DETECTION LIMITS AND STATISTICAL SEPARABILITY OF TREE RING WATERMARKS IN RECTIFIED FLOW-BASED TEXT-TO-IMAGE GENERATION MODELS

Ved Umrajkar*
Department of Mathematics
Indian Institute of Technology Roorkee
v_umrajkar@ma.iitr.ac.in

Aakash Kumar Singh*
Mehta Family School of Data Science and Artificial Intelligence
Indian Institute of Technology Roorkee
aakash_ks@mfs.iitr.ac.in

ABSTRACT

Tree-Ring Watermarking is a significant technique for authenticating AI-generated images. However, its effectiveness in rectified flow-based models remains unexplored, particularly given the inherent challenges of these models with noise latent inversion. Through extensive experimentation, we evaluated and compared the detection and separability of watermarks between SD 2.1 and FLUX.1-dev models. By analyzing various text guidance configurations and augmentation attacks, we demonstrate how inversion limitations affect both watermark recovery and the statistical separation between watermarked and unwatermarked images. Our findings provide valuable insights into the current limitations of Tree-Ring Watermarking in the current SOTA models and highlight the critical need for improved inversion methods to achieve reliable watermark detection and separability. The official implementation, dataset release and all experimental results are available at this **link**.

1 INTRODUCTION

The rapid advancement of generative AI models has raised pressing concerns about the authenticity and provenance of digital content. While watermarking techniques for AI-generated images have emerged as a promising solution, their effectiveness heavily depends on reliable detection and clear separability between watermarked and non-watermarked content. Recent approaches like Tree Ring Watermarking (Wen et al., 2024) have shown promise, but their effectiveness remains unexplored for newer architectures.

Recent advances in text-conditioned generative models, particularly rectified flow models, have demonstrated remarkable capabilities in high-resolution image synthesis. Unlike traditional diffusion models, rectified flows model transportation between distributions through linear interpolation of marginals, enabling efficient sampling with fewer discretization steps. However, the implications of these architectural differences on watermarking mechanisms remain unexplored.

This work investigates watermark detection and separability in flow-based generative models, focusing on the FLUX model. We analyze two critical aspects: the reconstruction and detection of embedded watermarks through noise latent inversion, and the statistical separability between watermarked and non-watermarked distributions under various attack scenarios. Our findings demonstrate that while flow-based models present unique challenges for watermark detection, careful consideration of model configuration and inversion methodology can achieve reliable separation.

2 RELATED WORKS

Watermarking Approaches. Random seed modification watermarks like Tree Ring (Wen et al., 2024) and RingID (Ci et al., 2024) embed a known key into the noise latent that is the starting point for image generation using diffusion. The effectiveness of these approaches has been systematically

*Equal contribution. Data Science Group, IIT Roorkee.

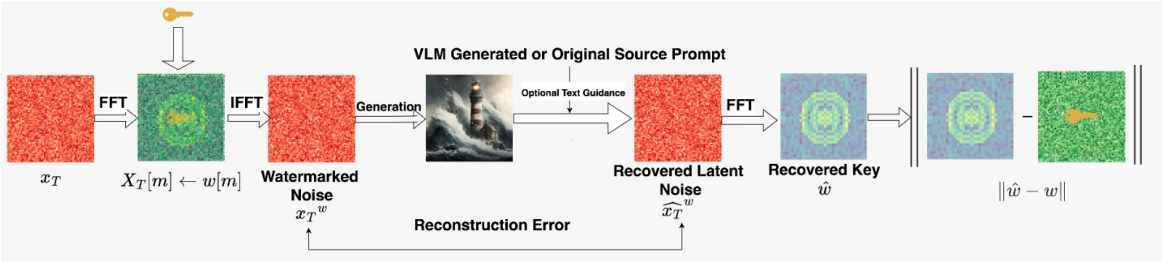

Figure 1: Watermarking workflow for both FLUX and Stable Diffusion

evaluated through benchmarks like Waves (An et al.), which provides standardized attack scenarios for robustness assessment.

Inversion Methods. Recent work has advanced latent inversion techniques, with Hong et al. (2024) demonstrating significantly improved Tree-ring watermark detection using higher-order inversion algorithms compared to naive DDIM inversion. While their work showed remarkable detection accuracy on traditional diffusion models using DPM-Solver++ (Lu et al., 2022), the effectiveness of these techniques on newer rectified flow-based architectures remains unexplored. Our work extends this analysis to flow-based models, providing insights into watermark detection across different architectures.

3 Methodology

3.1 Preliminaries

Notation Let x_t denote the noisy image at timestep t, with x_0 and x_T representing the generated image and initial noise latent respectively. In frequency domain, X_T denotes the Fourier transform of x_T. The data and noise distributions are denoted by π_0 and $\pi_1 \sim \mathcal{N}(0, I)$ respectively. We denote the parameters of a neural network by θ that can be used for adequate prediction targets and c for the text prompt used to guide the text-to-image generation models.

Generation and Inversion Framework Image generation involves producing an image x_0 from random noise x_T, while inversion aims to reconstruct the original noise latent ($\hat{x}_T$) from an input generated latent. The noise map obtained from inversion should generate the exact same image x_0 by sampling using diffusion. Both processes involve solving Ordinary Differential Equations (ODEs) through numerical integration, which can be done using first-order methods, such as Euler's method. This is usually the case with models like Flux, which follow a more linear trajectory. We focus on two primary approaches: traditional Denoising Diffusion Models (DDMs) and the newer Rectified Flow Transformer models (Liu et al., 2023; Lipman et al., 2022; Liu, 2022; Esser et al., 2024).

FLUX Transformers trained with the flow-matching objective have recently achieved state-of-the-art results in image generation (Esser et al., 2024). We utilize the open weights FLUX.1-dev model, which employs a Diffusion Transformer (DiT) (Peebles & Xie, 2023) architecture and differs fundamentally from traditional DDMs like Stable Diffusion (Rombach et al., 2022b) in its approach to generation and inversion. FLUX is based on rectified flows, which construct a transportation between the source distribution π_1 (typically standard Gaussian) and the target data distribution π_0 through the following ODE:

$$\frac{dx_t}{dt} = v_t(x_t, t, c)dt, \quad X_0 \sim \pi_0, \quad t \in [0, 1]$$

where v_t is a time-dependent velocity field parameterized by the neural network. A key property of rectified flows is that the marginal distribution at time t follows a linear interpolation between x_0 and x_1:

$$x_t \sim (1 - t)x_0 + tx_1$$

Algorithm 1 Tree Ring Watermarking Procedure

Require: Image dimensions (h, w), watermark channel c_w, radius r, batch size b, seed s
Ensure: Watermarked noise x_T, watermark key w, watermark mask m
1: $\boldsymbol{x_T} \sim \mathcal{N}(0, I)$ *(Sample initial Gaussian noise)*
2: Generate watermark mask m using radius r and channel c_w
3: Generate watermark key w using pattern and seed s
4: Compute FFT of noise: $X_T \leftarrow \text{FFT}(\boldsymbol{x_T})$
5: Apply watermark: $\hat{X_T}[m] \leftarrow w[m]$
6: Compute inverse FFT: $x_T^w \leftarrow \text{IFFT}(\hat{X_T})$
return x_T^w, w, m

This property enables efficient sampling with relatively few discretization steps. For generation, the ODE is solved forward, while inversion uses the backward Euler method:

$$\mathbf{x}_{t_i} = \mathbf{x}_{t_{i-1}} - (t_i - t_{i-1})\, \mathbf{v}_\theta(\mathbf{x}_{t_i}, t_i, c)$$

This contrasts with the usual first-order (naive) DDIM inversion:

$$x_{t+1} = \sqrt{\bar{\alpha}_{t+1}}\, \hat{x}_0^t + \sqrt{1 - \bar{\alpha}_{t+1}}\, \epsilon_\theta(x_t, \sigma_t, c)$$

For more details and a thorough discussion, refer Appendix Section A.

3.2 Approach

Tree-Ring Watermark Embedding The Tree-Ring watermark embedding follows a Fourier space modification approach:

$$\mathbf{x}_T = \mathcal{F}^{-1}(\mathbf{X}_T), \quad \text{where } \mathbf{X}_T[m] = w[m] \tag{1}$$

Here, w represents the ring-pattern watermark key, m the circular mask in channel $\mathcal{C}_w$, and $\mathcal{F}^{-1}$ the inverse Fourier transform. The corresponding recovered key is denoted by $\hat{w}$ which is obtained from the fourier transform of the recovered watermarked noise latent, i.e. $\hat{w} = \mathcal{F}(\widehat{x_T}^w)$.

The complete watermarking procedure is detailed in Algorithm 1.

VLM Generated Prompt for Inversion Guidance For real-world scenarios where original prompts might be unavailable, we employ Qwen2-VL-2B-Instruct (Wang et al., 2024) to generate image-grounded captions as alternative prompts. This approach enables evaluation of both prompt-free and prompt-guided inversion scenarios.

Evaluation We evaluate watermark separability by analyzing the distribution of Fourier space distances $d = \|\hat{w} - w\|$ between reconstructed ($\hat{w}$) and original (w) watermark patterns. To quantify the statistical separation between different configurations (with/without prompts, with/without attacks), we compute the Symmetric KL Divergence between their respective distance distributions: $\mathcal{D}_{\text{SKL}}(P\|Q) = \frac{1}{2}[\mathcal{D}_{\text{KL}}(P\|Q) + \mathcal{D}_{\text{KL}}(Q\|P)]$, where P and Q represent the distance distributions for different experimental configurations.

4 Experiments

4.1 Experimental Setup

To ensure consistency, we used a fixed global random seed for generating initial latents, enabling reproducibility across models and configurations. The same watermark key, derived from this seed, was applied throughout the study. Additionally, we adopted a uniform timestep schedule for both sampling and inversion, which appeared to improve the inversion accuracy. We perform the experiments on the test partition of the open source Stable Diffusion Prompts Dataset (Santana, 2023). For all our experiments we have used a fixed Classifier-free guidance (Bansal et al.) of 3.5.

Table 1: Watermark Extraction Metrics

Model	**Configuration**	Latent Noise Reconstruction		$\frac{1}{n}\sum \lvert\hat{w}_i - w_i\rvert$
		NMAE $\frac{\lvert\hat{w}-w\rvert_1}{\lvert w\rvert_1}$	**NMSE** $\frac{\lvert\hat{w}-w\rvert_2^2}{\lvert w\rvert_2^2}$	
FLUX.1-dev	No Attack (No Prompt)	$0.303_{0.056}$	$0.106_{0.042}$	$22.772_{0.550}$
	No Attack (With Prompt)	$0.232_{0.046}$	$0.063_{0.029}$	$22.123_{0.351}$
	No Attack (With VLM)	$0.290_{0.050}$	$0.096_{0.037}$	$22.701_{0.477}$
	Blur (No Prompt)	$\mathbf{1.259_{0.016}}$	$\mathbf{1.594_{0.040}}$	$37.514_{1.066}$
	Blur (With Prompt)	$\mathbf{1.261_{0.017}}$	$\mathbf{1.597_{0.042}}$	$37.718_{1.049}$
	Noise (No Prompt)	$\mathbf{1.325_{0.043}}$	$\mathbf{1.772_{0.018}}$	$38.656_{1.372}$
	Noise (With Prompt)	$\mathbf{1.343_{0.043}}$	$\mathbf{1.821_{0.106}}$	$39.309_{1.302}$
SD 2.1 base	No Attack (No Prompt)	$0.345_{0.060}$	$0.132_{0.047}$	$45.601_{1.702}$
	No Attack (With Prompt)	$0.338_{0.061}$	$0.128_{0.047}$	$45.070_{1.749}$

Note: Subscripts denote standard deviations. Last column represents average L1 distance in Fourier space.

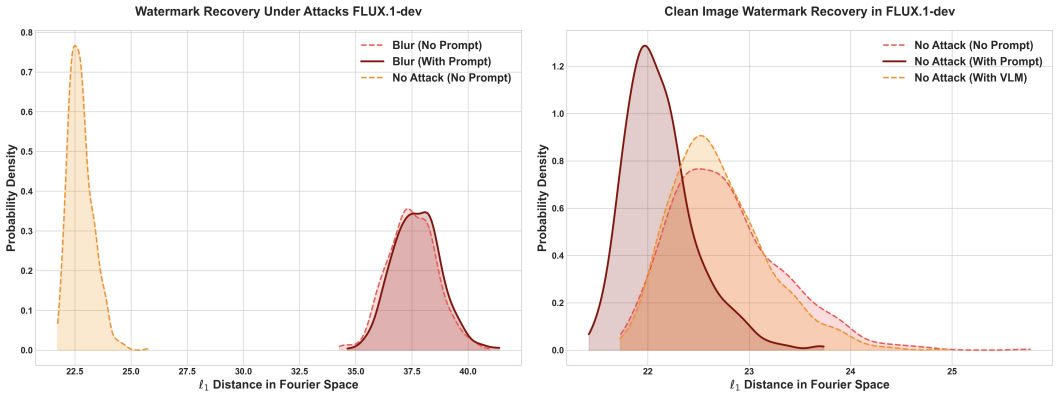

Figure 2: Distribution of watermark distances in Fourier space. Attacked scenarios show the distribution of the fourier space distance under noise, and blur manipulations. It can be clearly seen that in non-attacked scenarios, the prompt guidance plays a significant role in accurate inversion. We note that in attack scenarios the distance in the fourier space is drastically increased for FLUX.1-dev.

4.2 Results and Analysis

Clean Images. Our experiments with non-attacked scenarios reveal that exact prompt guidance during inversion yields the lowest reconstruction error in both Fourier and spatial domains however, in case of attacked images the exact prompt guidance does not aid in reconstruction. Interestingly, FLUX.1-dev demonstrates superior latent noise reconstruction for clean images compared to the baseline model SD 2.1 (Rombach et al., 2022b). However, this advantage diminishes significantly under attacked scenarios, where the separability between watermarked and non-watermarked distributions becomes drastically reduced.

Table 2: AUC comparison for watermark detection under different attacks

Model	Blurring	Noise
SD 2.1 base (DDIM)	0.999	0.944
FLUX.1-dev (RF)	**0.888**	**0.662**

Table 3: Watermark Detection Performance for clean images with FLUX.1-dev

Configuration	TPR@1%FPR	AUC	Thresholds at FPR		
			1%	5%	10%
No Prompt	1.000	0.989	36.950	37.538	37.978
With Prompt	1.000	0.999	36.949	37.524	37.993

The use of VLM-generated prompts demonstrates a noteworthy but constrained improvement in watermark detection. While these semantically derived prompts show marginal benefits in distribution separability and offer performance intermediate between exact prompt and no-prompt configurations, they fail to match the effectiveness of exact prompt guidance. This suggests that while semantic understanding from VLMs can aid reconstruction, **precise prompt matching remains crucial for optimal watermark recovery**.

DDIM (SD 2.1 base) exhibits robust separation between watermarked and non-watermarked images through naive inversion, maintaining consistent performance regardless of prompt guidance. This behavior contrasts significantly with FLUX.1-dev, where reconstruction quality demonstrates marked sensitivity to the presence and accuracy of prompt guidance.

Attacked Scenarios. Under attacked scenarios, the application of noise and blur perturbations significantly compromises the watermark detection capability in FLUX.1-dev as shown in 1. This degradation is particularly evident in the Fourier domain, where the characteristic ring patterns become increasingly difficult to distinguish from background frequencies. This behavior stands in stark contrast with DDIM, where previous work by(Wen et al., 2024)has demonstrated that latent noise reconstruction maintains high fidelity even under various attack scenarios.

Impediments to Watermark Recovery The observed differences in watermark recovery between FLUX.1-dev and traditional diffusion models stem from fundamental architectural and training methodology differences. Flux employs a Multimodal Diffusion Transformer (MM DiT) architecture where text and image information are deeply entangled throughout the network, unlike older diffusion models' UNet architecture where text conditioning occurs primarily through cross-attention layers. This architectural difference makes image generation in Flux more fundamentally dependent on prompt information. Additionally, Flux uses a T5 text encoder with different latent characteristics than the CLIP encoder used in stable diffusion models, further altering information flow through the model. Most importantly, the rectified flow training objective optimizes for straight paths between source and target distributions, prioritizing efficient forward sampling at the expense of invertibility. This straightened path inherently discards information that would be useful during inversion. Higher-order numerical methods might offer incremental improvements, but cannot fully overcome these fundamental architectural limitations.

5 Conclusion and Future Work

Our study reveals fundamental differences in watermark detection and recovery capabilities across DDIM (SD 2.1 base) and FLUX.1-dev architectures. Most notably, we find that the diffusion transformer model FLUX.1-dev exhibits a strong dependency on prompt guidance for accurate reconstruction and watermark recovery, differing significantly from DDIM-based models like Stable Diffusion 2.1, which achieve reliable separation between watermarked and non-watermarked images even without prompt guidance and under attacks. Our analysis demonstrates that detection accuracy in FLUX.1-dev degrades significantly under attacked scenarios, underscoring the need for more robust inversion techniques. A qualitative visualization of image reconstruction is provided in Figure 3.

These findings highlight several critical directions for future research: developing improved inversion techniques specifically for rectified flow-based generative models, and crafting approaches to increase robustness of popular watermarking techniques over image manipulation attacks while maintaining watermark effectiveness.

REFERENCES

Bang An, Mucong Ding, Tahseen Rabbani, Aakriti Agrawal, Yuancheng Xu, Chenghao Deng, Sicheng Zhu, Abdirisak Mohamed, Yuxin Wen, Tom Goldstein, et al. Waves: Benchmarking the robustness of image watermarks. In *Forty-first International Conference on Machine Learning*.

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal Guidance for Diffusion Models. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 843–852. IEEE. doi: 10.1109/CVPRW59228.2023.00091. URL https://ieeexplore.ieee.org/document/10208653/.

Hai Ci, Pei Yang, Yiren Song, and Mike Zheng Shou. Ringid: Rethinking tree-ring watermarking for enhanced multi-key identification. In *European Conference on Computer Vision*, pp. 338–354. Springer, 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024.

FLUX. Flux. https://github.com/black-forest-labs/flux, 2024.

FLUX.1-dev. Black-forest-labs/FLUX.1-dev · Hugging Face. URL https://huggingface.co/black-forest-labs/FLUX.1-dev.

Seongmin Hong, Kyeonghyun Lee, Suh Yoon Jeon, Hyewon Bae, and Se Young Chun. On exact inversion of dpm-solvers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7069–7078, 2024.

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

Qiang Liu. Rectified flow: A marginal preserving approach to optimal transport, 2022. URL https://arxiv.org/abs/2209.14577.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *International Conference on Computer Vision*, pp. 4195–4205, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022a.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, June 2022b.

Gustavo Santana. Stable-diffusion-prompts, Mar 2023. URL https://huggingface.co/datasets/Gustavosta/Stable-Diffusion-Prompts.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution, 2024. URL https://arxiv.org/abs/2409.12191.

Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-rings watermarks: Invisible fingerprints for diffusion images. *Advances in Neural Information Processing Systems*, 36, 2024.

A Appendix: Diffusion Models and Flow Matching

A.1 Latent Diffusion Models

Latent Diffusion Models (Rombach et al., 2022a) (LDMs) operate in the compressed latent space of an autoencoder rather than directly in pixel space. The autoencoder consists of an encoder $\mathcal{E}$ that maps images $x \in \mathbb{R}^{H\times W\times 3}$ to a lower-dimensional latent representation $z = \mathcal{E}(x) \in \mathbb{R}^{h\times w\times c}$, and a decoder $\mathcal{D}$ that reconstructs the image from latents.

The diffusion process occurs entirely in this latent space, offering two key advantages: reduced computational complexity due to lower dimensionality, and the ability to leverage semantic compression from the autoencoder. Given a noise schedule $\{\beta_t\}_{t=1}^T$ and defining $\bar{\alpha}_t = \prod_{i=1}^t (1-\beta_i)$, the forward process adds noise to the latents:

$$z_t = \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \tag{2}$$

where $z_0 = \mathcal{E}(x)$ is the encoded latent. The model learns to predict the noise component using a neural network $\epsilon_\theta(z_t, t)$ trained with the objective:

$$\mathcal{L}_{\text{simple}} = \mathbb{E}_{t,z_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(z_t, t)\|_2^2\right] \tag{3}$$

After the diffusion and denoising process, the final latent z_0 is decoded to obtain the image: $x = \mathcal{D}(z_0)$. The LDM architecture's latent space dimensions vary across implementations. The FLUX dev model uses a VAE with latent dimensions $(16, h/8, w/8)$ where h, w are the input image dimensions, allowing for flexible resolution generation. In contrast, Stable Diffusion 2.1 base model employs a fixed latent dimension of $(4, 64, 64)$

A.2 DDIM Sampling and Inversion

Denoising Diffusion Implicit Models (DDIM) provide a deterministic framework for generating images through the reverse diffusion process. Unlike standard diffusion models, DDIM defines a non-Markovian reverse process that enables deterministic trajectories between noise and images.

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{x}_0^t + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_\theta(x_t, t) \tag{4}$$

where $\hat{x}_0^t$ represents the predicted clean image:

$$\hat{x}_0^t = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}} \tag{5}$$

For inversion, DDIM maps a given image x_0 back to noise x_T using:

$$x_{t+1} = \sqrt{\bar{\alpha}_{t+1}}\hat{x}_0^t + \sqrt{1-\bar{\alpha}_{t+1}}\epsilon_\theta(x_t, t) \tag{6}$$

This naïve DDIM inversion can be interpreted as forward Euler integration starting from $t = 0$. While computationally efficient, it can accumulate errors over multiple steps.

A.3 Rectified Flow and Flow Matching

Rectified Flow (RF) facilitates the transition between the data distribution π_0 and Gaussian noise distribution π_1 along a straight path. This is achieved by learning a forward-simulating system defined by the ODE:

$$d\mathbf{x}_t = \mathbf{v}_\theta(\mathbf{x}_t, t)dt, \quad t \in [0, 1] \tag{7}$$

which maps $\mathbf{x}_1 \sim \pi_1$ to $\mathbf{x}_0 \sim \pi_0$. In practice, the velocity field $\mathbf{v}$ is parameterized by a neural network with parameters θ.

During training, given empirical observations of two distributions $\mathbf{x}_0 \sim \pi_0$, $\mathbf{x}_1 \sim \pi_1$ and $t \in [0, 1]$, the forward process of rectified flow is defined by a simple linear combination:

$$\mathbf{x}_t = t\mathbf{x}_1 + (1 - t)\mathbf{x}_0 \tag{8}$$

which can be written in differential form as:

$$d\mathbf{x}_t = (\mathbf{x}_1 - \mathbf{x}_0)dt \tag{9}$$

Consequently, the training process optimizes the network by solving the least squares regression problem:

$$\min_\theta \int_0^1 \mathbb{E}\left[\|(\mathbf{x}_1 - \mathbf{x}_0) - \mathbf{v}_\theta(\mathbf{x}_t, t)\|^2\right] dt \tag{10}$$

For sampling, the ODE equation 9 is discretized and solved using the Euler method. The model starts with a Gaussian noise sample $\mathbf{x}_{t_N} \sim \mathcal{N}(0, I)$. Given a series of N discrete timesteps $t = \{t_N, \ldots, t_0\}$, the model iteratively applies:

$$\mathbf{x}_{t_{i-1}} = \mathbf{x}_{t_i} + (t_{i-1} - t_i)\mathbf{v}_\theta(\mathbf{x}_{t_i}, t_i) \tag{11}$$

For inversion, the backward Euler method is used:

$$\mathbf{x}_{t_i} = \mathbf{x}_{t_{i-1}} - (t_i - t_{i-1})\mathbf{v}_\theta(\mathbf{x}_{t_i}, t_i) \tag{12}$$

The RF model can generate high-quality images in much fewer timesteps compared to DDPM, owing to the nearly linear transition trajectory established during training.

A.4 Higher-Order Inversion Methods

Recent work has introduced exact inversion techniques using higher-order ODE solvers. For DDIM, the backward Euler method provides more accurate inversion by solving:

$$\hat{z}_{t_{i-1}} = \hat{z}_{t_i} - (t_i - t_{i-1})v_\theta(\hat{z}_{t_i}, t_i) \tag{13}$$

This can be improved through gradient descent steps:

$$\nabla_{\hat{z}_{t_{i-1}}} \|\hat{z}_{t_i} - z'_{t_i}\|^2 \tag{14}$$

where z'_{t_i} is computed using:

$$z'_{t_i} \leftarrow \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}}\hat{z}_{t_{i-1}} - \alpha_{t_i}(e^{-h_i} - 1)z_0(\hat{z}_{t_{i-1}}, t_{i-1}) \tag{15}$$

The DPM-Solver++ framework generalizes this to higher orders using the exponential integrator:

$$\begin{aligned} x_{t_i} = \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}}x_{t_{i-1}} + \sigma_{t_i}\sum_{n=0}^{k-1} x_\theta^{(n)}(x_{\lambda_{t_{i-1}}}, \lambda_{t_{i-1}}) \\ \cdot \int_{\lambda_{t_{i-1}}}^{\lambda_{t_i}} \frac{e^\lambda(\lambda - \lambda_{t_{i-1}})^n}{n!}d\lambda \end{aligned} \tag{16}$$

where $\lambda_t = \log(\alpha_t/\sigma_t)$ is the log-SNR and k represents the order of the solver.

B Appendix: Experimental Details

B.1 Model Configurations and Sampling Parameters

We conducted our experiments using carefully calibrated configurations for both FLUX.1-dev and Stable Diffusion 2.1 models. The key parameters were selected to balance generation quality with computational efficiency while maintaining fair comparison conditions across models.

B.1.1 FLUX-dev Configuration

For the FLUX-dev model, we employed the following parameters:

- Number of sampling steps: 28 steps for both generation and inversion processes
- Guidance scale: 3.5 (classifier-free guidance)
- Sampling method: Euler solver for ODE integration
- Timestep scheduling: Uniform spacing between t=0 and t=1

The relatively lower number of steps (28) for FLUX-dev is justified by its efficient rectified flow training objective and Euler integration scheme, which allows for larger step sizes while maintaining generation quality.

B.1.2 Stable Diffusion 2.1 Configuration

For SD2.1 with DDIM sampling, we used:

- Number of sampling steps: 50 steps for both generation and inversion processes
- Guidance scale: 3.5 (matching FLUX-dev for comparative analysis)
- Sampling method: DDIM deterministic sampling
- Timestep scheduling: Default DDIM schedule

The higher number of steps (50) for DDIM sampling is necessary for finer granularity in the diffusion process timestep discretization.

These configurations were held constant across all experiments to ensure consistency and reproducibility of our results. The parameters were validated through preliminary experiments to ensure they produced high-quality generations while maintaining reasonable computational requirements.

B.1.3 Evaluation Framework

We quantify watermark robustness through the following metrics:

- Fourier Space $\mathcal{L}_1$ Distance: Measures discrepancy between reconstructed ($\hat{w}$) and original (w) watermark patterns in frequency domain:

$$\mathcal{L}_1(w, \hat{w}) = \sum_i |w_i - \hat{w}_i|$$

- Normalized Error Metrics: For assessing reconstruction accuracy:

$$\text{NMSE} = \frac{\|\hat{w} - w\|_2^2}{\|w\|_2^2}, \quad \text{NMAE} = \frac{\|\hat{w} - w\|_1}{\|w\|_1}$$

- Symmetric KL Divergence: Quantifies distributional differences between guided (P) and non-guided (Q) reconstructions:

$$\mathcal{D}_{\text{SKL}}(P\|Q) = \frac{1}{2}[\mathcal{D}_{\text{KL}}(P\|Q) + \mathcal{D}_{\text{KL}}(Q\|P)]$$

  where $\mathcal{D}_{\text{KL}}(P\|Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$

The Fourier space separation metrics obtained for both FLUX.1-dev and SD 2.1 base are listed in 4

Table 4: Distribution Analysis of Watermarked vs Non-watermarked (Non-Attacked Images)

Model	**Image Type**	$\|\|\hat{w} - w\|\|$	**Symmetric KLD**
FLUX.1-dev	Watermarked	$22.77_{0.550}$	$18.00_{0.067}$
	Non-watermarked	$39.395_{1.111}$	
SD 2.1 base	Watermarked	$45.601_{1.702}$	$17.81_{0.081}$
	Non-watermarked	$79.263_{2.308}$	

Note: L1 Distance measured in Fourier space. Symmetric KLD computed between watermarked and non-watermarked distributions. Subscripts denote standard deviations.

B.2 Qualitative Results

To ensure experimental reproducibility, we maintained a consistent global random seed when generating initial latents across all experiments. The identical watermark key was employed throughout all tests, and we implemented a uniform timestep schedule for both sampling and inversion processes, as our preliminary tests demonstrated this approach significantly enhanced inversion quality.

A notable observation from our experiments is that, even when using identical prompts, images generated from the original and reconstructed noise latents show perceptible differences, as illustrated in Figure 4.

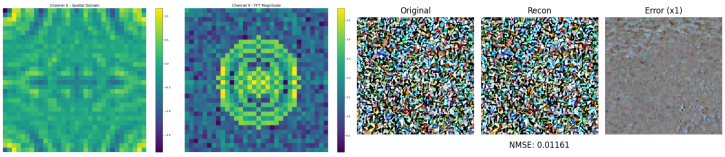

Figure 3: Visualization of noise reconstruction in spatial and frequency domains. Left: Channel 0 of the latent noise in spatial domain averaged over 100 samples, showing the characteristic noise pattern. Center: Magnitude of the 2D Fourier transform of Channel 0, revealing the circular watermark pattern in frequency space. Right: Original noise, reconstructed noise, and their difference (error magnified by 1×) for a representative sample, with NMSE of 0.01161.

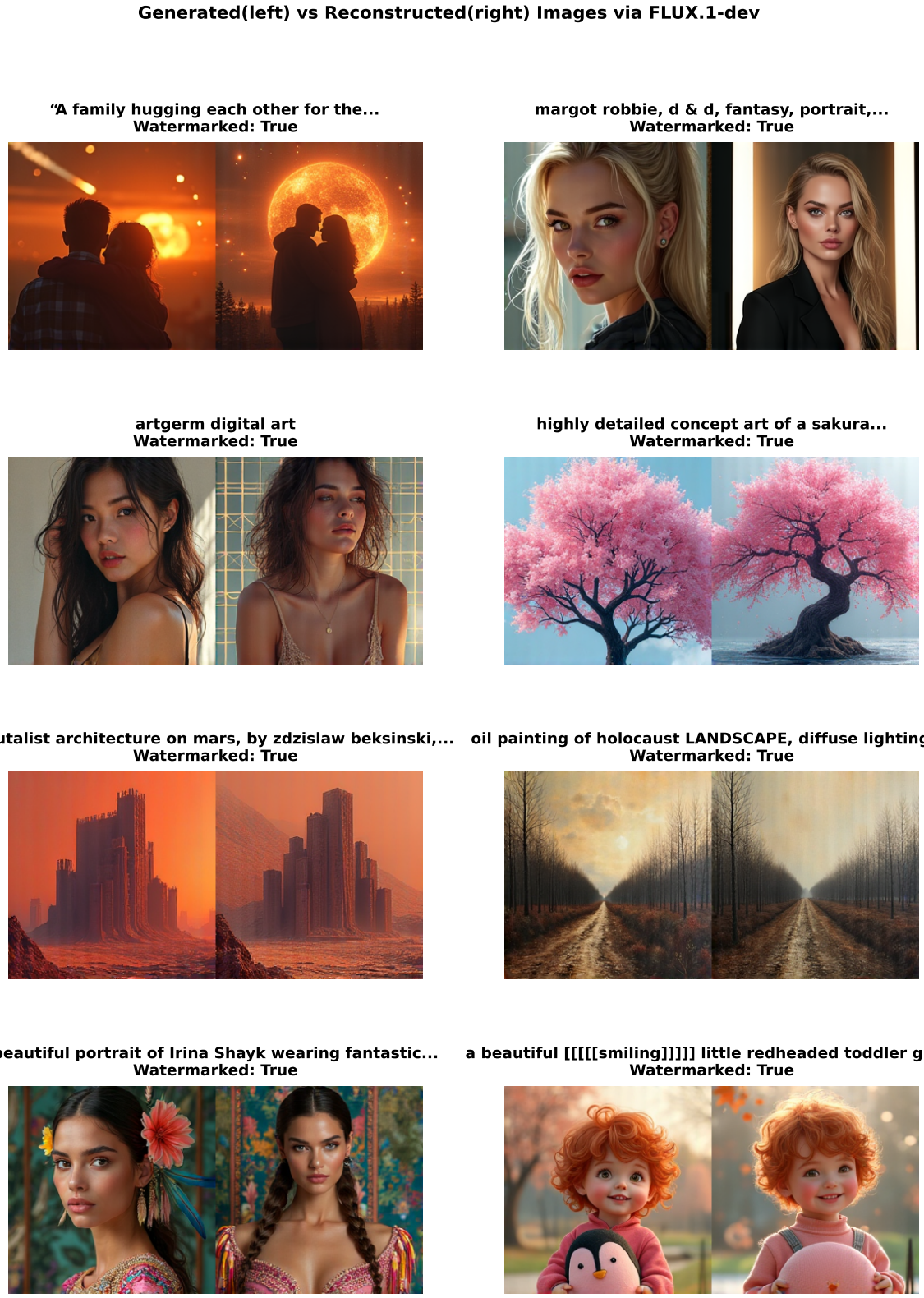

Figure 4: Image generation results from the reconstructed initial noise using FLUX.1-dev. Despite using identical prompts, notable differences can be observed between original generations (left) and those from reconstructed noise (right).