Rethinking in Spikes: Mitigating Hallucinations in MDLMs with Step-Aware Decoding

Zhongxing Xu[1*] **Zhonghua Wang**[1*] **Zhe Qian**[1] **Shiyan Su**[1] **Ming Hu**[1]
Xiaocheng Zou[4] **Wei Feng**[1] **Mingquan Lin**[5] **Yuyin Zhou**[3]
Yifan Peng[2] **Hamid Rezatofighi**[1] **Zongyuan Ge**[1]
[1] Monash University [2] Cornell University [3] UC Santa Cruz
[4] Northeastern University [5] University of Minnesota
https://mdlm-steprefine.github.io/

Abstract

Recent advancements in multimodal diffusion language models (MDLMs) have exhibited strong global modeling capabilities through iterative refinement. We observe that mask uncertainty follows an overall downward trend during iterative denoising, while a few denoising steps exhibit localized entropy spikes closely associated with hallucinations. We argue that anomalous steps can be identified by measuring the entropy deviation of each denoising step from its neighboring temporal window. We hypothesize that discrete token commitment discards the distributional information at masked positions, limiting the continued interaction between candidate semantics and the context during entropy spikes. With this goal, we present **StepRefine**, a plug-and-play decoding strategy that leverages semantic context to achieve reliable denoising. The core of our method lies in a step-aware switch between discrete denoising and latent refinement. The model employs probability-weighted embeddings at entropy spike steps to preserve diverse reasoning hypotheses and performs multiple rounds of continuous refinement. Meanwhile, a refinement controller monitors distributional stability to reduce unnecessary loops and switch the model back to standard discrete denoising. Through extensive experiments, StepRefine demonstrates significant hallucination mitigation across different MDLMs on multimodal benchmarks.

1 Introduction

Diffusion language models [40, 98, 19] replace autoregressive left-to-right generation with iterative denoising, forming an alternative paradigm for sequence generation. Current diffusion language models typically adopt a discrete masked diffusion mechanism [54, 90, 105, 58], in which multiple masked positions are predicted in parallel and progressively committed to discrete tokens. This paradigm has recently been extended to multimodal settings. Multimodal diffusion language models (MDLMs) [92, 93, 39, 89, 65] leverage bidirectional attention over visual and textual inputs to model global context, demonstrating strong potential in perception and reasoning tasks.

Despite these advantages, MDLMs remain susceptible to hallucination. Existing inference-time methods for diffusion language models primarily focus on efficiency, such as cache reuse [27, 79, 107] and denoising scheduling [46, 43, 29], but these methods typically do not improve sequence reliability. Other decoding strategies attempt to improve quality by modifying token commitment policies, for example, by re-masking unreliable tokens [22, 25] or introducing continuous token representations [103, 21, 10]. However, these methods typically treat uncertainty uniformly across denoising steps and do not explicitly account for temporal irregularities in the denoising trajectory.

*Equal contribution. Correspondence: {zhongxing.xu, zhonghua.wang, zongyuan.ge}@monash.edu

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

In this paper, we examine hallucination in MDLMs from the perspective of step-wise uncertainty dynamics across denoising steps. A natural expectation is that predictive uncertainty should steadily decrease as the sequence becomes progressively less masked and the context becomes more complete. Our analysis shows that the average entropy of the masked sequence indeed decreases. However, this decrease is not monotonic. A small number of steps show sharp entropy increases relative to their local temporal window. We refer to these steps as entropy spikes.

Through statistical analysis, entropy spike steps are rare but highly correlated with hallucinated tokens in the final response. We observe that entropy spike steps exhibit a non-uniform distribution along the denoising trajectory, mainly occurring in the early-to-middle stages. At these stages, a large portion of the sequence remains masked, leaving the context information still limited. Moreover, we find that replacing standard confidence-based decoding with random decoding at entropy spike steps causes little performance change, whereas applying the same perturbation at ordinary steps results in clear degradation. At entropy spike steps, confidence signals may become less discriminative, making discrete decoding less reliably grounded in the current context. Therefore, our findings suggest that identifying entropy spike steps and applying contextual refinement before discrete commitment is crucial to mitigating hallucinations.

In this work, we propose StepRefine, a plug-and-play decoding strategy that leverages dense contextual cues to improve denoising reliability. StepRefine adaptively switches between standard discrete denoising and latent probabilistic refinement based on the entropy dynamics of each step from its local temporal window. Specifically, StepRefine computes the average mask entropy at each denoising step and detects entropy spikes by comparing the current entropy change against trends within a local window. When such spikes are detected, the model switches to a probabilistic refinement mode, constructs probability-weighted token embeddings that implicitly preserve diverse hypotheses, and promotes sufficient interaction between candidate semantics and the context through iterative refinement. In addition, a refinement controller monitors distributional stability to reduce unnecessary iterations and switch the model back to standard discrete denoising when appropriate.

Extensive experiments show that StepRefine significantly reduces hallucinations across multiple MDLM models on diverse reasoning benchmarks, demonstrating its effectiveness. Our main contributions are as follows:

- We analyze the relationship between entropy spike steps and hallucinations in multimodal diffusion models from the perspective of uncertainty dynamics across denoising steps.
- We propose StepRefine, a plug-and-play decoding approach that mitigates hallucinations through entropy spike detection and step-aware mode switching.
- Extensive evaluations across four MDLM backbones and diverse multimodal benchmarks show the superior performance of StepRefine, offering an effective solution for multimodal diffusion hallucinations.

2 Preliminary

An MDLM takes an image and a textual prompt as input and generates text through iterative denoising. Starting from a sequence of all mask tokens, the model predicts candidate tokens for all masked positions in parallel at each step. A remasking strategy then commits a subset of these candidates as final tokens while resetting the rest to masked for the next step, until all positions are finalized. The raw image is encoded and projected into a sequence of vision tokens $\mathbf{v}$, while the textual prompt is tokenized into a sequence of text tokens $\mathbf{q}$. The generation sequence is initialized as $\mathbf{x}_T = [\mathrm{MASK}]^L$, where L is a predefined generation budget, $[\mathrm{MASK}]$ is a special mask token from the vocabulary $\mathcal{V}$, and T is the total number of denoising steps. At denoising step $t \in \{T, T-1, \ldots, 1\}$, the complete multimodal input sequence is $\mathbf{s}_t = \mathbf{v} \oplus \mathbf{q} \oplus \mathbf{x}_t$, where $\mathbf{x}_t = (x_t^1, x_t^2, \ldots, x_t^L)$ is the generation sequence at step t and x_t^i is the token at position i.

Discrete Decoding. Let $\mathcal{M}_t = \{i \mid x_t^i = [\mathrm{MASK}]\}$ denote the set of masked positions at step t. At each denoising step, the MDLM D_θ takes $\mathbf{s}_t$ as input and predicts per-position token distributions over all positions in parallel. Discrete greedy decoding then selects the most likely tokens for each masked position:

$$\mathbf{p}_t = D_\theta(\mathbf{s}_t) \in \Delta(\mathcal{V})^L, \qquad \hat{x}_t^i = \arg\max_{v \in \mathcal{V}} \mathbf{p}_t^i(v), \quad i \in \mathcal{M}_t, \tag{1}$$

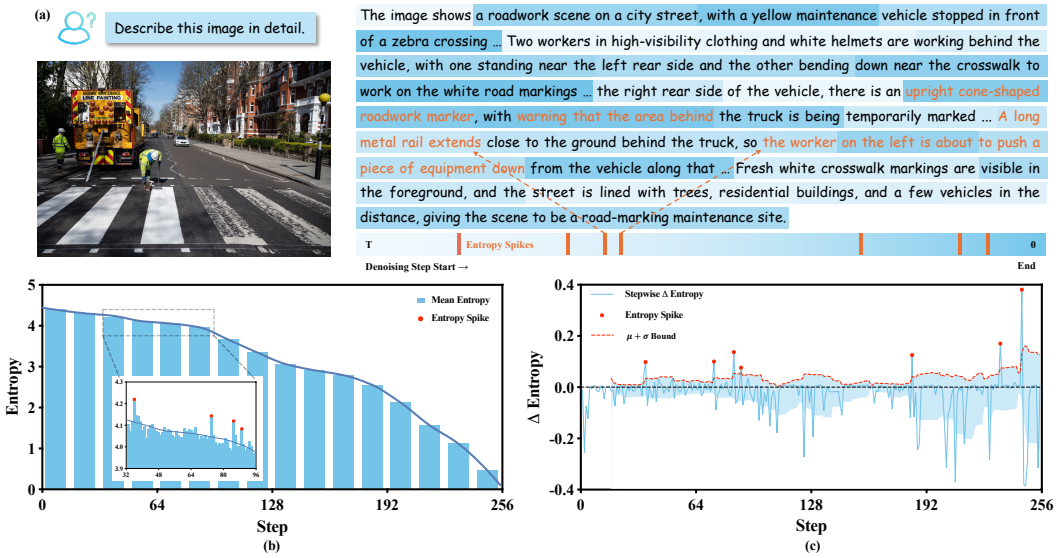


Figure 1: Entropy spikes in denoising dynamics. (a) A representative example illustrating the correlation between entropy spikes and hallucinations. Tokens committed at steps with an entropy spike closely correspond to hallucinated content in the response. (b) Mean mask entropy decreases steadily across denoising, while a few steps show sharp rises that break the local trend, which we refer to as entropy spikes. (c) A step is flagged as an entropy spike when its stepwise entropy change exceeds the $\mu + \sigma$ bound from the calibration window.

where $\Delta(\mathcal{V})^L$ denotes a sequence of L probability simplices over $\mathcal{V}$, and $\mathbf{p}_t^i$ is the prediction distribution at position i. Discrete decoding produces a candidate token $\hat{x}_t^i$ at each masked position, while the remasking step determines which candidates are actually committed.

Latent Refinement. Let $\mathbf{e}(v) \in \mathbb{R}^d$ denote the embedding of token v, where d is the embedding dimension. Discrete decoding collapses the predictive distribution $\mathbf{p}_t^i$ into a single token, discarding its distributional information. Latent refinement instead converts the prediction distribution into a probability-weighted embedding and further refines it in the continuous representation space:

$$\tilde{\mathbf{e}}_t^i = \mathbb{E}_{v \sim \mathbf{p}_t^i}\left[\mathbf{e}(v)\right] \in \mathbb{R}^d, \quad i \in \mathcal{M}_t, \tag{2}$$

where $\mathbb{E}$ denotes the expectation under the distribution $\mathbf{p}_t^i$. The resulting soft embedding $\tilde{\mathbf{e}}_t^i$ replaces the $[\mathrm{MASK}]$ embedding at position i in the next forward pass, allowing candidate semantics to further interact with the context before discrete token decoding.

Remasking. Although discrete decoding produces candidate tokens for all masked positions at each step, only a subset of $\hat{x}_t^i$ is committed into the generation sequence. A remasking strategy selects an update set $\mathcal{U}_t \subseteq \mathcal{M}_t$, where the chosen positions are committed and the rest stay masked. Common choices include random sampling that selects positions uniformly, low-confidence remasking that commits the highest-confidence positions according to a step-wise schedule on $|\mathcal{U}_t|$, and top-k marginal selection that commits a fixed number of k positions with the largest marginal probability per step. The generation sequence is then updated by committing selected positions while leaving the others unchanged:

$$x_{t-1}^i = \begin{cases} \hat{x}_t^i, & i \in \mathcal{U}_t, \\ x_t^i, & i \notin \mathcal{U}_t. \end{cases} \tag{3}$$

The masked-position set is accordingly updated as $\mathcal{M}_{t-1} = \mathcal{M}_t \setminus \mathcal{U}_t$. Through repeated prediction and remasking, the model progressively refines an initially fully masked sequence $\mathbf{x}_T$ into a fully decoded sequence $\mathbf{x}_0$.

3 Methodology

We propose StepRefine, a step-aware decoding strategy for mitigating hallucinations in MDLMs. Section 3.1 presents entropy spike detection, along with an analysis of entropy spikes. Section 3.2

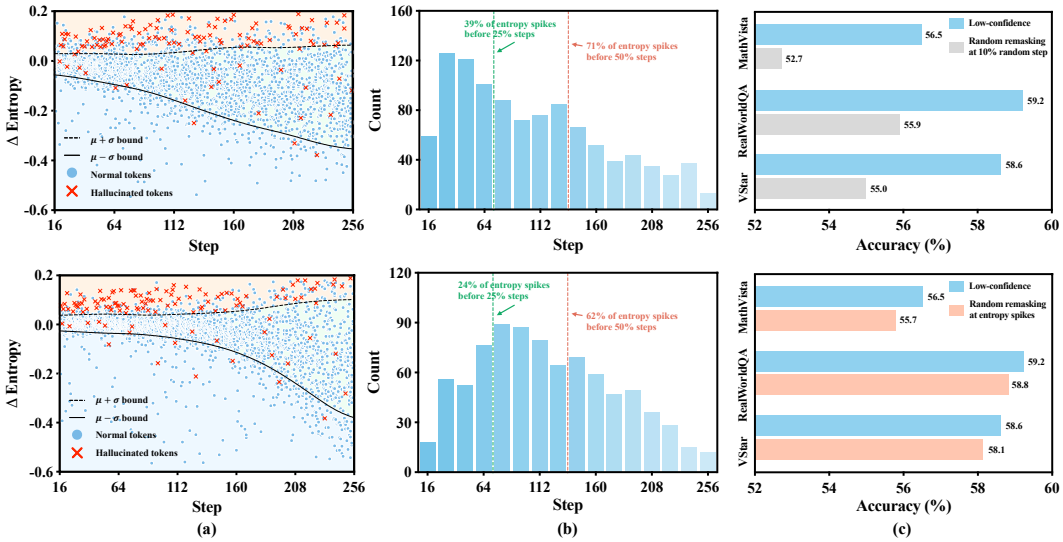


Figure 2: Entropy spikes characterization across decoding steps. (a) Token-level scatter of stepwise entropy change on LLaDA-V (top) and Dream-VL (bottom). Roughly 80% of hallucinated tokens fall in the area above $\mu + \sigma$. (b) Distribution of entropy spikes across denoising steps on LLaDA-V (top) and Dream-VL (bottom). On both models, entropy spikes mostly occur in the early-to-middle decoding stage, when few tokens have been committed and the available context is still limited. (c) Effect of random remasking on performance using LLaDA-V. Top: replacing low-confidence remasking with random remasking at 10% of randomly selected steps causes a substantial drop on three datasets. Bottom: applying the same random remasking only at entropy spike steps yields negligible change. At entropy spike steps, confidence signals may become less discriminative, making discrete commitment decisions less reliably grounded in the current contextual information.

introduces step-aware refinement switching with adaptive termination. For clarity, Algorithm 1 provides the pseudocode for the decoding process of StepRefine.

3.1 Entropy Spikes Detection

During iterative denoising, the mean mask entropy over masked positions decreases as the model gradually commits tokens (Fig. 1b). However, the decrease is not monotonic. A small number of steps exhibit sharp local increases that break the trend, which we refer to as entropy spikes. We measure this entropy at each denoising step t, defined as the expected Shannon entropy of $\mathbf{p}_t^i$ over masked positions:

$$\bar{H}_t = \mathbb{E}_{i\sim\mathbf{U}(\mathcal{M}_t)}\left[-\sum_{v\in\mathcal{V}} \mathbf{p}_t^i(v)\log \mathbf{p}_t^i(v)\right], \tag{4}$$

where $\mathbf{U}(\mathcal{M}_t)$ denotes the uniform distribution over the masked positions $\mathcal{M}_t$. A step is then classified as an entropy spike if its entropy change exceeds the local upper bound:

$$\Delta\bar{H}_t \triangleq \bar{H}_t - \bar{H}_{t+1}, \qquad \mathcal{T}_{\text{spike}} \triangleq \left\{\, t \mid \Delta\bar{H}_t > \mu_t + \sigma_t \,\right\}, \tag{5}$$

where μ_t and σ_t are the mean and standard deviation of the w preceding stepwise entropy changes along the denoising trajectory, computed over $\{\Delta\bar{H}_{t+1}, \ldots, \Delta\bar{H}_{t+w}\}$. As shown in Fig. 1c, this criterion captures steps where mean mask entropy deviates sharply from the local denoising trend.

Entropy Spike Analysis. Figure 2(a) demonstrates that entropy spike steps are relatively rare along the denoising trajectory, while approximately 80% of hallucinated tokens are located within the entropy spike region. Fig. 2(b) further shows the distribution of entropy spikes across the denoising process. Entropy spikes are mainly concentrated in the early-to-middle stages: about 71% occur within the first 50% of steps in LLaDA-V, while about 62% occur within the first 50% of steps in Dream-VL. This suggests that entropy spikes are non-uniformly distributed and are more likely to occur when many positions remain masked and the available context is still insufficient. Low-confidence remasking is an effective deterministic strategy [54], which preferentially preserves

Algorithm 1 StepRefine

Require: MDLM D_θ, embedding matrix $\mathbf{W}_{\text{emb}}$, denoising budget T, generation length L, window size w, stability threshold ϵ, prompt $\mathbf{x}_{\text{prompt}}$

Initialize $\mathbf{x}_T \leftarrow \text{concat}(\mathbf{x}_{\text{prompt}}, [\text{MASK}]^L)$, $\mathbf{E}_T \leftarrow \text{Embed}(\mathbf{x}_T)$, $t \leftarrow T$
while $t \geq 1$ **do**
  $\mathbf{p}_t \leftarrow D_\theta(\mathbf{E}_t)$
  Compute $\bar{H}_t$, $\Delta\bar{H}_t$, μ_t, σ_t, $k \leftarrow 0$ ▷ forward pass + spike detection
  if $\Delta\bar{H}_t > \mu_t + \sigma_t$ **then** ▷ $t \in \mathcal{T}_{\text{spike}}$
    $\tilde{\mathbf{p}}_{t,0} \leftarrow \mathbf{p}_t$
    repeat
      $k \leftarrow k+1$, $\mathbf{E}_t[\mathcal{M}_t] \leftarrow \tilde{\mathbf{p}}_{t,k-1}[\mathcal{M}_t]\,\mathbf{W}_{\text{emb}}$, $\tilde{\mathbf{p}}_{t,k} \leftarrow D_\theta(\mathbf{E}_t)$ ▷ StepRefine iteration
    until $\overline{D}_{\text{KL}}^{(k)} < \epsilon$ ▷ exit on distributional stability
    $\mathbf{p}_t \leftarrow \tilde{\mathbf{p}}_{t,k}$
  end if
  Decode $\hat{x}_t^i \leftarrow \arg\max_{v\in\mathcal{V}} \mathbf{p}_t^i(v)$ for $i \in \mathcal{M}_t$, select $\mathcal{U}_t \subseteq \mathcal{M}_t$
  Update $\mathbf{x}_{t-1}$, $\mathbf{x}_{t-1-k} \leftarrow \mathbf{x}_{t-1}$, $t \leftarrow t-1-k$, $\mathbf{E}_t \leftarrow \text{Embed}(\mathbf{x}_t)$ ▷ commit, then skip k indices
end while
return $\mathbf{x}_0$

high-confidence tokens. Fig. 2(c) shows that applying the same random-remasking perturbation causes a clear performance drop at randomly selected 10% steps, but only a minor change at entropy spike steps. At entropy spike steps, confidence signals may become less discriminative, making discrete commitment decisions less reliably grounded in the current contextual information.

3.2 Step-Aware Mode Switch

StepRefine adopts a step-aware switching strategy that alternates between standard discrete decoding and latent refinement, gated by the spike detection criterion (Eq. 5). At each denoising step t, it first performs a standard forward pass with $\mathbf{e}([\text{MASK}])$ as input at masked positions to obtain the predictive distribution $\mathbf{p}_t^i$ for $i \in \mathcal{M}_t$ and the corresponding spike indicator. At ordinary steps $t \notin \mathcal{T}_{\text{spike}}$, it follows the standard discrete decoding path. At entropy spike steps $t \in \mathcal{T}_{\text{spike}}$, StepRefine activates latent refinement, which performs iterative forward passes within the same denoising step before discrete decoding. Let $\tilde{\mathbf{p}}_{t,0}^i \triangleq \mathbf{p}_t^i$ denote the initial distribution from the standard forward pass. At each refinement iteration $k \geq 1$, StepRefine constructs a probability-weighted embedding $\tilde{\mathbf{e}}_{t,k}^i$ from the previous distribution $\tilde{\mathbf{p}}_{t,k-1}^i$ and replaces the $[\text{MASK}]$ embedding as input to the next forward pass, which produces an updated distribution $\tilde{\mathbf{p}}_{t,k}^i$. Formally,

$$\tilde{\mathbf{e}}_{t,k}^i = \mathbb{E}_{v\sim\tilde{\mathbf{p}}_{t,k-1}^i}[\mathbf{e}(v)], \quad i \in \mathcal{M}_t,\ k = 1, 2, \ldots. \tag{6}$$

Together with the forward pass, Eq. 6 forms a recursive refinement loop between $\tilde{\mathbf{p}}_{t,k}^i$ and $\tilde{\mathbf{e}}_{t,k}^i$ at each masked position. It carries distributional information about candidate tokens into subsequent forward passes, letting the soft embedding interact with the visual-textual context before discrete decoding.

To prevent unnecessary refinement, the refinement controller monitors the change between consecutive predictive distributions. At iteration $k \geq 1$, we compute the change as the average Kullback-Leibler (KL) divergence ($D_{\text{KL}}(\cdot \,\|\, \cdot)$) at masked positions:

$$\overline{D}_{\text{KL}}^{(k)} = \mathbb{E}_{i\sim\mathbf{U}(\mathcal{M}_t)}\left[\, D_{\text{KL}}\big(\tilde{\mathbf{p}}_{t,k}^i \,\|\, \tilde{\mathbf{p}}_{t,k-1}^i\big)\,\right]. \tag{7}$$

The iteration terminates once $\overline{D}_{\text{KL}}^{(k)} < \epsilon$, where $\epsilon > 0$ is a stability threshold. The final refined distribution $\tilde{\mathbf{p}}_{t,k}^i$ is then used for standard discrete decoding and remasking to commit tokens and obtain $\mathbf{x}_{t-1}$. To maintain a fixed inference budget, the additional forward passes for refinement are counted against the total denoising budget of T. If k refinement passes are used at step t, the next iteration begins at index $t-1-k$ rather than $t-1$. Thus, the total number of forward passes remains the same.

Table 1: Benchmark results of StepRefine with different MDLMs across four benchmark categories. Scores are reported for MMHalu (0–6) and Bingo (1–5); accuracy is reported for all other benchmarks.

Benchmark	LLaDA-V-8B					LLaDA-O-8B		Dream-VL-7B		LaViDA-L-8B	
	Base	CoTA	ReMix	WINO	**StepRefine**	Base	**StepRefine**	Base	**StepRefine**	Base	**StepRefine**
General Reasoning & Understanding											
VStar	55.5	57.1	56.5	56.5	**58.6** (+3.1)	57.1	**58.1** (+1.0)	69.1	**72.8** (+3.7)	48.7	**51.3** (+2.6)
MMVP	40.7	42.7	43.3	44.0	**45.3** (+4.6)	44.7	**47.3** (+2.6)	32.0	**35.3** (+3.3)	42.3	**46.7** (+4.4)
MMEval-Pro	68.4	67.9	63.5	68.5	**71.6** (+3.2)	57.9	**60.8** (+2.9)	68.5	**70.9** (+2.4)	55.3	**59.8** (+4.5)
VMCBench	73.1	73.8	71.0	74.7	**74.4** (+1.3)	69.8	**73.0** (+3.2)	71.8	**73.6** (+1.8)	61.5	**63.5** (+2.0)
RealWorldQA	57.5	58.3	58.8	58.6	**59.2** (+1.7)	56.0	**57.3** (+1.3)	64.1	**65.5** (+1.4)	49.2	**52.8** (+3.6)
Hallucination Benchmarks											
MMHalu	3.35	3.39	3.25	3.43	**3.50** (+2.5)	3.76	**4.01** (+4.2)	3.54	**3.68** (+2.3)	2.38	**2.54** (+2.7)
Bingo	2.71	2.93	2.65	2.96	**2.93** (+4.4)	2.90	**3.08** (+3.6)	3.31	**3.46** (+3.0)	2.35	**2.56** (+4.2)
POPE-R	89.6	89.4	90.3	89.8	**91.5** (+1.9)	85.3	**86.1** (+0.8)	82.6	**85.7** (+3.1)	66.8	**69.1** (+2.3)
Mathematical Reasoning											
MathVision	25.6	27.3	26.4	28.8	**29.1** (+3.5)	20.8	**23.7** (+2.9)	26.7	**30.9** (+4.2)	26.1	**28.9** (+2.8)
MathVista	54.7	55.2	54.6	55.9	**56.5** (+1.8)	64.8	**67.1** (+2.3)	57.8	**59.6** (+1.8)	47.0	**47.2** (+0.2)
Scientific Reasoning											
MMK12-Phys	37.6	38.2	38.8	38.0	**39.4** (+1.8)	24.4	**25.2** (+0.8)	35.8	**38.0** (+2.2)	33.0	**33.4** (+0.4)
MMK12-Chem	36.2	37.6	39.8	38.0	**40.4** (+4.2)	23.4	**25.2** (+1.8)	31.4	**34.6** (+3.2)	35.2	**36.4** (+1.2)

4 Experiments

4.1 Experimental Settings

Baselines and Evaluation Benchmarks. We evaluate StepRefine on four representative MDLMs (LLaDA-V-8B [92], LLaDA-O-8B [93], Dream-VL-7B [89], and LaViDa-L-8B [39]). We evaluate on a diverse suite of multimodal benchmarks covering four general and domain-specific reasoning categories: (i) General reasoning & understanding on VStar [77], MMVP [66], MMEval-Pro [24], VMCBench [100], RealWorldQA [80]; (ii) General hallucination assessment on MMHalu [62], Bingo [14], POPE [42], (iii) Mathematical reasoning on MathVision [69], MathVista [47]; and (iv) Scientific reasoning on the MMK12 [52] Physics and Chemistry subsets.

Implementation Details. Across all backbones and benchmarks, we decode with generation length $L = 256$, total denoising steps $T = 256$, and low-confidence remasking. The stability threshold ϵ is set to 0.01. Details of the compared methods are provided in Appendix B.1.

4.2 Main Results

Benchmark Evaluation. We compare StepRefine against three training-free decoding strategies for MDLMs, including CoTA [101], ReMix [91], and WINO [22], as shown in Table 1. On LLaDA-V-8B, StepRefine improves average accuracy on general reasoning by +2.8% over the baseline and matches or outperforms competing decoding strategies on every general reasoning benchmark. It also improves on the hallucination benchmarks, yielding gains of +2.5% on MMHalu and +4.4% on Bingo. These improvements generalize across all backbones and benchmark categories. Every backbone shows positive gains, with average improvements ranging from +2.3% to +2.8%. These

Table 2: Compatibility of StepRefine with existing dLLM acceleration methods. We report accuracy for VStar and MMVP, score for MMHalu and Bingo, along with inference speedup and throughput (*Tput*, tokens/s). Speedups are measured relative to the unaccelerated LLaDA-V baseline.

Method	VStar			MMVP			MMHalu			Bingo		
	Acc	Speedup	Tput	Acc	Speedup	Tput	Score	Speedup	Tput	Score	Speedup	Tput
LLaDA-V	55.5	1.00×	2.75	40.7	1.00×	6.61	3.35	1.00×	2.75	2.71	1.00×	3.33
Fast-dLLM	52.4	8.45×	23.21	37.3	4.23×	27.94	3.09	7.75×	21.37	2.58	6.83×	22.75
+ StepRefine	**56.5**	8.47×	23.39	**42.0**	4.25×	28.05	**3.22**	7.79×	21.41	**2.64**	6.86×	22.84
dLLM-Cache	53.9	2.69×	7.39	42.7	2.02×	13.35	3.29	2.66×	7.32	2.44	2.87×	9.58
+ StepRefine	**57.1**	2.73×	7.51	**44.7**	2.02×	13.37	**3.36**	2.72×	7.48	**2.47**	2.88×	9.60
dKV-Cache	54.5	2.42×	6.63	42.0	1.96×	12.94	3.33	2.41×	6.64	2.41	2.66×	8.84
+ StepRefine	**56.0**	2.48×	6.82	**43.4**	1.98×	13.10	**3.41**	2.49×	6.85	**2.50**	2.71×	9.03

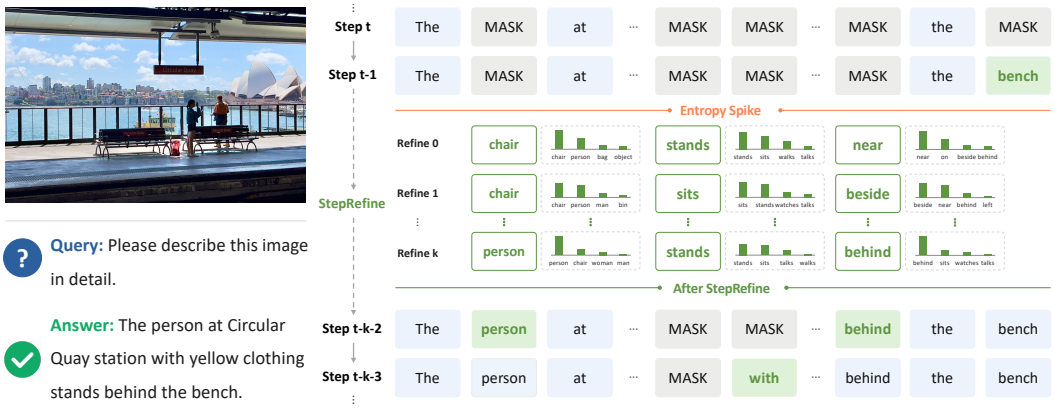


Figure 3: Example visualization of StepRefine. At an entropy spike step, StepRefine reshapes the predicted distribution at masked positions, shifting the prediction toward a more context-consistent candidate, which is then committed. More qualitative examples are provided in Appendix C.

results suggest that the benefit of entropy-spike-aware refinement is robust across MDLM backbones rather than specific to any single model. Additional results are reported in Appendix B.3.

Compatibility with Acceleration Methods. We further evaluate StepRefine in combination with three acceleration methods, including Fast-dLLM [75], dLLM-Cache [44], dKV-Cache [51]. StepRefine is orthogonal to these methods by design. Each refinement round, StepRefine operates on the embedding at masked positions through standard forward passes, leaving the cache and feature-reuse schedules used by each accelerator unchanged during ordinary steps. Table 2 shows that, plugged into each method, StepRefine largely maintains the speedups of these acceleration methods while improving performance by +2.9% on VStar and +1.6% on MMHalu. Throughput is also preserved or slightly increased after adding StepRefine, confirming that refinement does not introduce hidden latency. The additional speedup over each accelerator's baseline is due to the budget compensation mechanism. The k refinement rounds at an entropy spike step are charged against the remaining step budget, replacing k later steps rather than extending the total compute.

Qualitative Analysis. Figure 3 illustrates a representative StepRefine decoding trajectory, which alternates between standard denoising steps and entropy spike steps. Within each spike step, we plot the predicted distributions at masked positions across the refinement rounds. As refinement proceeds, these distributions sharpen as the entropy of the predicted distribution decreases, and the prediction converges on the candidate that is finally committed. Notably, entropy spike steps are precisely the

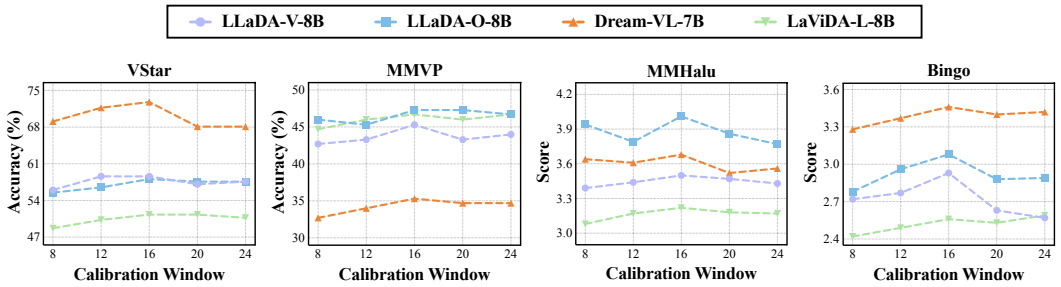


Figure 4: Comparisons of model performance on four datasets under different calibration window sizes. The window provides the running statistics of the entropy deviation used to detect spike steps.

points where a single greedy commit is unreliable. Each refinement round feeds the soft embeddings derived from the previous distributions back into the model, letting competing candidates interact with the surrounding context.

4.3 Ablation Study

Effect of Calibration Window Size. We study the effect of the calibration window size w used for entropy spike detection. As shown in Fig. 4, accuracy improves as the window increases up to a moderate size ($w = 16$ in our experiments), and then oscillates. When the window is small, the entropy-deviation estimate is dominated by step-to-step noise and produces unstable triggers. When the window is large, entropy deviations are averaged across denoising stages with different uncertainty profiles, blurring the distinction between stable steps and steps with entropy spike. Taken together, these suggest that a moderate window provides a reasonable trade-off in our setting.

Adaptive versus Fixed Refinement Scheduling. We compare StepRefine's adaptive trigger against three fixed schedules that insert latent refinement at uniformly spaced denoising steps. Fig. 5(a) reveals that the adaptive trigger outperforms all three fixed schedules. Entropy spike steps neither follow a uniform spacing nor occur at the same positions across samples, so a periodic schedule may fail to capture spikes in some inputs while spending refinement on already-stable steps in others. In contrast, our adaptive trigger conditions both the position and frequency of refinement on local entropy deviation, applying refinement only at steps with anomalous uncertainty.

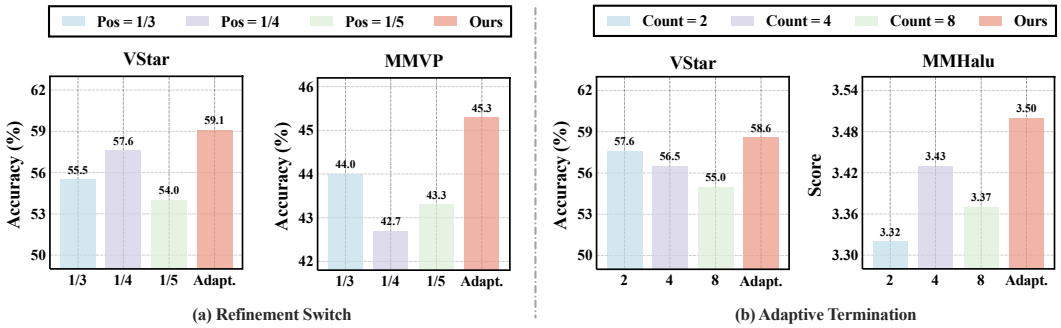


Figure 5: Ablations on StepRefine's adaptive trigger and termination criteria. (a) Comparison between StepRefine's adaptive trigger and three fixed schedules that insert latent refinement at uniformly spaced denoising steps. (b) Comparison between StepRefine's adaptive termination criterion and fixed numbers of refinement rounds.

Effect of the Refinement Controller. We compare StepRefine's adaptive termination criterion against fixed numbers of refinement rounds. Fig 5(b) shows that the adaptive scheme outperforms every fixed budget on both datasets, while no single fixed value performs consistently best across datasets. A single fixed budget allocates an equal number of refinement rounds to every spike step, regardless of whether the prediction has already converged. The adaptive criterion instead allocates refinement to each entropy spike step only while its predictive distribution continues to change, and terminates when it stabilizes.

Table 3: Ablations of decoding settings for LLaDA-V-8B on MMHalu. (a) Baseline vs. StepRefine across remasking strategies. (b) Baseline vs. StepRefine across step budgets and generation lengths.

(a) Remasking strategy

Strategy	Base	**StepRefine**
Random	2.97	**3.08**
Low Confidence	3.35	**3.50**
Top-k Margin	3.27	**3.41**

(b) Step budget at two generation lengths

Method	$L = 256$			$L = 128$		
	T=64	T=128	T=256	T=32	T=64	T=128
Base	3.13	3.31	3.35	3.22	3.30	3.45
StepRefine	**3.28**	**3.37**	**3.50**	**3.34**	**3.41**	**3.53**

Robustness across Decoding Settings. StepRefine remains robust across diverse decoding settings. As shown in Table 3(a), StepRefine improves over the baseline under three remasking strategies with absolute MMHalu gains of +3.7% to +4.5%. Table 3(b) further shows consistent improvements across two generation lengths with three step budgets each, with MMHalu gains ranging from +1.8% to +4.8% across all six configurations. These results indicate that StepRefine generalizes across decoding configurations without per-setting tuning. The gain also holds at tight budgets, indicating that StepRefine remains effective even when total compute is constrained.

5 Related Work

Multimodal diffusion LLM. Diffusion large language models (dLLMs) [54, 90, 105, 28, 1] leverage their inherent bidirectional attention and iterative denoising process to exhibit strong global context modeling capability and flexible reasoning potential. In recent years, this paradigm has been further extended to multimodal settings. Multimodal diffusion models [20, 38, 65, 6, 26, 74, 89, 34, 12] are typically built upon pretrained dLLMs and incorporate visual representations to handle vision-language tasks. For example, methods such as LLaDA-V [92], LaViDa [39], and Dimple [94] align image features into the same embedding space as text tokens through visual instruction tuning, enabling joint modeling of visual and linguistic information. Another line of methods [88, 63, 59], such as LLaDA-O [93], Lumina-dimoo [84], and LaViDa-O [37], further combines fine-tuning and reinforcement learning to jointly model discretized visual and textual representations for unified multimodal generation and understanding. In addition, methods such as DiffusionVL [95] and Fudoki [67] transform autoregressive models into the discrete diffusion paradigm through diffusion fine-tuning, providing another path for multimodal diffusion modeling.

Decoding strategy in dLLM. Despite the advantages of parallel generation and global modeling offered by discrete masked diffusion, the inference efficiency [55, 53, 23, 18, 78, 75, 9, 3] and generation quality of diffusion large language models largely depend on the decoding strategy. One line of work leverages cache reuse [32, 27, 71, 61, 51, 44] and token pruning [45, 82, 41, 8, 4] to alleviate the computational bottlenecks introduced by global attention. Another line of research focuses on optimizing sampling [16, 81, 5, 35, 15, 31, 68, 56, 70, 30], early stopping [36, 33, 7], and step scheduling [49, 99, 17, 73, 17, 99, 2, 50] to reduce redundant denoising iterations. Existing approaches for mitigating hallucinations in diffusion models include self-corrective decoding [22, 25, 64, 11, 60, 83, 104] and prior constraints [108, 13, 76, 57, 85, 96, 87], which improve model reliability by reducing error accumulation. Prior work replaces the binary mask-to-token transition with continuous token representations at each decoding step [91, 106, 103, 10, 21, 102]. We examine hallucination in MDLMs from the perspective of step-wise uncertainty dynamics during denoising. Based on this observation, we propose StepRefine, a plug-and-play decoding method that mitigates hallucinations by detecting entropy spike steps and selectively applying latent refinement.

6 Conclusion

In this work, we extensively study hallucination in multimodal diffusion language models by looking at how predictive uncertainty evolves across denoising steps. We observe that during iterative denoising, MDLMs show sharp local increases in mean mask entropy at a small number of steps, which we call entropy spikes. These steps correlate with hallucinated tokens and mainly arise in the early-to-middle stages of denoising, where confidence-based decoding becomes less reliably

grounded in the current context. Building on this observation, we introduce StepRefine, a training-free decoding strategy that detects entropy spikes via local trend deviation, switches to iterative latent refinement at those steps, and adaptively terminates once the predictive distribution stabilizes. Across four MDLM backbones and multiple benchmarks comprehensively covering general reasoning, hallucination, mathematical, and scientific reasoning, StepRefine obtains improved accuracy and can be combined with existing acceleration methods. In summary, our findings suggest that focusing refinement on a small number of high-uncertainty steps yields measurable improvements in MDLM decoding without additional training.

Acknowledgments

ZX was supported by the Australian Government Research Training Program (RTP) Scholarship. This work was supported by the MAVERIC supercomputing platform at Monash University. This work was supported by the NVIDIA Singapore team through the provision of computational resources.

References

[1] Inclusion AI, Tiwei Bie, Haoxing Chen, Tieyuan Chen, Zhenglin Cheng, Long Cui, Kai Gan, Zhicheng Huang, Zhenzhong Lan, Haoquan Li, et al. Llada2. 0-uni: Unifying multimodal understanding and generation with diffusion large language model. *arXiv preprint arXiv:2604.20796*, 2026.

[2] Wenrui Bao, Zhiben Chen, Dan Xu, and Yuzhang Shang. Learning to parallel: Accelerating diffusion large language models via learnable parallel decoding. *arXiv preprint arXiv:2509.25188*, 2025.

[3] Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. *arXiv preprint arXiv:2505.24857*, 2025.

[4] Shuochen Chang, Xiaofeng Zhang, Qingyang Liu, and Li Niu. D3tom: Decider-guided dynamic token merging for accelerating diffusion mllms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 19961–19969, 2026.

[5] Jian Chen, Yesheng Liang, and Zhijian Liu. Dflash: Block diffusion for flash speculative decoding. *arXiv preprint arXiv:2602.06036*, 2026.

[6] Jiayi Chen, Wenxuan Song, Pengxiang Ding, Ziyang Zhou, Han Zhao, Feilong Tang, Donglin Wang, and Haoang Li. Unified diffusion vla: Vision-language-action model via joint discrete denoising diffusion process. *arXiv preprint arXiv:2511.01718*, 2025.

[7] Qiguang Chen, Hanjing Li, Libo Qin, Dengyun Peng, Jinhao Liu, Jiangyi Wang, Chengyue Wu, Xie Chen, Yantao Du, and Wanxiang Che. Beyond surface reasoning: Unveiling the true long chain-of-thought capacity of diffusion large language models. *arXiv preprint arXiv:2510.09544*, 2025.

[8] Xinhua Chen, Sitao Huang, Cong Guo, Chiyue Wei, Yintao He, Jianyi Zhang, Hai Li, Yiran Chen, et al. Dpad: Efficient diffusion language models with suffix dropout. *arXiv preprint arXiv:2508.14148*, 2025.

[9] Zigeng Chen, Gongfan Fang, Xinyin Ma, Ruonan Yu, and Xinchao Wang. dparallel: Learnable parallel decoding for dllms. *arXiv preprint arXiv:2509.26488*, 2025.

[10] Zigeng Chen, Gongfan Fang, Xinyin Ma, Ruonan Yu, and Xinchao Wang. Dmax: Aggressive parallel decoding for dllms. *arXiv preprint arXiv:2604.08302*, 2026.

[11] Ziyu Chen, Xinbei Jiang, Peng Sun, and Tao Lin. Optimizing decoding paths in masked diffusion models by quantifying uncertainty. *arXiv preprint arXiv:2512.21336*, 2025.

[12] Shuang Cheng, Yihan Bian, Dawei Liu, Linfeng Zhang, Qian Yao, Zhongbo Tian, Wenhai Wang, Qipeng Guo, Kai Chen, Biqing Qi, et al. Sdar: A synergistic diffusion-autoregression paradigm for scalable sequence generation. *arXiv preprint arXiv:2510.06303*, 2025.

[13] Zicong Cheng, Ruixuan Jia, Jia Li, Guo-Wei Yang, Meng-Hao Guo, and Shi-Min Hu. Improving variable-length generation in diffusion language models via length regularization. *arXiv preprint arXiv:2602.07546*, 2026.

[14] Chenhang Cui, Yiyang Zhou, Xinyu Yang, Shirley Wu, Linjun Zhang, James Zou, and Huaxiu Yao. Holistic analysis of hallucination in gpt-4v (ision): Bias and interference challenges. *arXiv preprint arXiv:2311.03287*, 2023.

[15] Yihong Dong, Zhaoyu Ma, Xue Jiang, Zhiyuan Fan, Jiaru Qian, Yongmin Li, Jianha Xiao, Zhi Jin, Rongyu Cao, Binhua Li, et al. Saber: An efficient sampling with adaptive acceleration and backtracking enhanced remasking for diffusion language model. *arXiv preprint arXiv:2510.18165*, 2025.

[16] Sicheng Feng, Zigeng Chen, Xinyin Ma, Gongfan Fang, and Xinchao Wang. dvoting: Fast voting for dllms. *arXiv preprint arXiv:2602.12153*, 2026.

[17] Hengyu Fu, Baihe Huang, Virginia Adams, Charles Wang, Venkat Srinivasan, and Jiantao Jiao. From bits to rounds: Parallel decoding with exploration for diffusion language models. *arXiv preprint arXiv:2511.21103*, 2025.

[18] Yifeng Gao, Ziang Ji, Yuxuan Wang, Biqing Qi, Hanlin Xu, and Linfeng Zhang. Self speculative decoding for diffusion large language models. *arXiv preprint arXiv:2510.04147*, 2025.

[19] Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. Ssd-lm: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11575–11596, 2023.

[20] Zhihao He, Tieyuan Chen, Kangyu Wang, Ziran Qin, Yang Shao, Chaofan Gan, Shijie Li, Zuxuan Wu, and Weiyao Lin. Vidlada: Bidirectional diffusion large language models for efficient video understanding. *arXiv preprint arXiv:2601.17868*, 2026.

[21] Michael Hersche, Samuel Moor-Smith, Thomas Hofmann, and Abbas Rahimi. Soft-masked diffusion language models. *arXiv preprint arXiv:2510.17206*, 2025.

[22] Feng Hong, Geng Yu, Yushi Ye, Haicheng Huang, Huangjie Zheng, Ya Zhang, Yanfeng Wang, and Jiangchao Yao. Wide-in, narrow-out: Revokable decoding for efficient and effective dllms. *arXiv preprint arXiv:2507.18578*, 2025.

[23] Jianuo Huang, Yaojie Zhang, Yicun Yang, Benhao Huang, Biqing Qi, Dongrui Liu, and Linfeng Zhang. Mask tokens as prophet: Fine-grained cache eviction for efficient dllm inference. *arXiv preprint arXiv:2510.09309*, 2025.

[24] Jinsheng Huang, Liang Chen, Taian Guo, Fu Zeng, Yusheng Zhao, Bohan Wu, Ye Yuan, Haozhe Zhao, Zhihui Guo, Yichi Zhang, et al. Mmevalpro: Calibrating multimodal benchmarks towards trustworthy and efficient evaluation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4805–4822, 2025.

[25] Zemin Huang, Yuhang Wang, Zhiyang Chen, and Guo-Jun Qi. Don't settle too early: Self-reflective remasking for diffusion language models. *arXiv preprint arXiv:2509.23653*, 2025.

[26] Yatai Ji, Teng Wang, Yuying Ge, Zhiheng Liu, Sidi Yang, Ying Shan, and Ping Luo. From denoising to refining: A corrective framework for vision-language diffusion model. *arXiv preprint arXiv:2510.19871*, 2025.

[27] Yuchu Jiang, Yue Cai, Xiangzhong Luo, Jiale Fu, Jiarui Wang, Chonghan Liu, and Xu Yang. d ^{2} cache: Accelerating diffusion-based llms via dual adaptive caching. *arXiv preprint arXiv:2509.23094*, 2025.

[28] Jaehyeong Jo and Sung Ju Hwang. Continuous diffusion model for language modeling. *arXiv preprint arXiv:2502.11564*, 2025.

[29] Bumjun Kim, Dongjae Jeon, Dueun Kim, Wonje Jeung, and Albert No. Rainbow padding: Mitigating early termination in instruction-tuned diffusion llms. *arXiv preprint arXiv:2510.03680*, 2025.

[30] Seo Hyun Kim, Sunwoo Hong, Hojung Jung, Youngrok Park, and Se-Young Yun. Klass: Kl-guided fast inference in masked diffusion models. *arXiv preprint arXiv:2511.05664*, 2025.

[31] Fanheng Kong, Jingyuan Zhang, Yahui Liu, Zirui Wu, Yu Tian, Guorui Zhou, et al. Accelerating diffusion llm inference via local determinism propagation. *arXiv preprint arXiv:2510.07081*, 2025.

[32] Guanghao Li, Zhihui Fu, Min Fang, Qibin Zhao, Ming Tang, Chun Yuan, and Jun Wang. Diffuspec: Unlocking diffusion language models for speculative decoding. *arXiv preprint arXiv:2510.02358*, 2025.

[33] Jinsong Li, Xiaoyi Dong, Yuhang Zang, Yuhang Cao, Jiaqi Wang, and Dahua Lin. Beyond fixed: Training-free variable-length denoising for diffusion large language models. *arXiv preprint arXiv:2508.00819*, 2025.

[34] Lijiang Li, Zuwei Long, Yunhang Shen, Heting Gao, Haoyu Cao, Xing Sun, Caifeng Shan, Ran He, and Chaoyou Fu. Omni-diffusion: Unified multimodal understanding and generation with masked discrete diffusion. *arXiv preprint arXiv:2603.06577*, 2026.

[35] Miao Li, Hanyang Jiang, Sikai Cheng, Hengyu Fu, Yuhang Cai, Baihe Huang, Tinghan Ye, Xuanzhou Chen, and Pascal Van Hentenryck. Plan, verify and fill: A structured parallel decoding approach for diffusion language models. *arXiv preprint arXiv:2601.12247*, 2026.

[36] Pengxiang Li, Yefan Zhou, Dilxat Muhtar, Lu Yin, Shilin Yan, Li Shen, Soroush Vosoughi, and Shiwei Liu. Diffusion language models know the answer before decoding. *arXiv preprint arXiv:2508.19982*, 2025.

[37] Shufan Li, Jiuxiang Gu, Kangning Liu, Zhe Lin, Zijun Wei, Aditya Grover, and Jason Kuen. Lavida-o: Elastic large masked diffusion models for unified multimodal understanding and generation. *arXiv preprint arXiv:2509.19244*, 2025.

[38] Shufan Li, Jiuxiang Gu, Kangning Liu, Zhe Lin, Zijun Wei, Aditya Grover, and Jason Kuen. Sparse-lavida: Sparse multimodal discrete diffusion language models. *arXiv preprint arXiv:2512.14008*, 2025.

[39] Shufan Li, Konstantinos Kallidromitis, Hritik Bansal, Akash Gokul, Yusuke Kato, Kazuki Kozuka, Jason Kuen, Zhe Lin, Kai-Wei Chang, and Aditya Grover. Lavida: A large diffusion language model for multimodal understanding. *arXiv preprint arXiv:2505.16839*, 2025.

[40] Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in neural information processing systems*, 35:4328–4343, 2022.

[41] Yang Li, Han Meng, Chenan Wang, and Haipeng Chen. Dip: Dynamic in-context planner for diffusion language models. *arXiv preprint arXiv:2601.03199*, 2026.

[42] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 292–305, 2023.

[43] Enshu Liu, Xuefei Ning, Yu Wang, and Zinan Lin. Ni sampling: Accelerating discrete diffusion sampling by token order optimization. *arXiv preprint arXiv:2604.18471*, 2026.

[44] Zhiyuan Liu, Yicun Yang, Yaojie Zhang, Junjie Chen, Chang Zou, Qingyuan Wei, Shaobo Wang, and Linfeng Zhang. dllm-cache: Accelerating diffusion large language models with adaptive caching. *arXiv preprint arXiv:2506.06295*, 2025.

[45] Lingkun Long, Yushi Huang, Shihao Bai, Ruihao Gong, Jun Zhang, Ao Zhou, and Jianlei Yang. Focus-dllm: Accelerating long-context diffusion llm inference via confidence-guided context focusing. *arXiv preprint arXiv:2602.02159*, 2026.

[46] Guanxi Lu, Hao Mark Chen, Yuto Karashima, Zhican Wang, Daichi Fujiki, and Hongxiang Fan. Adablock-dllm: Semantic-aware diffusion llm inference via adaptive block size. *arXiv preprint arXiv:2509.26432*, 2025.

[47] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*, 2023.

[48] Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-gps: Interpretable geometry problem solving with formal language and symbolic reasoning. *arXiv preprint arXiv:2105.04165*, 2021.

[49] Lizhuo Luo, Shenggui Li, Yonggang Wen, and Tianwei Zhang. Dsb: Dynamic sliding block scheduling for diffusion llms. *arXiv preprint arXiv:2602.05992*, 2026.

[50] Omer Luxembourg, Haim Permuter, and Eliya Nachmani. Plan for speed: Dilated scheduling for masked diffusion language models. *arXiv preprint arXiv:2506.19037*, 2025.

[51] Xinyin Ma, Runpeng Yu, Gongfan Fang, and Xinchao Wang. dkv-cache: The cache for diffusion language models. *arXiv preprint arXiv:2505.15781*, 2025.

[52] Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu, Botian Shi, Wenhai Wang, Junjun He, Kaipeng Zhang, et al. Mm-eureka: Exploring visual aha moment with rule-based large-scale reinforcement learning. *arXiv preprint arXiv:2503.07365*, 2025.

[53] Quan Nguyen-Tri, Mukul Ranjan, and Zhiqiang Shen. Attention is all you need for kv cache in diffusion llms. *arXiv preprint arXiv:2510.14973*, 2025.

[54] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

[55] Daisuke Oba, Danushka Bollegala, Masahiro Kaneko, and Naoaki Okazaki. Stopping computation for converged tokens in masked diffusion-lm decoding. *arXiv preprint arXiv:2602.06412*, 2026.

[56] Xiaojing Qi, Lun Du, Xinyuan Zhang, Lanning Wei, Tao Jin, and Da Zheng. Hierarchy decoding: A training-free parallel decoding strategy for diffusion large language models. In *The Fourteenth International Conference on Learning Representations*.

[57] Yeongbin Seo, Dongha Lee, Jaehyung Kim, and Jinyoung Yeo. Fast and fluent diffusion language models via convolutional decoding and rejective fine-tuning. *arXiv preprint arXiv:2509.15188*, 2025.

[58] Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37:103131–103167, 2024.

[59] Qingyu Shi, Jinbin Bai, Zhuoran Zhao, Wenhao Chai, Kaidong Yu, Jianzong Wu, Shuangyong Song, Yunhai Tong, Xiangtai Li, Xuelong Li, et al. Muddit: Liberating generation beyond text-to-image with a unified discrete diffusion model. *arXiv preprint arXiv:2505.23606*, 2025.

[60] Yingte Shu, Yuchuan Tian, Chao Xu, Yunhe Wang, and Hanting Chen. Deferred commitment decoding for diffusion language models with confidence-aware sliding windows. *arXiv preprint arXiv:2601.02076*, 2026.

[61] Yuerong Song, Xiaoran Liu, Ruixiao Li, Zhigeng Liu, Zengfeng Huang, Qipeng Guo, Ziwei He, and Xipeng Qiu. Sparse-dllm: Accelerating diffusion llms with dynamic cache eviction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 33038–33046, 2026.

[62] Zhiqing Sun, Sheng Shen, Shengcao Cao, Haotian Liu, Chunyuan Li, Yikang Shen, Chuang Gan, Liang-Yan Gui, Yu-Xiong Wang, Yiming Yang, et al. Aligning large multimodal models with factually augmented rlhf. In *Annual Meeting of the Association for Computational Linguistics*, 2024.

[63] Alexander Swerdlow, Mihir Prabhudesai, Siddharth Gandhi, Deepak Pathak, and Katerina Fragkiadaki. Unified multimodal discrete diffusion. *arXiv preprint arXiv:2503.20853*, 2025.

[64] Runchu Tian, Junxia Cui, Xueqiang Xu, Feng Yao, and Jingbo Shang. Finish first, perfect later: Test-time token-level cross-validation for diffusion large language models. *arXiv preprint arXiv:2510.05090*, 2025.

[65] Ye Tian, Ling Yang, Jiongfan Yang, Anran Wang, Yu Tian, Jiani Zheng, Haochen Wang, Zhiyang Teng, Zhuochen Wang, Yinjie Wang, et al. Mmada-parallel: Multimodal large diffusion language models for thinking-aware editing and generation. *arXiv preprint arXiv:2511.09611*, 2025.

[66] Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. Eyes wide shut? exploring the visual shortcomings of multimodal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9568–9578, 2024.

[67] Jin Wang, Yao Lai, Aoxue Li, Shifeng Zhang, Jiacheng Sun, Ning Kang, Chengyue Wu, Zhenguo Li, and Ping Luo. Fudoki: Discrete flow-based unified understanding and generation via kinetic-optimal velocities. *arXiv preprint arXiv:2505.20147*, 2025.

[68] Kangyu Wang, Zhiyun Jiang, Haibo Feng, Weijia Zhao, Lin Liu, Jianguo Li, Zhenzhong Lan, and Weiyao Lin. Creditdecoding: Accelerating parallel decoding in diffusion large language models with trace credits. *arXiv preprint arXiv:2510.06133*, 2025.

[69] Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with math-vision dataset. *Advances in Neural Information Processing Systems*, 37:95095–95169, 2024.

[70] Wen Wang, Bozhen Fang, Chenchen Jing, Yongliang Shen, Yangyi Shen, Qiuyu Wang, Hao Ouyang, Hao Chen, and Chunhua Shen. Time is a feature: Exploiting temporal dynamics in diffusion language models. *arXiv preprint arXiv:2508.09138*, 2025.

[71] Xu Wang, Chenkai Xu, Yijie Jin, Jiachun Jin, Hao Zhang, and Zhijie Deng. Diffusion llms can do faster-than-ar inference via discrete diffusion forcing. *arXiv preprint arXiv:2508.09192*, 2025.

[72] Zeqing Wang, Gongfan Fang, Xinyin Ma, Xingyi Yang, and Xinchao Wang. Sparsed: Sparse attention for diffusion language models. *arXiv preprint arXiv:2509.24014*, 2025.

[73] Qingyan Wei, Yaojie Zhang, Zhiyuan Liu, Puyu Zeng, Yuxuan Wang, Biqing Qi, Dongrui Liu, and Linfeng Zhang. Accelerating diffusion large language models with slowfast sampling: The three golden principles. *arXiv preprint arXiv:2506.10848*, 2025.

[74] Junjie Wen, Minjie Zhu, Jiaming Liu, Zhiyuan Liu, Yicun Yang, Linfeng Zhang, Shanghang Zhang, Yichen Zhu, and Yi Xu. dvla: Diffusion vision-language-action model with multimodal chain-of-thought. *arXiv preprint arXiv:2509.25681*, 2025.

[75] Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. *arXiv preprint arXiv:2505.22618*, 2025.

[76] Jingxuan Wu, Zhenglin Wan, Xingrui Yu, Yuzhe Yang, Yiqiao Huang, Ivor Tsang, and Yang You. Time-annealed perturbation sampling: Diverse generation for diffusion language models. *arXiv preprint arXiv:2601.22629*, 2026.

[77] Penghao Wu and Saining Xie. V?: Guided visual search as a core mechanism in multimodal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13084–13094, 2024.

[78] Shutong Wu and Jiawei Zhang. Free draft-and-verification: Toward lossless parallel decoding for diffusion large language models. *arXiv preprint arXiv:2510.00294*, 2025.

[79] Tianyi Wu, Xiaoxi Sun, Yanhua Jiao, Yulin Li, Yixin Chen, Yun-Hao Cao, Yi-Qi Hu, and Zhuotao Tian. Dynamic-dllm: Dynamic cache-budget and adaptive parallel decoding for training-free acceleration of diffusion llm. In *The Fourteenth International Conference on Learning Representations*.

[80] X.AI. Grok-2 beta release, 2024. Accessed: 2024.

[81] Yanzheng Xiang, Lan Wei, Yizhen Yao, Qinglin Zhu, Hanqi Yan, Chen Jin, Philip Alexander Teare, Dandan Zhang, Lin Gui, Amrutha Saseendran, et al. Stop the flip-flop: Context-preserving verification for fast revocable diffusion decoding. *arXiv preprint arXiv:2602.06161*, 2026.

[82] Zhongyu Xiao, Zhiwei Hao, Jianyuan Guo, Yong Luo, Jia Liu, Jie Xu, and Han Hu. Streaming-dllm: Accelerating diffusion llms via suffix pruning and dynamic decoding. *arXiv preprint arXiv:2601.17917*, 2026.

[83] Yi Xin, Siqi Luo, Qi Qin, Haoxing Chen, Kaiwen Zhu, Zhiwei Zhang, Yangfan He, Rongchao Zhang, Jinbin Bai, Shuo Cao, et al. dmllm-tts: Self-verified and efficient test-time scaling for diffusion multi-modal large language models. *arXiv preprint arXiv:2512.19433*, 2025.

[84] Yi Xin, Qi Qin, Siqi Luo, Kaiwen Zhu, Juncheng Yan, Yan Tai, Jiayi Lei, Yuewen Cao, Keqi Wang, Yibin Wang, et al. Lumina-dimoo: An omni diffusion large language model for multi-modal generation and understanding. *arXiv preprint arXiv:2510.06308*, 2025.

[85] Zhen Xiong, Yujun Cai, Zhecheng Li, and Yiwei Wang. Unveiling the potential of diffusion large language model in controllable generation. *arXiv preprint arXiv:2507.04504*, 2025.

[86] Weiye Xu, Jiahao Wang, Weiyun Wang, Zhe Chen, Wengang Zhou, Aijun Yang, Lewei Lu, Houqiang Li, Xiaohua Wang, Xizhou Zhu, et al. Visulogic: A benchmark for evaluating visual reasoning in multi-modal large language models. *arXiv preprint arXiv:2504.15279*, 2025.

[87] Kaisen Yang, Jayden Teoh, Kaicheng Yang, Yitong Zhang, and Alex Lamb. Improving sampling for masked diffusion models via information gain. *arXiv preprint arXiv:2602.18176*, 2026.

[88] Ling Yang, Ye Tian, Bowen Li, Xinchen Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. Mmada: Multimodal large diffusion language models. *arXiv preprint arXiv:2505.15809*, 2025.

[89] Jiacheng Ye, Shansan Gong, Jiahui Gao, Junming Fan, Shuang Wu, Wei Bi, Haoli Bai, Lifeng Shang, and Lingpeng Kong. Dream-vl & dream-vla: Open vision-language and vision-language-action models with diffusion language model backbone. *arXiv preprint arXiv:2512.22615*, 2025.

[90] Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

[91] Yushi Ye, Feng Hong, Huangjie Zheng, Xu Chen, Zhiyong Chen, Yanfeng Wang, and Jiangchao Yao. Rejection mixing: Fast semantic propagation of mask tokens for efficient dllm inference. *arXiv preprint arXiv:2602.22868*, 2026.

[92] Zebin You, Shen Nie, Xiaolu Zhang, Jun Hu, Jun Zhou, Zhiwu Lu, Ji-Rong Wen, and Chongxuan Li. Llada-v: Large language diffusion models with visual instruction tuning. *arXiv preprint arXiv:2505.16933*, 2025.

[93] Zebin You, Xiaolu Zhang, Jun Zhou, Chongxuan Li, and Ji-Rong Wen. Llada-o: An effective and length-adaptive omni diffusion model. *arXiv preprint arXiv:2603.01068*, 2026.

[94] Runpeng Yu, Xinyin Ma, and Xinchao Wang. Dimple: Discrete diffusion multimodal large language model with parallel decoding. *arXiv preprint arXiv:2505.16990*, 2025.

[95] Lunbin Zeng, Jingfeng Yao, Bencheng Liao, Hongyuan Tao, Wenyu Liu, and Xinggang Wang. Diffusionvl: Translating any autoregressive models into diffusion vision language models. *arXiv preprint arXiv:2512.15713*, 2025.

[96] Jiebin Zhang, Zhenghan Yu, Liang Wang, Nan Yang, Eugene J Yu, Zheng Li, Yifan Song, Dawei Zhu, Xingxing Zhang, Furu Wei, et al. Learning to draft: Adaptive speculative decoding with reinforcement learning. *arXiv preprint arXiv:2603.01639*, 2026.

[97] Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pages 169–186. Springer, 2024.

[98] Yizhe Zhang, Jiatao Gu, Zhuofeng Wu, Shuangfei Zhai, Joshua Susskind, and Navdeep Jaitly. Planner: Generating diversified paragraph via latent language diffusion model. *Advances in Neural Information Processing Systems*, 36:80178–80190, 2023.

[99] Yu Zhang, Xinchen Li, Jialei Zhou, Hongnan Ma, Zhongwei Wan, Yiwei Shi, Duoqian Miao, Qi Zhang, and Longbing Cao. Swordsman: Entropy-driven adaptive block partition for efficient diffusion language models. *arXiv preprint arXiv:2602.04399*, 2026.

[100] Yuhui Zhang, Yuchang Su, Yiming Liu, Xiaohan Wang, James Burgess, Elaine Sui, Chenyu Wang, Josiah Aklilu, Alejandro Lozano, Anjiang Wei, et al. Automated generation of challenging multiple-choice questions for vision language model evaluation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 29580–29590, 2025.

[101] Qiyan Zhao, Xiaofeng Zhang, Shuochen Chang, Qianyu Chen, Xiaosong Yuan, Xuhang Chen, Luoqi Liu, Jiajun Zhang, Xu-Yao Zhang, and Da-Han Wang. Context tokens are anchors: Understanding the repetition curse in dmllms from an information flow perspective. *arXiv preprint arXiv:2601.20520*, 2026.

[102] Huangjie Zheng, Shansan Gong, Ruixiang Zhang, Tianrong Chen, Jiatao Gu, Mingyuan Zhou, Navdeep Jaitly, and Yizhe Zhang. Continuously augmented discrete diffusion model for categorical generative modeling. *arXiv preprint arXiv:2510.01329*, 2025.

[103] Linhao Zhong, Linyu Wu, Bozhen Fang, Tianjian Feng, Chenchen Jing, Wen Wang, Jiaheng Zhang, Hao Chen, and Chunhua Shen. Beyond hard masks: Progressive token evolution for diffusion language models. *arXiv preprint arXiv:2601.07351*, 2026.

[104] Linhao Zhong, Linyu Wu, Wen Wang, Yuling Xi, Chenchen Jing, Jiaheng Zhang, Hao Chen, and Chunhua Shen. Efficient self-evaluation for diffusion language models via sequence regeneration. *arXiv preprint arXiv:2603.02760*, 2026.

[105] Fengqi Zhu, Zebin You, Yipeng Xing, Zenan Huang, Lin Liu, Yihong Zhuang, Guoshan Lu, Kangyu Wang, Xudong Wang, Lanning Wei, et al. Llada-moe: A sparse moe diffusion language model. *arXiv preprint arXiv:2509.24389*, 2025.

[106] Qinglin Zhu, Yizhen Yao, Runcong Zhao, Yanzheng Xiang, Amrutha Saseendran, Chen Jin, Philip Teare, Bin Liang, Yulan He, and Lin Gui. Latent refinement decoding: Enhancing diffusion-based language models by refining belief states. *arXiv preprint arXiv:2510.11052*, 2025.

[107] Zijian Zhu, Fei Ren, Zhanhong Tan, and Kaisheng Ma. Es-dllm: Efficient inference for diffusion large language models by early-skipping. *arXiv preprint arXiv:2603.10088*, 2026.

[108] Haomin Zuo, Yidi Li, Luoxiao Yang, and Xiaofeng Zhang. Diffusion-cam: Faithful visual explanations for dmllms. *arXiv preprint arXiv:2604.11005*, 2026.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our contributions are summarized in the Abstract and Introduction sections.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss failure cases in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Please see the setup in Sec.4.1 and detailed experimental settings in appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code has been publicly released.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The detailed results can be found in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The detailed results can be found in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The detailed results can be found in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have read the NeurIPS Code of Ethics and confirm that our research complies with these ethical standards.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Please see the Appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All the thing in this paper credited and are the license and terms of use explicitly mentioned and properly respected.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: This paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The method development in this paper does not involve LLMs as any components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.

Appendix

A Use of LLMs

LLMs were used as writing and editing assistants in this study, with their role limited to improving grammar, spelling, clarity, and stylistic consistency. They did not contribute to the scientific content of the work, including research ideation, experimental design, data analysis, result interpretation, or any core intellectual contributions. All substantive content and scientific contributions are entirely attributable to the authors.

B Additional Experiments

B.1 Baseline

All experiments are conducted on a single server with 8 NVIDIA H100 GPUs (80GB). To benchmark our method against recent training-free decoding strategies for diffusion-based models, we reproduce three representative baselines. All baselines are implemented under the same backbone and evaluation pipeline. We follow the original method designs and reported hyper-parameter settings whenever available to ensure a fair comparison.

- **Wide-In, Narrow-Out (WINO) [22]:** WINO enables revokable decoding for DLLMs through a parallel draft-and-verify mechanism. At each decoding step, it first drafts multiple tokens using a lenient confidence threshold, then verifies previously decoded tokens with an auxiliary shadow block and re-masks low-confidence tokens for refinement. We follow the original semi-autoregressive decoding setting and use the reported drafting and verification thresholds.
- **Context Tokens are Anchors (CoTA) [101]:** CoTA mitigates cache-induced repetition in dMLLMs by preserving the information flow of context tokens. It enhances attention to nearby context tokens through a distance-aware intervention and adjusts decoding scores using deep-layer context-token entropy to suppress uncertain candidates. We integrate CoTA with the corresponding cache setting and follow the original hyper-parameter configuration.
- **Rejection Mixing (ReMix) [91]:** ReMix accelerates DLLM decoding by introducing a continuous mixing state between masked and finalized token states. Undecoded positions are updated with probability-weighted embeddings to propagate semantic information, while unstable positions are rejected back to the masked state based on distributional divergence. We follow the original confidence threshold, mixing ratio, and rejection threshold settings.

We also examine the compatibility of our method with recent training-free inference acceleration methods for diffusion language models. These methods target different computational bottlenecks during diffusion decoding, such as feature reuse, KV-cache approximation, sparse attention, and delayed cache updates. Since our method only introduces an intra-step soft refinement before discrete token commitment, these acceleration methods can be applied during regular denoising steps and resumed after refinement.

- **dLLM-Cache [44]:** dLLM-Cache accelerates diffusion language model inference by reusing intermediate features across adjacent denoising steps. It applies long-interval caching to static prompt tokens and adaptive short-interval updates to response tokens, where dynamically changing positions are selected according to value-vector similarity. When combined with our method, dLLM-Cache can be used during regular decoding steps, while soft-refinement steps are handled with an isolated temporary cache before refreshing the main cache after token commitment.
- **Fast-dLLM [75]:** Fast-dLLM improves diffusion LLM inference through block-wise approximate KV caching and confidence-aware parallel decoding. Its cache mechanism reuses prefix, or prefix-and-suffix, KV states within a decoding block, while its parallel decoding strategy commits only high-confidence tokens to reduce dependency errors. In our setting, Fast-dLLM can operate during normal decoding, whereas soft refinement temporarily disables cache reuse and then returns the refined logits to the standard confidence-aware commitment rule.

- **SparseD [72]:** SparseD reduces the quadratic attention cost in diffusion language models by using head-specific sparse attention patterns. It first applies full attention in early denoising steps, then pre-computes sparse query-key patterns and reuses them in later steps. Since SparseD modifies the attention computation rather than the discrete decoding trajectory, it is naturally compatible with our method; during soft refinement, one may either retain the sparse attention pattern or conservatively fall back to full attention.
- **dKV-Cache [51]:** dKV-Cache introduces a delayed KV-cache mechanism for diffusion language models by caching only decoded tokens whose representations have become relatively stable. To avoid instability immediately after token commitment, it delays cache insertion by one denoising step and periodically refreshes cached states. When integrated with our method, dKV-Cache remains active in regular denoising steps, while refinement uses a temporary soft-state cache; newly committed tokens after refinement follow the original one-step delayed caching rule.

B.2 Benchmarks

In this section, we briefly describe each benchmark used in our evaluation, grouped into four categories: general reasoning, hallucination, mathematical reasoning, and scientific reasoning.

B.2.1 General Reasoning & Understanding

- **V*Bench (vstar_bench) [77]**: a multimodal benchmark built on diverse real-world images that probes visual understanding, spatial reasoning, and fine-grained attribute recognition in VLMs.
- **RealWorldQA [80]**: a collection of real-world photographs (many taken from vehicle viewpoints) paired with short-answer questions, targeting basic spatial understanding and everyday scene reasoning.
- **MMVP (MultiModal Visual Patterns) [66]**: 300 carefully crafted questions over "CLIP-blind" image pairs, testing whether models can tell apart subtle visual-pattern differences that standard visual encoders typically miss.
- **MMEval-Pro [24]**: a calibrated multimodal benchmark that augments existing multiple-choice datasets (e.g., MMMU, ScienceQA, MathVista) with perception and knowledge-anchor questions, forming question triplets that suppress text-only shortcuts.
- **VMCBench [100]**: a unified multiple-choice benchmark ($\sim$9k questions) that consolidates 20 VQA datasets via AutoConverter, enabling consistent and scalable evaluation of VLMs across diverse visual-linguistic scenarios.

B.2.2 Hallucination Benchmark

- **MMHalu (MMHal-Bench) [62]**: a general-purpose hallucination benchmark built on diverse image-question pairs, measuring how often models generate visually inconsistent or fabricated content in realistic settings.
- **Bingo (Bias and Interference Challenges) [14]**: a diagnostic benchmark with image-text challenge sets that target bias-driven errors (region, OCR, factual) together with interference-induced hallucinations.
- **POPE (Polling-based Object Probing Evaluation) [42]**: an object-hallucination probe built from balanced yes/no questions about object existence, providing a stable, instruction-agnostic measure of image-faithful generation.

B.2.3 Mathematical Reasoning

- **MathVision (MATH-Vision) [69]**: 3,040 visual math problems drawn from real competitions, spanning 16 disciplines and 5 difficulty levels, to assess multimodal mathematical reasoning in exam-style settings.
- **MathVista [47]**: a unified benchmark of 6,141 examples from 28 existing datasets together with three new subsets (IQTest, FunctionQA, PaperQA), evaluating mathematical reasoning over diagrams, plots, and scientific figures.

- **MathVerse [97]**: 2,612 diagram-based math problems expanded into ∼15k multimodal variants, enabling fine-grained evaluation of whether MLLMs actually reason over diagrams rather than rely on text alone.
- **VisuLogic [86]**: 1,000 single-choice visual reasoning problems across six categories (quantitative shifts, spatial relations, attribute comparisons, etc.), explicitly designed to rule out language-only shortcuts.
- **Geometry3K [48]**: 3,002 multiple-choice Euclidean geometry problems paired with diagrams and formalized structural annotations, covering high-school to competition-level difficulty.

B.2.4 Scientific Reasoning

- **MMK12 [52]**: a K-12 multimodal reasoning dataset from the MM-Eureka framework; its evaluation split contains 500 multiple-choice questions per discipline (mathematics, physics, chemistry, biology), covering foundational scientific reasoning.

B.3 Additional Benchmark Results

Table 4 reports StepRefine results on the remaining benchmarks that are not included in the main paper. The same trend holds: StepRefine consistently improves over the baseline across all backbones.

Benchmark	LLaDA-V-8B		LLaDA-O-8B		Dream-VL-7B		LaViDA-L-8B	
	Base	**StepRefine**	Base	**StepRefine**	Base	**StepRefine**	Base	**StepRefine**
Hallucination Benchmarks								
POPE-P	87.9	**88.7** (+0.8)	83.8	**84.9** (+1.1)	81.7	**84.5** (+2.8)	64.9	**66.1** (+1.2)
POPE-A	86.2	**87.5** (+1.3)	83.3	**83.8** (+0.5)	80.1	**83.4** (+3.3)	61.7	**64.8** (+3.1)
Mathematical Reasoning								
MathVerse	34.1	**35.3** (+1.2)	38.6	**42.6** (+4.0)	31.4	**34.9** (+3.5)	32.1	**33.9** (+1.8)
Geometry3K	20.9	**22.3** (+1.4)	26.6	**29.5** (+2.9)	37.4	**39.1** (+1.7)	15.3	**15.7** (+0.4)
VisuLogic	24.5	**27.7** (+3.2)	35.2	**36.3** (+1.1)	24.8	**27.1** (+2.3)	23.1	**25.9** (+2.8)
MMK12-Math	38.8	**42.0** (+3.2)	26.8	**28.0** (+1.2)	42.8	**45.6** (+2.8)	36.2	**39.0** (+2.8)
Scientific Reasoning								
MMK12-Bio	39.6	**40.2** (+0.6)	27.4	**28.8** (+1.4)	31.8	**35.8** (+4.0)	38.0	**41.4** (+3.4)

Table 4: Additional benchmark results of StepRefine across the four backbones. Accuracy is reported for all benchmarks.

B.4 Extensive ablations on model parameter setting

In this section, we provide additional ablations on decoding hyper-parameters that complement the LLaDA-V-8B results in Table 3 of the main paper. We extend the remasking-strategy and step-budget studies to Dream-VL-7B (Table 5). All numbers are MMHalu scores in $[0, 6]$.

Table 5 mirrors the main-paper layout on Dream-VL-7B. (a) shows that StepRefine consistently improves the MMHalu score over the baseline under all three remasking strategies, with absolute gains ranging from $+2.3\%$ to $+3.7\%$. (b) varies the step budget $T_{\max}$ at two generation lengths $L \in \{128, 256\}$; StepRefine improves the score in every $(L, T_{\max})$ configuration tested. The trends agree with the LLaDA-V-8B results, indicating that the gains transfer across backbones.

B.5 Compatibility with other acceleration methods

We additionally report the compatibility of StepRefine with SparseD [72]. As shown in Table 6, SparseD + StepRefine preserves SparseD's speedup and improves accuracy on every benchmark.

Strategy	Base	StepRefine
Random	2.73	**2.95**
Low Confidence	3.47	**3.68**
Top-k Margin	3.25	**3.39**

(a) Remasking strategy

Method	$L = 256$			$L = 128$		
	T=64	T=128	T=256	T=32	T=64	T=128
Base	3.37	3.54	3.47	3.18	3.69	3.58
StepRefine	**3.55**	**3.63**	**3.68**	**3.72**	**3.86**	**3.67**

(b) Step budget at two generation lengths

Table 5: Ablations of decoding settings for Dream-VL-7B on MMHalu. (a) Baseline vs. StepRefine across remasking strategies. (b) Baseline vs. StepRefine across step budgets and generation lengths.

Benchmark	SparseD			SparseD + StepRefine		
	Acc	Speedup	Tput	Acc	Speedup	Tput
VStar	56.0	1.15×	3.17	**58.6**	1.18×	3.25
MMVP	39.3	1.39×	9.08	**40.7**	1.42×	9.29
MMHalu	3.37	1.38×	3.76	**3.46**	1.42×	3.88
Bingo	2.41	1.28×	4.28	**2.42**	1.30×	4.33

Table 6: Compatibility of StepRefine with SparseD across four benchmarks. Speedups are measured relative to the unaccelerated LLaDA-V baseline.

C Visualizations

C.1 Additional Examples of Entropy Spike Detection

To complement Figure 1, we provide two additional samples in Figure 6. Each sample shows the mean mask entropy across denoising steps, with the detected entropy spikes marked along the trajectory.

C.2 General and Domain-specific Multimodal Reasoning Visualization

Figure 7 presents more qualitative examples of StepRefine's decoding trajectories on general reasoning and hallucination benchmarks, covering a broader range of question types and visual scenes. For domain-specific settings, additional decoding visualizations on mathematical and scientific reasoning benchmarks are shown in Figure 8.

C.3 Challenge in Specific Scenarios

We further include a case in Figure 9, where intra-step refinement is unable to recover the correct answer in particularly challenging scenarios.

D Annotation Process

(1) GPT-4o pre-annotation. We used GPT-4o as a preliminary annotator with explicit rule-based annotation criteria to identify hallucinated tokens in model responses based on the input image and question. Our annotation criteria were as follows: content was labeled correct if it was explicitly grounded in visual evidence and consistent with the context; content was labeled hallucinated if it could not be verified from the image or contradicted the image, including incorrect statements about objects, attributes, quantities, actions, or relationships.

(2) Human verification. We partitioned all pre-annotated samples into two disjoint subsets, each of which was independently reviewed by three authors. Given access to the original image and the complete model response, each author applied the same annotation criteria to (i) inspect and correct the token-level pre-annotations and (ii) identify hallucinations potentially missed by GPT-4o. For

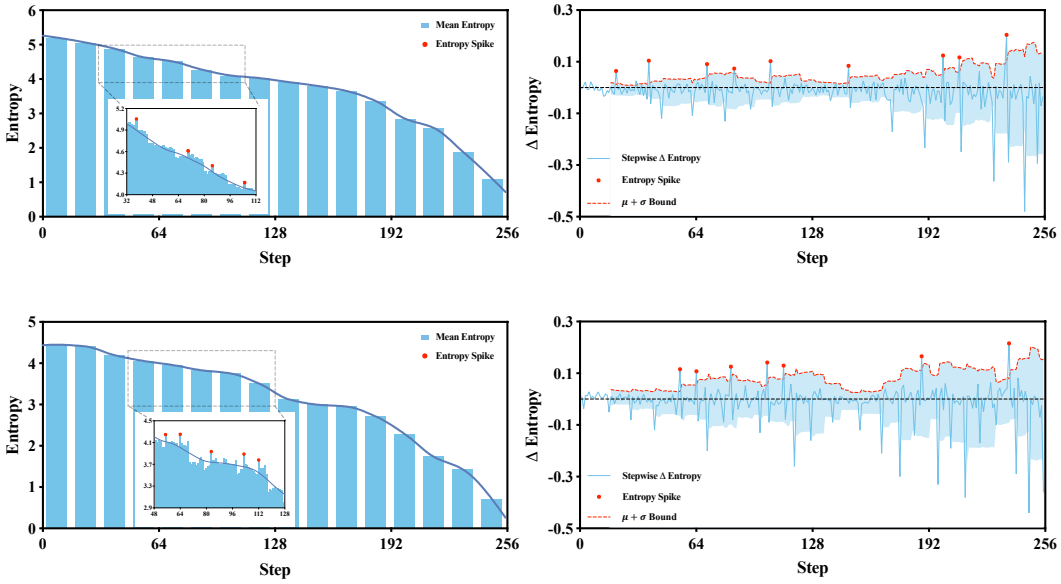


Figure 6: Additional examples of entropy spike detection on two samples. We plot the mean mask entropy across denoising steps and mark the steps flagged as entropy spikes by the $\mu + \sigma$ threshold over the local calibration window.

Model	**LLaDA-V-8B**		**Dream-VL-7B**	
Dataset	p-value	95% CI	p-value	95% CI
VStar	0.031	[1.05, 5.76]	0.039	[1.05, 6.81]
MMVP	0.007	[1.67, 8.00]	0.041	[0.67, 6.33]
MMEval-Pro	< 0.001	[2.62, 3.77]	< 0.001	[1.95, 2.87]
VMCBench	0.019	[0.30, 2.30]	0.003	[0.70, 3.00]
RealWorldQA	0.029	[0.26, 3.14]	0.013	[0.39, 2.48]
MathVision	< 0.001	[2.53, 4.47]	< 0.001	[3.36, 5.07]
MathVista	< 0.001	[1.30, 2.33]	< 0.001	[1.34, 2.28]
MMK12-Phys	0.035	[0.40, 3.40]	0.043	[0.20, 4.20]
MMK12-Chem	0.005	[1.40, 7.00]	0.017	[0.80, 5.60]

Table 7: p-values and 95% confidence intervals across datasets.

ambiguous cases (e.g., plausible inferences or vague prompts), we adopted a conservative criterion: content was labeled correct only when it was directly and unambiguously supported by visible evidence.

E Statistical Significance

To assess the reliability of the reported gains, we have conducted a statistical significance analysis for the LLaDA-V and Dream-VL results across different benchmarks, as reported in Table 7.

F Broader Impacts

All experiments are conducted on publicly released models and benchmarks; no personal or sensitive data are collected or inferred, and all qualitative examples are drawn from publicly available images. Our approach does not involve human-subject interaction and does not raise specific ethical concerns beyond those typically associated with AI model development and evaluation.

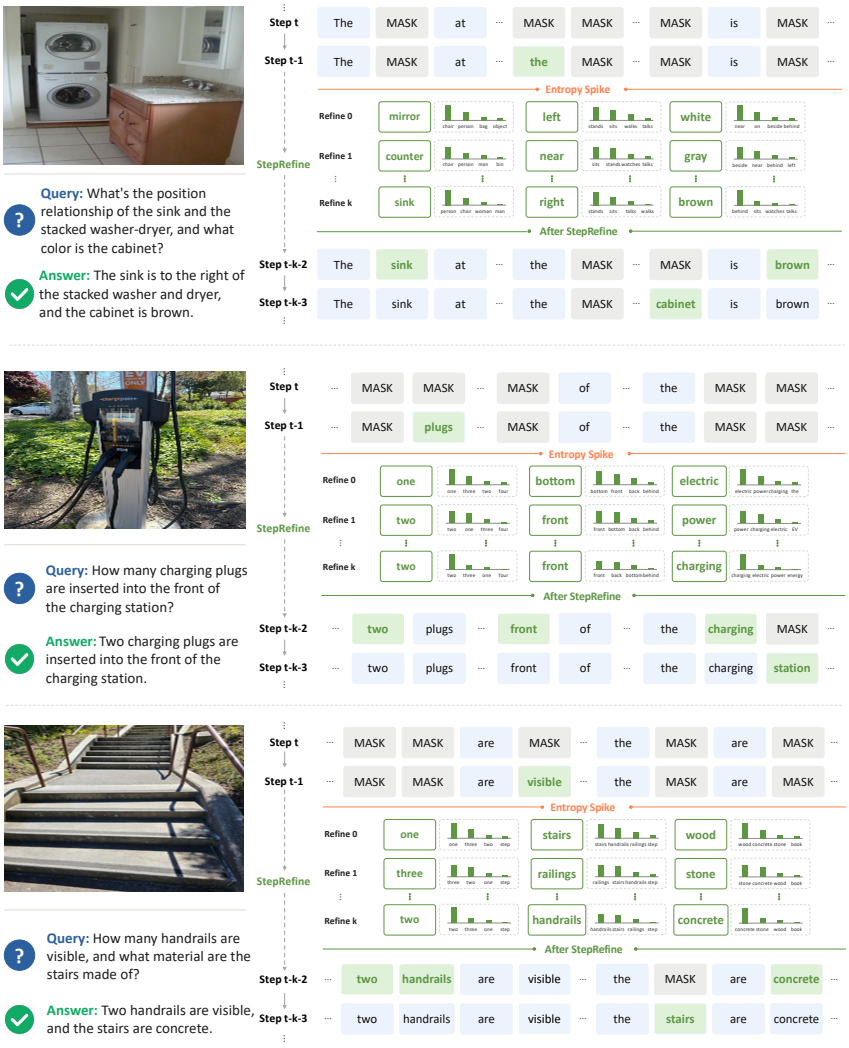


Figure 7: Example visualization of StepRefine on general reasoning & hallucination benchmarks.

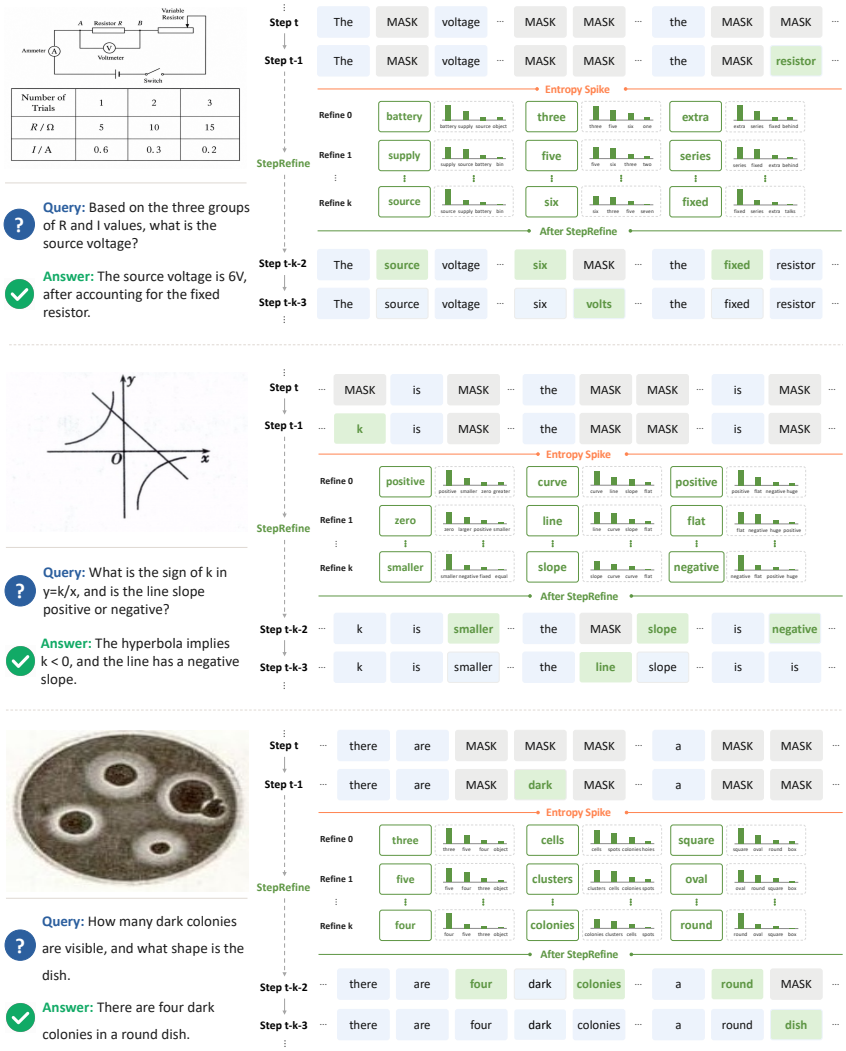


Figure 8: Example visualization of StepRefine on mathematical & scientific benchmarks.

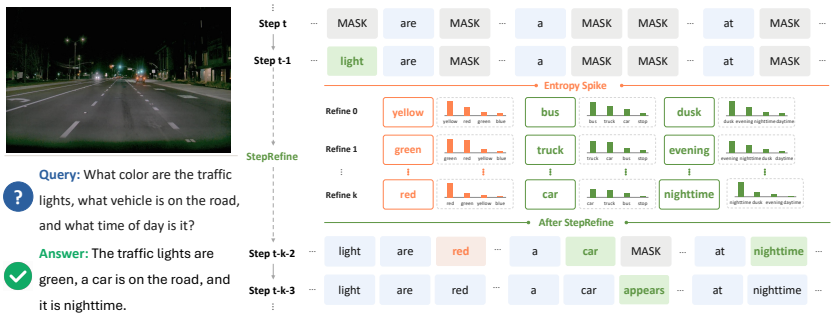


Figure 9: Example visualization of StepRefine on a challenging case where intra-step refinement fails to recover the correct answer.