CSFlow: Aligning Flow Matching with Human Contrast Sensitivity

Malgorzata Galinska[1*] **Bart Pogodzinski**[1*] **Jan Eric Lenssen**[1,2]

[1]Max Planck Institute for Informatics
[2]Saarland University, Saarland Informatics Campus, Germany

{mgalinsk, bpogodzi, jlenssen}@mpi-inf.mpg.de

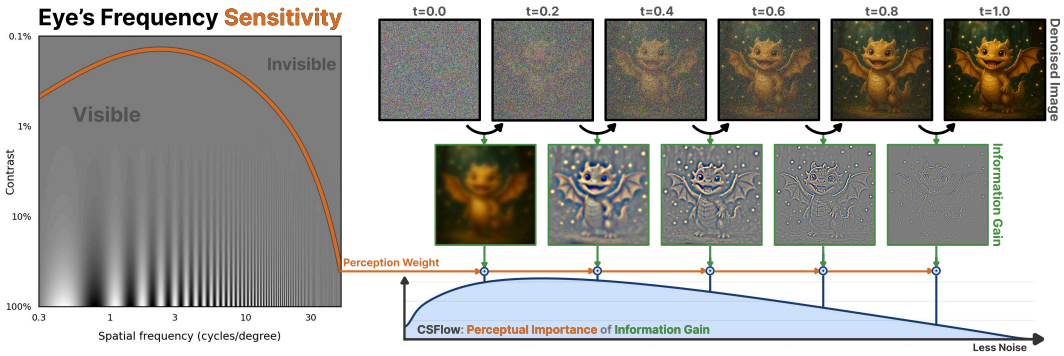

Figure 1: Contrast Sensitive Flow (CSFlow) connects the human eye's Contrast Sensitivity Function (CSF) [Barten, 2003] (left) to the iterative denoising steps of flow matching (right), providing a weighting scheme grounded in human vision. Information gain in Fourier space of each denoising step is weighted by the perceptual importance of generated frequencies, allocating more capacity to frequencies to which the human eye is more sensitive.

Abstract

We introduce Contrast Sensitive Flow (CSFlow), a weighting scheme that connects the human eye's Contrast Sensitivity Function (CSF) to the iterative denoising steps of flow matching. Because real-world images concentrate signal at low spatial frequencies, these components reach high signal-to-noise ratio earlier during continuous diffusion than high-frequency components. When generating images with diffusion or flow matching models, this induces a soft autoregressive structure in Fourier space, where coarse image content stabilizes before fine detail. Meanwhile, the human visual system is unequally sensitive to spatial frequencies: very low and very high frequencies require significantly higher contrast to be perceived. We for the first time merge these observations through two contributions: (1) a metric that estimates which frequencies are generated at each reverse flow interval and (2) timestep weights obtained by aligning the frequencies generated at each noise level with human contrast sensitivity. We validate our contributions experimentally showing that these weights can improve generative performance by lowering FID by up to 12.15%, increasing Inception Score by up to 3.2% and improving GenEval scores by up to 3.0% using inference-only timestep modification or short fine-tuning. Qualitatively, we find that our CSFlow weights lead to better visual realism and less cartoonish appearance of generated images. In a two-alternative forced-choice study, participants preferred CSFlow generations over the baseline in 65% of comparisons. Additional visualisations, code, models and the example CSFlow weights are available on our project website[2].

[*]Equal contribution.
[2]gosiagal.github.io/CSFlow

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

1 Introduction

Diffusion [Sohl-Dickstein et al., 2015, Ho et al., 2020] and flow matching [Lipman et al., 2023, Liu et al., 2023] models have recently become some of the most successful paradigms for image synthesis, enabling the generation of diverse and high-quality images [Ma et al., 2026]. Although model architectures, training objectives, etc., can differ, one of the *most common* objectives is to produce high-quality images that are viewed and judged by humans, with some limited training and inference compute budget.

Research on the human eye shows that there exists a *window of well-noticable spatial frequencies* for human perception (c.f. Fig 1, left) [Barten, 2003]. Certain image variations, such as changes in very low or high frequency components (e.g., very coarse gradients or texture details), are largely invisible to the human eye, even if the contrast is high. Meanwhile, other frequencies, mostly located in intermediate frequency bands, are easy to spot even with low contrast. Consequently, unnatural representation of those low-mid-range frequencies is the most noticeable to the human eye.

Recent works in generative modeling with diffusion and flow models strongly connect the different stages of the reverse process to the reconstruction of different spatial frequencies [Dieleman, 2024, Falck et al., 2025]. Over the course of the reverse process, images are generated in a coarse-to-fine manner, where information is added in low frequencies first before adding high frequency details towards the end. In a setting with a limited compute budget and model capacity, this poses the question of whether an image generation model should allocate the budget to reconstruct frequencies that are invisible to us humans, or if it should rather prioritize allocating its capabilities to the most visible ones. With this work, we are the first to connect these two phenomena and show that weighting the information gain in reverse flow matching steps to favor steps that are important for human perception (c.f. Fig. 1, right) can improve the quality of generated images.

Concretely, we propose Contrast Sensitive Flow (CSFlow), a procedure to calculate *perception-driven weights* for flow matching. The weighting can be applied in two ways: (1) to modify the step sizes during inference, increasing the approximation accuracy of important intervals of the generation process; (2) to weight the probability of sampling certain t during model training, favoring the allocation of more model capacity to specific generation intervals. To this end, we describe a metric to estimate which frequencies are generated in which parts of the flow matching reverse process and how contrast sensitive weights can subsequently be obtained from the resulting connection.

We conduct experiments on representatives of current state-of-the-art pixel-space flow matching models, i.e., PixelGen [Ma et al., 2026], PixelREPA [Shin et al., 2026] and JiT [Li and He, 2025], evaluating the GenEval [Ghosh et al., 2023], FID, and Inception score, which represent visual quality obtained by applying our training weights and inference step sizes. Our findings suggest that perception-driven weighting during training and inference consistently improves the visual quality of generation by up to 12.15% in FID, 3.2% in Inception Score, and 3.0% in GenEval scores. Qualitatively, we find that our generated images appear to be more realistic and less *cartoonish* than those from the baselines, and a two-alternative forced-choice study confirms that human viewers prefer them in 65% of cases. Our concrete contributions are the following:

- A closed-form metric $\partial_t r_{\text{signal}}(f, t)$ that estimates which spatial frequencies are *being generated* in each interval of the reverse flow. To our knowledge, we are the first to describe explicitly which frequency content is generated at a specific t.
- **CSFlow**: perception-driven timestep weights, obtained by aligning the frequencies generated at each noise level with human contrast sensitivity.
- Two drop-in applications of these weights: as as a non-uniform **inference** step grid and a **training** timestep distribution. Neither of them requires new training losses or architecture changes.

2 Related Work

Diffusion models can change the relative importance of denoising stages through the noise schedule, distribution of timesteps (t), explicit loss weights, and the choice of the prediction target (ϵ, x_{clean}, v). Those have been implemented across many works with varying backgrounds and motivation.

DDPM [Ho et al., 2020] derives timestep-dependent loss terms from the ELBO, but obtains better sample quality with a simplified denoising objective that drops the ELBO-derived weighting. Im-

proved DDPM [Nichol and Dhariwal, 2021] introduces a cosine noise schedule, which changes how signal is retained across the timesteps and makes the time discretization leading to improved results. EDM [Karras et al., 2022] shows that the treatment of different noise levels is shaped by separable design choices, including the training noise-level distribution, the denoising loss weight, network parametrization, and the sampling-time discretization. Min-SNR-γ [Hang et al., 2023] treats the diffusion as a multi-task-learning problem and notices that training convergence and final model quality can improve with loss weighted according to the clamped signal-to-noise ratios, preventing the high-loss (and thus high gradient) regions from overpowering the optimization – resulting in down-weighing high-noise t in x_{clean}-prediction and downweighing the low-noise regions in ϵ-prediction. EDiff-I [Balaji et al., 2022] takes a different approach by training an ensemble of denoisers for individual noise intervals, essentially mitigating the issue some noise levels overpowering others during training, at the cost of additional training and system complexity.

High-resolution and flow models further modify timestep importance through schedule shifts and non-uniform sampling. Simple Diffusion [Hoogeboom et al., 2023] argues that increasing image resolution changes the effective SNR of low-frequency structure: averaging over more pixels reduces the noise variance of coarse image content. They therefore shift the noise schedule for high-resolution pixel-space diffusion so that the *effective* SNR at a reference resolution is preserved. Stable Diffusion 3 [Esser et al., 2024] applies a related resolution-dependent timestep shift in latent-space rectified flows, selecting the shift through inference-time sampling experiments and then using it during both high-resolution training and inference. It also studies non-uniform training-time timestep sampling, including logit-normal sampling, which emphasizes intermediate timesteps and vanishes near the endpoints. Schedule On the Fly [Ye et al., 2025] targets inference-time compute allocation by adding a next noise level predictor via reinforcement learning. All of the above methods aim to improve the understanding and learnability of diffusion models but do not align it with human perception.

P2 [Choi et al., 2022] aims to align the generation task to human perception by observing that late denoising stages mostly correspond to noise-cleanup phase, where no visible-to-humans generative decisions are left to be made. They heuristically propose loss weights, which are lower for late denoising stages. While the approach shares the general motivation with our work, our method is grounded in a mathematical connection, deriving timestep weighting directly from the contrast sensitivity function, rather than from coarse observations.

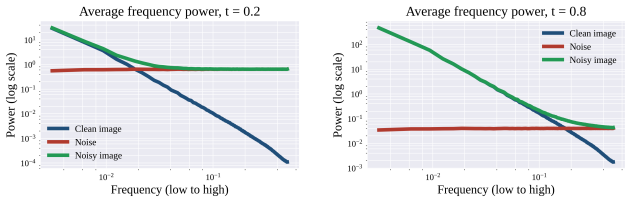

Figure 2: Comparison of average frequency strengths between two denoising stages: $t = 0.2$ – early stage, mostly noise and $t = 0.8$ – late stage, mostly data. Natural image power spectrum follows a power law and noise power spectrum is constant. The noisy image magnitude (green) aligns with the clean image (blue) and noise magnitude (red) on different ranges depending on the t. The later the stage, the higher frequency bands are reconstructed in the noisy image.

3 Preliminaries

Flow Matching. The flow matching model defines a continuous denoising trajectory $\mathbf{x}_t$, $t \in [0, 1]$, which transforms a noise distribution at $t = 0$ into the data distribution at $t = 1$. In flow matching [Lipman et al., 2023], a neural network $v_\theta(\mathbf{x}_t, t)$ is trained to predict the velocity of this trajectory, typically using an objective of the form:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_1}\left[\left|v_\theta(\mathbf{x}_t, t) - \mathbf{u}_t\right|^2\right], \tag{1}$$

where $\mathbf{u}_t$ is the target velocity conditioned on x_1. Thus, t specifies the denoising state at which the model is trained, and the distribution used to sample t during training pressures the model to allocate more of its capacity to some noise levels. At inference, the continuous flow is approximated numerically by a time point grid:

$$0 = t_0 < t_1 < \cdots < t_N = 1, \qquad \Delta t_i = t_{i+1} - t_i, \tag{2}$$

and the ODE $\frac{d\mathbf{x}_t}{dt} = v_\theta(\mathbf{x}_t, t)$ is solved over these intervals. Consequently, the distribution of training times and the discretization of inference times determine how computational and modeling capacity is allocated along the denoising trajectory.

Frequency content of images. The frequency domain provides a representation of images by describing them in terms of their spatial frequency components rather than their pixel intensities, making it useful for image analysis and compression because it highlights patterns such as edges, textures, and repetitive structures that may be less obvious in pixel space. To decompose an image into spatial frequencies that create it, we use the *two-dimensional Discrete Fourier Transform* (2DDFT).

One property of natural images that is crucial to our method is that their power spectra, on average, follow a power law, with low frequencies having orders of magnitude stronger signals than the high frequencies. As discussed in [Dieleman, 2024, Falck et al., 2025], this results in imbalanced corruption of the frequency information during the diffusion forward process (c.f. Fig. 2): even weak noise can destroy the highest-frequency information, but a much stronger noise is necessary to destroy the lower frequencies. As a result, during the generation step at some noise level t, only a narrow frequency band of frequencies can be considered "being generated", since lower frequencies already have too high a signal-to-noise (SNR) ratio to meaningfully change, while the higher ones will remain completely overpowered by the noise after the step.

Human frequency sensitivity. The human visual system is not equally sensitive to all spatial frequencies: usually very smooth gradients (low frequencies) and tiny textural details (high frequencies) become hard to spot. An established way to measure it is via the Contrast Sensitivity Function (CSF), which measures how much contrast is necessary for an average human eye to notice that a pattern is shown. One of the most commonly used CSF models was proposed in [Barten, 2003], which can be observed in Fig. 1. For more details on the CSF, see Appendix B.

4 Contrast Sensitive Flow

In this section, we explain how we incorporate human frequency sensitivity into the image generation of diffusion models. First, we derive *perception-driven weights* that indicate the importance of every generation stage with respect to human sensitivity to the frequencies being reconstructed in these stages. We start by analyzing how the recoverable frequency content evolves over time during the denoising process in Sec. 4.1 and how much information is gained in each step in Sec. 4.2. Then, we incorporate the CSF and distribute weights over time intervals in Sec. 4.3. In Sec. 4.4, we then explain how these weights can be used in practice to bias the model towards more important denoising stages.

4.1 Recoverable frequency content

During denoising, different spatial frequencies become distinguishable from noise at different stages. We therefore want to estimate, for each frequency f and time t, how much of the spectral power present in the noisy image is attributable to the image component rather than to noise.

Let $\mathbf{x}_{clean} \sim D$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\mathbf{x}_t = a_t \mathbf{x}_{clean} + b_t \boldsymbol{\epsilon}$, where a_t and b_t define the noise schedule. For a frequency f, we denote the corresponding 2D DFT coefficient as $F_f(\cdot)$, power as $P_f(\mathbf{x}) = |F_f(\mathbf{x})|^2$, and define the expected clean image power $S_f = \mathbb{E}\left[P_f(\mathbf{x}_{clean})\right]$.

The quantity we would like to measure is the fraction of expected spectral power in $\mathbf{x}_t$ that comes from the image component, i.e., the fraction of retained image signal at denoising state t:

$$r_{\text{signal}}(f, t) = \frac{\mathbb{E}\left[P_f(a_t \mathbf{x}_{clean})\right]}{\mathbb{E}\left[P_f(\mathbf{x}_t)\right]}. \tag{3}$$

Values close to 0 indicate that the frequency is noise-dominated (not generated), while values close to 1 indicate that it is image-dominated and will not significantly change with further denoising. The following theorem shows how to measure r_{signal} in closed-form.

Theorem 1 (Closed form of retained signal)**.** *Assume that $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$ is independent of $\mathbf{x}_{clean}$ and S_f, a_t, and b_t are defined as above. Then,*

$$r_{\text{signal}}(f, t) = \frac{\mathbb{E}\left[P_f(a_t \mathbf{x}_{clean})\right]}{\mathbb{E}\left[P_f(\mathbf{x}_t)\right]} = \frac{a_t^2 S_f}{a_t^2 S_f + b_t^2}. \tag{4}$$

The proof is given in App G. r_{signal} can also be seen as a normalized signal-to-noise (SNR). With

$$SNR_f(t) = \frac{P_f(a_t \cdot \mathbf{x}_{clean})}{P_f(b_t \cdot \epsilon)} = \frac{a_t^2 S_f}{b_t^2}, \tag{5}$$

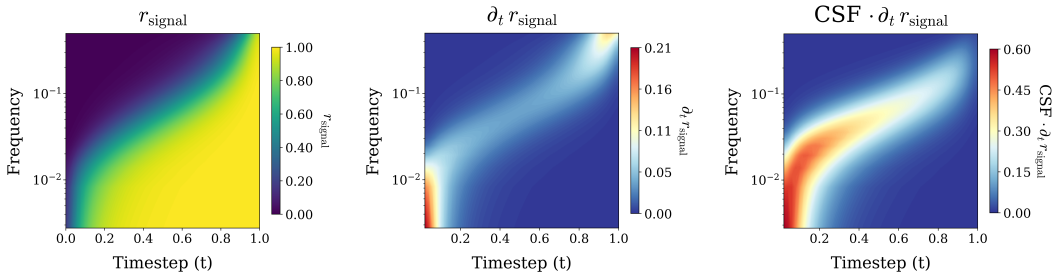

Figure 3: r_{signal} (left), $\partial_t r_{\text{signal}}$ (middle) and CSF $\cdot\, \partial_t r_{\text{signal}}$ (right) calculated using ImageNet 256×256 training dataset, 50 linear time values t and linear noise schedule $a_t = t,\ b_t = 1 - t$. Frequencies are plotted in the log space. In the plots $t = 0$ corresponds to pure noise and $t = 1$ to the clean image, therefore the denoising process happens from left to right. First two plots show that the denoising process reconstructs spatial frequencies *hierarchically*, in a coarse-to-fine manner. In the last plot, high frequency values of Δr_{signal} are suppressed while low and mid-range frequencies are amplified, which aligns with the CSF.

we have

$$r_{\text{signal}}(f,t) = \frac{SNR_f(t)}{1 + SNR_f(t)}. \tag{6}$$

Unlike raw SNR, $r_{\text{signal}}(f,t) \in [0,1]$ holds, making it convenient for comparing how much of a frequency should be already *generated* at a specific noise level t.

4.2 Denoising step information gain from r_{signal}

While $r_{\text{signal}}(f,t)$ describes how recoverable a frequency f is at a single denoising state, we are interested in which frequencies *become* recoverable during each denoising interval. We therefore define the stepwise increase

$$\Delta r_{\text{signal}}(f,t,\Delta t) = r_{\text{signal}}(f,t+\Delta t) - r_{\text{signal}}(f,t). \tag{7}$$

We use Δr_{signal} as a metric for the amount of *newly generated* frequency content over the interval $[t, t+\Delta t]$. Thus, Δr_{signal} indicates the *expected fraction of total frequency information we gain* at a step from t to $t+\Delta t$. Since r_{signal} is non-decreasing in our noise schedule (and all commonly used), $\Delta r_{\text{signal}}(f,t,\Delta t) \geq 0$. We can also define a continuous version of Δr_{signal} as $\partial_t r_{\text{signal}}(t,f)$ indicating how frequencies *gain information* at timestep t.

Figure 3 visualizes r_{signal} and $\partial_t r_{\text{signal}}$. Lower spatial frequencies become data-dominated earlier in the denoising process, while higher frequencies become recoverable later. This shows that the denoising process reconstructs data frequency content hierarchically, in a coarse-to-fine manner. We include the implementation details of estimating the weights in Appendix C.

4.3 CSF-based interval weighting

Our goal is to assign larger weights to denoising intervals in which the newly recovered data-attributable frequencies are more important for human perception. We quantify perceptual importance using the Contrast Sensitivity Function (CSF). Since CSF is defined over frequencies in cycles per degree of visual angle, while our image frequencies are measured in cycles per pixel, we first convert image frequency bins to cycles per degree before evaluating CSF. We assume a pixel size of 0.0114 cm and a viewing distance of 50 cm. The conversion of units is described in Appendix B.2.

Recall that $\Delta r_{\text{signal}}(f,t,\Delta t)$ measures how much data-attributable content at frequency f becomes newly recoverable during the interval $[t, t+\Delta t]$. In continuous notation, this corresponds to the instantaneous increase $\partial_t r_{\text{signal}}(f,t)$. We define the *raw perceptual weight* at time t as the CSF-weighted average of the newly recovered frequency content:

$$\widetilde{w}_{\text{CSFlow}}(t) = \frac{\int_\Omega \partial_t r_{\text{signal}}(f,t) \cdot CSF(f)\, df}{\int_\Omega \partial_t r_{\text{signal}}(f,t)\, df}. \tag{8}$$

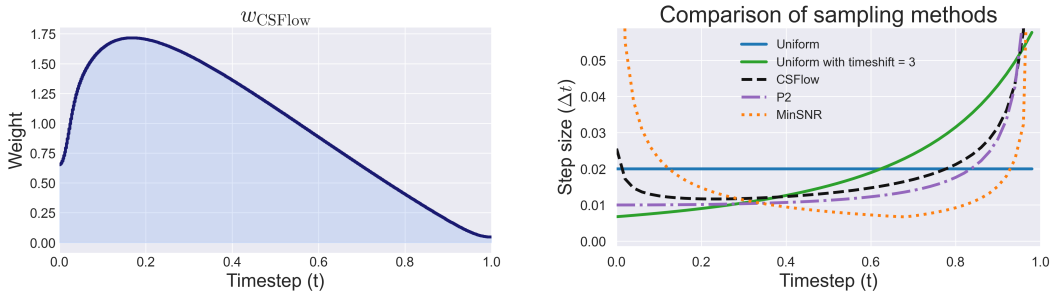

Figure 4: Resulting w_{CSFlow} (left) and a comparison of sampling methods during inference including our weighted method (right). In the plots, the denoising process happens from left to right. Our pure weights strongly bias the model towards early to mid-range generation stages.

Here, Ω denotes the set of frequency bins. The denominator makes the score relative: intervals are weighted according to *which* frequencies are recovered, rather than simply how much total spectral content is recovered. Equivalently, $\widetilde{w}_{\text{CSFlow}}(t)$ is the average CSF value over the frequencies that become recoverable around time t. I.e., if $CSF(f) \equiv 1$, all t receive the same raw weight.

We normalize these scores over the denoising trajectory:

$$w_{\text{CSFlow}}(t) = \frac{\widetilde{w}_{\text{CSFlow}}(t)}{\int_0^1 \widetilde{w}_{\text{CSFlow}}(\tau)\, d\tau}. \tag{9}$$

In practice, we approximate Eq. 8 with finite differences using $\Delta r_{\text{signal}}(f, t, \Delta t)$, and approximate the integral in Eq. 9 as a sum over the chosen timestep grid.

Fig. 4 (left) shows the resulting weights for an example setting. The largest weights occur in the early-to-middle part of the denoising process, indicating that these timesteps recover the frequencies most aligned with human visual sensitivity. Those continuous weights integrate to 1, so they can directly be used as a noise levels distribution during training.

4.4 Getting timesteps from weights

After the normalization in Eq. 9, w_{CSFlow} integrates to one over $[0, 1]$ and can be treated as a density over denoising time, with cumulative distribution

$$CDF(t) = \int_0^t w_{\text{CSFlow}}(\tau)\, d\tau. \tag{10}$$

We use an inverse of this CDF in two places. For **training**, we draw $u \sim \mathcal{U}(0, 1)$ and set $t = CDF^{-1}(u)$, which samples noise levels proportionally to w_{CSFlow}. For **inference**, we evaluate the same inverse CDF at even grid of quantiles,

$$\tilde{t}_k = CDF^{-1}\left(\tfrac{k}{N}\right), \qquad k = 0, \ldots, N, \tag{11}$$

which gives a grid of N steps satisfying $\int_{\tilde{t}_k}^{\tilde{t}_{k+1}} w_{\text{CSFlow}}(t)\, dt = 1/N$ for every k. In regions with large w_{CSFlow} the CDF rises faster, so the inverse gets flatter and the steps are smaller. The grid is more dense in the intervals that recover perceptually important frequencies. Fig. 4 (right) shows the resulting step sizes against uniform and time-shifted grids. We provide more details on obtaining a smooth monotone inverse CDF, as well as some implementation details in the Appendix C.

5 Experiments

To test the weights, we incorporate them in inference and short finetuning of different flow matching models and compare our setup to its original training/inference. We also compare our weighting influence to the commonly adopted min-SNR-γ [Hang et al., 2023] and similarly motivated P2 [Choi

et al., 2022] weights. For text-to-image generation, we report results on GenEval [Ghosh et al., 2023]. For class-to-image generation, we report FID [Heusel et al., 2017] and Inception Score (IS) [Salimans et al., 2016]. We run all single experiments 5 times with different seeds and report the mean score with two standard deviations (95% confidence interval).

In all experiments, we calculate our weights based on the chosen model's training dataset and forward noise schedule, and then interpolate between the CSFlow weights and the model's base weights according to the interpolation parameter α:

$$w_{final}(t) = \alpha \cdot w_{CSFlow}(t) + (1 - \alpha) \cdot w_{base}(t), \tag{12}$$

one for training (α_1) and one for inference (α_2). We provide further details on the alphas used in the experiments in App. C.6 and show their impact during inference in Fig. 7.

5.1 Model assumptions

Our derivation assumes that the noising and denoising process is defined directly in pixel space. This is important because r_{signal} relates the noise level at time t to recoverable spatial frequencies in the image. In latent-space models, noise is applied to learned latent variables, whose x-y coordinates may not correspond explicitly to pixel-space coordinates. Therefore, the connection between denoising time and recoverable image frequencies is not well defined, and CSFlow cannot be directly applied (we define it as a possible extension to latent models in App. H.2). For this reason, we evaluate our method on pixel-space generative models, namely PixelGen-XXL and PixelGen-XL [Ma et al., 2026], PixelREPA-H [Shin et al., 2026] and JiT-H [Li and He, 2025]. In all of our experiments, we use the original inference setups (e.g., CFG configuration, number of inference steps) and vary only the timestep weighting in finetuning or inference.

5.2 Text-to-Image generation

Setup. For text-to-image generation we use PixelGen-XXL/16 [Ma et al., 2026] at 512×512 resolution. This model is pretrained with ImageNet for a total of $280K$ steps and further finetuned with high-quality instruction-tuning dataset BLIP3o-60k [Chen et al., 2025a] for $40K$ steps. We calculate our weights based on BLIP3o-60k dataset statistics. In inference, we use Adams-2nd solver with 25 steps with the modified-weighted timestep grid. In finetuning comparisons, we train all models for 7500 steps on the BLIP3o-60k dataset, with weighted noise level distributions and unmodified baselines.

Table 1: Effect of our weighting scheme on text-to-image generation (PixelGen-XXL/16, 512×512). We report the overall GenEval score. The upper block applies each weighting scheme directly, the lower block interpolates it with the model's base weights and reports the best performing configuration; the corresponding α_1, α_2 are listed in Tab. 3. Best result per column and block in **bold**.

Method	Inference-only	Finetuning-only	Finetuning + Inf.
Baseline	0.791 ± 0.0036	0.789 ± 0.0048	0.789 ± 0.0048
base weights			
P2	0.778 ± 0.0048	0.809 ± 0.0062	**0.806** ± 0.0010
MinSNR	0.765 ± 0.0046	0.786 ± 0.0022	0.758 ± 0.0052
CSFlow (ours)	**0.798** ± 0.0050	**0.810** ± 0.0008	0.797 ± 0.0022
with interpolation			
P2	0.792 ± 0.0080	0.809 ± 0.0062	0.806 ± 0.0010
MinSNR	0.788 ± 0.0096	0.786 ± 0.0022	0.788 ± 0.0026
CSFlow (ours)	**0.798** ± 0.0050	**0.810** ± 0.0008	**0.813** ± 0.0044

In our main results, we use all weighting schemes as is. Additionally, we find that alpha-interpolated schemes (Eq. 12) can improve the benchmark scores. In our additional experiments, we interpolate weights with the original setup (uniform and time-shifted in inference and logit normal in training), obtaining mixed-distributions, and report the best performing α_1, α_2 setups.

Quantitative results. Table 1 reports the performance of our CSFlow-modifications in text-to-image generation using GenEval benchmark. The PixelGen-XXL inference-weighted model achieves GenEval score of 0.798 surpassing its unmodified version. Finetuning the model with our weighted time sampling method (or using interpolation $\alpha_1 = 0.8$ and $\alpha_2 = 0.7$) further improves the model, achieving GenEval score of 0.810 and 0.813.

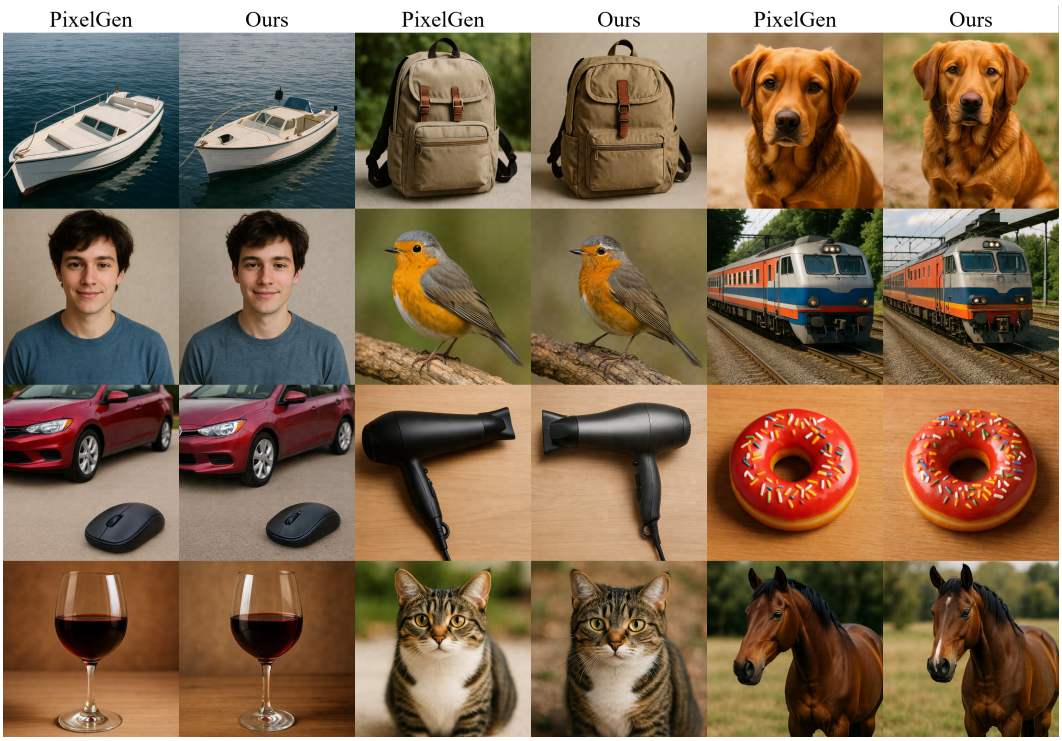

Figure 5: Comparison of the base PixelGen-XXL/16 model (left) with our weighted version (with $\alpha_1 = 0.8$ and $\alpha_2 = 0.7$) (right) in text-to-image generation. All images are 512×512 resolution. We observe that with our weights, samples often yield much less cartoonish-looking generations. Best viewed zoomed in. More results can be found in the appendix.

Qualitative results. Figure 5 shows chosen image pairs that compare the generations of the baseline model and our best configuration model. We observe that our model generates **more realistic** images in terms of their textures: while the baseline model suffers from oversmoothing objects and surfaces, we find that our weighting in finetuning and inference fixes that issue, making the images look less artificial and cartoonish.

5.3 Class-to-Image generation

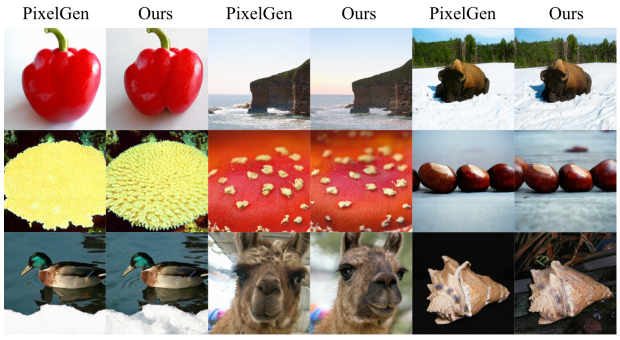

Figure 6: Comparison of the base PixelGen-XL/16 model (left) with our weighted version with $\alpha_2 = 0.5$) (right) in class-to-image generation. All images are 256×256 resolution. While findings are generally the same, the differences are less noticeable in lower quality images.

Setup. For class conditional generation, we use three models: JiT-H/16 [Li and He, 2025], PixelREPA-H/16 [Shin et al., 2026], and PixelGen-XL/16 [Ma et al., 2026]. All models are trained on ImageNet 256×256 [Deng et al., 2009, Russakovsky et al., 2015], JiT and PixelREPA for 600 epochs and PixelGen for 160 epochs. We use the same dataset for analyzing the average data frequency content and use it to calculate our weights. JiT is a relatively straightforward flow model that predicts the clean image $\mathbf{x}_{clean}$ itself and does not use any tokenizer, pre-training or extra losses. PixelREPA extends JiT with a modified REPA-style alignment loss, replacing REPA's per-token MLP projector with a Masked Transformer Adapter. PixelGen instead uses REPA with a regular per-token MLP and output-space perceptual losses on the predicted clean image: LPIPS (VGG) [Zhang et al., 2018] and a cosine loss on last-layer DINOv2-B [Oquab et al., 2024] patch features. In inference, we interpolate our weights with uniform ones and test the resulting non-

Table 2: Effect of our weights on class-to-image generation (ImageNet 256×256 with CFG). We report FID and Inception Score, each block reports one metric. All weighting schemes are interpolated with the base weights at the best performing α per method; values are listed in Tab. 3. Best result per column and block in **bold**.

	JiT-H			PixelREPA-H	PixelGen-XL
Method	Inference-only	Finetuning-only	Finetuning + Inf.	Inference-only	Inference-only
FID ↓					
Baseline	1.89 ± 0.022	1.89 ± 0.038	1.89 ± 0.038	1.81 ± 0.030	1.88 ± 0.034
P2	1.84 ± 0.022	**1.86** ± 0.042	**1.78** ± 0.030	1.62 ± 0.008	1.89 ± 0.022
MinSNR	1.84 ± 0.022	1.90 ± 0.026	1.86 ± 0.042	1.68 ± 0.018	1.88 ± 0.026
CSFlow (ours)	**1.80** ± 0.020	1.89 ± 0.010	**1.78** ± 0.040	**1.59** ± 0.010	**1.85** ± 0.030
Inception Score ↑					
Baseline	297.2 ± 3.20	294.1 ± 3.98	294.1 ± 3.98	316.6 ± 2.30	292.7 ± 6.68
P2	297.4 ± 3.52	298.9 ± 2.78	302.9 ± 2.06	316.7 ± 4.68	295.3 ± 2.84
MinSNR	**302.6** ± 1.72	290.8 ± 1.88	291.6 ± 1.46	**320.9** ± 1.30	294.0 ± 4.16
CSFlow (ours)	299.7 ± 3.58	**299.2** ± 3.56	**303.5** ± 2.76	320.8 ± 4.66	**301.5** ± 2.54

uniform step sizes. We use Heun sampler with 50 steps for all setups. In training, we sample from a modified-weighted distribution. We finetune the JiT-H model for 2 epochs.

Quantitative results. Table 2 reports the performance of our weighted-modifications in class-to-image generation using FID and IS metrics. The PixelREPA-H inference-weighted model achieves the FID score of 1.59 and IS score of 320.8, significantly improving the baseline model (FID 1.81, IS 316.6). The JiT-H and PixelGen-XL inference-weighted models also improve scores of the baseline model (FID 1.89→1.80 and 1.88→1.85, IS 297.2→299.7 and 292.7→301.5). To emphasize, by **only introducing our weights in inference**, we surpass the JiT baseline model, as well as its largest version JiT-G/16 with over 1B more parameters (FID 1.82, IS 292.6 reported by JiT authors). By combining finetuning and weighted-inference in JiT, we achieve even better FID of 1.78 and IS 303.5.

Qualitative results. On ImageNet generation, we observe similar effects as in high resolution text-to-image generation: more realistic textures and better color tones and contrast. However, due to the general low quality of generated images, the effect is less significant. In Figure 6 we present pair comparisons between the baseline model and our modified version of it that reflect our observations.

5.4 Analysis

We further analyze the GenEval scores of different methods over different interpolation states between original and proposed weights in Fig. 7. We can see that, in contrast to other weighting schemes, CSFlow weighting in inference tend to improve scores when approaching $\alpha_2 = 1.0$ – directly applied weights. Fig. 8 shows a visual analysis of two examples when varying $\alpha_1 = \alpha_2$. We observe that the realism gradually increases over the trajectory. We note though, that while this pattern of improvement over higher alpha is consistent in our GenEval experiments, in our ImageNet experiments interpolated alphas have shown significantly better performance, with the best reported in Tab. 3.

Frequency reconstruction. We also verify that our method leads to improvements in modeling the overall frequency distribution. Comparing the radially averaged power spectral density (RAPSD) of generated images against that of the data, we find that the log-space power spectrum error of CSFlow generations is lower than the baseline's across the whole spectrum, and that the reduction is largest in the mid-frequency band where the CSF is high (a 33% to 51% relative reduction there, against 1% to 6% at the spectral extremes). The full analysis is given in Appendix E.

5.5 User preference study

To test whether the improvement in visual realism is visible to human observers, we conducted a two-stage preference study. In the first stage, participants voted on the **correctness** of individual generated images taken from the full pool of the original and CSFlow-weighted PixelGen-XXL generations. We then filtered out image pairs, in which either the baseline or our generation was marked as incorrect (e.g. missing legs, artifacted). Filtering rates were balanced across the two models (54% for ours, 55% for the baseline, no statistically significant difference), so the comparison

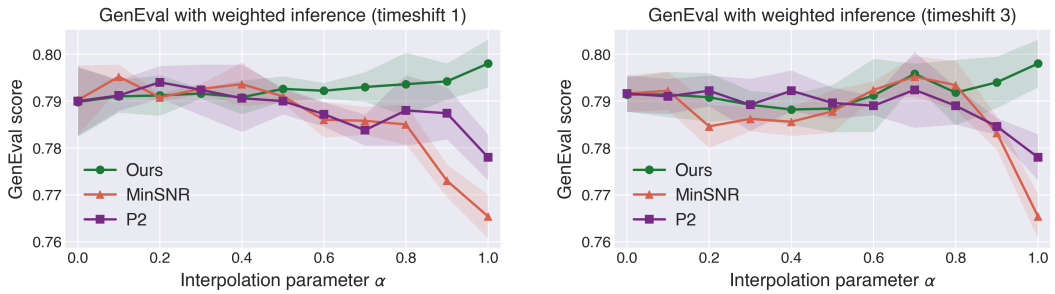

Figure 7: Change in GenEval score across interpolation parameter α in inference for all three perception-driven methods. We interpolate with uniform (left) and shifted uniform weights (right). Our method surpasses other two in the best setups (when using weights directly with $\alpha = 1.0$). Point values show the mean GenEval score and shaded regions show 95% confidence interval.

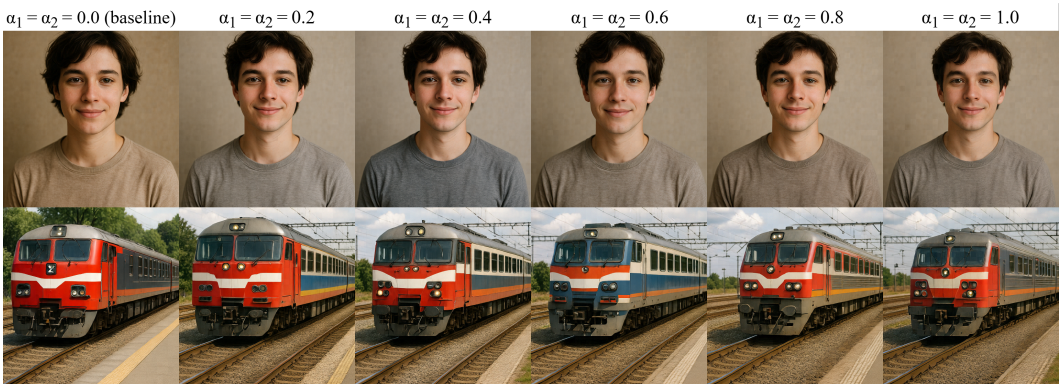

Figure 8: Change in generated images for increasing α, interpolating from uniform weights $\alpha_1 = \alpha_2 = 0.0$ to our perception-driven weights $\alpha_1 = \alpha_2 = 1.0$ for two fixed seeds. We observe that images become visually more realistic when increasing α. Best viewed zoomed in.

is not biased by the filter. In the second stage, the retained pairs were shown in randomized order and left/right position, and participants chose the image they preferred in terms of **quality**, **realism** and **aesthetics**. Across 393 pairwise judgments, our generations were preferred in $\mathbf{65}\%$ of comparisons (95% Wilson confidence interval: $[60.0\%, 69.4\%]$). Furthermore, 69.4% of 49 participants preferred our generations in the majority of the pairs they judged, against 20.4% for the baseline, thus pair-level preference was not driven by a small number of strongly opinionated users. The full protocol, verbatim instructions and more analysis are given in Appendix D.

6 Conclusion

We introduced Contrast Sensitive Flow (CSFlow), a procedure to weigh the information gain of individual steps in flow matching using human contrast sensitivity. We are the first to explicitly describe which frequencies are *being generated* at specific t's. In experiments on text-to-image and class-to-image generation tasks, we were able to verify that the introduction of such weights leads to quantitative improvements in training-free and finetuning settings. Qualitatively, images generated with the derived weights show higher visual realism.

Limitations and Future Work. While CSFlow improves the visual realism of generated images, we find that it does not impact the number of large geometric errors (e.g. missing legs) which are sometimes made by the base models. This suggests that the reasons for these types of errors are different from the suboptimal weighting of timesteps. A conceptual limitation of our weighting derivation is the required assumption of a typical screen and screen distance, leading to the necessity of calibration if fully optimal weights are desired. A promising venture for the future would be to analyze how image frequencies translate into common VAE latent spaces to transfer appropriate weighting schemes to latent image generation.

Acknowledgments

Jan Eric Lenssen and Bart Pogodzinski are supported by the German Research Foundation (DFG) - 556415750 (Emmy Noether Programme, project: Spatial Modeling and Reasoning).

Disclosures

The authors declare no competing interests.

References

Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Qinsheng Zhang, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, Tero Karras, and Ming-Yu Liu. eDiff-I: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022.

Peter Barten. Formula for the contrast sensitivity of the human eye. In *Image Quality and System Performance*, volume 5294 of *Proceedings of SPIE*, pages 231–238, 2003. doi: 10.1117/12.537476.

Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, Le Xue, Caiming Xiong, and Ran Xu. BLIP3-o: A family of fully open unified multimodal models—architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025a.

Junyu Chen, Dongyun Zou, Wenkun He, Junsong Chen, Enze Xie, Song Han, and Han Cai. DC-AE 1.5: Accelerating diffusion model convergence with structured latent space. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19628–19637, October 2025b.

Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11472–11481, 2022.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Sander Dieleman. Diffusion is spectral autoregression, 2024. URL `https://sander.ai/2024/09/02/spectral-autoregression.html`. Blog post.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 12606–12633. PMLR, 2024.

Fabian Falck, Teodora Pandeva, Kiarash Zahirnia, Rachel Lawrence, Richard Turner, Edward Meeds, Javier Zazo, and Sushrut Karmalkar. A fourier space perspective on diffusion models. *arXiv preprint arXiv:2505.11278*, 2025.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. GenEval: An object-focused framework for evaluating text-to-image alignment. In *Advances in Neural Information Processing Systems*, volume 36, 2023. Datasets and Benchmarks Track.

Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. Efficient diffusion training via Min-SNR weighting strategy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7441–7451, 2023.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, pages 6626–6637, 2017.

Jonathan Ho, Ajay N. Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851, 2020.

Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. Simple diffusion: End-to-end diffusion for high resolution images. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 13213–13232. PMLR, 2023.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, volume 35, pages 26565–26577, 2022.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. *arXiv preprint arXiv:2511.13720*, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023.

Zehong Ma, Ruihan Xu, and Shiliang Zhang. PixelGen: Pixel diffusion beats latent diffusion with perceptual loss. *arXiv preprint arXiv:2602.02493*, 2026.

Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8162–8171. PMLR, 2021.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115 (3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Evan Ruzanski and V. Chandrasekar. Scale filtering for improved nowcasting performance in a high-resolution X-band radar network. *IEEE Transactions on Geoscience and Remote Sensing*, 49 (6):2296–2307, 2011. doi: 10.1109/TGRS.2010.2103946.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, volume 29, pages 2226–2234, 2016.

Jaeyo Shin, Jiwook Kim, and Hyunjung Shim. Representation alignment for just image transformers is not easier than you think, 2026. URL `https://arxiv.org/abs/2603.14366`.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265. PMLR, 2015.

Zilyu Ye, Zhiyang Chen, Tiancheng Li, Zemin Huang, Weijian Luo, and Guo-Jun Qi. Schedule on the fly: Diffusion time prediction for faster and better image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, 2018.

A Broader Impact

Our work presents a method that improves the realism of generated images. Methods for realistic image generation have the potential to be misused; for example, for the creation of deepfake images. Our contribution is of conceptual nature with proof of concept experiments. We do not provide usable software that goes beyond the capabilities of state-of-the-art image generators from industry.

B Contrast Sensitivity Function (CSF)

B.1 CSF definition

CSF is defined for spatial frequencies measured in cycles per degree of visual angle. We use Barten's Barten [2003] formulation and parameters given below.

$$\text{CSF}(f) = \frac{M_{\text{opt}}(f)}{k}\left[\frac{2}{T}\left(\frac{1}{X_0^2}+\frac{1}{X_{\max}^2}+\frac{f^2}{N_{\max}^2}\right)\left(\frac{1}{npE}+\frac{\phi_0}{1-\exp\left(-\left(\frac{f}{u_0}\right)^2\right)}\right)\right]^{-1/2} \tag{13}$$

$$M_{\text{opt}}(f) = \exp\left(-2\pi^2\sigma^2 f^2\right) \tag{14}$$

$$\sigma = \frac{0.5}{60}, \qquad k = 3.0, \qquad T = 0.1, \tag{15}$$

$$X_0 = 60, \qquad X_{\max} = 12, \qquad N_{\max} = 15, \tag{16}$$

$$n = 0.03, \qquad p = 1.2 \times 10^6, \qquad E = 500, \tag{17}$$

$$\phi_0 = 3 \times 10^{-8}, \qquad u_0 = 7 \tag{18}$$

In the digital images spatial frequencies are defined in the cycles per pixel domain and depend on the physical size of a pixel and viewing distance to the screen. In all usages of CSF we assume the user is looking at the generated images on a regular size laptop in a viewing distance of 50 cm and a pixel size of 0.0114 cm which translates to approximately 3cm and 6 cm image edge length in 256×256 and 512×512 resolutions respectively.

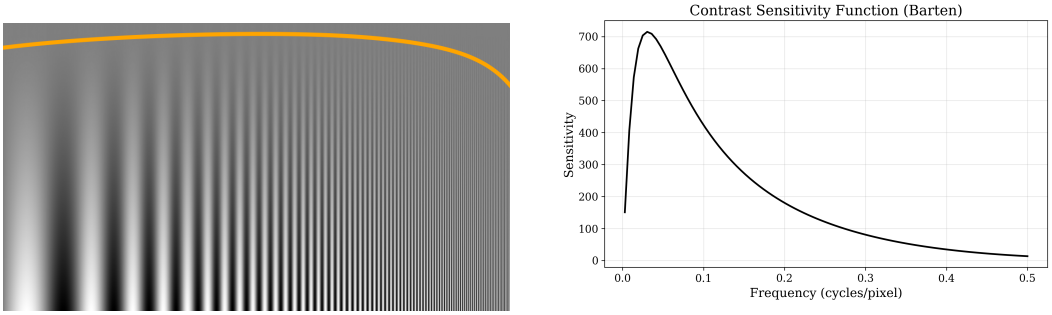

Figure 9: Visualization of CSF (left) and CSF values for spatial frequencies in cycles/pixel domain (right). We plot CSF in log-log space on top of spatial frequencies to visualize the idea of CSF. We assume viewing distance of $40\ cm$ and physical size of the plot $7\ cm$ ($\approx \frac{1}{3}$ of an A4 page).

B.2 Conversion from cycles per pixel to cycles per degree

The CSF is defined over spatial frequencies in cycles per degree of visual angle, while our Fourier frequencies are measured in cycles per pixel. Let p be the physical pixel size and d the viewing distance. The visual angle subtended by one pixel is

$$\theta_{\text{pix}} = 2\arctan\left(\frac{p}{2d}\right)\cdot\frac{180}{\pi} \quad \text{degrees/pixel.}$$

Therefore, an image frequency f measured in cycles per pixel corresponds to

$$f_{\text{cpd}} = \frac{f}{\theta_{\text{pix}}}$$

cycles per degree. In our experiments, we use $p = 0.0114$ cm and $d = 50$ cm.

C Technical details

C.1 Estimating the data power spectrum

The retained signal $r_{\text{signal}}(f, t)$ requires the expected image power spectrum

$$S_f = \mathbb{E}\left[|F_f(\mathbf{x}_{clean})|^2\right]. \tag{19}$$

We estimate S_f from the model's training dataset using the Radially Averaged Power Spectral Density (RAPSD) Ruzanski and Chandrasekar [2011]. We use RAPSD because our weighting depends only on the spatial frequency magnitude, not on the orientation of the corresponding Fourier component.

For an image $\mathbf{x}_{clean}$, let

$$P_{\mathbf{x}_{clean}}(u, v) = |F_{\mathbf{x}_{clean}}(u, v)|^2 \tag{20}$$

denote the two-dimensional DFT power at frequency coordinate (u, v). For a radial frequency bin f, let $\mathcal{A}_f$ be the set of Fourier coordinates whose distance from the origin falls into that bin. The RAPSD is

$$\text{RAPSD}_{\mathbf{x}_{clean}}(f) = \frac{1}{|\mathcal{A}_f|} \sum_{(u,v)\in\mathcal{A}_f} P_{\mathbf{x}_{clean}}(u, v). \tag{21}$$

We then estimate the expected image power as

$$S_f = \frac{1}{|D|} \sum_{\mathbf{x}_{clean}\in D} \text{RAPSD}_{\mathbf{x}_{clean}}(f). \tag{22}$$

Using this estimate, the retained signal is computed as

$$r_{\text{signal}}(f, t) = \frac{a_t^2 S_f}{a_t^2 S_f + b_t^2}. \tag{23}$$

C.2 Computing CSF-based weights

In the main text, we define the CSF-based weights in continuous time using the instantaneous increase

$$\partial_t r_{\text{signal}}(f, t).$$

The raw perceptual score at time t is

$$\widetilde{w}_{\text{CSFlow}}(t) = \frac{\int_\Omega \partial_t r_{\text{signal}}(f, t) \cdot CSF(f)\, df}{\int_\Omega \partial_t r_{\text{signal}}(f, t)\, df}, \tag{24}$$

where Ω denotes the set of radial frequency bins. This quantity is the average CSF value over the frequencies that become newly recoverable around time t.

We normalize the raw scores to have mean one over the denoising trajectory:

$$w_{\text{CSFlow}}(t) = \frac{\widetilde{w}_{\text{CSFlow}}(t)}{\int_0^1 \widetilde{w}_{\text{CSFlow}}(\tau)\, d\tau}. \tag{25}$$

Since the denoising interval has length one, this gives

$$\int_0^1 w_{\text{CSFlow}}(t)\, dt = 1,$$

so the average weight over time is unchanged. In practice, we compute interval-level weights on a timestep grid

$$0 = t_0 < t_1 < \cdots < t_T = 1, \qquad \Delta t_i = t_{i+1} - t_i.$$

For each interval, we approximate the integral of $\partial_t r_{\text{signal}}$ using the finite difference

$$\Delta r_{\text{signal}}(f, t_i, \Delta t_i) = r_{\text{signal}}(f, t_{i+1}) - r_{\text{signal}}(f, t_i). \tag{26}$$

This is consistent with the continuous formulation because

$$\Delta r_{\text{signal}}(f, t_i, \Delta t_i) = \int_{t_i}^{t_{i+1}} \partial_t r_{\text{signal}}(f, t)\, dt.$$

The discrete raw score is then

$$\widetilde{w}_{\text{CSFlow},i} = \frac{\sum_{f\in\Omega} \Delta r_{\text{signal}}(f, t_i, \Delta t_i) \cdot CSF(f)}{\sum_{f\in\Omega} \Delta r_{\text{signal}}(f, t_i, \Delta t_i)}. \tag{27}$$

Finally, we normalize the interval weights to have mean one over time:

$$w_{\text{CSFlow},i} = \frac{\widetilde{w}_{\text{CSFlow},i}}{\sum_{j=0}^{T-1} \Delta t_j \widetilde{w}_{\text{CSFlow},j}}. \tag{28}$$

For uniformly spaced intervals, this reduces to division by the empirical mean:

$$w_{\text{CSFlow},i} = \frac{\widetilde{w}_{\text{CSFlow},i}}{\frac{1}{T}\sum_{j=0}^{T-1} \widetilde{w}_{\text{CSFlow},j}}.$$

C.3 Constructing the smooth inverse CDF

The CSF-based weights define a time-dependent weighting function $w_{\text{CSFlow}}(t)$. After mean-one normalization, we have

$$\int_0^1 w_{\text{CSFlow}}(t)\, dt = 1,$$

so $w_{\text{CSFlow}}(t)$ can be interpreted as a probability density over denoising time. We define its cumulative distribution function as

$$CDF(t) = \int_0^t w_{\text{CSFlow}}(\tau)\, d\tau.$$

In practice, w_{CSFlow} is computed on a finite timestep grid

$$0 = t_0 < t_1 < \cdots < t_T = 1.$$

We approximate the CDF at the grid points using numerical integration:

$$CDF_i = \frac{\sum_{j=0}^{i-1} w_{\text{CSFlow}}(t_j)\Delta t_j}{\sum_{j=0}^{T-1} w_{\text{CSFlow}}(t_j)\Delta t_j}, \qquad \Delta t_j = t_{j+1} - t_j.$$

The denominator ensures that $CDF_T = 1$ numerically. We then construct a smooth monotone approximation of the inverse CDF from the pairs

$$(CDF_i, t_i)_{i=0}^{T}.$$

We denote this approximation by CDF^{-1}_{smooth}.

C.4 Training modification: weighted time sampling

For training, we sample timesteps from the CSF-weighted time distribution using inverse transform sampling. We first sample

$$u \sim \mathcal{U}(0, 1),$$

and then set

$$t = CDF^{-1}_{\text{smooth}}(u).$$

This biases training toward denoising times with larger $w_{\text{CSFlow}}(t)$, i.e. toward intervals where the newly recovered frequencies are more perceptually important.

C.5 Inference modification: weighted step sizes

For inference, we use the same smooth inverse CDF to construct a non-uniform sampling grid. Given N inference steps, we evaluate the inverse CDF at evenly spaced quantiles:

$$q_k = \frac{k}{N}, \qquad k = 0, 1, \ldots, N,$$

and define

$$\tilde{t}_k = CDF^{-1}_{\text{smooth}}(q_k).$$

The resulting grid $\{\tilde{t}_k\}_{k=0}^{N}$ has step sizes

$$\Delta\tilde{t}_k = \tilde{t}_{k+1} - \tilde{t}_k.$$

Since larger values of $w_{\text{CSFlow}}(t)$ make the CDF increase faster, the inverse CDF changes more slowly in those regions. Thus, evenly spaced quantiles produce smaller timestep gaps where w_{CSFlow} is large, resulting in a denser inference grid in perceptually important denoising intervals.

C.6 Interpolation parameters

In some cases (both for our method and the baselines), we found that using the derived weights provided a too-strong bias in the training distribution, or the inference step-sizes, causing performance degradation. To fairly compare the methods, we interpolate the weights with the model's original training timestep distribution or step-sizes from original paper (e.g. uniform or timeshifted), and report the best performing setup per method. In Table 3 we report the α_1 (training), α_2 (inference) values that gave us the best performance for each of the methods, which we used to obtain the main results in Tables 1 and 2.

Table 3: Interpolation parameters for the results reported in Table 1 and Table 2. A value of 1.0 means the weighting scheme was applied directly, without interpolation.

Model	Method	Inference-only α_2	Training-only α_1	Training + Inference (α_1, α_2)
PixelGen-XXL/16	P2	0.7	1.0	(1.0, 1.0)
	MinSNR	0.4	1.0	(1.0, 0.7)
	CSFlow (ours)	1.0	1.0	(0.8, 0.7)
JiT-H/16	P2	0.3	1.0	(1.0, 0.4)
	MinSNR	0.3	1.0	(1.0, 0.2)
	CSFlow (ours)	0.4	1.0	(0.4, 0.4)
PixelREPA-H/16	P2	0.5	–	–
	MinSNR	0.4	–	–
	CSFlow (ours)	0.5	–	–
PixelGen-XL/16	P2	0.3	–	–
	MinSNR	0.2	–	–
	CSFlow (ours)	0.5	–	–

We believe that although our weights match the visual importance, they are not guaranteed to be optimal in terms of generating the best images. While some frequencies might be irrelevant for the human eye (and thus are heavily down-weighted by our method), they might still be relevant for the flow model itself. For example, if very low frequencies are not modeled correctly (e.g. images are often too bright, too blue, etc.) then, in intermediate denoising states, the noisy image might become out of distribution for the model. By interpolating with base weights, we search for weights that put more emphasis on more important frequencies without losing too much capacity for other ones.

There is also a second, inference-specific reason when the weights are used only to place the step grid. There, the benefit of taking smaller steps in a perceptually important region is bounded by the quality of the flow the model has already learned: beyond some step density, smaller steps bring no benefits. On the other hand, reducing the density further at the less important regions will likely further harm the generation quality, since inference trajectories are not straight, and coarse ODE steps may cause a large integration error. The weights interpolation lets us balance these effects.

D User preference study

This appendix gives the full protocol and analysis for the study summarized in Section 5.5.

D.1 Protocol

Both stages were implemented as web applications and distributed by link to volunteer participants at our institution who are not connected to this work. The study was fully anonymous: preferences cannot be linked to any personal identifier. In both stages, participants were shown a brief description of the study and voluntarily proceeded to the task.

The two models used in this study were PixelGen-XXL/16 in its original version and with CSFlow weighted finetuning and inference ($\alpha_1 = 0.8, \alpha_2 = 0.7$, our best performing model on the GenEval). Participants were instructed to use a laptop or a larger display. Viewing conditions were otherwise not controlled, and all images were shown at their original resolution of 512×512.

Stage 1: quality filtering. The first stage is a single-image classification task whose purpose is to remove failed generations: images with large geometric errors, distorted elements, etc. For every session, 15 random pair indices are sampled from the full pool of generations across both methods (original and with CSFlow) and shown one at a time in random order. Since both methods' generations used the same prompt and seed, these image pairs contain approximately the same content. Within a session, pairs were sampled without replacement, so the pairs may repeat across different participants' sessions. The instruction was:

> *Your task is to decide whether the image is **correct** – contains **no** artifacts (e.g. **no** additional animal legs, **no** missing legs, the **right number** of heads or fingers), **no** distorted elements (e.g. merged two objects), has the **right geometry**, has the **correct text**, etc. Odd colors or textures are acceptable.*
> *For each image, choose:*
> ***Correct**: image looks acceptable, may contain unnatural texture, shifted color, unreadable small text, etc.*
> ***Incorrect**: image has visual errors, e.g. missing limbs, distorted objects, too many heads/fingers, etc.*

An image pair was passed to Stage 2 only if both images were independently marked *Correct*. Of 381 candidate pairs, 126 passed this filter.

Stage 2: pairwise preference. The second stage is a two-alternative forced-choice task on the pairs retained from Stage 1. For each trial, both the pair and its left/right assignment were randomized. Within a session, pairs were sampled without replacement; they may repeat across different participants' sessions. The instruction was:

> *Your task is to choose which of the two images (Image A and Image B) you prefer in terms of its quality, realism and aesthetics.*

In this stage, 49 participants took part and 393 pairwise judgments were collected. No participants or judgments were removed as outliers at either stage.

D.2 Filtering balance

Because Stage 2 operates only on pairs that survived Stage 1, a filter that passed one model's images more often than the other's could bias the comparison in Stage 2. We analysed the results, and it was not the case: the per-image pass rate was 54% for CSFlow generations and 55% for the baseline. A two-proportion z-test finds no significant difference ($z = 0.39$, $p = 0.699$; 95% confidence interval on the difference $[-5.7, +8.5]$ percentage points).

D.3 Results

Table 4 summarizes the results. Participants preferred CSFlow generations in 65.0% of all judgments, with a 95% confidence interval of $[60.0\%, 69.4\%]$ (Wilson score). Aggregating by participant rather

Table 4: User study results. Pair-level preference is the fraction of all judgements favouring CSFlow; participant-level figures count each participant once, according to whether they favoured a model in the majority of the pairs they judged.

Measure	Value
Pair-level	
Preference for CSFlow	65.0%
95% Wilson score interval	$[60.0\%,\ 69.4\%]$
Participant-level	
Preferred CSFlow	69.4%
Preferred baseline	20.4%
Tied	10.2%
Per-participant preference rate	
Mean / median	0.655 / 0.666
Standard deviation	0.250
Side-choice balance	
Left / right	48.2% / 51.8%

than by judgment, 69.4% of the participants preferred CSFlow generations in the majority of pairs they judged, 20.4% preferred the baseline and 10.2% were tied. The mean per-participant preference rate was 0.655 (median 0.666, standard deviation 0.250), indicating that the effect is distributed across observers rather than produced by a small number of strongly opinionated ones. Side-choice rates were balanced at 48.2% left against 51.8% right, giving no evidence of positional bias.

E Frequency reconstruction analysis

Flow and diffusion models often don't accurately model all aspects of the data distribution. The cartoonish-look discussed in Section 5.4 can intuitively be associated with a weaker power of the mid-high frequencies. To quantify that, we analyze how well our method models the marginal frequency spectrum (visualized in Figure 2) compared to the base model and inference.

We compare the radially averaged power spectral density (RAPSD, discussed in Appendix C.1) of generated images against that of the data. For a method m and radial frequency bin f we report the log-space error of the marginal power spectrum,

$$E_m(f) = \left|\log \mathrm{RAPSD}_m(f) - \log \mathrm{RAPSD}_D(f)\right|, \tag{29}$$

where RAPSD_m is averaged over generations and RAPSD_D over the dataset. We work in log space, because natural image power spectra span several orders of magnitude across the bands (Fig. 2), so an absolute error would be dominated by the lowest frequencies.

Both analyses were conducted on ImageNet-256 using finetuned JiT-H with CFG. For the baseline we use the model finetuned for the same number of steps with a uniform inference grid; for ours we use the CSFlow-weighted JiT-H model that performed best in terms of FID. We used 5 runs to estimate the mean and standard deviation of the scores.

Fig. 10 shows that CSFlow lowers the log-space spectrum error at every frequency, but the reduction is strongly non-uniform: it is largest in the mid-frequency band around $1/32$ to $1/8$ cycles per pixel, where the CSF retains between 47% and 100% of its peak value, and smallest at the two extremes, where the CSF is lower. In relative terms the reduction is 32.7% to 51.0% in the mid band against 1.0% at the lowest and 6.2% at the highest frequency. Overall, our method then produces improved modeling of the power spectrum, but the gains are higher in the middle-frequencies, where the CSF weight is high.

F Sensitivity of the CSFlow weights

The CSF is defined over cycles per degree of visual angle, so converting it to the image frequency axis requires assumptions about how the image is viewed (Section B.2). This appendix examines how

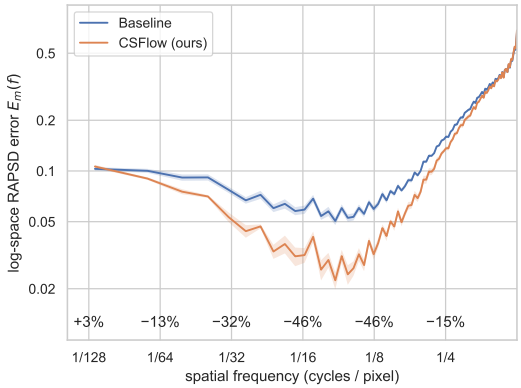

Figure 10: Log-space RAPSD error against spatial frequency for the baseline and CSFlow, with ± 1 standard deviation bands over 5 runs. Our method shows improvements across almost all frequencies, with the highest gains in the medium freqiency range. Note that the fluctuations of the curve may be caused by the aggregation of the radial frequency bins in RAPSD.

much the resulting weights depend on those assumptions. The short answer is that the three viewing parameters reduce to fewer degrees of freedom than they appear to have, and that the dependence is a shift along the log-frequency axis rather than a change in shape.

F.1 Image size and viewing distance

The weights depend on viewing distance, image size and image resolution. Of these, the first two are not independent: doubling the physical image size is exactly equivalent to halving the viewing distance, since both enter the conversion only through the visual angle subtended by one pixel. This is the usual scale ambiguity, and it means there is one parameter here, not two.

Changing that parameter rescales the arguments of the CSF multiplicatively. Because the CSF is conventionally plotted against log frequency, this corresponds to translating the frequency window along the CSF curve, leaving its shape intact. Table 5 shows the effect: configurations 2 and 3 produce identical weights, as the equivalence predicts.

F.2 Resolution

Resolution behaves differently. Increasing the resolution while holding image size and viewing distance fixed raises the effective SNR of the low-frequency content that was already present, because coarse structure is now averaged over more pixels – the observation that motivates the resolution-dependent schedule shifts of Simple Diffusion [Hoogeboom et al., 2023] and Stable Diffusion 3 [Esser et al., 2024].

In our setting, however, a change in resolution cannot be reduced to such a timestep shift. Increasing the resolution also introduces frequencies that did not exist in the lower-resolution image, which extends the range of arguments passed to the CSF. The effect on the weights therefore depends on the CSF's behaviour in a region it was not previously evaluated on, and cannot be computed exactly without additional assumptions about the empirically estimated CSF. Configuration 4 in Table 5 shows that halving the resolution gives weights close to, but distinct from, the size and distance changes.

F.3 Other display parameters

Display parameters that scale contrast uniformly across frequencies, such as brightness, have no effect on the weights. Our weighting depends only on the *relative* sensitivity across frequencies, through the normalization in Eq. 9, so a frequency-independent factor cancels.

Table 5: Mean w_{CSFlow} per interval of denoising time under four viewing configurations. Configuration 1 uses the assumptions of the main paper at 512×512. Configurations 2 and 3 are identical, as image size and viewing distance enter only through their ratio. Halving the resolution (4) is close to, but not the same as, either.

Configuration	$[0.0, 0.2)$	$[0.2, 0.4)$	$[0.4, 0.6)$	$[0.6, 0.8)$	$[0.8, 1.0)$
1. Paper assumptions, 512×512	1.34	1.55	1.18	0.71	0.23
2. $2\times$ image size	0.84	1.31	1.29	1.03	0.53
3. $\frac{1}{2}\times$ viewing distance	0.84	1.31	1.29	1.03	0.53
4. $\frac{1}{2}\times$ resolution (256)	0.91	1.36	1.28	0.97	0.49

G Proof of Theorem 1, Closed form of retained signal

Proof. By linearity of the DFT,

$$F_f(\mathbf{x}_t) = a_t F_f(\mathbf{x}_{clean}) + b_t F_f(\boldsymbol{\epsilon}). \tag{30}$$

Therefore,

$$\mathbb{E}[P_f(\mathbf{x}_t)] = \mathbb{E}\left[\left|a_t F_f(\mathbf{x}_{clean}) + b_t F_f(\boldsymbol{\epsilon})\right|^2\right] \tag{31}$$

$$= a_t^2 \mathbb{E}\left[|F_f(\mathbf{x}_{clean})|^2\right] + b_t^2 \mathbb{E}\left[|F_f(\boldsymbol{\epsilon})|^2\right] \tag{32}$$

$$+ 2a_t b_t \operatorname{Re} \mathbb{E}\left[F_f(\mathbf{x}_{clean})\overline{F_f(\boldsymbol{\epsilon})}\right]. \tag{33}$$

Since $\boldsymbol{\epsilon}$ is independent of $\mathbf{x}_{clean}$ and zero-mean, the cross term vanishes. By the flat unit expected power assumption (true for $\epsilon \sim \mathcal{N}(0, 1)$),

$$\mathbb{E}\left[|F_f(\boldsymbol{\epsilon})|^2\right] = 1. \tag{34}$$

Thus,

$$\mathbb{E}[P_f(\mathbf{x}_t)] = a_t^2 S_f + b_t^2. \tag{35}$$

Furthermore,

$$\mathbb{E}\left[P_f(a_t \mathbf{x}_{clean})\right] = a_t^2 S_f. \tag{36}$$

Hence,

$$\frac{\mathbb{E}\left[P_f(a_t \mathbf{x}_{clean})\right]}{\mathbb{E}\left[P_f(\mathbf{x}_t)\right]} = \frac{a_t^2 S_f}{a_t^2 S_f + b_t^2}. \tag{37}$$

□

H Extensions: model-dependent signal gain and equalized-gain objectives

The CSFlow weights described in the Section 4 are derived directly from the forward process. They depend only on the noise schedule a_t, b_t and the dataset power spectrum S_f via Theorem 1 and no model is evaluated to obtain them. The retained signal describes *what information is present* in real images after noising them to some t (rather than what could be reliably recovered using a denoising model). This is also why they do not transfer to latent models: in a latent flow there is no direct way to assess which pixel-space spectral information remains after noising to t, because a noisy latent is out of distribution for most decoders. As a result, analysing just the forward process could be misleading in case of latent models, as it may show that adding even small amounts of noise can remove all the information (since noisy latents are OOD), even though even a simple denoising model could reliably remove the noise and recover the information.

This appendix describes two future works that may follow from taking the denoiser into account.

H.1 A model-dependent signal gain metric

Instead of what survives noising, let's consider what a *trained* model can restore. For a diffusion or flow model with parameters θ, and for each image in a large evaluation set, we:

1. encode the image (omitted for pixel-space models),
2. noise the representation to level t,
3. denoise it back to $t = 1$ using the flow/diffusion model,
4. decode the result (omitted for pixel-space models).

We can then take the RAPSD of the reconstruction error between the output and the original image to see the radial spectrum of error. This, normalized by the RAPSD of the original image itself, tells us which radial frequencies did the model not reconstruct accurately. Then we need to average it over the data and normalize between two reference points. The first reference is the error at $t = 0$, where the representation is pure noise and nothing can be restored. The second, at $t = 1$ is the error of the autoencoder itself, where no information has been lost to noise (for a pixel-space model this second reference is a zero vector). It is our lower bound of the information loss. The result, which we denote $\hat{r}^{\theta}_{\mathrm{signal}}(f, t)$, takes values in $[0, 1]$ and indicates how much of the original signal at frequency f the model can restore from noise level t.

This quantity is conceptually close to the retained signal $r_{\mathrm{signal}}(f, t)$ of Theorem 1, but differs in one important part: it depends on a specific trained denoiser (θ). The information does not need to be present in $\mathbf{x}_t$ to be recovered, as long as the model can provide it from its learned prior. For example, let's consider a recognizable logo on a shirt. At a sufficiently high noise level the representation may no longer carry the logo's fine geometric details or textures, yet a model that has seen that specific logo multiple times can restore it correctly, since the coarse structure and surrounding context make it very clear which logo should it be. In such a case r_{signal} would report that the detail is destroyed while $\hat{r}^{\theta}_{\mathrm{signal}}$ would report that it is still recoverable.

$\hat{r}^{\theta}_{\mathrm{signal}}$ can be used in the same ways as r_{signal}. It can be also visualized as in Fig. 3 and its time derivative can measure the information gain per denoising step under that specific model, so it can be substituted for $\partial_t r_{\mathrm{signal}}$ in Eqs. 8 and 9 to obtain model-specific weights. That can be useful in two ways. First, in model analysis: the resulting map shows at which stage a given model commits to each level of detail. This can also be useful in comparing different latent spaces. Second, in inference-time acceleration: if a band of highest frequencies is already fully restorable at a moderately high noise level, then there is no reason to do more than one step there.

Nevertheless, we expect some potential problems. Most importantly, $\hat{r}^{\theta}_{\mathrm{signal}}$ is likely to show considerable overlap between the frequencies recovered at different noise levels, as many autoencoders distribute structural (low frequency) and detail (high frequency) information across channels with similar magnitude, and so both may end up being generated at the same time [Chen et al., 2025b]. Moreover, a spectral analysis of the final output may not reveal the non-spectral "decisions" that were made in early stages (eg. some semantic information about the species of a dog). Furthermore, the $\hat{r}^{\theta}_{\mathrm{signal}}$ weights depend on and are precise only for a specific model, and thus if one would want to use such timestep weights during training, the timestep weights themselves would also need to be updated over time.

H.2 Equal-weight latent space

Using Section 4.4 we build the timestep grid so that each step gets an equal CSFlow weight. A different approach would be to train a representation in which the weight is already equal across timesteps under a linear schedule.

To do that, one would need to repurpose the $\hat{r}^{\theta}_{\mathrm{signal}}$ introduced in the previous section. In this case, for computational feasibility (as this needs to happen during autoencoder training) we should consider working on individual-sample approximations of the $\hat{r}^{\theta}_{\mathrm{signal}}$. Explicitly, we can define the pixel-space reconstruction error map as:

$$z_1 = E_\theta(x_{raw}) \quad \text{Encode the image} \tag{38}$$
$$z_t = z_1 \cdot t + (1-t) \cdot \epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{1}) \quad \text{Noise the latent} \tag{39}$$
$$\hat{z}_1 = G_\phi(z_t, t) \quad \text{Denoise the latent} \tag{40}$$
$$\hat{x} = D_\theta(\hat{z}) \quad \text{Decode the image} \tag{41}$$
$$m_{x,t} = x_{raw} - \hat{x} \quad \text{Get the pixel-space error map} \tag{42}$$

with x_{raw} being the original pixel-space image, E_θ and D_θ defining the encoder and decoder, and G_ϕ the denoiser. We can then get the reconstruction error's spectral magnitude and it's radial average as:

$$m_{f,x,t} = F_f(m_{x,t}) \quad \text{2D Fourier transform of the error} \tag{43}$$
$$R_{x,t}(f) = \frac{RAPSD_{m_{f,x,t}}(f)}{RAPSD_x(f)} \quad \text{vector of reconstruction error ratio for each radial frequency} \tag{44}$$
$$R_{CSF,x,t} = \int_\Omega R_{x,t}(f) \cdot CSF(f) df \quad \text{Get a scalar CSF-weighted frequency information loss/error} \tag{45}$$

That allows us to get a single score for a specific image x how much human-vision-relevant spectral information has been removed by noising it to a noise level t. Then enforcing linearity on $R_{CSF,x,t}$ w.r.t t would ensure that each equal-sized denoising step of the latent diffusion/flow matching reduces the CSF-weighted spectral error by the same value.

We can enforce this linearity by enforcing the following constraint

$$\forall_{t,x,\epsilon} \frac{R_{CSF,x,t} - R_{CSF,x,1}}{R_{CSF,x,0} - R_{CSF,x,1}} = t \tag{46}$$

Satisfying that constraint is likely implausible, since it would require exactly linear information gain in the latent space, but it could be encouraged by the following regularization loss on θ.

$$\mathcal{L}_{CSF\text{-linear}} = \| \frac{R_{CSF,x,t} - sg(R_{CSF,x,1})}{sg(R_{CSF,x,0}) - sg(R_{CSF,x,1})} - t \|_2, \tag{47}$$

with sg denoting stopgrad. We believe it to be an interesting and promising future work direction that could enforce the equal perception-weighted information gain of each denoising step in latent-space diffusion and flow models.

I More qualitative results

More uncurated qualitative results can be viewed in Figure 11.



Figure 11: Pair comparisons between PixelGen-XXL/16 and CSFlow-weighted version (ours). Image pair indices were sampled randomly.

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: [N/A]

   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: [N/A]

   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: [Yes]

Justification: [N/A]

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: [N/A]

   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: All data and information is publicly available. Code reproducing the experiments will be released upon publication.

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: [N/A]

   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: Experiment results are averaged over five runs, with confidence interval set to 2 (plots) and 3 (result tables) standard deviations. User study results also report confidence intervals and statistical significance.

   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: [No]

   Justification: The computational resources required are identical to those of the used baseline models.

   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: [N/A]

   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: A broader impact section is given in 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: We base our experiments on existing models and data, which are already publicly available.

    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: [N/A]

    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: No 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: [Yes]

    Justification: User study detailed description is included in the Appendix.

    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: [No]

    Justification: 2AFC visual preference studies are treated as minimal-risk within our institution and do not require per-protocol ethics board submission.

    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: [N/A]

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.