How to Sample Your Diffusion: Parametric Score Rescaling for Steering Sampling Diversity

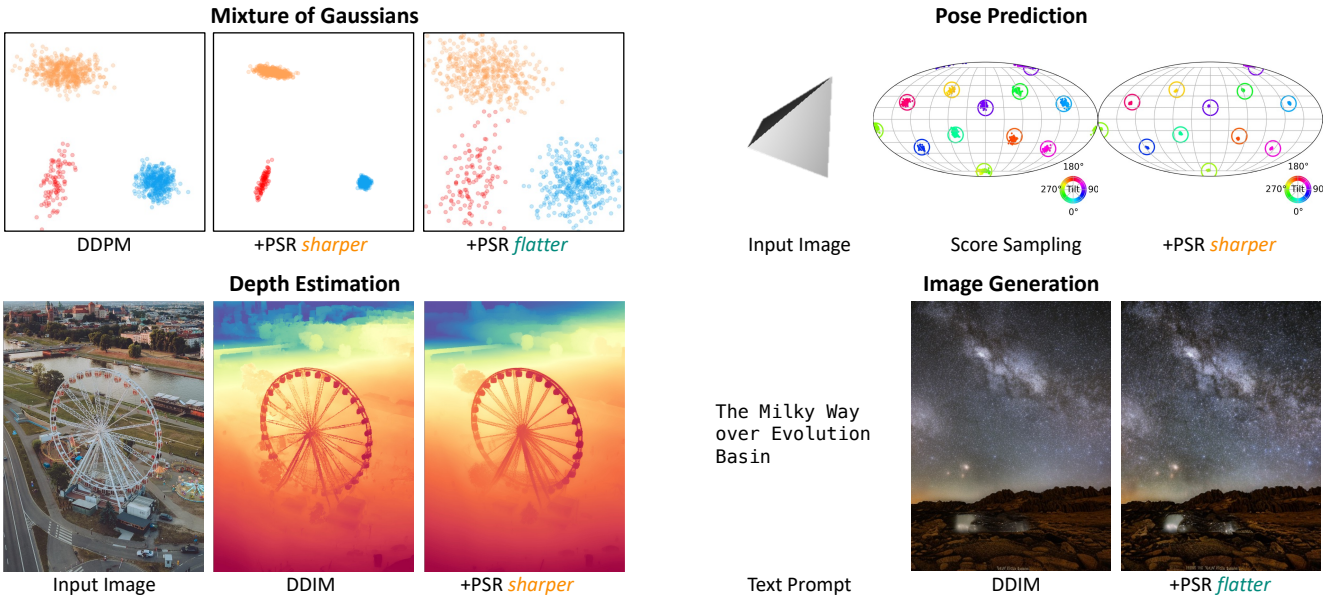


Figure 1. We present **Parametric Score Rescaling** (PSR), an approach to steer the sampling distribution in a diffusion model to be sharper or flatter than the training distribution (top-left). We use PSR to modify the sampling of pre-trained diffusion models across different tasks (pose prediction, depth estimation, and image generation) and find that it yields consistent improvements.

Abstract

We present a mechanism to steer the sampling diversity of denoising diffusion models, allowing users to sample from a sharper or broader distribution than the training distribution. We derive a (time-dependent) two-parameter 'score rescaling' function that can be used to scale the predicted noise during the diffusion sampling process. Notably, this approach does not require any finetuning or alterations to training strategy, and can be applied to any off-the-shelf diffusion model. We first validate our framework on toy 2D data, and then demonstrate its application for diffusion models trained across four disparate tasks – pose estimation, depth prediction, image generation, robot manipulation. We find that across these tasks, our approach allows sampling from sharper (or flatter) distributions, yielding performance gains e.g. *depth prediction models benefit from sampling more likely depth estimates, whereas image generation models perform better when sampling a (slightly) flatter distribution.*

1. Introduction

Denoising diffusion models have become ubiquitous across computer vision, enabling applications such as generation, perception, and interaction. Given training data $\{\mathbf{x}^n\}$, they can model the underlying data distribution $p_\theta(\mathbf{x})$ (or $p_\theta(\mathbf{x}|\mathbf{c})$ from training tuples $\{(\mathbf{x}^n, \mathbf{c}^n)\}$). At inference, these models then allow drawing samples $\mathbf{x} \sim p_\theta(\mathbf{x})$, *e.g.* to generate novel images.

However, in certain applications, we may not want to truly sample the modeled distribution. For example, when predicting depth from RGB input, we may want the more likely estimate(s) as output. In contrast, an artist exploring design choices may want the trained image generative model to yield more diverse samples even if they maybe somewhat less likely in the data. In this work, we ask whether we can 'steer' diffusion models to output more likely (or conversely, more diverse) samples. More specifically, *given any trained diffusion model, can we adapt its sampling process to trade-off sample diversity and likeli-*

hood at inference?

Towards answering this question, we note that a denoising diffusion model learns a family of score functions $\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)$ corresponding to the data distribution with varying levels of (time-dependent) noise. We study the case of isotropic gaussian data, and derive a 'score rescaling' function (determined by two parameters in addition to the diffusion schedule) that can blur/sharpen the sampling distribution by rescaling the predicted scores from a trained diffusion model. We then analyze the applicability of our 'parametric score rescaling' (PSR) formulation for broader settings, demonstrating that it can allow tuning the sampling diversity of generic diffusion models.

We perform experiments to highlight the broad applicability of PSR, studying four different applications of denoising diffusion models – pose estimation, depth prediction, image generation, and robot manipulation. Across these applications, we show that PSR can improve the performance of pre-trained diffusion models *e.g.* allowing more precise depth and pose inference, or enabling image generation to better match real data distribution.

Prior Art. We are of course not the first to consider the likelihood-diversity trade-off in sampling generative models. For example, the technique of 'temperature scaling' [1–3], initially used to calibrate classification networks, is often adopted for steering sampling from auto-regressive models by varying the temperature of the predicted next token distribution. However, as Shih *et al*. [4] demonstrated, this 'greedy' approach does not represent a temperature scaling of the joint distribution, and presented a technique for fine-tuning autoregressive (and diffusion) models for temperature-scaled inference.

While PSR can be similarly thought of as a temperature-scaling approach for sampling the joint distribution in diffusion models, it *does not require any training/finetuning*, and to our knowledge, PSR is the first such training-free technique for tuning sampling from diffusion models. Although classifier-free guidance (CFG) [5] can have similar effects in certain scenarios, we note that there are some fundamental differences. First, CFG cannot be applied for unconditional diffusion. Even for conditional diffusion, it requires changes to the training procedure (condition dropout when training) and cannot be leveraged for models trained without this protocol. Finally, is not mathematically equivalent to altering the sharpness of the modeled distribution which. Another interesting alternative to CFG suggested by Karras *et al*. [6] is to use a 'bad version of the diffusion model for guidance, and while their formulation does improve generation fidelity, it is not a probabilistically grounded mechanism for steering diversity and also requires multiple copies of a diffusion model.

2. Formulation

In this section, we derive a mechanism to steer the sampling process in denoising diffusion, effectively allowing sampling from a broader/narrower version of the distribution learned by a given (pre-trained) diffusion model. We first introduce some notation and review preliminaries of denoising difusion models. We then formalize the task of 'steerable' sampling and derive our parametric score rescaling formulation.

Notations and Conventions. We denote a diffusion model as ϵ_θ, and assume the model predicts noise *i.e.* $\hat{\epsilon} = \epsilon_\theta(\mathbf{x}_t, t)$, but note that this is a matter of convenience as different predictions are interchangeable (see supplementary). Finally, while we only discuss unconditional distributions and diffusion models in the text below, our formulation is equally applicable for the conditional setting.

Preliminaries. A denoising diffusion generates samples by reversing a forward process that adds noise to data $\mathbf{x}_0$:

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

A diffusion model ϵ_θ can be trained by learning to predict the added noise *i.e.* minimizing $\mathbb{E}_{\mathbf{x}_0,\epsilon}\|\epsilon_\theta(\mathbf{x}_t, t) - \epsilon\|^2$. Under this training objective, a diffusion model learns to approximate the score function of the (noisy) data:

$$\nabla_{\mathbf{x}_t} \log\ p(\mathbf{x}_t) \approx -\frac{\epsilon_\theta(\mathbf{x}_t, t)}{\sigma_t} \tag{2}$$

Given a trained model ϵ_θ, one can obtain samples from the (approximated) data distribution $p(\mathbf{x}_0)$ by following a reverse process that begins with $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, I)$ and iteratively denoises it, for example via DDIM [7] inference:

$$\mathbf{x}_{t-1} = \alpha_{t-1} \frac{\mathbf{x}_t - \sigma_t \epsilon_\theta(\mathbf{x}_t, t)}{\alpha_t} + \sigma_{t-1} \epsilon_\theta(\mathbf{x}_t, t) \tag{3}$$

2.1. Problem Statement

Given a training dataset $\{\mathbf{x}^n\}$, a denoising diffusion model ϵ_θ can approximate the underlying (unknown) data distribution $p(\mathbf{x}_0)$ and allow generating novel samples. In this work, we ask whether we can alter the sampling process such that the generated samples are not from $p(\mathbf{x}_0)$, but from a 'sharper' or 'flatter' version of it. To formalize this, we assume that the data distribution $p(\mathbf{x}_0)$ can be considered as a mixture of (*an unknown set of*) gaussians (while this is a strong assumption, we show empirically that the resulting approach is broadly applicable across tasks):

$$p(\mathbf{x}_0) \equiv \sum_m w_m \mathcal{N}(\mathbf{x}_0; \mu_m, \Sigma_m)$$

We can then define a family of corresponding 'sharper' or 'flatter' distributions (parametrized by k):

$$\bar{p}_k(\mathbf{x}_0) \equiv \sum_m w_m \mathcal{N}(\mathbf{x}_0; \mu_m, \frac{1}{k}\Sigma_m)$$

Intuitively, $\bar{p}_k(\mathbf{x}_0)$ represents a distribution where the variance near each local mode in the data distribution is scaled by $\frac{1}{k}$, with $k > 1$ leading to a 'sharper' distribution and $k < 1$ a 'flatter' one compared to the original. Using the above definition, we can formalize our problem statement as follows:

> Given a diffusion model ϵ_θ trained to approximate a (unknown) data distribution $p(\mathbf{x}_0)$, can we construct a diffusion model $\bar{\epsilon}_\theta$ that would produce samples from $\bar{p}_k(\mathbf{x}_0)$?

We note that this task formulation is different from a temperature scaling of the joint distribution as the mixture weights are unchanged *i.e.* we can view our formulation as seeking a mechanism to flatten/sharpen the samples around 'local' modes in a data distribution while preserving the 'global' distribution of the samples.

2.2. Parametric Score Rescaling

Toward answering the above question, we observe that just as ϵ_θ approximates the score of the noisy data *i.e.* $\epsilon_\theta(\mathbf{x}_t, t) \approx -\sigma_t \nabla_{\mathbf{x}_t} \log\ p(\mathbf{x}_t)$ (Eq. 2), $\bar{\epsilon}_\theta$ should similarly approximate the score for the (noisy versions of) data sampled from the target distribution $\bar{p}_k(\mathbf{x}_0)$. Specifically, if we consider $\bar{p}_k(\mathbf{x}_t|\mathbf{x}_0) \equiv \mathcal{N}(\alpha_t \mathbf{x}_0, \sigma_t^2 I)$ to represent the forward diffusion process on samples from $\bar{p}_k(\mathbf{x}_0)$, we would expect $\bar{\epsilon}_\theta(\mathbf{x}_t, t) \approx -\sigma_t \nabla_{\mathbf{x}_t} \log\ \bar{p}_k(\mathbf{x}_t)$. This leads us to the key insight that relating the score $\nabla_{\mathbf{x}_t} \log\ p(\mathbf{x}_t)$ to $\nabla_{\mathbf{x}_t} \log\ \bar{p}_k(\mathbf{x}_t)$ would allow deriving $\bar{\epsilon}_\theta(\mathbf{x}_t, t)$ from $\epsilon_\theta(\mathbf{x}_t, t)$. More specifically:

> If there is a linear transformation $\mathcal{T}$ s.t. $\nabla_{\mathbf{x}_t} \log\ \bar{p}_k(\mathbf{x}_t) \equiv \mathcal{T}(\nabla_{\mathbf{x}_t} \log\ p(\mathbf{x}_t))$, then $\bar{\epsilon}_\theta(\mathbf{x}_t, t) \equiv \mathcal{T}(\epsilon_\theta(\mathbf{x}_t, t))$

We show that such a $\mathcal{T}$ (which refer to as a 'score rescaling function') can indeed be derived for a simple scenario, and then analyze the formulation in more generic settings.

Score Rescaling for Isotropic Gaussian Data. We consider data drawn from an isotropic gaussian distribution $\mathbf{x_0} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma^2 I)$. Under the forward diffusion process (Eq. 1), the noisy data distribution $p(\mathbf{x_t})$ can also be shown to be a gaussian:

$$p(\mathbf{x_t}) \equiv \mathcal{N}(\alpha_t \boldsymbol{\mu}, (\alpha_t^2 \sigma^2 + \sigma_t^2) I)$$

We can also derive the score under the above distribution:

$$\nabla_{\mathbf{x}_t} \log\ p(\mathbf{x}_t) = -\frac{\mathbf{x}_t - \alpha_t \boldsymbol{\mu}}{\alpha_t^2 \sigma^2 + \sigma_t^2} \tag{4}$$

Similarly, if we define a corresponding 'sharper' or 'flatter' data distribution $\bar{p}_k(\mathbf{x}_0) \equiv \mathcal{N}(\boldsymbol{\mu}, \frac{1}{k}\sigma^2 I)$, we can compute its score as follows:

$$\nabla_{\mathbf{x}_t} \log\ \bar{p}_k(\mathbf{x}_t)) = -\frac{\mathbf{x}_t - \alpha_t \boldsymbol{\mu}}{\frac{\alpha_t^2}{k}\sigma^2 + \sigma_t^2} \tag{5}$$

Denoting by s_t the signal-to-noise ratio $\frac{\alpha_t^2}{\sigma_t^2}$ in denoising diffusion, we can combine Eq. 4 and Eq. A to derive $\mathcal{T}$ for isotropic gaussian data $\mathcal{N}(\boldsymbol{\mu}, \sigma^2 I)$:

$$\nabla_{\mathbf{x}_t} \log\ \bar{p}_k(\mathbf{x}_t) = \frac{s_t \sigma^2 + 1}{s_t \frac{\sigma^2}{k} + 1} \nabla_{\mathbf{x}_t} \log\ p(\mathbf{x}_t) \tag{6}$$

Note that $k = 1.0$ recovers the original diffusion model.

Mixture of Gaussians. While we derived the 'score rescaling' function for a single (isotropic) gaussian, we can show that is also a valid approximation if the data distribution is a mixture of *well-separated* isotropic gaussians *i.e.* for most points in space, a single mixture component dominates. We analyze this further in the supplementary, but intuitively, the score function in such a scenario can be approximated as the (weighted) score from the closest gaussian, and the above derivation still (approximately) holds.

Although this is not valid for anisotropic gaussians or mixtures of 'nearby' gaussians, we empirically show (in Section 3) that such a score rescaling still yields the desired flattening/sharpening of the sampling distribution.

Steering Diffusion Inference. We can operationalize the above derivation (Eq. 8) into a simple algorithm for steering the sampling from a pre-trained diffusion model ϵ_θ. Assuming the diffusion forward process has signal-to-noise ratio s_t, we can define 'score rescaling' function via two user-defined parameters k and σ:

> **Sampling with Parametric Score Rescaling** (k, σ)
>
> Given a diffusion model ϵ_θ, substitute its noise prediction with:
>
> $$\bar{\epsilon}_\theta(\mathbf{x}_t, t) = r_t(k, \sigma)\ \epsilon_\theta(\mathbf{x}_t, t),$$
>
> where rescaling factor $r_t(k, \sigma) := \frac{s_t \sigma^2 + 1}{s_t \frac{\sigma^2}{k} + 1}$

We note that this is a one line change that can be easily integrated into the sampling process in any off-the-shelf diffusion model! We first analyze its effects in sampling simple 2D data and then apply it to diffusion models trained across different tasks.

3. Analysis

In this section, we seek to gain insight into the empirical behavior of `PSR` as well as compare it to alternate strategies for tuning sampling sharpness. We do so by analyzing

the generated samples when applying PSR and other techniques for 'toy' 2D distribution $p(\mathbf{x}_0)$, where we can compare the generated samples to those under the (analytically computed) 'ground-truth' target distribution $\bar{p}_k(\mathbf{x}_0)$.

3.1. Setup

We use a mixture of gaussians for 2D distribution $p(\mathbf{x}_0)$. Formally, $p(\mathbf{x}_0) \sim \sum_i \pi_i N(\mu_i, \Sigma_i)$, where π_i represents mixture weight, and μ_i, Σ_i describe each gaussian parameters. Given a mixture of gaussians, we can analytically derive its score function, which we directly use for diffusion steps instead of training a neural network. We compare PSR with two other alternatives: (1) temperature scaling [1–3] in an autoregressive model and (2) naive noise scaling in diffusion [4], explained below.

Temperature Scaling in Autoregressive Model (AR). Autoregressive models the 2D data distribution as $p(x, y) \sim p(x)p(y|x)$, where $\mathbf{x}_0 = (x, y)$. $p(x)$ and $p(y|x)$ are also a gaussian mixture and can be computed analytically. Temperature scaling changes the sampling distribution of autogressive model as $x \sim p^\tau(x)$ and $y \sim p^\tau(y|x)$. We use ground truth analytical solutions for $p(x)$ and $p(y|x)$ during sampling. The limitation of this approach is that it does not apply temperature scaling on the "joint" distribution but only on the marginal distribution.

Naive Noise Scaling (NNS). DDPM[8] samples data using

$$\mathbf{x}_{t-1} = \alpha_t \frac{\mathbf{x}_t - \sigma_t \epsilon_\theta(\mathbf{x}_t, t)}{\alpha_t} + \sqrt{\sigma_{t-1}^2 - v_t^2}\epsilon_\theta(\mathbf{x}_t, t) + v_t \epsilon$$

, where $v_t = \sigma_{t-1}/\sigma_t \sqrt{1 - \alpha_t^2/\alpha_{t-1}^2}$, and $\epsilon \sim \mathcal{N}(\mathbf{0}, I)$. Naive noise scaling [4] tunes the sampling sharpness by increasing/reducing the variance of noise added at each denoising step of DDPM, i.e. $\hat{\epsilon} = \epsilon/k'$ at each step. Intuitively, by increasing/reducing the variance of noise, the variance of DDPM samples increases/decreases, respectively. We denote the factor of variance change as k'. However, k' has no clear probabilistic interpretation of its effects and is mode seeking.

3.2. Results

Isotropic Gaussian Mixture. We first test PSR on the simplest setup, where $p(\mathbf{x}_0)$ is a mixture of equally weighted isotropic gaussian with equal variance σ^2. We compare PSR with other sampling approaches previously explained. Fig.3 presents the results, where the first two columns display $p(\mathbf{x}_0)$ and $\bar{p}_k(\mathbf{x}_0)$, respectively, while the subsequent columns correspond to different sampling methods. When $k = 10.0$, PSR successfully reduces the variance of each mode while preserving their weights. In contrast, even in this simple setting, other methods sample non-uniformly from each mode. As previously explained, temperature scaling changes the marginal distribution, causing

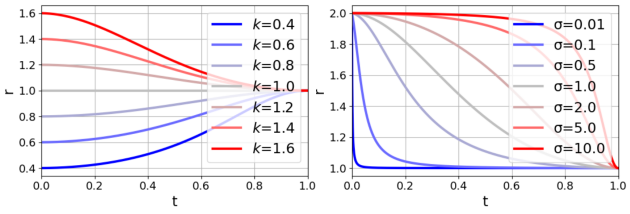


Figure 2. **Effect of σ and k on rescaling factor r_t.** Fixed $\sigma = 1.0$, varying k(left). Fixed $k = 2.0$, varying σ(right).

the weight of the Gaussian with x coordinate not overlapping with others to reduce. The mode-seeking behavior of naive noise scaling also leads to an uneven distribution across modes. Similarly, when $k = 0.5$, PSR successfully increases the variance across modes, and although the other methods also similarly increase variance, we do find that the weights from PSRare more uniform.

Anisotropic Gaussian Mixture. Under the more generic distribution where $p(\mathbf{x}_0)$ is an uneven mixture of anisotropic Gaussians, σ only approximates the variance of data. Nevertheless, as shown in Fig.4, PSR consistently creates a sharper distribution when $k > 1$ and a flatter distribution when $k < 1$ across varying values of σ, following our intention. This shows that even though we derived PSR under isotropic Gaussian data of variance σ, PSR can be leveraged in more diverse scenarios.

3.3. Interpreting Rescaling Hyperparameters

Beyond interpreting σ as the variance of the data, and k as a control factor for adjusting variance at each local mode, we provide an alternative interpretation of their roles using the rescaling factor r_t. This becomes particularly crucial when extending PSR to more complex domains, where multiple modes often exhibit varying and anisotropic variance, as demonstrated in the previous experiment. In Fig. 2, we plot the r_t against k and σ. Note that k now indicates the max/min of the rescaling factor r_t. And as $t \to 0$, signal-to-noise ratio $s_t \to \infty$, $r_t \to k$. Meanwhile, σ indicates how early we want to steer the sampling process. The larger σ, the earlier the sampling is steered. A very small σ let us use the original diffusion sampling ($r_t \approx 1.0$) and only steer the last few denoising steps.

4. Applications

We demonstrate the broad applicability and effectiveness of PSR by applying it to a diverse set of real-world applications, spanning perception (pose estimation and depth estimation in Sec. 4.1 and 4.2), generation (image synthesis in Sec. 4.3), and interaction (robotic manipulation in Sec. 4.4).

4.1. Pose Prediction

We first evaluate PSR on object pose prediction from a single image. We focus on predicting $SO(3)$ rotations of ob-

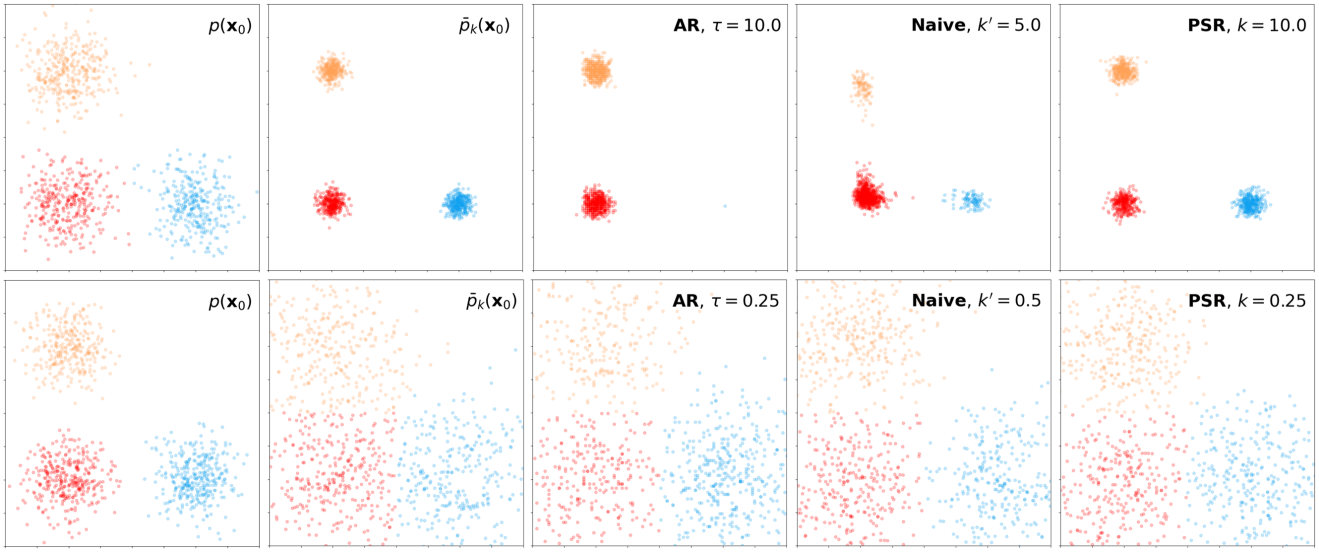


Figure 3. **Comparison on uniform mixture of 2D isotropic gaussians. AR** - temperature scaling, **Naive** - naive noise scaling (NNS), **PSR** - ours. When $k = 10.0$ (top), AR and NNS tend to converge to a subset of modes (orange, red) and (red), respectively. When $k = 0.25$(bottom), the trend becomes opposite. AR biases the samples more on the mode it lost at $k = 10.0$(blue), and NNS also focuses on the modes it was less focusing at $k = 10.0$(orange, blue). **PSR** preserves weights uniformly at both $k = 10.0, 0.25$.

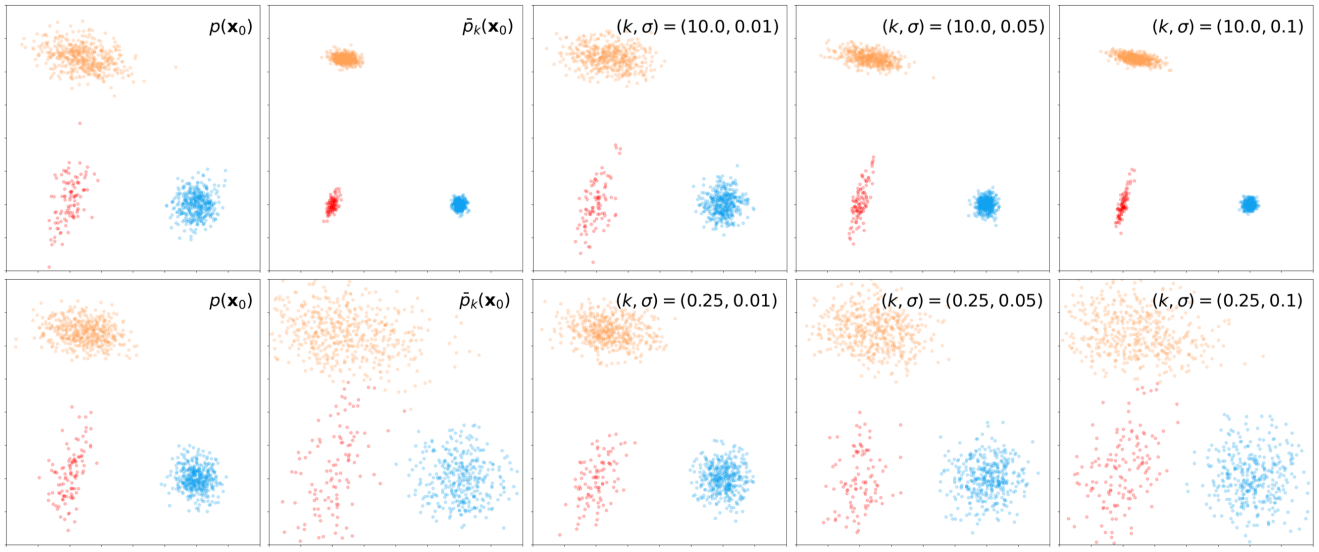


Figure 4. **PSR on nonuniform mixture of 2D anisotropic gaussians.** The mixture weight of $p(\mathbf{x}_0)$ is 0.1(red), 0.5(orange), and 0.4(blue). When $k = 10.0$(top), **PSR** consistently makes the distribution sharper, while making the distribution flatter at $k = 0.25$(bottom), across all σ values.

jects using the SYMSOL dataset [9], which contains geometric shapes with a high order of symmetries. The inherent ambiguities arising from object symmetries necessitate modeling a multi-modal distribution. Previous work [10–14] show that diffusion models can effectively model and sample from such multi-modal distributions and predict accurate poses. Since better pose predictions require samples close to the ground truth modes, we apply PSR to the $SO(3)$ diffusion model following the setup of [12] to sample a sharper distribution using k values greater than 1.

We visualize the effect of PSR in Fig. 5 where we show the sampled poses on an example image from SYMSOL. PSR samples poses more concentrated around ground truth modes (the circle centers) than score sampling used in [12]. Moreover, we evaluate PSR quantitatively in Tab. 1. PSR predictions have lower average error and higher ac-

Table 1. **Quantitative results for pose estimation.** We show the mean error of predictions in degrees, and percentage accuracy of prediction within a threshold of 0.1, 0.2, 0.5, and 1 degree. We use $(k, \sigma) = (7.0, 0.5)$ for PSR and $k' = 40$ for naive noise scaling.

	Error (deg) ↓	Acc@ (deg) ↑			
		0.1	0.2	0.5	1.0
Score Sampling	0.444	1.37	9.44	68.33	97.91
+ Naive Noise Scaling(40)	**0.350**	**3.43**	**20.02**	**84.98**	**99.10**
+ PSR (7.0, 0.5)	0.356	3.02	18.52	84.05	99.00

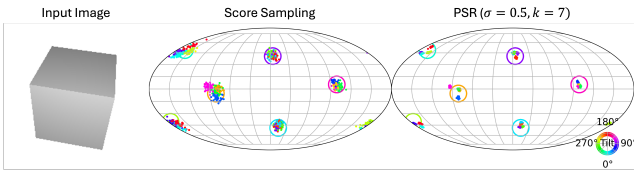


Figure 5. **Predicted poses on SYMSOL**. We show predicted poses (right) on an input image (left) and notice PSR reduces prediction error. Each sample is a dot on the sphere, positioned by its first canonical axis, with color indicating rotation. Circles denote ground truth poses. We modify the location of samples to exaggerate error by a factor of 15 to show visual difference given plotting constraints.

curacy under a range of accuracy thresholds compared to score sampling, highlighting the benefits of predicting close to modes. Furthermore, we find that naive noise scaling also reduces pose error, achieving a performance slightly better than PSR on SYMSOL. This is a strong sign that steering the sampling distribution during test time can, in fact, improve model performance. However, we note that PSR remain robust and applicable over many tasks and sampling methods where naive noise scaling is not possible.

Finally, we ablate the mean prediction error over σ and k (see supplementary figure and details). PSR consistently reduces the prediction error across a wide range of $k \in (1, 40]$ compared to the baseline ($k = 1$) and reaches the optimal at $k \approx 7$.

4.2. Depth Estimation

The task of monocular depth estimation is inherently challenging due to its uncertainty—an object could appear large but be far away, or small but close. Several methods [15–17] addresses this by using diffusion models. We chose Marigold [17] in our experiment, which fine-tuned a pre-trained text-to-image diffusion model for depth estimation, achieving impressive results. However, individual samples may be suboptimal due to the stochastic nature of diffusion-based sampling and the ambiguity of depth estimation. To mitigate this, Marigold introduces a test-time ensembling method that aggregates multiple samples using a pixel-wise median, significantly improving final performance at the cost of increased computational expense.

As shown in Tab. 2, PSR enhances prediction accuracy by sampling from a sharper distribution, ensuring more probable samples given the input image. Notably, PSR is complementary to the ensembling—we achieve similar performance on ETH3D [18] when ensembling with 2 samples compared to the DDIM result with an ensemble size of 10. We also include some qualitative comparisons in Fig. 6. Note that PSR predicts cleaner depth compared to DDIM, especially in regions with high uncertainty (as highlighted in the figure). We also show the effect of σ and k on the AbsRel metric in supplementary. Compared with the DDIM sample ($k = 1$), PSR demonstrates consistent performance gain in various (k, σ) configurations.

Table 2. **Quantitative Evaluation of Depth Estimation.** PSR improves the results without additional computation and outperforms the naive baseline. With the ensemble technique, PSR further improves the performance. With an ensemble size of 2, PSR and get compatible results to the optimal performance (DDIM w/ensemble=10). We use $(k, \sigma) = (1.3, 10.0)$ for PSR.

	NYUv2 [19]		ETH3D[18]	
	AbsRel ↓	δ_1 ↑	AbsRel ↓	δ_1 ↑
DDIM	6.0	95.9	7.1	90.4
+ Naive Noise Scaling	5.85	**96.0**	6.82	95.6
+ PSR	**5.84**	**96.0**	**6.68**	**95.7**
+ PSR (w/ ensemble = 2)	5.67	96.2	6.50	95.8
DDIM (w/ ensemble=10)	5.5	96.4	6.5	96.0

Table 3. **Results for Image Generation**. PSR improves image fidelity (FID) and text-alignment (CLIP) consistently across multiple sampling methods—DDIM, DDPM, and EulerDiscrete.

	FID ↓	CLIP ↑
DDIM	21.28	33.54
+ PSR (0.95, 1.0)	**20.05**	**33.61**
DDPM	22.81	33.66
+ Naive Noise Scaling (0.96)	19.87	33.68
+ PSR (0.9, 1.0)	**19.57**	**33.77**
EulerDiscrete	22.11	33.54
+ PSR (0.95, 1.0)	**19.95**	**33.61**

4.3. Image Generation

In addition to pose and depth prediction, diffusion models [8, 20–23]—are among the most powerful and widely used approaches for image generation. We examine the effect of steering the sampling distribution of diffusion models for diversity versus likelihood with PSR in image generation. Unlike pose and depth estimation, which requires high precision, image generation is a more creative task where sampling from flatter distributions helps to recover more pleasing images with more high frequency details. We use Stable Diffusion v2 [21] and evaluate FID [24, 25] and CLIP [26] against a 5k image subset from LAION Aesthetics [27]

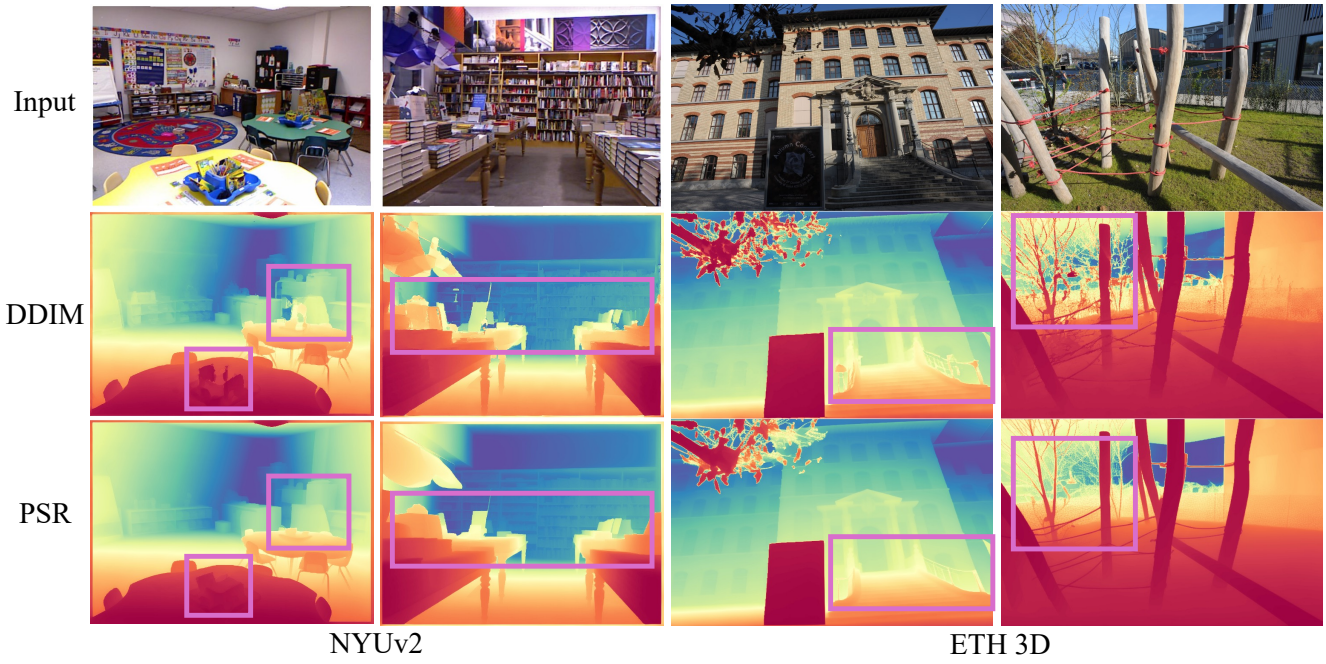


Figure 6. **Qualitative Depth Estimation Comparisons.** Compared to DDIM, PSR with $k > 1$ predicts cleaner depth in the regions with high uncertainty (highlighted by pink boxes).

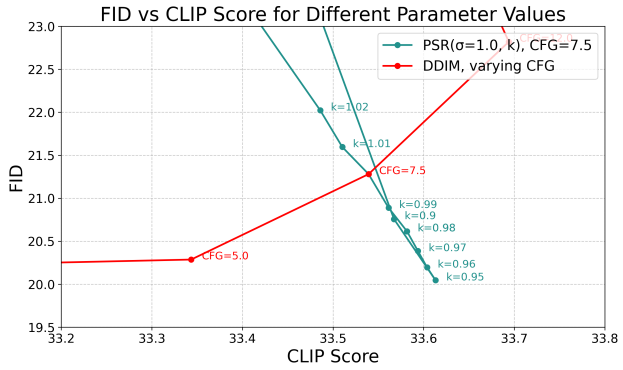


Figure 7. **FID vs CLIP Score.** PSR achieves better text-alignment (CLIP) and image fidelity (FID), improving upon the Pareto frontier of CFG tuning, which trades off between FID and CLIP.

across different configurations of CFG, PSR parameter k, and sampling methods. We fixed the number of sampling steps to 50. In Fig. 7, we see adjusting CFG makes a trade-off between text-alignment and image fidelity—higher CFG increases CLIP score at the cost of worse FID. Meanwhile, PSR allows for additional tuning beyond the Pareto frontier of CFG. Fixing CFG at 7.5, $k = 0.95$ achieves better FID and CLIP score compared to DDIM sampling.

Moreover, we show that PSR is capable of improving both FID and CLIP scores across different diffusion sampling methods in Tab. 3. While naive noise scaling with $k' < 1$ also improves upon the DDPM baseline, PSR remains better while being applicable to sampling methods that do not support naive noise scaling, such as DDIM.

Qualitatively, we observe in Fig. 8 that lower ks lead to images with more high-frequency detail (in the extreme case more noisy), and higher ks lead to a smoother image. We infer that using a smaller k flattens the modeled distribution and allows better coverage of the desirable image space. Overall, our results highlight that the control over the likelihood-diversity trade-off enabled by ours is beneficial in image generation.

4.4. Robotic Manipulation

Lastly, we examine the applicability of PSR on predicting robot actions, with a focus on robotic manipulation. In particular, we use the diffusion policy [28–31] as the diffusion backbone, where a policy is trained for each task. We use checkpoints provided by Chi *et al*. [28], containing both transformer (Diffusion Policy-T) and U-Net (Diffusion Policy-C). It is worth mentioning that the problem is formulated as sequential decision-making, which differs from previous domains. In particular, the policy only models a short-horizon or future actions rather than the full length of the task episode. Hence, we are steering the sampling distribution of the short-horizon actions, rather than a distribution of the long-horizon actions [4], which may cause undesirable effects across the episode.

We choose two complex manipulation tasks (Transport, Tool-Hang) [32, 33] and one simple 2D manipulation task

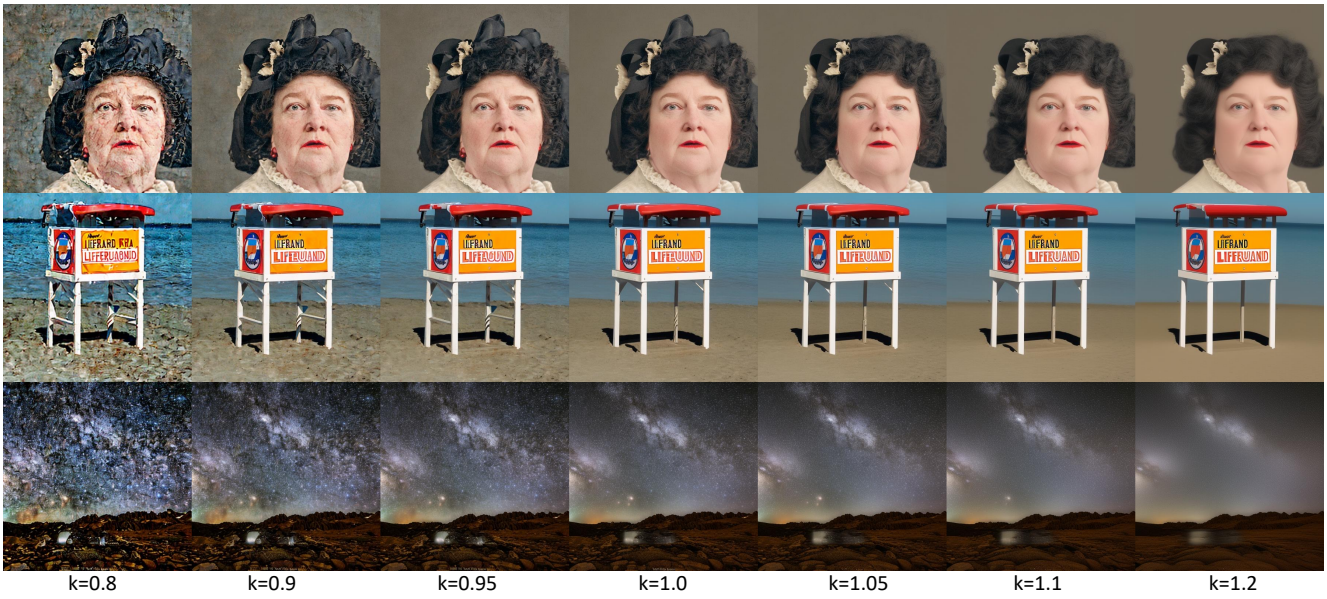


Figure 8. **Qualitative Results for Varying K.** PSR allows for tuning the generated outputs to be more diverse and detailed (lower k) or more smooth and likely (higher k). While neither extreme is desirable, we notice a slightly smaller of $k = 0.95$ gives pleasing images with more details.

(Push-T) in simulation with state observation, with multi-human data (mh) for transport and proficient human data (ph) for the remaining two. Results are in Table. 4. Without any further training, PSR consistently improves performance, especially when the base policy performance is low. Compared to other domains, optimal k is smaller than 1.0 for some tasks (Transport, Tool Hang) and larger than 1.0 for others (Push T). This is because each base policy has a different level of proficiency. For a nonproficient base policy, sampling diverse actions with $k < 1.0$ tends to help escape the local minimum, while for a proficient base policy, $k > 1.0$ tends to help. Moreover, for certain tasks and models, both $k > 1$ and $k < 1$ show performance gain. As $k > 1$ and $k < 1$ are both steering the sampling process, changing the sampling distribution may help to escape the local minimum, even for $k > 1$. See supplementary for details.

5. Discussion

We presented PSR, an approach to alter the sampling distribution for a pre-trained diffusion model. While we demonstrated its efficacy across several (toy and real) tasks, there are fundamental limitations worth highlighting. First, unlike temperature scaling, PSR can only alter the 'local' sampling *e.g.* PSR does not change the weights of the components in a gaussian mixture, only the variance. Moreover, while PSR does empirically steer the sampling diversity in generic scenarios, the theoretical guarantees are limited to simpler settings and one may be able to derive a better algorithm for different distributions. Nevertheless, as PSR can be readily applied to any off-the-shelf diffusion model, we believe it would a generally useful technique for the community to explore and also hope this work inspires further investigations on how one can controllably vary the sampling distribution of a pre-trained diffusion model.

Table 4. **Results for Robotic Manipulation.** Success rates are computed across 3 seeds, 3 checkpoints, and 150 different environment initial conditions (1350 episodes in total), where we compute the mean and std over seeds. The results are computed with best (k, σ) and k' for both PSR and Naive Noise Scaling.

	Transport	ToolHang	Push-T
Model	mh	ph	ph
DiffusionPolicy-C	63.2 ± 0.3	57.0 ± 0.4	90.1 ± 0.4
+ PSR	$\mathbf{66.9 \pm 1.4}$	$\mathbf{63.3 \pm 1.3}$	$\mathbf{90.6 \pm 0.004}$
+ Naive Noise Scaling	64.6 ± 0.7	58.4 ± 1.1	90.2 ± 0.6
DiffusionPolicy-T	41.4 ± 2.7	87.7 ± 1.7	90.2 ± 0.5
+ PSR	$\mathbf{45.9 \pm 2.4}$	$\mathbf{89.3 \pm 0.9}$	$\mathbf{90.7 \pm 0.5}$
+ Naive Noise Scaling	45.3 ± 1.9	88.7 ± 0.9	90.3 ± 0.3

References

[1] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *ICML*, 2017. 2, 4

[2] Jeremy Nixon, Michael W Dusenberry, Linchuan Zhang, Ghassen Jerfel, and Dustin Tran. Measuring calibration in deep learning. In *CVPR workshops*, 2019.

[3] Shrey Desai and Greg Durrett. Calibration of pre-trained transformers. In *EMNLP*, 2020. 2, 4

[4] Andy Shih, Dorsa Sadigh, and Stefano Ermon. Long horizon temperature scaling. In *ICML*, 2023. 2, 4, 7

[5] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. 2022. 2

[6] Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. *Advances in Neural Information Processing Systems*, 37:52996–53021, 2024. 2

[7] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 2

[8] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 4, 6

[9] Kieran A Murphy, Carlos Esteves, Varun Jampani, Srikumar Ramalingam, and Ameesh Makadia. Implicit-pdf: Non-parametric representation of probability distributions on the rotation manifold. In *Proceedings of the 38th International Conference on Machine Learning*, pages 7882–7893, 2021. 5

[10] Adam Leach, Sebastian M Schmon, Matteo T. Degiacomi, and Chris G. Willcocks. Denoising diffusion probabilistic models on SO(3) for rotational alignment. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*, 2022. 5

[11] Yesukhei Jagvaral, Francois Lanusse, and Rachel Mandelbaum. DIFFUSION GENERATIVE MODELS ON SO(3), 2023.

[12] Tsu-Ching Hsiao, Hao-Wei Chen, Hsuan-Kung Yang, and Chun-Yi Lee. Confronting ambiguity in 6d object pose estimation via score-based diffusion on se(3). In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 352–362, 2024. 5

[13] Jianyuan Wang, Christian Rupprecht, and David Novotny. PoseDiffusion: Solving pose estimation via diffusion-aided bundle adjustment. 2023.

[14] Jason Y Zhang, Amy Lin, Moneish Kumar, Tzu-Hsuan Yang, Deva Ramanan, and Shubham Tulsiani. Cameras as rays: Pose estimation via ray diffusion. In *International Conference on Learning Representations (ICLR)*, 2024. 5

[15] Yiquan Duan, Xianda Guo, and Zheng Zhu. Diffusiondepth: Diffusion denoising approach for monocular depth estimation. In *European Conference on Computer Vision*, pages 432–449. Springer, 2024. 6

[16] Saurabh Saxena, Abhishek Kar, Mohammad Norouzi, and David J Fleet. Monocular depth estimation using diffusion models. *arXiv preprint arXiv:2302.14816*, 2023.

[17] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 6

[18] Thomas Schops, Johannes L Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3260–3269, 2017. 6

[19] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 6

[20] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023. 6

[21] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 6

[22] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[23] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 6

[24] 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. *Advances in neural information processing systems*, 30, 2017. 6

[25] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On aliased resizing and surprising subtleties in gan evaluation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11410–11420, 2022. 6

[26] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021. 6

[27] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in neural information processing systems*, 35:25278–25294, 2022. 6

[28] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *IJRR*, 2023. 7, 2

[29] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. In *RSS*, 2024.

[30] Yanjie Ze, Zixuan Chen, Wenhao Wang, Tianyi Chen, Xialin He, Ying Yuan, Xue Bin Peng, and Jiajun Wu. Generalizable humanoid manipulation with improved 3d diffusion policies. *arXiv preprint arXiv:2410.10803*, 2024.

[31] Tsung-Wei Ke, Nikolaos Gkanatsios, and Katerina Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. In *CoRL*, 2024. 7

[32] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *CoRL*, 2021. 7

[33] Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Soroush Nasiriany, and Yifeng Zhu. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293*, 2020. 7

[34] Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011. 1

[35] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022. 1

How to Sample Your Diffusion: Parametric Score Rescaling for Steering Sampling Diversity

Supplementary Material

A. Proofs and Discussions

Relating Score Function to Noise Prediction. In a diffusion model, we define a forward process s.t. $\mathbf{x}_t = \alpha_t \mathbf{x}_0 + \sigma_t \epsilon$, and trained the model ϵ_θ to predict the noise given $\mathbf{x}_t$. Defining $\bar{\mathbf{x}}_t \equiv \frac{\mathbf{x}_t}{\alpha_t}$, we get:

$$\bar{\mathbf{x}}_t = \mathbf{x}_0 + \frac{\sigma_t}{\alpha_t}\epsilon$$

Following Tweedie's formula [34], we can show that the noise prediction is an estimate of the score for the variable $\bar{\mathbf{x}}_t$:

$$\nabla_{\bar{\mathbf{x}}_t} \log\ p(\bar{\mathbf{x}}_t) = -\frac{\epsilon_\theta(\mathbf{x}_t, t)}{\frac{\sigma_t}{\alpha_t}}$$

We can also show that:

$$\nabla_{\bar{\mathbf{x}}_t} \log\ p(\bar{\mathbf{x}}_t) = \nabla_{\bar{\mathbf{x}}_t} \log\ p(\mathbf{x}_t) = \alpha_t \nabla_{\mathbf{x}_t} \log\ p(\mathbf{x}_t)$$

Combining the above, we obtain:

$$\nabla_{\mathbf{x}_t} \log\ p(\mathbf{x}_t) = -\frac{\epsilon_\theta(\mathbf{x}_t, t)}{\sigma_t} \tag{7}$$

Application for $\mathbf{x}_0$ and v prediction diffusion models. Our method can be applied to all types of diffusion models, although we assumed a noise prediction network in the main paper. For several parameterizations for training a score-based model, e.g. denoising prediction ($\mathbf{x}_{0,\theta}$), noise prediction (ϵ_θ), and v-prediction (v_θ) [35], one can easily show their equivalence given $\mathbf{x}_t$:

$$\epsilon_t = \frac{\mathbf{x}_t - \alpha_t \mathbf{x}_{0,\theta}(\mathbf{x}_t, t)}{\sigma_t}$$
$$\epsilon_t = \alpha_t v_\theta(\mathbf{x}_t, t) + \sigma_t \mathbf{x}_t$$

. After obtaining $\bar{\epsilon_t} = r_t \epsilon_t$, one can have:

$$\bar{\mathbf{x}}_0 = (\mathbf{x}_t - \sigma_t \bar{\epsilon_t})/\alpha_t$$
$$\bar{v}_t = (\bar{\epsilon}_t - \sigma_t \mathbf{x}_t)/\alpha_t$$

Mixture of Isotropic Gaussians. The probability density function(PDF) of a mixture of isotropic gaussians is given by:

$$p(\mathbf{x}) \sim \sum_{i=1}^{K} \pi_i \mathcal{N}(\mathbf{x}; \mu_i, \sigma_i^2 I)$$

where I is the identity matrix, and $\mathcal{N}(\mathbf{x}; \mu_i, \sigma_i^2 I)$ is the PDF of a multivariate Gaussian distribution. From above, the score function can be derived as follows:

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) = \nabla_{\mathbf{x}} \log \left[\sum_{i=1}^{K} \pi_i \mathcal{N}(\mathbf{x}; \mu_i, \sigma_i^2 I) \right]$$

$$= \frac{\sum_{i=1}^{K} \pi_i \nabla_{\mathbf{x}} \mathcal{N}(\mathbf{x}; \mu_i, \sigma_i^2 I)}{\sum_{i=1}^{K} \pi_i \mathcal{N}(\mathbf{x}; \mu_i, \sigma_i^2 I)}$$

$$= \frac{\sum_{i=1}^{K} \pi_i \left(-\frac{1}{\sigma_i^2}(\mathbf{x} - \mu_i) \right) \mathcal{N}(\mathbf{x}; \mu_i, \sigma_i^2 I)}{\sum_{i=1}^{K} \pi_i \mathcal{N}(\mathbf{x}; \mu_i, \sigma_i^2 I)}$$

when each gaussians are well separated, at each point $\mathbf{x}$, one gaussian dominates the density. Suppose the dominating gaussian to be $\mathcal{N}(\mathbf{x}; \mu_j, \sigma^2 I)$. Then we can approximate the score function as below.

$$\mathcal{N}(\mathbf{x}; \mu_j, \sigma_j^2 I) >> \mathcal{N}(\mathbf{x}; \mu_i, \sigma_i^2 I) \ for\ j \neq i.$$

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) = \frac{\sum_{i=1}^{K} \pi_i \left(-\frac{1}{\sigma_i^2}(\mathbf{x} - \mu_i) \right) \frac{\mathcal{N}(\mathbf{x}; \mu_i, \sigma_i^2 I)}{\mathcal{N}(\mathbf{x}; \mu_j, \sigma_j^2 I)}}{\sum_{i=1}^{K} \pi_i \frac{\mathcal{N}(\mathbf{x}; \mu_i, \sigma_i^2 I)}{\mathcal{N}(\mathbf{x}; \mu_j, \sigma_j^2 I)}}$$

$$\simeq -\frac{1}{\sigma_i^2}(\mathbf{x} - \mu_i)$$

, which is a score function of $\mathcal{N}(\mathbf{x}; \mu_j, \sigma_j^2 I)$.

In a diffusion model where $p(\mathbf{x})$ follows mixture of isotropic gaussians as $p(\mathbf{x}_0) \sim \sum_{i=1}^{K} \pi_i \mathcal{N}(\mathbf{x}; \mu_i, \sigma_i^2 I)$, the noisy distribution $p(\mathbf{x}_t)$ also follows a mixture of isotropic gaussians, as below.

$$p(\mathbf{x}_t) \sim \sum_{i=1}^{K} \pi_i \mathcal{N}(\mathbf{x}; \alpha_t \mu_i, (\alpha_t^2 \sigma_i^2 + \sigma_t^2) I)$$

, and $\bar{p}_k(\mathbf{x}_t)$ can be expressed as below.

$$\bar{p}_k(\mathbf{x}_t) \sim \sum_{i=1}^{K} \pi_i \mathcal{N}(\mathbf{x}; \alpha_t \mu_i, \frac{(\alpha_t^2 \sigma_i^2 + \sigma_t^2) I}{k})$$

Assuming $\mathcal{N}(\mathbf{x}; \mu_j, \sigma^2 I)$ is the dominating gaussian at x_t, we obtain:

$$\nabla_{\mathbf{x}_t} \log\ p(\mathbf{x}_t)) \simeq -\frac{\mathbf{x}_t - \alpha_t \mu_j}{\alpha_t^2 \sigma_j^2 + \sigma_t^2}$$

$$\nabla_{\mathbf{x}_t} \log\, \bar{p}_k(\mathbf{x}_t)) \simeq -\frac{\mathbf{x}_t - \alpha_t \mu_j}{\frac{\alpha_t^2}{k}\sigma_j^2 + \sigma_t^2}$$

$$\nabla_{\mathbf{x}_t} \log\, \bar{p}_k(\mathbf{x}_t) = \frac{s_t \sigma_j^2 + 1}{s_t \frac{\sigma_j^2}{k} + 1} \nabla_{\mathbf{x}_t} \log\, p(\mathbf{x}_t) \tag{8}$$

As a result, if a mixture of isotropic gaussians is well-separated, we can derive a score rescaling actor analogous to r_t. If all gaussians have a same variance σ^2, we get same formulation to PSR.

B. More Results and Analysis

2D Toy Data. We experiment on more diverse 2D gaussian mixtures, which manifests the benefit of PSR over temperature scaling on autoregressive model and naive noise scaling. Results are in Fig.11 and Fig.12.

Pose Estimation. We show more pose prediction results in Fig. 13. PSR predicts tighter samples around the ground truth mode, which can be observed by the low spread of sampled poses compared to score sampling. We also include more depth samples and comparisons Fig. 9, and see that increasing k, across different σ, yields consistent improvement in pose accuracy.

Depth Estimation. We also show the effect of σ and k on the AbsRel metric in Fig. 10. Compared with the DDIM sample ($k = 1$), PSR demonstrates consistent performance gain in various (k, σ) configurations. We also include more depth samples and comparisons Fig. 14. A consistent improvement of PSR result can be observed, compared to the DDIM samples.

Image Generation. We show additional qualitative results for image generation in Fig. 15. We show PSR with varying k and highlight the control over the expression of high frequency details. Additionally, we show examples with DDIM and varying CFG. The significant change in image composition across CFG highlights that CFG changes the global sampling distribution as it injects negative or null prompts, leading to large changes in image semantics, which is orthogonal to PSR which steers the sharpness and flatness of the sampling distribution. Additionally, we show the effect of (k, σ) in Fig. 16 on FID and CLIP scores. PSR achieves better scores over various configurations of parameters. Interestingly, while DDPM performs worse than DDIM without any scaling, DDPM with PSR achieves better scores than DDIM with PSR.

Robot Manipulation. Here we explain the experiment details and additional results. We search over the parameters $k \in \{0.7, 0.9, 1.1, 1.3, 1.5\}$ and $\sigma \in \{0.01, 0.025, 0.5, 0.1, 0.25, 0.5, 1.0, 2.5, 5.0, 10.\}$ for PSR. For naive noise scaling, we search over $k' \in \{0.1, 0.5, 0.7, 0.9, 1.1, 1.3, 1.5, 5.0, 10.0\}$. Additionally, we increase the threshold of task success in Push-T from 0.95 to 0.99. The threshold indicates the minimum proportion of T-shape within the target T shape to be success.

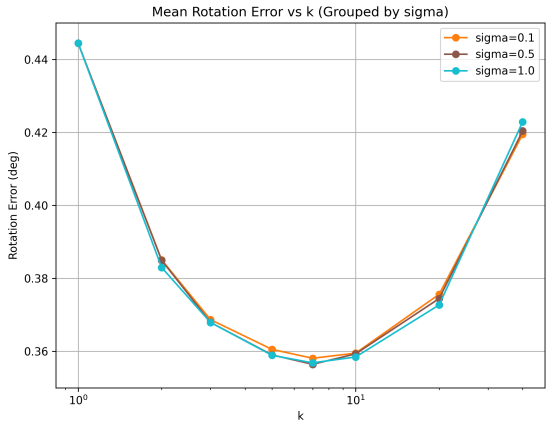


Figure 9. **Effects of (k, σ) on pose estimation.** PSR with various (k, σ) configurations effectively ourperforms the baseline sampling method $k = 1$. While PSR is not sensitive to σ in pose estimation, PSR reaches optimal performance with $k \approx 7$.

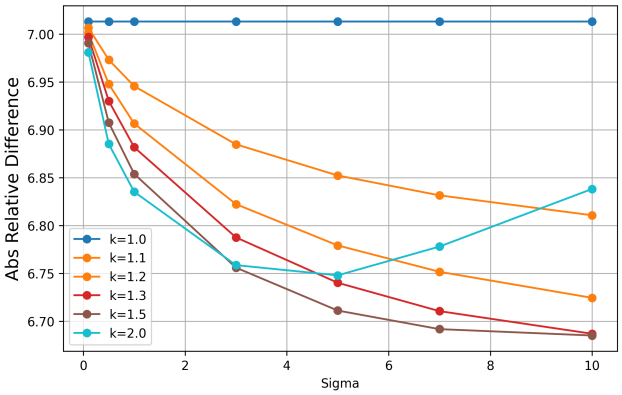


Figure 10. **Effects of (k, σ) on depth estimation.** Comparing with DDIM sample ($k = 1$), PSR demonstrates consistent performance gains in various (k, σ) configurations.

Quantitative results on how PSR performs with varying (k, σ) are in Fig.17 and Fig.18, with optimal (k, σ) noted. Moreover, a qualitative comparison of Diffusion Policy [28] and PSR are in the supplementary material, where we show the rollout videos of each method with the same initial configuration.

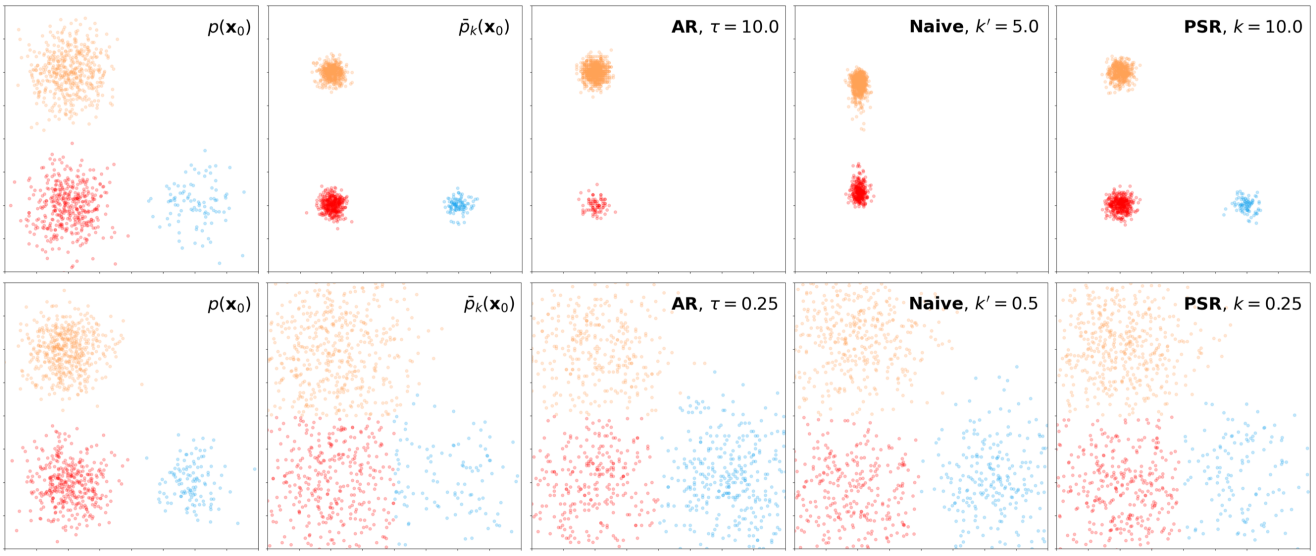


Figure 11. **Comparison on nonuniform mixture of 2D isotropic gaussians. AR** - temperature scaling, **Naive** - naive noise scaling(NSS), **PSR** - ours. The mixture weight of $p(\mathbf{x}_0)$ is 0.4(red), 0.5(orange), and 0.1(blue). When $k = 10.0$ (top), AR and NNS tend to converge to a subset of modes (orange, red). When $k = 0.25$(bottom), the trend becomes the opposite. AR and NNS biases the samples more on the mode it lost at $k = 10.0$(blue), while the desired weight for blue is 0.1. **PSR** preserves weights at both $k = 10.0, 0.25$.

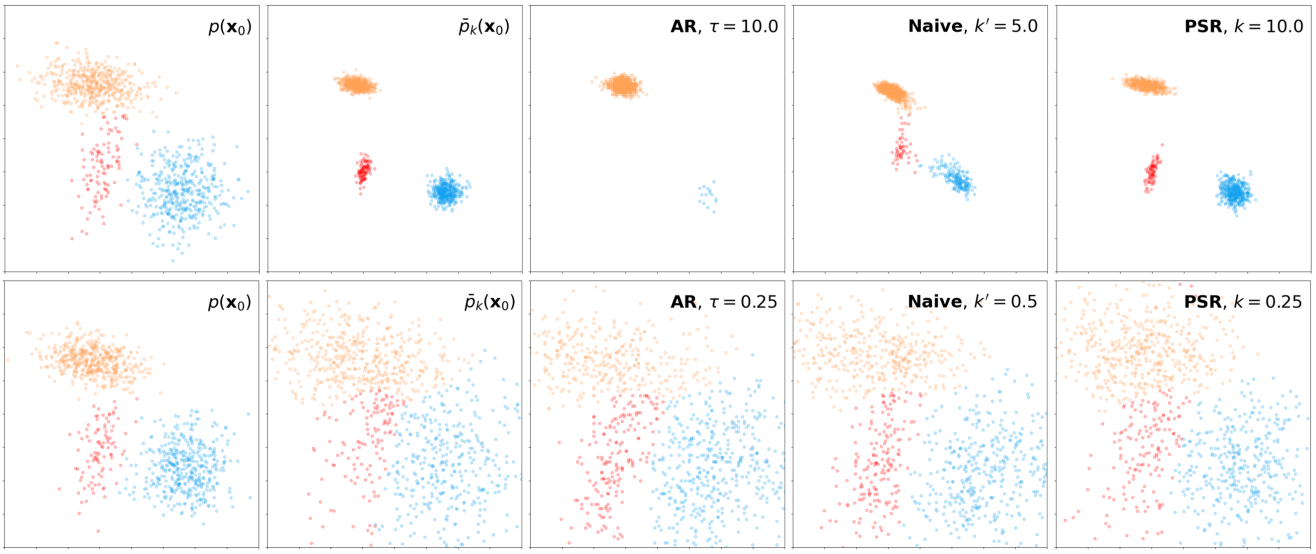


Figure 12. **Comparison on nonuniform mixture of weakly-separated 2D anisotropic gaussians.** Notation is analogous to above. The mixture weight of $p(\mathbf{x}_0)$ is 0.1(red), 0.5(orange), and 0.4(blue). Note that the gaussians are not separated well in the case, making the setup challenging. When $k = 10.0$ (top), AR and NNS tend to converge to a subset of modes. Additinally, NNS tends to generate samples that are close to mean of each modes, which is undesirable. When $k = 0.25$(bottom), AR biases the samples more on the red mode, while the desired weight for red is 0.1. **PSR** preserves weights and generated samples similar to $\bar{p}_k(\mathbf{x})$ at both k. We use $\sigma = \sqrt{0.025}$ for both k in PSR.

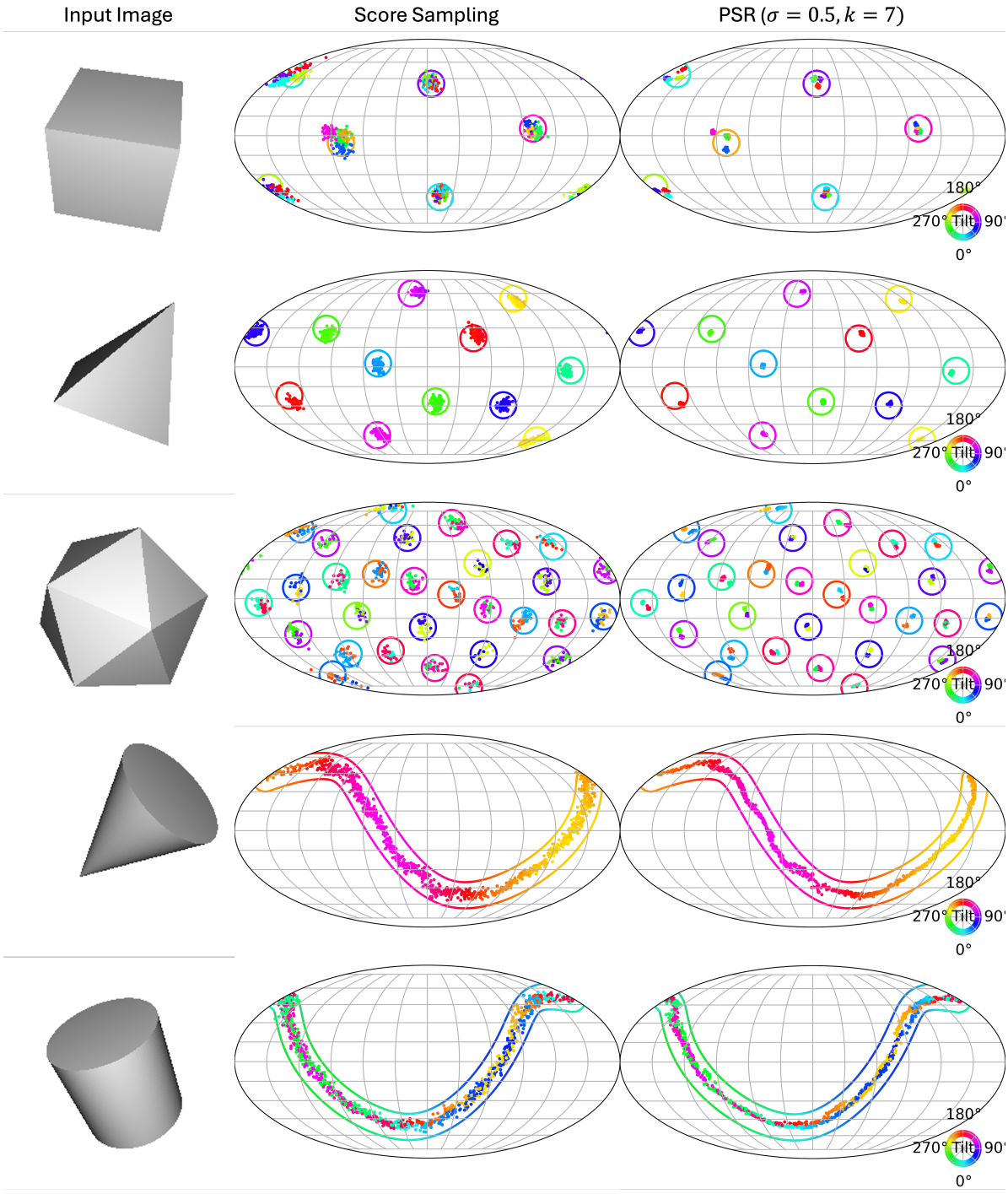


Figure 13. **More predicted poses on SYMSOL**. We show all 5 classes of shapes in SYMSOL. We use $\sigma = 1, k = 7$ for these visualizations. `PSR` consistently reduces prediction error across all classes compared to score sampling. We modify the location of samples to exaggerate error by a factor of 15 to show visual difference given plotting constraints.

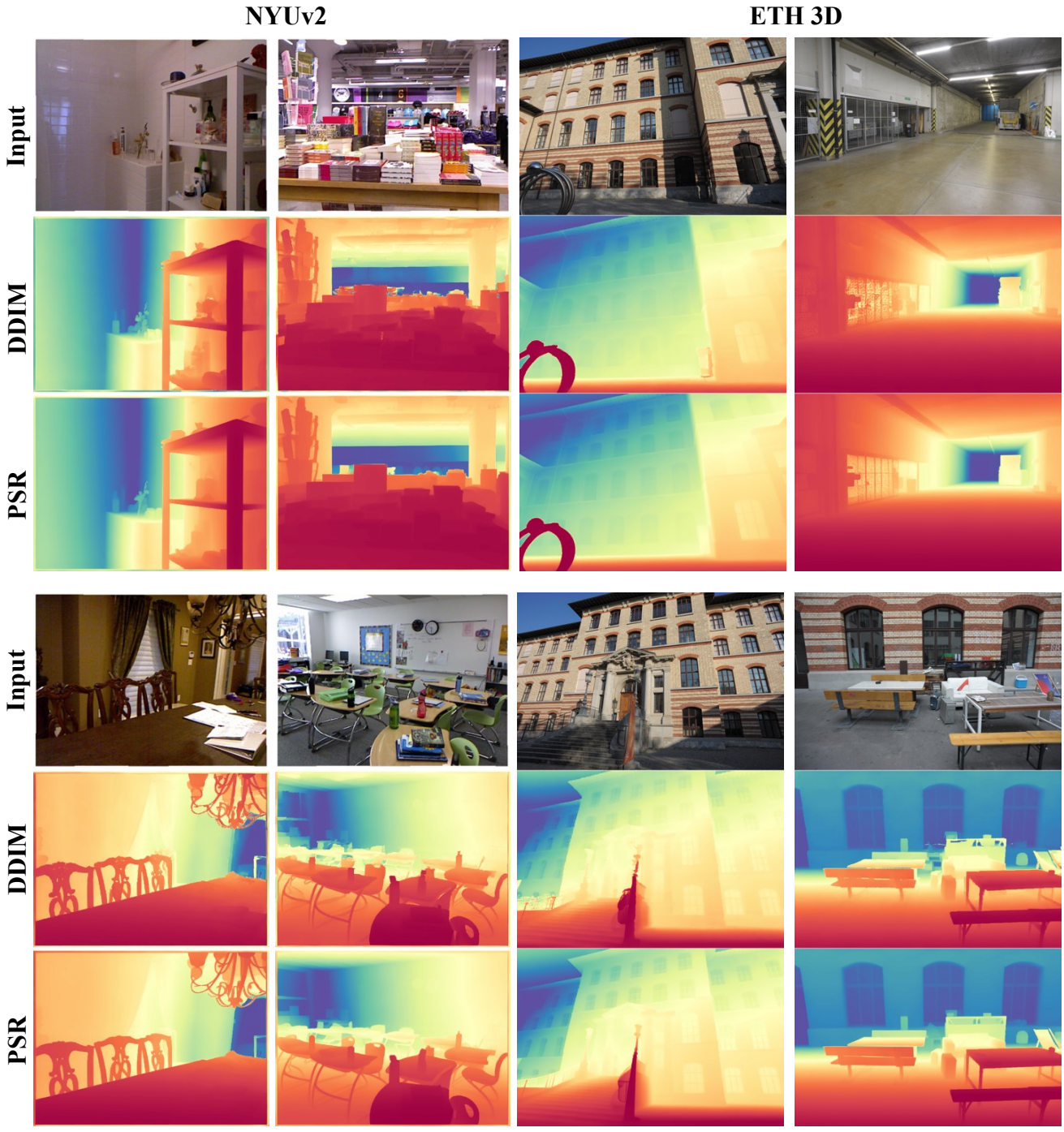


Figure 14. **More Depth Prediction Comparison.** We include more samples from NYUv2 and ETH3D. PSR demonstrates consistent improvement compared to the DDIM samples.

Figure 15. **More Image Generation Results.** We further show the effect of PSR in steering the sampling distribution of Stable Diffusion to be sharper or flatter. A larger k forces a sharper samplings distribution and creates a smoother image while a smaller k allows for more high frequency details. All PSR (k, 1.0) comparisons (TOP) use $\sigma = 1.0$ and CFG=7.5. When $k = 1.0$, the results are the same as DDIM sampling at CFG=7.5. We also show results for DDIM with varying CFG on (BOTTOM) and observe the CFG making trade-offs between text-alignment and image fidelity by changing the underlying distribution.

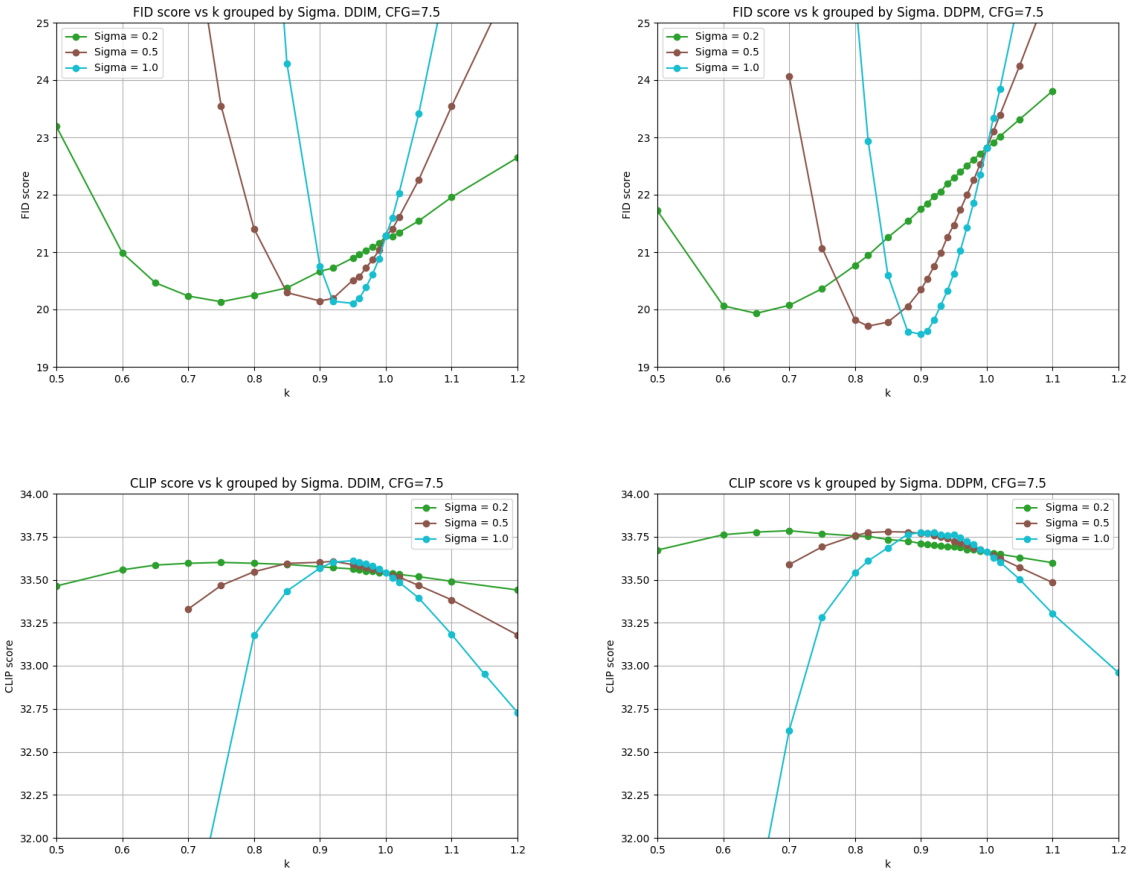
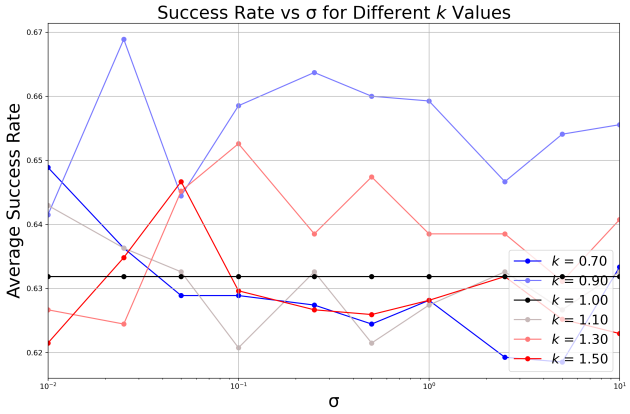
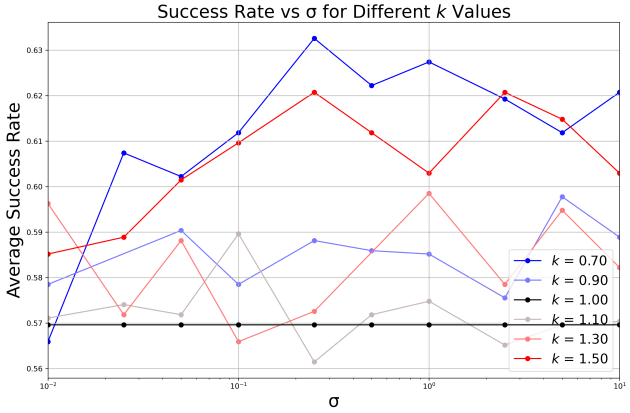


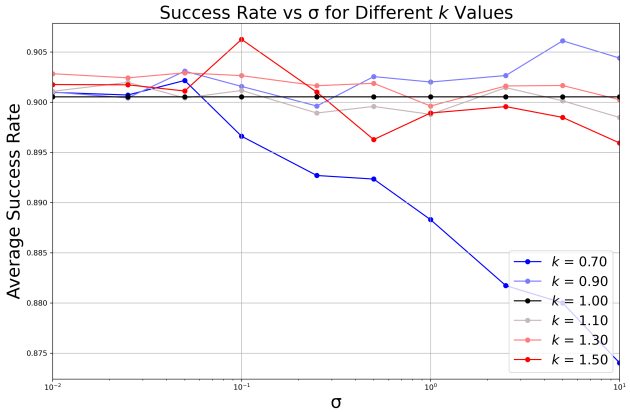
Figure 16. **Effects of (k, σ) on image generation.** The left column compares FID score over different (k, σ), while the right column compares CLIP score. The top row shows results with DDIM +PSR, while the bottom row shows results with DDPM+PSR. Our method consistently achieves better FID and CLIP score over various (k, σ) settings.



(a) Transport - mh. Best $(k, \sigma) = (0.9, 0.025)$.

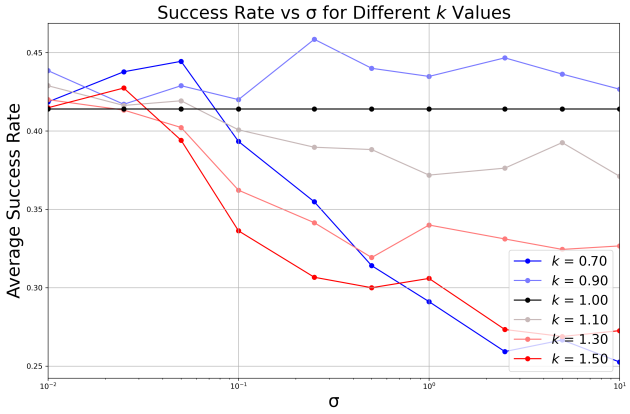


(b) Tool Hang - ph. Best $(k, \sigma) = (0.7, 0.25)$.

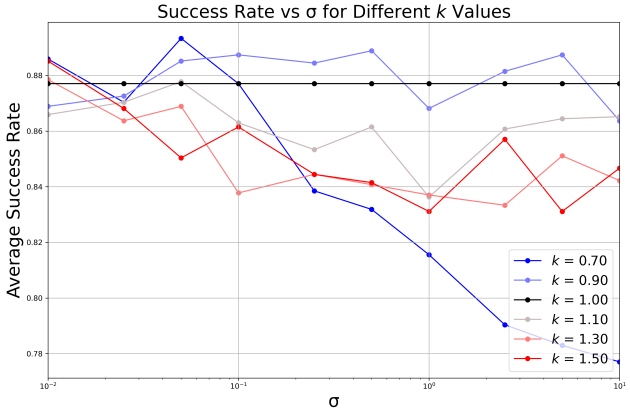


(c) Push-T - ph. Best $(k, \sigma) = (1.5, 0.1)$.

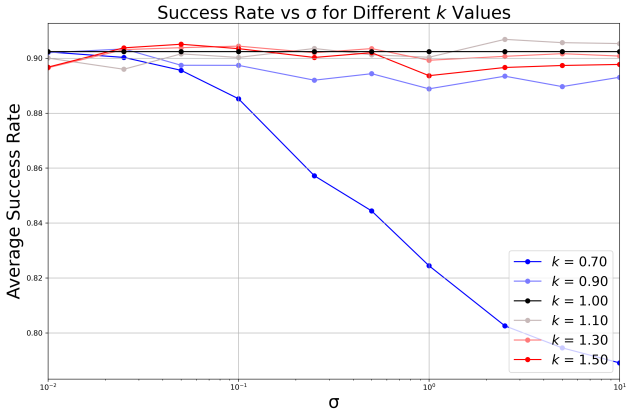
Figure 17. **Success rate for Diffusion Policy-C.** For tasks where Diffusion Policy(k=1.0) performs poorly(Transport, Tool Hang), optimal k is smaller than one, while for tasks where it performs well(Push-T), optimal k is larger than one. For some plots, both $k > 1$ and $k < 1$ outperform Diffusion Policy, as previously discussed.



(a) Transport - mh. Best $(k, \sigma) = (0.9, 0.25)$.



(b) Tool Hang - ph. Best $(k, \sigma) = (0.7, 0.05)$.



(c) Push-T - ph. Best $(k, \sigma) = (1.1, 2.5)$.

Figure 18. **Success rate for Diffusion Policy-T.** Similar as Diffusion Policy-C, $k > 1$ helps for Transport and Tool Hang, while $k > 1$ helps for Push-T. The trend is more clear compared to Diffusion Policy-C, as it is less the case where $k > 1$ and $k < 1$ both improve the performance.