( \frac{s_i - \theta}{\tau} \right) \quad (2)$$

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{policy}} + \lambda \times \mathcal{L}_{\text{prune}} \quad (3)$$

To enable trainability, we implement a differentiable approximation for the token pruning operation, allowing gradients to flow through the reduced token set. Compared to directly injecting binary masks into the transformer blocks—which masks out pruned tokens but still involves them in attention computations—our approach reduces the number of active tokens, yielding substantial GPU acceleration during both large-scale parallel training and evaluation. To achieve controllable pruning rates, we introduce a pruning loss for CTP. As defined in Eq. 2 and Eq. 3, the scoring network outputs importance scores  $s$ , compared with threshold  $\theta$  and passed through a sigmoid to produce an approximate pruning mask. Averaging the mask gives token density  $d$ . We design the pruning loss as a convex penalty on the token density  $d$ , minimized at the target density  $d^*$ . The resulting  $\mathcal{L}_{\text{prune}}$ , weighted by  $\lambda$ , is then added to the policy loss  $\mathcal{L}_{\text{policy}}$  to form the final objective.

#### IV. HARDWARE ARCHITECTURE

To fully exploit the potential of algorithmic optimizations and accelerate DP inference, we propose a novel accelerator architecture. An overview is shown in Fig.4. Our design targets efficient support for both convolution in the vision encoder and attention in the conditional diffusion, while also adapting to

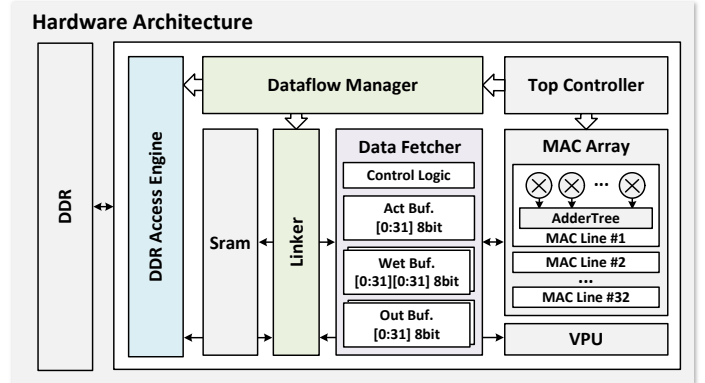


Fig. 4. Overview of our hardware architecture.

changes introduced by algorithm-level optimizations. Although both operators can be reduced to general matrix multiplications (GEMMs), their dataflow patterns and dimensional properties lead to substantially different arithmetic intensity (224 FLOPs/Byte for convolution and 32.8 FLOPs/Byte for attention, assuming 8-bit precision for both weights and activations, which is implemented in our deployment). An unoptimized architecture suffers either from up to 85% of DDR bandwidth waste or low PE utilization, which becomes a central challenge to address. To overcome this, we propose an architecture consisting of a PE array, a Vector Processing Unit (VPU), and a data management subsystem including the Data Fetcher, Dataflow Manager, and DDR Access Engine.

**PE Array.** As GEMMs dominate the workload, we adopt a regular  $M \times N$  MAC-line structure to balance throughput and energy efficiency. The PE array processes data in slices along the input and output channel dimensions, maximizing weight reuse and facilitating high-throughput pixel-wise sparse convolution. This design enables compact utilization of each MAC cell. Furthermore, token pruning is performed only once, after which the remaining tokens are stored in a compact, contiguous layout, eliminating the need for additional scheduling.

**Vector Processing Unit (VPU).** The VPU handles element-wise vector operations such as addition, subtraction, and multiplication. It also implements nonlinear activation functions (e.g., SiLU and ReLU) using piecewise-linear approximations. To efficiently support attention operators, we integrate a resource-efficient softmax unit based on an approximate computation method [36], achieving accuracy verified to be fully compatible with model requirements.

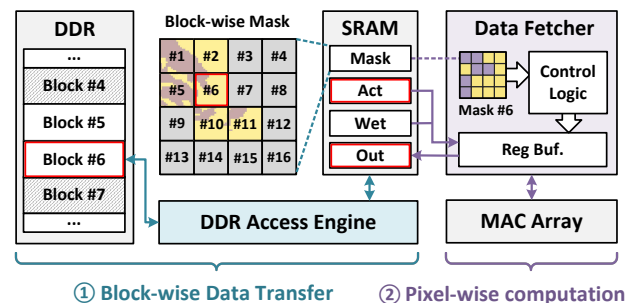


Fig. 5. Workflow of the DDR Access Engine and Data Fetcher in sparse convolution mode

**Data Fetcher and DDR Access Engine.** The Data Fetcher bridges on-chip SRAM and the PE array, streaming activations and weights and writing back outputs. Two operating modes are provided: (1) a regular mode optimized for structured GEMM in attention, and (2) a sparse mode tailored to pixel-level sparse convolution, where indices are derived from mask blocks. To match SRAM bandwidth with PE throughput, dual-register ping-pong buffering is employed for weights and outputs.

The DDR Access Engine, built on AXI bus interface, also supports both dense and sparse modes. As shown in Fig.5, the mask exhibits strong spatial locality from inter-frame similarity, with pixels requiring recomputation tending to cluster spatially. To exploit this, convolutional features are partitioned and stored in DDR at block granularity; only selected blocks are fetched, processed, and written back. In practice, we observe that some blocks, though not completely empty, are sparse enough to allow safe whole-block reuse without accuracy loss, greatly reducing memory traffic.

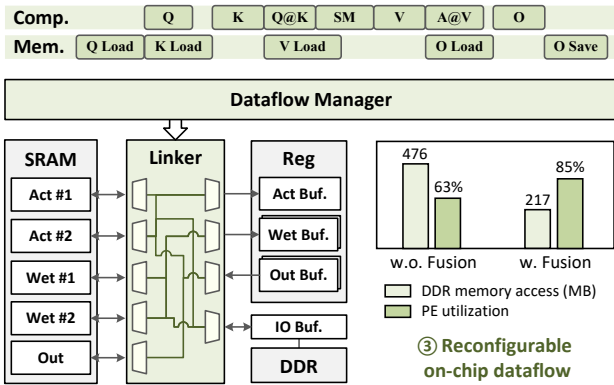


Fig. 6. Reconfigurable dataflow and operator scheduling in Attention mode (where comp. denotes computation and mem. denotes memory access), showing the collaboration between the Dataflow Manager and Linker, along with a comparison of DDR access and PE utilization before and after optimization.

**Dataflow Manager.** Convolution and attention layers show distinct patterns: convolution has rapidly growing channel dimensions (with peak weights at the MB level per layer), while attention causes regular but frequent off-chip traffic, limiting PE utilization. To address this, we instantiate additional SRAM for activations and weights. In convolution mode, extra SRAM acts as a scalable weight buffer to reduce off-chip transfers; in attention mode, it supports operator fusion and on-chip data reuse, balancing the compute-to-memory ratio and reducing DDR access.

As shown in Fig.6, a reconfigurable on-chip dataflow scheduler is implemented in the Dataflow Manager. The Linker module serves as a flexible interconnect among SRAM, registers, and DDR. Controlled by select signals, the Linker dynamically reconfigures into different physical connection networks for various compute and memory states. Combined with operator fusion, this reduces memory traffic by 54.5% and achieves a 1.35 $\times$  improvement in PE utilization for conditional diffusion.

**Other Design Details.** All computation units adopt 8-bit fixed-point format, enabled by verified quantization of the target model. The PE array consists of 32 lanes of 32-element MAC

lines, while the VPU supports 32-element vector operations. On-chip SRAM provides a total capacity of 464 KB, with each buffer instantiated as two banks to achieve a 256-bit width. Off-chip memory employs LPDDR4-2400, delivering 38.4 GB/s bandwidth—significantly lower than the 51–102 GB/s available on the Jetson Orin Nano edge GPU.

## V. EVALUATION

### A. Accuracy Evaluation

To evaluate the impact of our algorithmic optimizations on DP, we use 3D Diffusion Actor (3DDA) [6] as the baseline. The model can run as either a standalone policy or a downstream component in a dual-system setup, and its strong performance and extensibility make it well-suited for verification. We adopt the default configurations and pre-trained weights of both 3DDA and OpenHelix [19]. For benchmarking, we use the CALVIN ABC-D dataset [37], which offers diverse multi-task manipulation sequences with high-quality annotations and serves as a standard benchmark for policy learning.

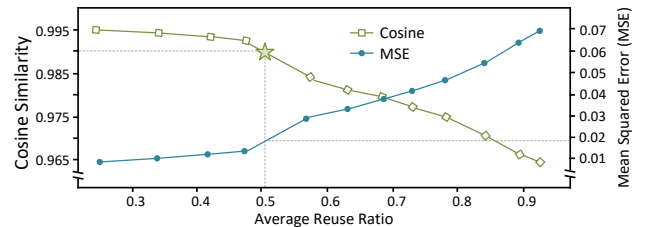


Fig. 7. Correlation between Reused Encoder-generated Visual Tokens and Ground Truth Tokens with respect to the Average Reuse Ratio,

We first evaluate the Self-guided Feature Reuse Mechanism integrated into 3DDA. This mechanism is controlled by a threshold on the binarized mask that determines the pixel reuse ratio in the vision encoder. Fig.7 shows visual token correlations at different reuse ratios with ground-truth tokens, measured by cosine similarity and mean squared error (MSE) to capture feature direction and value differences. The reuse ratio is defined as the weighted average of multi-layer masks over pixels. Results show that similarity remains high and changes slowly until the ratio reaches 50%, after which it drops sharply. In practical applications, we set the threshold at 52% with a cosine similarity of 0.990, and find no accuracy loss after fine-tuning.

TABLE I  
EVALUATION IN CALVIN ABC-D ENVIRONMENT.

Method	Task completed in a row (%) $\uparrow$					Avg. Len $\uparrow$
	1	2	3	4	5	
3DDA (baseline)	90	74	58	40	31	2.93
3DDA (Ours-22%)	92	78	57	41	32	3.00
Open-Helix (baseline)	93	79	64	52	42	3.31
Open-Helix (Ours-25%)	92	81	65	52	39	3.28

Next, we evaluate the Task-driven Token Pruning Strategy by varying the hyperparameter  $\lambda$  in Eq.3 to control the final pruning ratio. Our training strategy enables convergence with only 1/60 of the original epochs. Following the evaluation methodology in Open-Helix, for rapid assessment, we select the first 100 trials from the CALVIN dataset for repeated testing and report the averaged results. Performance is assessed in

TABLE II  
AREA AND POWER BREAKDOWN OF ON-CHIP MODULES

	PE Array	VPU	Control Logic	SRAM	Overall
Area ( $mm^2$ )	0.22	0.04	0.02	0.81	1.09
Power (mW)	124.4	10.9	6.7	207.9	349.9

both standalone and dual-system configurations, with results summarized in Table 1. We progressively compress the token size on both models through incremental testing, achieving a final compression of 22% for 3DDA and 25% for Open-Helix, with almost no accuracy degradation observed. Furthermore, we applied W8A8 quantization-aware training (QAT) to the final 3DDA model, enabling 8-bit deployment without any loss of accuracy compared to the original model.

### B. Area, Power, and Performance Evaluation

Our design is implemented in SystemVerilog and synthesized at 1 GHz using Synopsys Design Compiler and TSMC 28nm standard-cell library for area and power analysis. Off-chip memory performance is evaluated with DramSim3 [38] under an LPDDR4 configuration, reaching 36.0 GB/s on average across test cases, close to the 38.4 GB/s theoretical peak. Rarely invoked operators with poor hardware affinity are simulated on an edge CPU. Table 2 presents the area and power breakdown of our chip architecture. We further evaluate the average on-chip power consumption, which is 206.5 mW, while the average LPDDR power is 1262.4 mW, and total system power is about 1.5 W, far below the 7–15 W of Jetson Orin Nano.

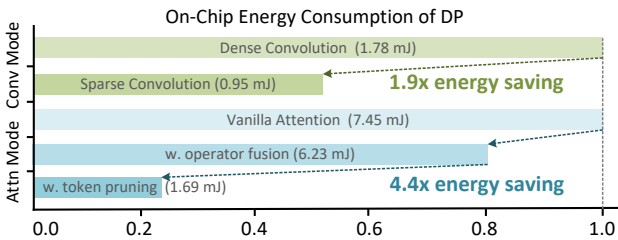


Fig. 8. Comparison of on-chip energy consumption in Convolution (Conv) and Attention (Attn) modes before and after optimization

To assess end-to-end performance, we develop a custom cycle-accurate simulator. We first analyze the computational patterns under both convolution and attention operators. Fig.8 illustrates the on-chip energy consumption for both conv and attn modes. After optimization, we achieve energy savings of 1.9 $\times$  for the conv mode and 4.4 $\times$  for the attn mode. We further tested the metrics of running 3DDA end-to-end. As shown in Fig.9, through our hardware-software co-optimization, the computational density ratio between the two main operators was balanced from 224/32.8 to 107/80.4. This allowed us to use relatively lower DRAM bandwidth while achieving higher PE utilization. The on-chip end-to-end latency achieved a 3.6 $\times$  speedup compared to the under-optimized architecture. For the accelerator system, the energy consumption per inference for a single trajectory was reduced by 2.8 $\times$ .

We also test inference on an NVIDIA RTX 3090 GPU using the CUDA Event API with native Python, and the results show a latency of 580 ms. In contrast, running the optimized 3DDA on our accelerator yields an end-to-end latency of 22.3 ms

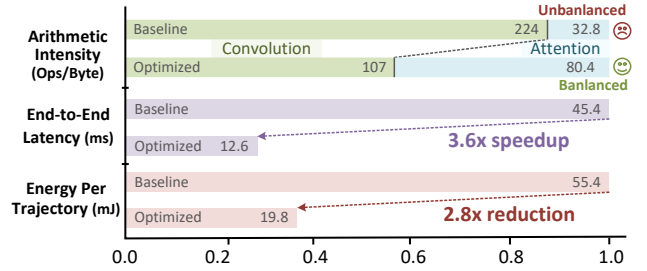


Fig. 9. Comparison of arithmetic intensity, end-to-end latency, and system energy consumption for 3DDA deployment before and after optimization.

(including 12.7 ms on our chip and 9.6 ms on edge CPU), corresponding to a response rate of 45 Hz. This demonstrates a 26 $\times$  speedup over the GPU baseline. Table III presents the hardware implementation results of our proposed architecture compared to existing Diffusion accelerators. It is evident that our architecture efficiently reduces on-chip area through the compatibility of data flows for both operators and high utilization of on-chip caches, while combining quantization techniques. With reuse and pruning considerations, our approach achieves higher sparse energy efficiency.

TABLE III  
COMPARISON OF DIFFUSION ACCELERATORS

Methods	ESSERC'24 [39]	ISSCC'25 [35]	VLSI'25 [30]	This Work
Application	Visual Generation	Generative AI	Robot Control	Robot Control
Support Model	CNN, DiT	CNN	CNN, DiT	CNN, DiT
Technology	28nm	3nm	28nm	28nm
Area ( $mm^2$ )	21.9	0.168	8.41	1.09
Frequency (MHz)	25-400	546	250-1250	1000
Precision	INT8-13, HYP8	INT8	INT8/4, FP4	INT8
Peak Performance (TOPS)	9.83	0.63	2-10 <sup>1</sup>	2.0-7.2 <sup>2</sup>
Energy Efficiency (TOPS/W)	4.96	12.38	17.6 <sup>1</sup>	9.6-34.7 <sup>2</sup>

1. with 80% input activation sparsity w/ 50% weight sparsity.
2. with 52% reuse ratio and 22% pruning ratio.

## VI. CONCLUSION

In this work, we tackle the latency and energy bottlenecks of Diffusion Policy (DP) on edge robots by exploiting intra- and inter-modal redundancies in multi-modal inputs. Our hardware-software co-design combines self- and cross-guided modal compression with a reconfigurable accelerator supporting sparse memory access and lightweight computation. Experiments show a 26 $\times$  speedup over NVIDIA RTX 3090 GPU with energy efficiency up to 34.7 TOPS/W and total power of only 1.5 W, enabling real-time, low-power visuomotor control on resource-constrained platforms.

## VII. ACKNOWLEDGEMENT

This research is supported by National Key Research and Development Program of China under Grant (2024YFB4405400) and the Deng Feng Fund.

## REFERENCES

- [1] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, p. 02783649241273668, 2023.
- [2] M. Reuss, Ö. E. Yağmurlu, F. Wenzel, and R. Lioutikov, "Multimodal diffusion transformer: Learning versatile behavior from multimodal goals," *arXiv preprint arXiv:2407.05996*, 2024.
- [3] X. Ma, G. Fang, and X. Wang, "Deepcache: Accelerating diffusion models for free," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024, pp. 15 762–15 772.
- [4] Y. He, L. Liu, J. Liu, W. Wu, H. Zhou, and B. Zhuang, "Ptqd: Accurate post-training quantization for diffusion models," *arXiv preprint arXiv:2305.10657*, 2023.
- [5] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.
- [6] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki, "3d diffuser actor: Policy diffusion with 3d scene representations," *arXiv preprint arXiv:2402.10885*, 2024.
- [7] E. Luhman and T. Luhman, "Knowledge distillation in iterative generative models for improved sampling speed," *arXiv preprint arXiv:2101.02388*, 2021.
- [8] T. Salimans and J. Ho, "Progressive distillation for fast sampling of diffusion models," *arXiv preprint arXiv:2202.00512*, 2022.
- [9] Y. Ren, X. Li, B. Li, X. Wang, M. Guo, S. Zhao, L. Zhang, and Z. Chen, "Moe-diffir: Task-customized diffusion priors for universal compressed image restoration," in *European Conference on Computer Vision*. Springer, 2024, pp. 116–134.
- [10] M. Reuss, J. Pari, P. Agrawal, and R. Lioutikov, "Efficient diffusion transformer policies with mixture of expert denoisers for multitask learning," *arXiv preprint arXiv:2412.12953*, 2024.
- [11] B. Huang, Y. Wang, X. Yang, Y. Luo, and Y. Li, "3d-vitac: Learning fine-grained manipulation with visuo-tactile sensing," *arXiv preprint arXiv:2410.24091*, 2024.
- [12] F. P. Bejarano, B. Jones, D. P. Moreno, J. Bowkett, P. G. Backes, and A. P. Schoellig, "Prodapt: Proprioceptive adaptation using long-term memory diffusion," *arXiv preprint arXiv:2503.00193*, 2025.
- [13] J. Ho, A. Jain, and P. Abbeel, "Denosing diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.
- [14] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *Advances in neural information processing systems*, vol. 34, pp. 8780–8794, 2021.
- [15] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu, "Rdt-1b: a diffusion foundation model for bimanual manipulation," *arXiv preprint arXiv:2410.07864*, 2024.
- [16] Y. Wu, Z. Chen, F. Wu, L. Chen, L. Zhang, Z. Bing, A. Swikir, S. Haddadin, and A. Knoll, "Tacdiffusion: Force-domain diffusion policy for precise tactile manipulation," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 11 831–11 837.
- [17] S. Yan, Z. Zhang, M. Han, Z. Wang, Q. Xie, Z. Li, Z. Li, H. Liu, X. Wang, and S.-C. Zhu, "M 2 diffuser: Diffusion-based trajectory optimization for mobile manipulation in 3d scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.
- [18] P. M. Scheikl, N. Schreiber, C. Haas, N. Freymuth, G. Neumann, R. Lioutikov, and F. Mathis-Ullrich, "Movement primitive diffusion: Learning gentle robotic manipulation of deformable objects," *IEEE Robotics and Automation Letters*, vol. 9, no. 6, pp. 5338–5345, 2024.
- [19] C. Cui, P. Ding, W. Song, S. Bai, X. Tong, Z. Ge, R. Suo, W. Zhou, Y. Liu, B. Jia *et al.*, "Openhelix: A short survey, empirical analysis, and open-source dual-system vla model for robotic manipulation," *arXiv preprint arXiv:2505.03912*, 2025.
- [20] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang *et al.*, "Gr00t n1: An open foundation model for generalist humanoid robots," *arXiv preprint arXiv:2503.14734*, 2025.
- [21] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.
- [22] W.-C. Fan, Y.-C. Chen, D. Chen, Y. Cheng, L. Yuan, and Y.-C. F. Wang, "Frido: Feature pyramid diffusion for complex scene image synthesis," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 37, no. 1, 2023, pp. 579–587.
- [23] X. Li, Y. Liu, L. Lian, H. Yang, Z. Dong, D. Kang, S. Zhang, and K. Keutzer, "Q-diffusion: Quantizing diffusion models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 17 535–17 545.
- [24] T. Castells, H.-K. Song, B.-K. Kim, and S. Choi, "Ld-pruner: Efficient pruning of latent diffusion models using task-agnostic insights," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 821–830.
- [25] Y. Zhang, E. Jin, Y. Dong, A. Khakzar, P. Torr, J. Stegmaier, and K. Kawaguchi, "Effortless efficiency: Low-cost pruning of diffusion models," *arXiv preprint arXiv:2412.02852*, 2024.
- [26] S. Ahmad, Q. Yang, H. Wang, R. K. Sitaraman, and H. Guan, "Diff-serve: Efficiently serving text-to-image diffusion models with query-aware model scaling," *arXiv preprint arXiv:2411.15381*, 2024.
- [27] S. Agarwal, S. Iqbal, and S. Mitra, "Prompt-aware scheduling for efficient text-to-image inferencing system," *arXiv preprint arXiv:2502.06798*, 2025.
- [28] Z. Xu, Z. Tang, J. Lou, Z. Yao, X. Xie, T. Wang, Y. Wang, and W. Jia, "Eat: Qos-aware edge-collaborative aigc task scheduling via attention-guided diffusion reinforcement learning," *arXiv preprint arXiv:2507.10026*, 2025.
- [29] A. Blattmann, R. Rombach, K. Oktay, J. Müller, and B. Ommer, "Retrieval-augmented diffusion models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 15 309–15 324, 2022.
- [30] S. Zhang, X. Chen, C. Peng, H. Yang, Y. Liu, and H. Jia, "A 94hz inference and 7.4 mj/epoch fine-tune edge soc for diffusion-based robot manipulation with speculation and disturbance enhancement," in *2025 Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*. IEEE, 2025, pp. 1–3.
- [31] Y. Qin, Y. Wang, X. Yang, Z. Zhao, S. Wei, Y. Hu, and S. Yin, "A 52.01 tflops/w diffusion model processor with inter-time-step convolution-attention-redundancy elimination and bipolar floating-point multiplication," in *2024 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*. IEEE, 2024, pp. 1–2.
- [32] R. Guo, L. Wang, X. Chen, H. Sun, Z. Yue, Y. Qin, H. Han, Y. Wang, F. Tu, S. Wei *et al.*, "20.2 a 28nm 74.34 tflops/w bf16 heterogenous cim-based accelerator exploiting denoising-similarity for diffusion models," in *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 67. IEEE, 2024, pp. 362–364.
- [33] Y. Jing, J. Zhou, Y. Sun, S. He, P. Chen, R. Huang, L. Ye, and T. Jia, "37.6 a 22nm 60.81 tflops/w diffusion accelerator with bandwidth-aware memory partition and bl-segmented compute-in-memory for efficient multi-task content generation," in *2025 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 68. IEEE, 2025, pp. 1–3.
- [34] S. Kim, J. Oh, J. So, Y. Choi, S. Kim, D. Im, G. Park, and H.-J. Yoo, "23.3 edgediff: 418.4 mj/inference multi-modal few-step diffusion model accelerator with mixed-precision and reordered group quantization," in *2025 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 68. IEEE, 2025, pp. 1–3.
- [35] S.-W. Hsieh, C.-H. Yuan, M.-H. Lin, P.-Y. Tsai, Y.-Y. Nian, C.-Y. Cheng, H.-W. Chih, P.-H. Chiang, M.-H. Chiang, Y.-J. Kuo *et al.*, "23.5 mae: A 3nm 0.168 mm<sup>2</sup> 576mac mini autoencoder with line-based depth-first scheduling for generative ai in vision on edge devices," in *2025 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 68. IEEE, 2025, pp. 414–416.
- [36] M. Wang, S. Lu, D. Zhu, J. Lin, and Z. Wang, "A high-speed and low-complexity architecture for softmax function in deep learning," in *2018 IEEE Asia Pacific conference on circuits and systems (APCCAS)*. IEEE, 2018, pp. 223–226.
- [37] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard, "Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7327–7334, 2022.
- [38] S. Li, Z. Yang, D. Reddy, A. Srivastava, and B. Jacob, "Drams3: A cycle-accurate, thermal-capable dram simulator," *IEEE Computer Architecture Letters*, vol. 19, no. 2, pp. 106–109, 2020.
- [39] S. Yoo, G. Ko, S. Ham, S. Kim, Y. Chen, and J.-Y. Kim, "A 28nm 4.96 tops/w end-to-end diffusion accelerator with reconfigurable hyper-precision and unified non-matrix processing engine," in *2024 IEEE European Solid-State Electronics Research Conference (ESSERC)*. IEEE, 2024, pp. 253–256.