INFERENCE-TIME SCALING IN DIFFUSION MODELS THROUGH ITERATIVE PARTIAL REFINEMENT

Taegu Kang
KAIST
rivertae9@kaist.ac.kr

Jaesik Yoon
KAIST & SAP
jaesik.yoon@kaist.ac.kr

Sungjin Ahn
KAIST & NYU
sungjin.ahn@kaist.ac.kr

ABSTRACT

Inference-time scaling has emerged as a major approach for improving reasoning capabilities, and has been increasingly applied to diffusion models. However, existing inference-time scaling methods for diffusion models typically rely on external verifiers or reward models to rank and select samples, limiting their scalability to settings where such evaluators are available and reliable. Moreover, while recent diffusion models perform *sequential* inference with region-wise, mixed-noise conditioning, inference-time scaling tailored to this setting remains relatively underexplored. We propose **Iterative Partial Refinement (IPR)**, an inference-time scaling method for sequential diffusion that requires no external verifier. Starting from an already-generated sample, IPR re-noises a subset of regions and regenerates them conditioned on the remaining regions, enabling the model to revise earlier decisions under a richer context than was available during the initial generation. This iterative partial refinement produces more globally consistent samples without external verification. On reasoning tasks requiring global constraint satisfaction, IPR consistently improves performance: on **MNIST Sudoku**, the valid solution rate increases from **55.8%** to **75.0%**. These results show that iterative partial refinement alone can serve as an effective inference-time scaling strategy for diffusion models in sequential, mixed-noise settings. Code is available at: https://github.com/ahn-ml/IPR

1 INTRODUCTION

Diffusion models are strong generators (Ho et al., 2020; Song et al., 2020), and a growing line of work studies *inference-time scaling*—improving generations by spending additional compute at test time (Singhal et al., 2025; Li et al., 2024b; Kim et al., 2025; Guo et al., 2025; Lee et al., 2025; Zhang et al., 2025b; He et al., 2025). Most existing approaches perform inference-time scaling by generating multiple candidates or trajectories and then ranking, selecting, or resampling them using an external reward model, verifier, or task-specific scoring function. While effective, this reliance restricts applicability to settings where such evaluators are unavailable or unreliable.

Recently, VFScale (Zhang et al., 2025a) takes an important step toward removing external evaluators by using the diffusion model's intrinsic energy as an internal scoring signal and scaling test-time compute through search over denoising trajectories. However, verifier-free scaling in this form is primarily developed for standard DDPM-style diffusion, where all regions are updated at the same noise level at each step.

Many realistic generation problems are inherently compositional: different parts of the output must agree with each other, so decisions made in one region constrain what is plausible in others. Such cross-region dependencies are often the norm rather than an exception, and they call for inference mechanisms that can exploit *asymmetric* conditioning between regions. Standard diffusion, with its uniform-noise updates, offers limited structure for exploiting such dependencies.

To address these limitations, recent works have moved toward modeling noise levels independently across different regions, decoupling the noise schedule from a single global timestep. This approach, which we refer to as **sequential diffusion** (Chen et al., 2024; Wewer et al., 2025; Wu et al., 2023; Zhang et al., 2024; Li et al., 2024a), allows different regions to evolve at varying noise levels so that

less-noisy regions provide stable context for generating noisier ones. By training on diverse region-wise noise configurations, these models learn mixed-noise conditional denoising behavior, making cross-region dependencies more explicit and enabling more structured conditional generation.

Despite this promise, inference-time scaling for sequential diffusion has received relatively less attention than its standard-diffusion counterpart. In particular, while sequential diffusion provides a natural mechanism for conditioning across regions, it remains unclear how to allocate additional inference-time compute to improve samples *without relying on external verifiers or rewards*. This gap motivates our work.

We propose **Iterative Partial Refinement (IPR)**, a simple inference-time scaling method for sequential diffusion models trained on arbitrary noise levels. Starting from an initial sample, IPR repeatedly selects a subset of regions, re-noises them, and regenerates them conditioned on the remaining regions. This operation allows inference to revisit earlier decisions and correct inconsistencies that would otherwise persist. IPR requires no additional training, no guidance, and no external verifier; it simply reuses the model's learned conditional distribution for self-improvement.

We evaluate IPR on globally constrained generation benchmarks: **MNIST Sudoku**, **Even Pixels**, and **Counting Polygons**. Across tasks, increasing the number of IPR iterations consistently improves constraint satisfaction and sample quality; on MNIST Sudoku HARD, IPR increases the valid solution rate from 55.8% to 75.0%. Overall, these results show that enabling revision during inference provides a simple way to scale sequential diffusion at inference time, improving global consistency without modifying the underlying model parameters.

Our contributions, framed in the context of recursive self-improvement, are as follows:

- **Inference-Time Refinement Strategy:** We introduce IPR, a method that revises generated samples by iteratively re-noising and regenerating sub-regions. By targeting the inference phase, we enable correction of global inconsistencies without permanent parameter updates.
- **Self-Improvement via Intrinsic Capability:** We demonstrate that the model can correct its own errors by leveraging its native mixed-noise conditional denoising capability. This approach achieves self-improvement using only the pre-trained model, avoiding the need for external reward models or verifiers.
- **Validation on Structured Generation:** We empirically validate IPR on three globally constrained benchmarks (MNIST Sudoku, Even Pixels, Counting Polygons), showing that simple iterative refinement significantly boosts constraint satisfaction rates and improves the quality of generations.

2 PRELIMINARY

2.1 DIFFUSION GENERATIVE MODELS

Let $\mathbf{x}_0 \in \mathbb{R}^D$ be a data sample drawn from $p_0(\mathbf{x}_0)$. Diffusion models (Ho et al., 2020; Song et al., 2020) and flow matching (Lipman et al., 2022) both construct a stochastic process $\{\mathbf{x}_t\}_{t=0}^{T}$ that continuously transforms data into noise. We refer to the continuous index $t \in [0, 1]$ as the *noise level*, where $t{=}0$ corresponds to clean data and $t{=}1$ to pure Gaussian noise. Given a noise schedule (α_t, σ_t), a pair of monotonic functions satisfying the boundary conditions $\alpha_0{=}1,\ \sigma_0{=}0$ and $\alpha_1{=}0,\ \sigma_1{=}1$ (i.e., the signal-to-noise ratio α_t/σ_t decreases monotonically from ∞ to 0), the noisy variable at noise level t is defined as the interpolation

$$\mathbf{x}_t = \alpha_t\, \mathbf{x}_0 + \sigma_t\, \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{1}$$

A neural network $\boldsymbol{\epsilon}_\theta$ is trained to recover the noise component via the denoising objective

$$\mathcal{L}(\theta) = \mathbb{E}_{t,\, \mathbf{x}_0,\, \boldsymbol{\epsilon}} \left[\left\| \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) - \boldsymbol{\epsilon} \right\|^2 \right]. \tag{2}$$

To generate samples, one draws $\mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and iteratively applies the learned reverse process $p_\theta(\mathbf{x}_{t-\Delta t} \mid \mathbf{x}_t)$, using either deterministic updates (Song et al., 2020) or stochastic sampling (Ho et al., 2020), progressing from $t{=}1$ down to $t{=}0$. We refer to this setup, where all regions share the same noise level and are denoised in parallel, as *standard diffusion* throughout the paper.

Figure 1: **Progressive refinement on MNIST Sudoku with IPR.** From left to right, Iterative Partial Refinement (IPR) progressively corrects erroneous cells (highlighted in red), resolving global constraint violations and converging to a valid Sudoku solution.

2.2 Sequential Diffusion Models

Unlike standard diffusion, where all regions share a single noise level, *sequential diffusion* (Chen et al., 2024; Wewer et al., 2025; Wu et al., 2023; Zhang et al., 2024; Li et al., 2024a) assigns each region its own noise level. Regions denoised earlier provide partially clean context for the remaining, noisier regions, making cross-region dependencies explicit.

Formally, let an output $\mathbf{x}$ be partitioned into N regions $\{x_1, \ldots, x_N\}$ (for images, each region is a patch). Let x_i^t denote region i at noise level t, and $\mathbf{t} = (t_1, \ldots, t_N)$ be a vector of per-region noise levels. Given a source configuration $\mathbf{t}'$, the model predicts the denoised output at a target configuration $\mathbf{t}$ ($t_i \leq t_i'$):

$$p_\theta\left(x_1^{t_1}, \ldots, x_N^{t_N} \,\middle|\, x_1^{t_1'}, \ldots, x_N^{t_N'}\right). \tag{3}$$

Training involves diverse pairs $(\mathbf{t}', \mathbf{t})$ so that the model learns to denoise regions conditioned on the rest. The choice of training noise distribution determines the model's capability: one can enforce a fixed generation order, or sample arbitrary mixed noise levels to enable the model to capture complex, order-agnostic dependencies.

Spatial Reasoning Models (SRMs) (Wewer et al., 2025) instantiate this framework for images, training on patch-wise mixed-noise configurations. At inference, SRMs select which patch to denoise next based on prediction uncertainty, denoising confident patches first to build reliable context, and we adopt SRMs as our sequential diffusion backbone.

3 Iterative Partial Refinement

Sequential diffusion follows a **one-pass generation trajectory**. Early in sampling, the context is formed under high noise and limited information, so stochastic updates may introduce small inconsistencies. Because later steps condition on this context without revisiting it, such inconsistencies can persist and propagate to the rest of the sequence. To address this limitation, we propose **Iterative Partial Refinement (IPR)**, an inference-time mechanism for sequential diffusion models trained on arbitrary noise levels. Starting from a generated sample, IPR repeatedly selects a subset of regions, adds noise to that subset, and regenerates it conditioned on the remaining regions.

Setup. We denote the output as $\mathbf{x} = (x_1, \ldots, x_N)$, where each x_i corresponds to a region. Sequential diffusion allows regions to reside at different noise levels, and the model p_θ is trained to perform conditional denoising under mixed noise levels.

Procedure. Given an initial sample $\mathbf{x}^{(0)}$ from standard sequential inference, IPR performs R refinement iterations. At each iteration $r = 1, \ldots, R$:

1. **Select:** Sample a random subset $\mathcal{M}^{(r)} \subset \{1, \ldots, N\}$ with $|\mathcal{M}^{(r)}| = \lfloor \alpha N \rfloor$, where α is the resampling ratio. Regions provided as fixed input conditions are excluded from the sampling pool.
2. **Re-noise:** Replace regions in $\mathcal{M}^{(r)}$ with random noise, i.e., sample $\mathbf{x}^{(r)}_{\mathcal{M}^{(r)}} \sim \mathcal{N}(0, I)$.

Algorithm 1 Iterative Partial Refinement (IPR)

Require: Trained sequential diffusion model p_θ, iterations R, resampling ratio α, fixed condition set $\mathcal{C}$

Initialize $\mathbf{x}^{(0)}$ via sequential diffusion inference
for $r = 1$ to R **do**
  Sample $\mathcal{M}^{(r)} \subset \{1, \dots, N\} \setminus \mathcal{C}$ uniformly at random, $|\mathcal{M}^{(r)}| = \lfloor \alpha(N - |\mathcal{C}|) \rfloor$
  Replace $\mathbf{x}^{(r-1)}_{\mathcal{M}^{(r)}}$ with random noise $\sim \mathcal{N}(0, I)$
  $\mathbf{x}^{(r)}_{\mathcal{M}^{(r)}} \sim p_\theta\big(\mathbf{x}_{\mathcal{M}^{(r)}} \mid \mathbf{x}^{(r-1)}_{\setminus \mathcal{M}^{(r)}}\big)$
  $\mathbf{x}^{(r)}_{\setminus \mathcal{M}^{(r)}} \leftarrow \mathbf{x}^{(r-1)}_{\setminus \mathcal{M}^{(r)}}$
end for
return $\mathbf{x}^{(R)}$

3. **Regenerate:** Conditionally regenerate the selected regions given the remaining context (where $\setminus \mathcal{M}^{(r)}$ denotes the complement $\{1, \dots, N\} \setminus \mathcal{M}^{(r)}$):

$$\mathbf{x}^{(r)}_{\mathcal{M}^{(r)}} \sim p_\theta\big(\mathbf{x}_{\mathcal{M}^{(r)}} \mid \mathbf{x}^{(r-1)}_{\setminus \mathcal{M}^{(r)}}\big), \tag{4}$$

$$\mathbf{x}^{(r)}_{\setminus \mathcal{M}^{(r)}} = \mathbf{x}^{(r-1)}_{\setminus \mathcal{M}^{(r)}}. \tag{5}$$

After R iterations, the final sample $\mathbf{x}^{(R)}$ is returned. If standard sequential inference requires T total denoising steps to generate all N regions, IPR adds $O(\alpha RT)$ steps, corresponding to regenerating αN regions per iteration over R rounds. Algorithm 1 summarizes the procedure.

Intuition. Repeating this process gives inference multiple opportunities to revise earlier mistakes. Re-noising part of the current sample undoes fixed decisions that were made under limited context, and lets the model regenerate that part given the now more complete surrounding regions. Over iterations, this can rebuild contexts closer to those seen during training, making it easier to correct inconsistencies and improve global consistency. In this way, reusing the model's own learned distribution for iterative refinement serves as an effective inference-time scaling strategy—achieving gains without requiring external verifiers or task-specific heuristics. A qualitative example of this progressive correction process on MNIST Sudoku is shown in Figure 1.

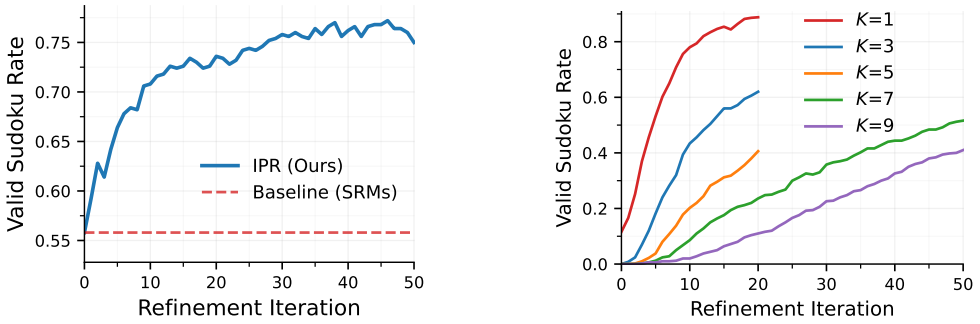
4 Experiments

4.1 Experimental Setup

Backbone and Benchmarks. We use **Spatial Reasoning Models (SRMs)** (Wewer et al., 2025) as the backbone, a general framework for sequential diffusion that supports patch-wise noise configurations. To ensure fair comparison, we adopt the same constrained generation benchmarks used in the original SRMs work: *MNIST Sudoku*, *Counting Polygons*, and *Even Pixels*. Each benchmark requires satisfying different types of global constraints.

Baseline and Evaluation. We compare IPR against the standard SRMs inference (without refinement) as the baseline. Common inference-time scaling strategies such as best-of-n sampling or pass@k require external verifiers to select or rank candidates; since our goal is to evaluate whether IPR can improve generation quality by better utilizing the learned distribution alone, we do not use such verifiers. All experiments use the same pretrained SRMs models without additional training.

Hyperparameters. Unless otherwise specified, we fix the resampling ratio $\alpha = 0.25$ and measure performance as a function of refinement iterations R. We use $R \leq 50$ in practice and do not employ early stopping. Detailed hyperparameters for each task are provided in Appendix D.



(a) Refinement on HARD Sudoku

(b) Refinement on K-Corrupted Sudoku

Figure 2: **IPR on MNIST Sudoku.** (a) Valid Sudoku rate improves consistently with IPR iterations on the HARD setting. (b) Recovery rate from corrupted grids with K initially swapped cell pairs, showing robustness even under severe corruption.

4.2 MNIST Sudoku

MNIST Sudoku follows standard Sudoku rules: each row, column, and 3×3 subgrid must contain the digits 1–9 without repetition. The key difference is that each cell is represented as a 28×28 MNIST digit image, requiring the model to generate visually coherent digits while satisfying combinatorial constraints. We use SRMs trained on 81 patches (one per cell) to generate the full grid (see Appendix C.2).

HARD Setting. We evaluate on the HARD setting, where only 0–26 cells are given as clues. During IPR, clue cells are fixed and excluded from resampling. As shown in Figure 2a, the baseline SRMs achieves a 55.8% valid Sudoku rate, while IPR progressively improves this to over 75%. This demonstrates that iteratively re-noising and regenerating parts of the sample allows the model to correct inconsistencies and improve global constraint satisfaction (see Figure 6).

Robustness to K-Corrupted Sudoku. To test whether IPR can recover from more severe inconsistencies, we start from valid Sudoku grids and introduce controlled corruption by randomly swapping K pairs of cells ($K \in \{1, 3, 5, 7, 9\}$), then allow all cells to be modified during refinement. As shown in Figure 2b, for small K, validity is quickly restored within a few iterations, while for larger K, recovery is slower but continues to improve up to 50 iterations. This confirms that IPR can recover global consistency even from severely corrupted states, given sufficient iterations (see Figure 7).

4.3 Counting Polygons

Counting Polygons requires generating a 128×128 image containing an FFHQ (Karras et al., 2019) face background, multiple polygons, and two digits. The digits specify the number of polygons and their vertex count, and the generated polygons must match these values exactly. This is an unconditional constrained generation task with no input conditions provided (see Appendix C.3 for details). We use SRMs trained on 256 patches of size 8×8.

Evaluation Metrics. We measure two metrics: *Number Match Accuracy*, which checks whether the generated digits match the actual polygon count and vertex type, and *Vertex Uniformity*, which checks whether all polygons in the image share the same number of vertices.

Results. As shown in Figure 3, both metrics improve with IPR iterations. The baseline achieves 15.4% on Number Match Accuracy and 98.8% on Vertex Uniformity; after 50 iterations, these improve to 27.4% (+12 pp) and 100%, respectively. The model adjusts the digits to match the polygons, iteratively refining the context to satisfy the constraints. Qualitatively, we also observe that the FFHQ background and the polygon shapes become clearer alongside constraint satisfaction, suggesting that IPR improves overall image coherence beyond just the constrained elements (see Figure 8).

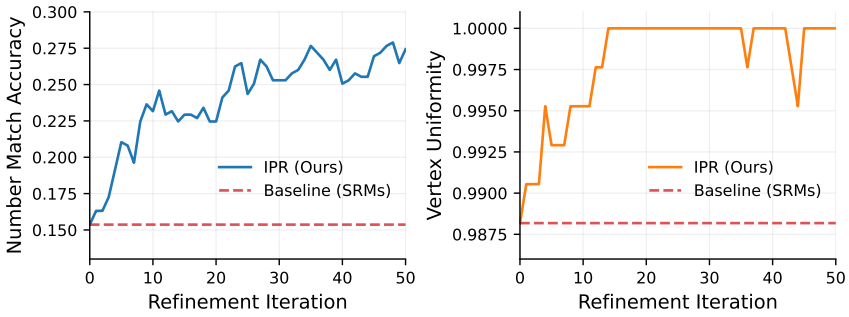


Figure 3: **IPR on Counting Polygons.** (Left) *Number Match Accuracy* improves as iterations increase. (Right) *Vertex Uniformity* reaches 100% within 50 iterations.

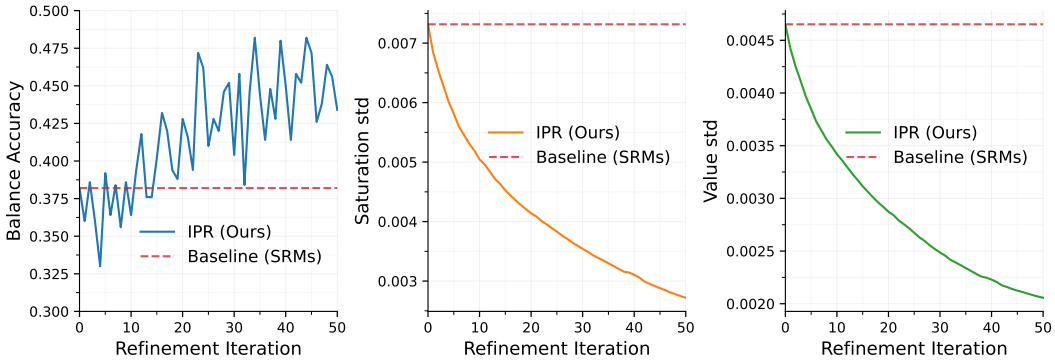


Figure 4: **IPR on Even Pixels.** Balance Accuracy (left) and color consistency measured by Saturation/Value std (middle, right) both improve with IPR iterations.

4.4 Even Pixels

Even Pixels requires generating a 32×32 image with exactly two colors, each occupying exactly half of the pixels. This is an unconditional constrained generation task that tests whether the model can satisfy a global counting constraint (see Appendix C.4). We use SRMs trained on patches of size 4×4.

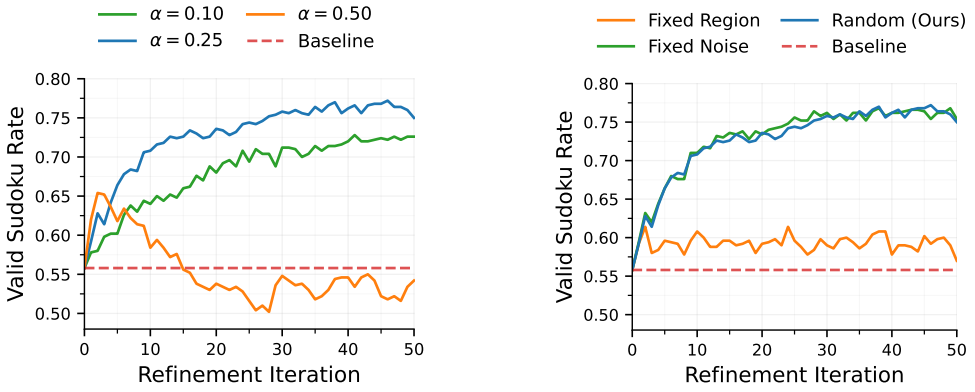
Evaluation Metrics. We evaluate two aspects: *Balance Accuracy* measures whether the two colors each occupy exactly 50% of the pixels, and *Saturation/Value Std* measures color consistency within each region—lower values indicate more uniform colors without noisy outliers.

Results. As shown in Figure 4, Balance Accuracy improves by approximately 5% after 50 IPR iterations. Beyond the balance constraint itself, the color consistency within each region also improves significantly. The Saturation/Value Std metrics capture this effect. Over 50 iterations, Saturation Std drops from 0.0073 to 0.0027 (63% reduction) and Value Std from 0.0047 to 0.0021 (55% reduction), indicating that IPR produces cleaner, more uniform colors within each region. These results show that IPR improves both constraint satisfaction and overall image quality (see Figure 9).

4.5 Ablation Study

We conduct ablation studies on MNIST Sudoku (HARD setting) to analyze the key design choices of IPR. Results are shown in Figure 5.

Resampling Ratio. The resampling ratio α controls the fraction of regions re-noised at each iteration. As shown in Figure 5a, setting $\alpha = 0.25$ yields the best performance, improving the valid Sudoku rate from 55.8% to 75.0% after 50 iterations. A smaller ratio ($\alpha = 0.10$) leads to slower but steady improvement, reaching 72.6%. Conversely, $\alpha = 0.50$ causes performance degradation



(a) Effect of resampling ratio α

(b) Effect of Fixed Noise and Region

Figure 5: **Ablation studies on IPR hyperparameters.** Valid Sudoku rate (%) on MNIST Sudoku HARD setting. (a) Resampling ratio $\alpha = 0.25$ performs best. (b) Fixed Noise performs comparably to random noise, while Fixed Region selection degrades performance.

to 54.2%, falling below the baseline. This indicates that resampling too large a fraction discards valuable context along with errors, undermining the refinement process.

Fixed Noise vs. Random Noise. We examine whether IPR requires fresh noise sampling at each iteration or can function with fixed noise. Instead of sampling from $\mathcal{N}(0, I)$ at each re-noising step, we fix the noise to the original noise used during the initial SRMs generation. As shown in Figure 5b, both variants achieve comparable performance, indicating that the choice of noise has minimal impact on IPR. This suggests that the *context* provided by already-generated regions, rather than the specific noise realization, is the primary factor driving refinement in sequential diffusion.

Fixed Region Selection. We further investigate whether the randomness in region selection is essential for IPR. Instead of randomly selecting regions to re-noise at each iteration, we fix the selection to the regions chosen in the first iteration. As shown in Figure 5b, this variant achieves only 57.0% at 50 iterations, showing negligible improvement over the baseline (55.8%). This confirms that varying the resampled regions across iterations is crucial for effective refinement: fixing the region selection prevents the model from propagating corrections across different regions, limiting the refinement to a static subset of the grid.

5 Related Work

A growing line of work studies inference-time scaling for diffusion models (Singhal et al., 2025; Li et al., 2024b; Yoon et al., 2025a;b; Lee et al., 2025; Zhang et al., 2025b; He et al., 2025; Zhang et al., 2025a). For example, Guo et al. (2025) proposes a tree-search-based path steering method that branches into multiple candidates at each denoising step and selects among them using a value function, while Lee et al. (2025) expands exploration by repeatedly selecting the top-K particles and cycling them back to an intermediate noise level. SMC/Feynman–Kac approaches similarly maintain multiple particles and perform reward-based reweighting and resampling at intermediate stages to bias sampling toward high-reward regions (Singhal et al., 2025; Kim et al., 2025). Despite their differences, these methods commonly rely on an external reward model or verifier (or an objective-based scoring function) to rank, select, or preserve better samples, which limits applicability to tasks where such verifiers are unavailable or unreliable.

In contrast, Zhang et al. (2025a) (VFScale) aims for verifier-free test-time scaling by using the diffusion model's intrinsic energy as an internal scoring signal. It trains the energy to correlate with sample quality, then scales test-time compute by searching over denoising trajectories guided by this score. However, VFScale ultimately relies on a single scalar score to drive search and selection. For problems with complex global constraints, such a score may not reliably capture constraint

satisfaction and long-range cross-region consistency, limiting the practical reach of score-driven scaling.

Unlike VFScale, **IPR** removes scoring and candidate selection altogether. Instead, IPR scales compute by repeatedly *editing* a single sample: it selectively re-noises a subset of regions and regenerates them conditioned on the remaining regions under the mixed-noise structure of sequential diffusion. By keeping parts of the sample revisable after they are generated, IPR provides verifier- and reward-free inference-time scaling tailored to globally constrained generation in sequential diffusion.

6 Discussion and Limitations

Computational cost. IPR increases inference cost linearly with the number of refinement iterations. With α=0.25 and R=50, the total computation is roughly $13\times$ that of a single generation pass. This trade-off is inherent to inference-time scaling: improved quality comes at the expense of additional compute. In practice, the cost can be controlled by choosing smaller R or α depending on the application's latency requirements.

Sensitivity to resampling ratio. The resampling ratio α must be carefully chosen. When α is too large (α=0.50), performance degrades below the baseline because too much useful context is discarded along with errors. This suggests that IPR's effectiveness relies on preserving a sufficient amount of correct context at each iteration to guide regeneration.

Future directions. IPR currently selects regions uniformly at random. More informed strategies—such as targeting regions that violate constraints or learning a state-dependent selection policy—may improve efficiency. Adapting the resampling ratio or re-noising strength over iterations is another avenue for finer-grained refinement.

7 Conclusion

We introduced **Iterative Partial Refinement (IPR)**, an inference-time scaling method for sequential diffusion models trained on arbitrary noise levels. IPR repeatedly selects a subset of regions, re-noises only the selected regions, and regenerates them conditioned on the remaining regions. This keeps parts of the sample revisable during sampling, providing a simple way to correct inconsistencies that can persist under the default one-pass generation trajectory. IPR requires no additional training, no guidance, and no external verifier; it leverages the model's mixed-noise denoising capability to revise a generated sample and improve global consistency at inference time.

Across three constrained generation benchmarks, IPR consistently improves constraint satisfaction—raising the valid Sudoku rate from 55.8% to over 75%, Number Match Accuracy from 15.4% to 27.4%, and reducing color outliers in Even Pixels—while also improving overall visual quality. By randomly varying which regions are resampled, IPR propagates contextual improvements across the entire sample over iterations, rather than refining only a fixed subset. The resampling ratio is critical to balance preserving useful context with allowing meaningful revision.

Acknowledgments

This work was supported by Basic Research Laboratory Program (No. RS-2024-00414822) through the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT). This work was also supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2024-00509279, Global AI Frontier Lab). We thank the members of the Machine Learning and Mind Lab (MLML) for their valuable discussions and assistance.

REFERENCES

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.

Yingqing Guo, Yukang Yang, Hui Yuan, and Mengdi Wang. Training-free guidance beyond differentiability: Scalable path steering with tree search in diffusion and flow models. *arXiv preprint arXiv:2502.11420*, 2025.

Haoran He, Jiajun Liang, Xintao Wang, Pengfei Wan, Di Zhang, Kun Gai, and Ling Pan. Scaling image and video generation via test-time evolutionary search. *arXiv preprint arXiv:2505.17618*, 2025.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

Sunwoo Kim, Minkyu Kim, and Dongmin Park. Test-time alignment of diffusion models without reward over-optimization. *arXiv preprint arXiv:2501.05803*, 2025.

Gyubin Lee, Bao N Nguyen Truong, Jaesik Yoon, Dongwoo Lee, Minsu Kim, Yoshua Bengio, and Sungjin Ahn. Adaptive inference-time scaling via cyclic diffusion search. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37: 56424–56445, 2024a.

Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gokcen Eraslan, Surag Nair, Tommaso Biancalani, Shuiwang Ji, Aviv Regev, Sergey Levine, et al. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. *arXiv preprint arXiv:2408.08252*, 2024b.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models. *arXiv preprint arXiv:2501.06848*, 2025.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Christopher Wewer, Bart Pogodzinski, Bernt Schiele, and Jan Eric Lenssen. Spatial reasoning with denoising models. *arXiv preprint arXiv:2502.21075*, 2025.

Tong Wu, Zhihao Fan, Xiao Liu, Hai-Tao Zheng, Yeyun Gong, Jian Jiao, Juntao Li, Jian Guo, Nan Duan, Weizhu Chen, et al. Ar-diffusion: Auto-regressive diffusion model for text generation. *Advances in Neural Information Processing Systems*, 36:39957–39974, 2023.

Jaesik Yoon, Hyeonseo Cho, Doojin Baek, Yoshua Bengio, and Sungjin Ahn. Monte carlo tree diffusion for system 2 planning. In *Forty-second International Conference on Machine Learning*, 2025a.

Jaesik Yoon, Hyeonseo Cho, Yoshua Bengio, and Sungjin Ahn. Fast monte carlo tree diffusion: 100x speedup via parallel sparse planning. *arXiv preprint arXiv:2506.09498*, 2025b.

Tao Zhang, Jia-Shu Pan, Ruiqi Feng, and Tailin Wu. Vfscale: Intrinsic reasoning through verifier-free test-time scalable diffusion model. *arXiv preprint arXiv:2502.01989*, 2025a.

Xiangcheng Zhang, Haowei Lin, Haotian Ye, James Zou, Jianzhu Ma, Yitao Liang, and Yilun Du. Inference-time scaling of diffusion models through classical search. *arXiv preprint arXiv:2505.23614*, 2025b.

Zihan Zhang, Richard Liu, Rana Hanocka, and Kfir Aberman. Tedi: Temporally-entangled diffusion for long-term motion synthesis. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024.

A USE OF LARGE LANGUAGE MODELS

We used large language models only for grammar checking and sentence correction.

B QUALITATIVE RESULTS

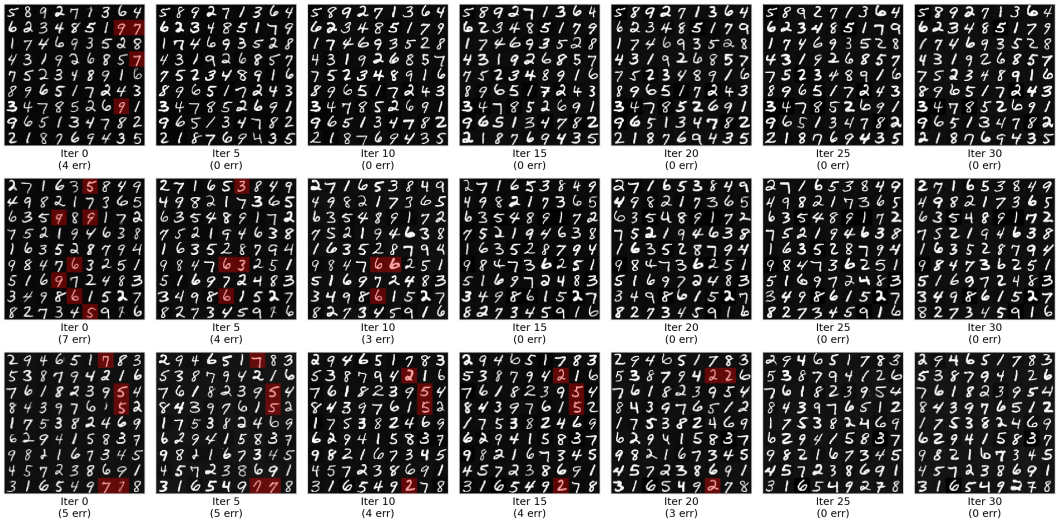


Figure 6: **Generated examples on MNIST Sudoku (HARD).** Each column shows the generated image at a different IPR iteration, progressing from left to right. Red backgrounds indicate cells that violate Sudoku constraints; violations decrease as refinement progresses.

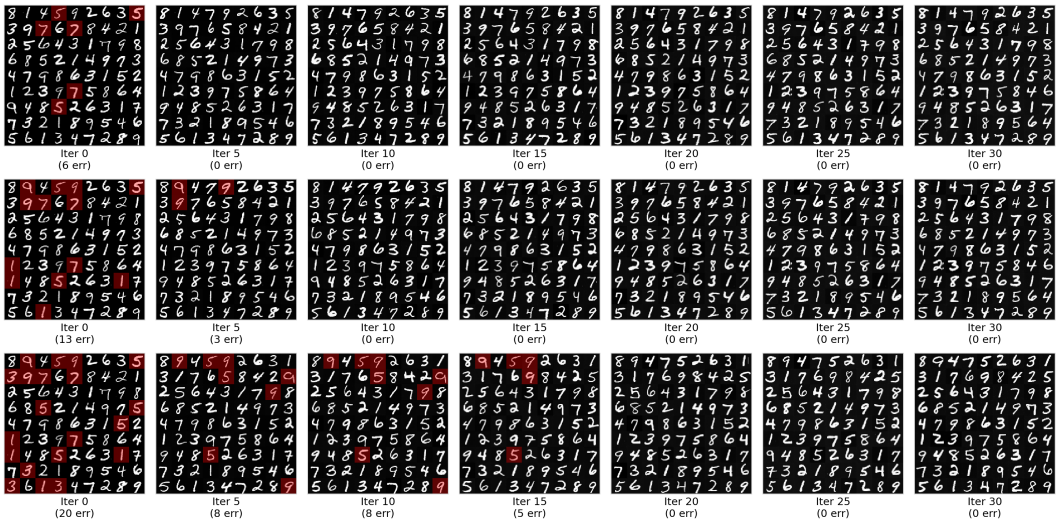


Figure 7: **Generated examples on K-Corrupted Sudoku ($K \in \{1, 3, 5\}$).** Each row corresponds to $K = 1, 3, 5$ from top to bottom. Each column shows the generated image at a different IPR iteration, progressing from left to right. Red backgrounds indicate cells that violate Sudoku constraints; violations decrease as refinement progresses.

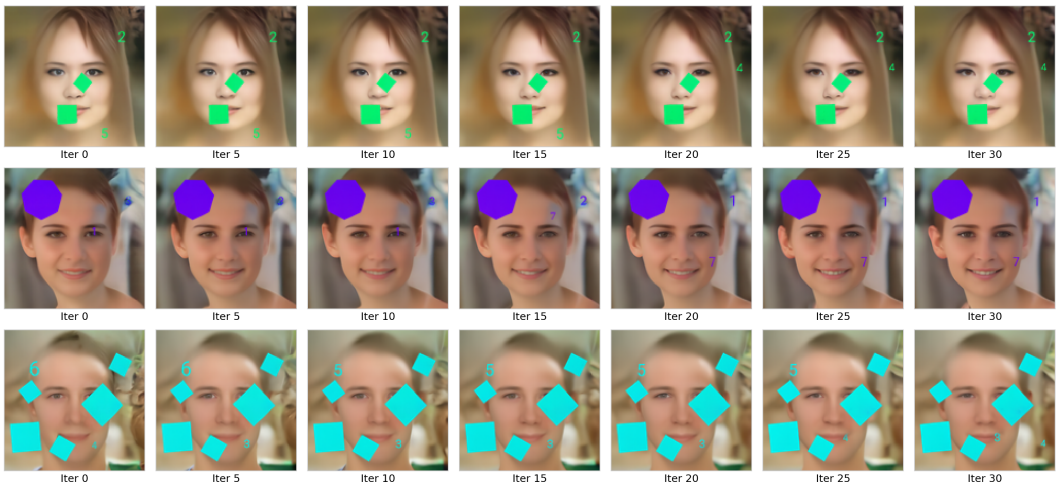


Figure 8: **Generated examples on Counting Polygons.** Each column shows the generated image at a different IPR iteration, progressing from left to right. The digit and polygon count converge to match the constraint.

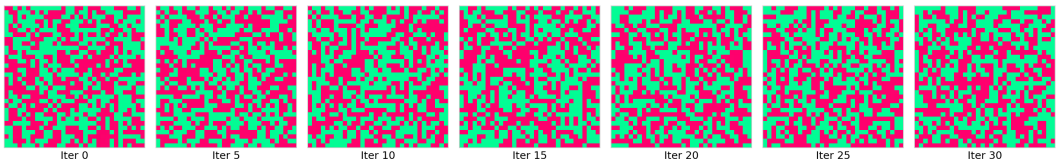


Figure 9: **Generated examples on Even Pixels.** Each column shows the generated image at a different IPR iteration, progressing from left to right. Color regions become more uniform and balanced across iterations.

C Experiment Details

C.1 Spatial Reasoning Models

We use SRMs (Wewer et al., 2025) as our sequential diffusion backbone for inference.

We use the pretrained models provided by the authors. Checkpoints were downloaded from `https://github.com/Chrixtar/SRM/releases`. The specific checkpoints used are:

- **MNIST Sudoku**: `ms1000_28/paper/checkpoints/last.ckpt`
- **Counting Polygons**: `cp_ffhq_8/paper/checkpoints/last.ckpt`
- **Even Pixels**: `ep_4/paper/checkpoints/last.ckpt`

C.2 MNIST Sudoku

C.2.1 Task Description

MNIST Sudoku is a constrained generation benchmark proposed by Wewer et al. (2025) that combines the visual complexity of handwritten digit generation with the combinatorial constraints of Sudoku. Each sample is a 9×9 grid where every cell contains a 28×28 MNIST digit image. The model must generate digits that are both visually realistic and satisfy standard Sudoku rules: each row, column, and 3×3 sub-grid must contain the digits 1–9 without repetition.

C.2.2 Experimental Setting

The benchmark supports varying difficulty by controlling the number of pre-filled hint cells N_{hint}. We adopt the **HARD** setting from Wewer et al. (2025), where N_{hint} is sampled uniformly from

$\{0, \ldots, 26\}$ for each sample. The positions of the hint cells are selected uniformly at random from the 81 cells in the grid, and the remaining $81 - N_{\text{hint}}$ cells are masked and must be generated by the model. This setting covers a wide range of difficulty, from nearly empty grids (purely generative) to moderately constrained puzzles (completion). During IPR, hint cells are fixed and excluded from resampling.

C.2.3 Evaluation Methodology

Since the output consists of images, we first map each generated cell to a digit class $d \in \{1, \ldots, 9\}$ using a pre-trained CNN classifier provided by the benchmark (Wewer et al., 2025). Let G denote the resulting 9×9 symbolic grid. A generated sample is considered valid if and only if G contains no duplicate digits in any row, column, or sub-grid. We report the **Sudoku Validity Rate**, i.e., the percentage of valid solutions over the test set.

C.3 Counting Polygons

C.3.1 Task Description

Counting Polygons is an unconditional constrained generation benchmark proposed by Wewer et al. (2025). The model must generate a 128×128 image containing an FFHQ face background, multiple polygons, and two digits. The two digits specify the number of polygons and their vertex count, respectively, and the generated polygons must match these values exactly. Crucially, no input conditions are provided: the model must jointly generate all components—background, polygons, and digits—such that they are internally consistent.

C.3.2 Training Dataset Construction

The training dataset is constructed by Wewer et al. (2025) as follows. Each sample is produced by overlaying randomly generated geometric polygons onto face images from the FFHQ dataset (Karras et al., 2019), which serve as diverse and realistic backgrounds:

- **Background**: A 128×128 face image is sampled from the FFHQ dataset.
- **Object Sampling**: The number of polygons (N_{poly}) and the number of vertices per polygon (N_{vert}) are sampled uniformly from predefined ranges (e.g., $N_{\text{vert}} \in [3, 7]$).
- **Appearance & Placement**: Polygons are drawn with random positions, scales, and rotations. To ensure visibility against the complex background, the fill color is dynamically selected by computing the HSV color histogram of the background image and choosing the hue with the minimum frequency, thereby maximizing color contrast.

C.3.3 Evaluation Methodology

Since this is an unconditional generation task, evaluation focuses on the internal consistency of the generated image. We employ a fine-tuned ResNet-50 classifier, provided by the benchmark (Wewer et al., 2025), to predict three attributes from the generated image: polygon count, vertex count, and uniformity. We report two metrics: *Number Match Accuracy*, which checks whether the generated digits match the actual polygon count and vertex type, and *Vertex Uniformity*, which checks whether all polygons in the image share the same number of vertices.

C.4 Even Pixels

C.4.1 Task Description

Even Pixels is a constrained generation benchmark designed to test the model's ability to satisfy precise constraints on pixel-level statistics. Each sample is a 32×32 image composed of exactly two distinct colors. The global constraint is that each color must occupy exactly 50% of the pixels.

C.4.2 Dataset Construction

The Even Pixels dataset (Wewer et al., 2025) is constructed to isolate the challenge of generating a balanced hue distribution while maintaining constant saturation and value.

- **Color Selection**: Images are generated in the HSV color space. A base hue h_1 is sampled uniformly from $[0, 0.5)$. The second hue is set to $h_2 = h_1 + 0.5$, ensuring the two colors are separated by 180° in the color wheel to maximize contrast. Saturation (S) and Value (V) are fixed to constant values (e.g., $S = 1.0, V = 0.7$) across the entire image.
- **Pixel Assignment**: A random binary mask $M \in \{0, 1\}^{32\times32}$ is generated such that $\sum_{i,j} M_{i,j} = 512$. Pixels corresponding to 0 are assigned h_1, and those corresponding to 1 are assigned h_2.

C.4.3 Evaluation Methodology

Evaluating the generated images requires verifying whether the two dominant colors appear in exactly equal proportions. The benchmark employs a histogram-based clustering approach to robustly count pixels for each color without assuming predefined color values.

Procedure.

1. **Color Space Conversion**: The generated RGB image is converted to the HSV color space. The analysis focuses on the Hue (H) channel.
2. **Histogram Analysis**: A histogram of the Hue channel (256 bins) is computed, and the two most prominent peaks, p_1 and p_2, are identified.
3. **Dynamic Thresholding**: Decision boundaries b_1, b_2 between the two colors are determined by finding the midpoints between the peaks in the circular Hue space:

$$b_1 = \frac{p_1 + p_2}{2}, \quad b_2 = \frac{p_1 + p_2 + 256}{2} \pmod{256}. \tag{6}$$

4. **Pixel Counting**: All pixels with hue values falling between b_1 and b_2 are assigned to one cluster. Let N_{c1} be the pixel count of this cluster.

Metrics. **Balance Accuracy** is defined as the percentage of images where the pixel count error $|N_{c1} - 512|$ is exactly 0. Additionally, **Saturation/Value Std** is measured to assess internal color consistency within each region.

D Inference Hyperparameters

We summarize the inference-time hyperparameters used in our experiments.

D.1 Parameter Definitions

- `init_overlap_ratio`: Controls the degree of scheduling overlap between regions during the initial generation (SRM). Larger values allow more concurrent denoising.
- `init_steps_per_patch`: The number of denoising steps allocated to each region during the initial generation.
- `ipr_overlap_ratio`: The scheduling overlap ratio used during IPR refinement.
- `ipr_steps_per_patch`: The number of denoising steps applied to each re-sampled region during IPR refinement.
- `stochasticity`: Controls the amount of randomness injected during diffusion sampling (for both initial generation and refinement).
- `resampling_ratio`: Controls the fraction of regions re-noised at each IPR iteration.
- `iteration`: The total number of IPR refinement iterations performed.

D.2 TASK-SPECIFIC SETTINGS

For all experiments, we fixed the following hyperparameters for each task and varied `resampling_ratio` and `iteration` to analyze their effects.

Table 1: Fixed inference hyperparameters for each task.

Hyperparameter	**MNIST Sudoku**	**Counting Polygons**	**Even Pixels**
`init_overlap_ratio`	0.0	0.9	0.9
`init_steps_per_patch`	3	10	30
`ipr_overlap_ratio`	0.8	0.9	0.9
`ipr_steps_per_patch`	10	10	30
`stochasticity`	0.5	0.5	0.5