Loss-Guided Diffusion Models for Plug-and-Play Controllable Generation

Jiaming Song [1] **Qinsheng Zhang** [1 2] **Hongxu Yin** [1] **Morteza Mardani** [1] **Ming-Yu Liu** [1] **Jan Kautz** [1]
Yongxin Chen [2] **Arash Vahdat** [1]

Abstract

We consider guiding denoising diffusion models with general differentiable loss functions in a plug-and-play fashion, enabling controllable generation without additional training. This paradigm, termed Loss-Guided Diffusion (LGD), can easily be integrated into all diffusion models and leverage various efficient samplers. Despite the benefits, the resulting guidance term is, unfortunately, an intractable integral and needs to be approximated. Existing methods compute the guidance term based on a point estimate. However, we show that such approaches have significant errors over the scale of the approximations. To address this issue, we propose a Monte Carlo method that uses multiple samples from a suitable distribution to reduce bias. Our method is effective in various synthetic and real-world settings, including image super-resolution, text or label-conditional image generation, and controllable motion synthesis. Notably, we show how our method can be applied to control a pretrained motion diffusion model to follow certain paths and avoid obstacles that are proven challenging to prior methods.

1. Introduction

Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021c) are widely used as foundation models (Bommasani et al., 2021) for generative modeling. They have been successfully applied to various domains, such as images (Rombach et al., 2022; Saharia et al., 2022; Balaji et al., 2022), 3D objects (Poole et al., 2022; Lin et al., 2022; Shue et al., 2022; Bautista et al., 2022), natural language (Li et al., 2022; Lovelace et al., 2022), audio (Kong et al., 2020), time series (Tashiro et al., 2021; Biloš et al., 2022), proteins (Wu et al., 2022; Qiao et al., 2022), and molecules (Xu et al., 2022b). In many domains (such as text-to-image generation), diffusion models can scale to large sets of paired data (on the order of billions, *e.g.*, Schuhmann et al. (2022)) and effectively perform conditional generation with techniques such as classifier(-free) guidance (Dhariwal & Nichol, 2021; Ho & Salimans, 2022).

To reduce the amount of training, one could also use unconditional diffusion models to perform certain conditional tasks, such as super-resolution with reconstruction guidance (Ho et al., 2022; Chung et al., 2022). While there are various plug-and-play (Venkatakrishnan et al., 2013) approaches that do not require additional training over noisy paired data, they are often limited to certain loss functions, such as the linear least squares loss (Anonymous, 2022).

In this paper, we consider the more general case where we only need the loss function to be differentiable. Our goal is to guide diffusion models with such loss functions in a plug-and-play fashion; we term this paradigm *Loss-Guided Diffusion* (LGD). This sampling-based approach can leverage efficient diffusion model sampling algorithms (Song et al., 2021a; Liu et al., 2022; Zhang & Chen, 2022; Lu et al., 2022). Unfortunately, the guidance term in LGD involves an expectation over an intractable probability distribution. Its success depends on whether we can efficiently approximate the term.

Existing methods, such as Diffusion Posterior Sampling (DPS, Chung et al. (2022)), approximate the expectation by replacing the ideal yet intractable distribution with a delta distribution centered at the denoising output from the diffusion model (Sec. 4.1). However, we show in a simple example that such approaches can significantly miscalculate the scale of the guidance term, which can be too large at high noise levels and too small at low noise levels (Sec. 4.2). This approximation bias can be attributed to the fact that the delta distribution is too far from the ideal one.

To this end, we propose LGD-MC, a Monte Carlo integration approach to reduce approximation errors. First, we replace the delta distribution with a smoother distribution. In Proposition 4.1, we explain why this could reduce bias in the asymptotic case. Then, we approximate the expectation by aggregating losses evaluated at multiple Monte Carlo samples (Sec. 4.3). Since most of the computing is spent on

*Equal contribution [1]NVIDIA Corporation [2]Georgia Institute of Technology. Correspondence to: Jiaming Song <jiamings@nvidia.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

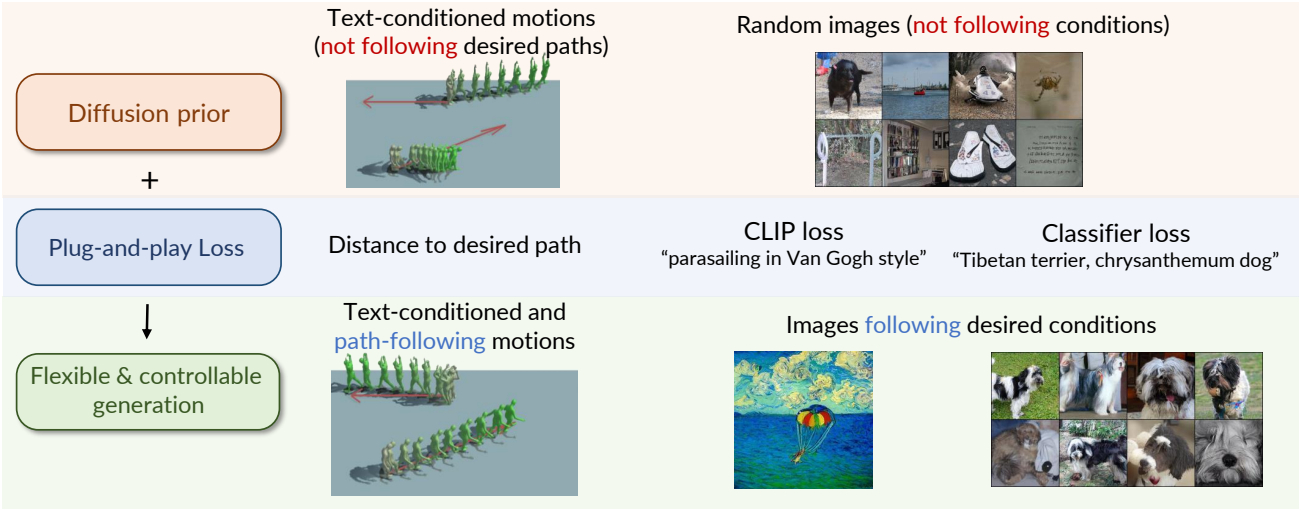

Figure 1. The idea behind **Loss-Guided Diffusion (LGD)**, a paradigm for plug-and-play controllable generation using diffusion models. Given a diffusion model serving as a prior (shown in the first row), we perform controllable generation, where the conditions are specified via a loss function independent of the diffusion model (shown in the second row). In addition, we wish to obtain loss guidance terms that allow us to leverage existing developments on accelerated diffusion sampling methods (Song et al., 2021a; Zhang & Chen, 2022; Lu et al., 2022), unlike in Graikos et al. (2022), which finds point estimates via stochastic optimization.

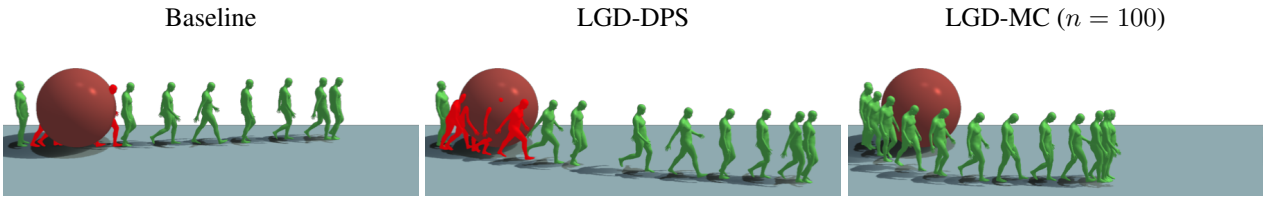

Figure 2. Samples for prompt ("walking") for obstacle avoidance in controllable motion synthesis (Sec. 6.3). The red poses indicate collision between the avatar and the ball obstacle while green ones indicate collision-free motions. LGD-MC (ours) can help synthesize collision-free motion trajectories, even when underlying motion diffusion models are trained in an obstacle-free environment.

backpropagating through the diffusion model, the additional samples introduce little computational overhead, and we can easily use hundreds of samples if the loss function is easy to evaluate.

We empirically demonstrate that LGD-MC is significantly more accurate than LGD with the point estimate and produce much better samples. We further show the effectiveness of LGD-MC over various real-world problems, such as image super-resolution, conditional image generation, and controllable motion synthesis. Notably, LGD-MC produces location-specific motions that follow a certain path or avoid obstacles, using a generic text-to-motion diffusion model within 100 timesteps.

In summary, our contributions are as follows.

- We study the problem of *Loss-Guided Diffusion* (LGD), which is a plug-and-play paradigm for efficient conditional generation using diffusion models.
- We illustrate the limitations of the existing solution that is based on point estimates using a concrete example.
- We propose a solution based on Monte-Carlo estimates and demonstrate its effectiveness on multiple domains.

2. Background

2.1. Diffusion Models

For a data distribution $p_0(\mathbf{x}_0)$ where $\mathbf{x}_0 \in \mathcal{X}$, a family of distributions $p_t(\mathbf{x}_t)$ can be defined by injecting *i.i.d.* Gaussian noise to data samples, such that $\mathbf{x}_t = \mathbf{x}_0 + \sigma_t \epsilon$ with $\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$ and σ_t monotonically increasing with respect to time $t \in [0, T]$. The score function $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ (*i.e.*, gradient of log-probability) can be learned via a denoising score matching objective (Vincent, 2011):

$$L_t(\theta) = \mathbb{E}_{\mathbf{x}_0, \mathbf{x}_t}[\|D_\theta(\mathbf{x}_t, t) - \mathbf{x}_0\|_2^2], \tag{1}$$

where $D_\theta : \mathcal{X} \times [0, T] \to \mathcal{X}$ is a time-conditioned neural network that tries to denoise the noisy sample $\mathbf{x}_t$. Assuming an infinite capacity of D_θ, the predictions of the optimal model are related to the score function via Tweedie's for-

mula (Efron, 2011):

$$\hat{\mathbf{x}}_t := D_\theta(\mathbf{x}_t, t) = \mathbf{x}_t + \sigma_t^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t), \quad (2)$$

which represents the minimum mean squared error (MMSE) estimator of $\mathbf{x}_0$ given $\mathbf{x}_t$ and σ_t.

Score-based diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) represent a class of models that solve a stochastic differential equation (Song et al., 2021c) using the neural network D_θ; see App. A.1 for more details.

Recently, there is a plethora of sampling methods developed for diffusion models, such as ones based on first-order ODE/SDE solvers (Song et al., 2021a), higher-order ODE solvers (Zhang & Chen, 2022; Karras et al., 2022; Lu et al., 2022; Dockhorn et al., 2022), or knowledge distillation models (Salimans & Ho, 2022; Meng et al., 2022; Xiao et al., 2022; Zheng et al., 2022). In this paper, we directly apply fast first-order ODE / SDE solvers, such as DDIM.

2.2. Guidance Methods for Conditional Generation

In many cases, we may wish to draw samples $\mathbf{x}_0$ subject to certain conditions $\mathbf{y}$. For diffusion models, the conditional score at time t can be obtained via Bayes' rule:

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y}) = \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t).$$

Classifier guidance (Dhariwal & Nichol, 2021) obtains the second term by training on paired $(\mathbf{x}_0, \mathbf{y})$ data. Diffusion Posterior Sampling (DPS, (Chung et al., 2022)) does not rely on paired training data and instead assumes that the conditional distribution on noiseless data, *i.e.*, a normalized probability distribution $p(\mathbf{y}|\mathbf{x}_0)$, is given. DPS approximates $p_t(\mathbf{y}|\mathbf{x}_t)$ as follows:

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \approx \text{DPS}(\mathbf{x}_t, \mathbf{y}) := \nabla_{\mathbf{x}_t} \log p(\mathbf{y}|\hat{\mathbf{x}}_t), \quad (3)$$

where $\hat{\mathbf{x}}_t$ is the MMSE estimator defined in Eq. 2. Essentially, DPS uses a point estimate from the diffusion model $\hat{\mathbf{x}}_t$ to replace $\mathbf{x}_0$; if $p(\mathbf{y}|\mathbf{x}_0)$ is differentiable *w.r.t.* the condition $\mathbf{x}_0$, so is the DPS approximation *w.r.t.* $\mathbf{x}_t$. Reconstruction guidance (Ho et al., 2022) is a special case of DPS where $p(\mathbf{y}|\mathbf{x}_0)$ is Gaussian. While DPS is effective on various non-linear inverse problems, its hyperparameters are tuned for every task and specific to 1000 timesteps, and they do not generalize well to other tasks or fewer timesteps (Chung et al. (2022), Appendix C and D).

3. Loss-Guided Diffusion

In this paper, we develop plug-and-play sampling methods for diffusion models. Compared to existing methods, we make weaker assumptions over the conditional probability distribution $p(\mathbf{y}|\mathbf{x}_0)$. This allows us to consider a wider range of problems for controllable generations.

For example, suppose that we have a motion diffusion model (Tevet et al., 2022) that is able to synthesize random text-conditioned human motions. In some cases, we also wish to specify certain target locations and some obstacles to avoid, which can be encoded in a general loss function. Here, a plug-and-play approach is more suitable for these types of flexible conditional generation tasks.

Target. We consider the plug-and-play conditional generation setting where we have a pre-trained diffusion model D_θ and a differentiable loss function $\ell_{\mathbf{y}} : \mathcal{X} \to \mathbb{R}$ with possible dependence on some conditions $\mathbf{y}$. Our goal is to sample from the following distribution:

$$p_0^{(\ell)}(\mathbf{x}_0|\mathbf{y}) = p_0(\mathbf{x}_0) \exp(-\ell_{\mathbf{y}}(\mathbf{x}_0))/Z, \quad (4)$$

where $Z = \int_{\mathbf{x}_0} p_0(\mathbf{x}_0) \exp(-\ell_{\mathbf{y}}(\mathbf{x}_0)) \mathrm{d}\mathbf{x}_0$ is a normalizing constant[1]. This can be treated as a posterior distribution, where the prior term is $p_0(\mathbf{x}_0)$ and the likelihood term is proportional to $\exp(-\ell_{\mathbf{y}}(\mathbf{x}_0))$. In the motion example, $p_0^{(\ell)}(\mathbf{x}_0|\mathbf{y})$ will not only draw samples that are reasonable motion sequences but also ones that are close to the desired path (as specified by the loss function).

Problem definition. To better leverage existing efficient samplers for diffusion models, we wish to derive guidance terms for different noise levels $t \in (0, T]$ given the loss function $\ell_{\mathbf{y}}(\mathbf{x}_0)$, which is defined only for clean data $\mathbf{x}_0$. Let us define $\mathbf{x}_t \sim p_t^{(\ell)}(\mathbf{x}_t|\mathbf{y})$ via the following process:

$$\mathbf{x}_t = \mathbf{x}_0 + \sigma_t \epsilon, \quad \mathbf{x}_0 \sim p_0^{(\ell)}(\mathbf{x}_0|\mathbf{y}), \quad \epsilon \sim \mathcal{N}(0, \boldsymbol{I}). \quad (5)$$

To draw samples from Eq. 4 using a diffusion model, the desired guidance term (which we call the "loss guidance term") for time t would simply be

$$\nabla_{\mathbf{x}_t} \log p_t^{(\ell)}(\mathbf{y}|\mathbf{x}_t) = \nabla_{\mathbf{x}_t} \log \left(p_t^{(\ell)}(\mathbf{x}_t|\mathbf{y}) / p_t^{(\ell)}(\mathbf{x}_t) \right). \quad (6)$$

This term is tremendously useful for sampling from the desired conditional distribution $p_0^{(\ell)}(\mathbf{x}_0|\mathbf{y})$, yet it is not immediately available to us. This motivates us to define the following estimation problem.

Definition 3.1 (The problem of Loss-Guided Diffusion.)**.** Given a diffusion model D_θ and a loss function $\ell_{\mathbf{y}}(\mathbf{x}_0)$ defined only for noiseless data $\mathbf{x}_0$, estimate the *loss guidance term* $\nabla_{\mathbf{x}_t} \log p_t^{(\ell)}(\mathbf{y}|\mathbf{x}_t)$ (Eq. 6), for all $t \in (0, T]$. Ideally, this should be done with a single query over D_θ.

With an estimated loss guidance term, we can perform conditional sampling with loss functions using existing diffusion model samplers. Of course, the quality of the estimation will affect the quality of the samples.

[1] We assume Z exists; if not, we can add $\mathbf{x}_0^2$ to the loss function to make the resulting distribution sub-Gaussian.

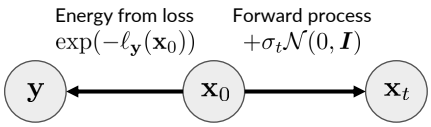

Figure 3. The graphical model over $\mathbf{x}_0$, $\mathbf{x}_t$, $\mathbf{y}$ is a Markov chain.

Challenges of LGD. Unfortunately, LGD is difficult since $p_t^{(\ell)}(\mathbf{y}|\mathbf{x}_t)$ is intractable in the most general case (Anonymous, 2022). To see why this is the case, we consider the graphical model for the variables $\mathbf{x}_0$, $\mathbf{x}_t$, and $\mathbf{y}$ in Fig. 3. Conditioned on $\mathbf{x}_0$, $\mathbf{y}$ and $\mathbf{x}_t$ are independent, so:

$$p_t^{(\ell)}(\mathbf{y}|\mathbf{x}_t) = \int_{\mathbf{x}_0} p(\mathbf{x}_0|\mathbf{x}_t) p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0) \mathrm{d}\mathbf{x}_0, \tag{7}$$

where the term $p(\mathbf{x}_0|\mathbf{x}_t)$ requires multiple iterations to approximate accurately with a diffusion model (for sampling or evaluating the likelihood of a single sample). The integral in Eq. 7 would thus be intractable if our goal is to use only one diffusion model evaluation for the guidance term, and thus one has to introduce further approximations.

4. Practical Algorithms

In this section, we discuss practical approximate algorithms for the LGD problem. In Sec. 4.1 and 4.2, we adapt an existing efficient method to LGD and discuss its limitations. In Sec. 4.3, we introduce a novel solution to address the limitation while keeping the efficiency.

4.1. Existing Approximations with DPS

First, we show that while DPS (Chung et al., 2022) assumes a normalized conditional probability $p(\mathbf{y}|\mathbf{x}_0)$ in DPS (*i.e.*, Gaussian and Poisson distributions), the same approximations can be used for unnormalized conditional probabilities $\exp(-\ell_{\mathbf{y}}(\mathbf{x}_0))$ as well. Since the normalizing constant Z does not depend on $\mathbf{x}_t$, we have the following:

$$\mathrm{DPS}(\mathbf{x}_t, \mathbf{y}) := \nabla_{\mathbf{x}_t} \log \frac{\exp(-\ell_{\mathbf{y}}(\hat{\mathbf{x}}_t))}{Z} = -\nabla_{\mathbf{x}_t} \ell_{\mathbf{y}}(\hat{\mathbf{x}}_t), \tag{8}$$

which is the gradient *w.r.t.* the loss evaluated at the MMSE estimate $\hat{\mathbf{x}}_t$ (Eq. 2).

4.2. DPS Miscalculates the Scale of Loss Guidance

Despite the simplicity of DPS, it can be a poor approximation to the loss guidance term needed for LGD, as we illustrate in the following example. In Fig. 4 (top left), we consider $p(\mathbf{x}_0)$ to be a mixture of two Gaussian distributions on one dimension, where we use $\mathbf{y}$ to denote the mixture component ($\mathbf{y} \in \{0, 1\}$), such that $p(\mathbf{x}_0|\mathbf{y} = 0) = \mathcal{N}(-1, 0.2^2)$ and $p(\mathbf{x}_0|\mathbf{y} = 1) = \mathcal{N}(1, 0.2^2)$. Both mixture components

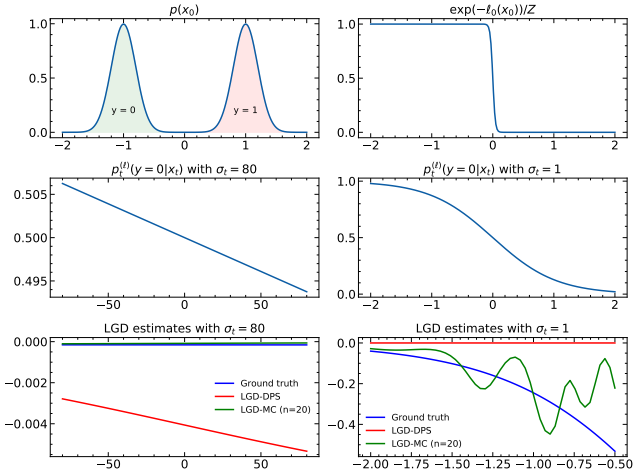

Figure 4. (*Top*) Left: the probability density function (PDF) of $p_0(\mathbf{x}_0)$, a mixture of Gaussian. Right: the exponential of the negative loss. (*Mid*) The probability of $\mathbf{y} = 0$ given $\mathbf{x}_t$ with σ_t being 1 or 80, respectively. (*Bottom*) For the LGD problem, the ground truth score, the score estimated by DPS, and the score estimated by Monte Carlo, with σ_t being 1 or 80, respectively. The Monte Carlo estimates become more accurate as n increases, *c.f.*, Fig. 8 in App. A.4.

have equal weights, so $p(\mathbf{y} = 0) = p(\mathbf{y} = 1) = 0.5$. Since $p_t(\mathbf{x}_t)$ are also mixtures of Gaussians, it becomes tractable to compute the value of loss guidance term, *i.e.*, $\nabla_{\mathbf{x}_t} \log p_t^{(\ell)}(\mathbf{y}|\mathbf{x}_t)$ for any value $t \in [0, T]$; examples for $\sigma_t = 1$ and $\sigma_t = 80$ are shown in Fig. 4.

From Fig. 4 (bottom), we see that approximations made with DPS (red) are quite different from the true guidance terms (blue). DPS approximations are too large when σ_t is large, but too small when σ_t is small. We explain this phenomenon as follows. Since the two Gaussian mixtures in $p_0(\mathbf{x}_0)$ are quite distinct, there is a clear decision boundary at around $\mathbf{x}_0 = 0$; the gradient of $\log p^{(\ell)}(\mathbf{y} = 0|\mathbf{x}_0)$ is large when $\mathbf{x}_0 \approx 0$ and become close to zero if $\mathbf{x}_0$ is far from the decision boundary. In DPS, one only uses the gradient of $\log p^{(\ell)}(\mathbf{y} = 0|\hat{\mathbf{x}}_t)$ where $\hat{\mathbf{x}}_t$ is the MMSE estimate (Eq. 2); the MMSE estimate is almost zero when σ_t is large, and far from zero if σ_t is small. Thus, DPS approximations to the loss guidance term in Eq. 7 are expected to be large when σ_t is large, and small when σ_t is small.

We further note that Chung et al. (2022) address this issue via a heuristic that divides the DPS estimates by the negative log-likelihood (NLL) of $p^{\ell}(\mathbf{y}|\mathbf{x}_0)$ times a constant (Chung et al. (2022), Appendix D.1). The NLL is large when σ_t is large (as the MMSE prediction is less accurate) and small when σ_t is small. Thus, this heuristic aligns with our findings about the relationship between the true loss guidance and DPS estimates.

4.3. Improving the Estimation with Monte Carlo

One could also interpret DPS as computing the score of the integral in Eq. 7 but with $p(\mathbf{x}_0|\mathbf{x}_t)$ replaced by a delta distribution around $\hat{\mathbf{x}}_t$. As the approximation error can largely be attributed to this point estimate, we consider an alternative Monte-Carlo approach that uses multiple samples to estimate the loss guidance term. Specifically, our approximation with n samples is as follows:

$$\mathrm{MC}_n(\mathbf{x}_t, \mathbf{y}) = \nabla_{\mathbf{x}_t} \log \left(\frac{1}{n} \sum_{i=1}^{n} \exp(-\ell_{\mathbf{y}}(\mathbf{x}^{(i)})) \right), \quad (9)$$

where $\mathbf{x}^{(i)} \sim q(\mathbf{x}_0|\mathbf{x}_t)$ are *i.i.d.* samples from an approximate inference distribution $q(\mathbf{x}_0|\mathbf{x}_t)$. Ideally, we want $q(\mathbf{x}_0|\mathbf{x}_t)$ to be as close to $p(\mathbf{x}_0|\mathbf{x}_t)$ as possible. In practice, to save compute, we consider $q(\mathbf{x}_0|\mathbf{x}_t) = \mathcal{N}(\hat{\mathbf{x}}_t, r_t^2 \boldsymbol{I})$, which is a Gaussian with mean being the MMSE estimate $\hat{\mathbf{x}}_t$ and covariance depending on a hyperparameter r_t. While this is by no means a highly accurate estimate, it is a viable option given that we can only afford a single evaluation of the diffusion model.

Motivations for our choice. We can motivate our choice of $q(\mathbf{x}_0|\mathbf{x}_t)$ via a variational perspective, where we can try to maximize the likelihood of $\mathbf{x}_0$ under the model $q(\mathbf{x}_0|\mathbf{x}_t)$:

$$\max_q \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}[\log q(\mathbf{x}_0|\mathbf{x}_t))]. \quad (10)$$

If we set $q(\mathbf{x}_0|\mathbf{x}_t) = \mathcal{N}(\mu(\mathbf{x}_t), r_t^2 \boldsymbol{I})$ as a Gaussian with a fixed covariance, Eq. 10 is equivalent to:

$$\min_\mu \mathbb{E}_{p(\mathbf{x}_0, \mathbf{x}_t)}[\|\mu(\mathbf{x}_t) - \mathbf{x}_0\|_2^2], \quad (11)$$

where the argmin for μ would satisfy $\mu(\mathbf{x}_t) = \hat{\mathbf{x}}_t$.

We illustrate the effectiveness of our Monte-Carlo approach (denoted as LGD-MC) using the earlier mixture of Gaussian example; we consider $r_t = \sigma_t / \sqrt{1 + \sigma_t^2}$, similar to (Ho et al., 2022; Anonymous, 2022). In Fig. 4 (bottom), we see that while the LGD-MC estimates (green) are not perfect due to the inaccurate $q(\mathbf{x}_0|\mathbf{x}_t)$, they are still much closer to the ground truth (blue) than DPS estimates (red) in terms of magnitude and the general trend. In Fig. 8 of App. A.4, we show that the LGD-MC approximation indeed becomes more accurate as n increases.

Computational efficiency of LGD-MC. While it may seem that taking multiple samples would significantly slow down the algorithm, it is not the case if the evaluation and differentiation of the loss function $\ell_{\mathbf{y}}(\mathbf{x}_0)$ is relatively cheap. To demonstrate this point, we show the computational graphs for the estimations used in DPS and MC in Fig. 5. In DPS, the gradient calculation is straightforward which backpropagates from the output node to the MMSE estimate and then to the input node $\mathbf{x}_t$. In LGD-MC, the backprogagation is largely similar, except that we now have multiple samples contributing to the output node. Nevertheless, we only need to backpropagate through the diffusion model once, which is often much more expensive to compute than the loss functions. In App. A.5, we show concrete empirical results to support our claims.

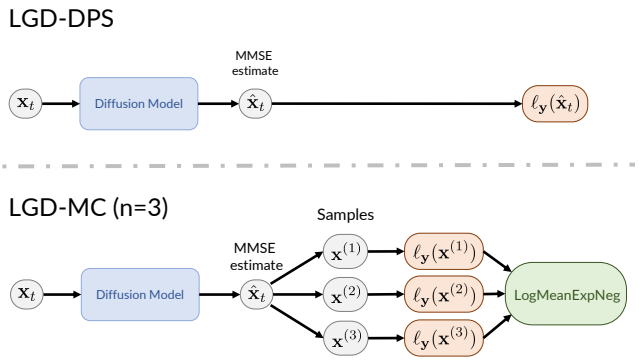

Figure 5. The computational graph for obtaining the DPS (*top*) and the MC (*bottom*) estimates, respectively. For MC, LogMeanExpNeg denotes the log-mean-exp-negative operation in Eq. 9. Note that the diffusion model is only called once in LGD-MC.

Bias correction. Since we may use a large number of samples, we are more concerned about bias, *i.e.*, the approximation error with our proposed $q(\mathbf{x}_0|\mathbf{x}_t)$ with infinitely many samples. In the following statement, we show that it is beneficial to have the proposed $q(\mathbf{x}_0|\mathbf{x}_t)$ closer to the (desired yet intractable) $p(\mathbf{x}_0|\mathbf{x}_t)$.

Proposition 4.1. *Assume that $p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0)$ is bounded for all $\mathbf{x}_0$ and $\mathbf{y}$, and given $t \in (0, T]$, $p(\mathbf{x}_0|\mathbf{x}_t)$ and $q(\mathbf{x}_0|\mathbf{x}_t)$ are supported on $\mathcal{X}$. Then $\forall \mathbf{x}_t \in \mathcal{X}$, there exists a constant c such that:*

$$\left| \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}[p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0)] - \mathbb{E}_{q(\mathbf{x}_0|\mathbf{x}_t)}[p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0)] \right| \quad (12)$$

$$\leq c \cdot TV(p(\mathbf{x}_0|\mathbf{x}_t), q(\mathbf{x}_0|\mathbf{x}_t)), \quad (13)$$

where $TV(p, q)$ is the total variation between p and q.

We present a proof in App. A.3. Compared to the $q(\mathbf{x}_0|\mathbf{x}_t)$ chosen by DPS, which is essentially a delta distribution, our choice would have a smaller total variation distance. Thus, our approximation could also have a lower bias, as suggested in Figs. 4 and 8.

5. Related Works

Inverse problem solvers with diffusion models. Diffusion models have been widely used to solve inverse problems in a plug-and-play manner, and are applied to natural

images (Kawar et al., 2021; Choi et al., 2021; Kawar et al., 2022a), medical images (Jalal et al., 2021; Chung & Ye, 2022; Song et al., 2021b), speech (Sawata et al., 2022), audio (Saito et al., 2022), and astronomy (Karchev et al., 2022). However, they are mainly focused on *linear* inverse problems with least-squares loss, and thus not directly applicable to nonlinear inverse tasks with generic loss functions as in LGD. DDRM (Kawar et al., 2022b) is also applied to certain non-differentiable but semi-linear inverse problems (*e.g.*, JPEG restoration), but it needs a suitable "pseudoinverse" operator.

Therefore, DPS (Chung et al., 2022; Ho et al., 2022) is currently one of the most viable solutions for solving the LGD problem for general loss functions. Our LGD-MC approach further improves upon DPS by considering multiple Monte-Carlo samples to estimate the expectation in Eq. 7, resulting in a more accurate approximation.

Comparing with existing paradigms. We compare LGD with classifier guidance (CG, Dhariwal & Nichol (2021)) and diffusion models as plug-and-play priors (D-PnP, Graikos et al. (2022)) in Tab. 5, App. A.2.

- **Plug-and-play** CG requires paired data $(\mathbf{x}_0, \mathbf{y})$ to be available during training, whereas for D-PnP and LGD the condition $\mathbf{y}$ only needs to be known at test time.
- **Solution** The stochastic optimization in D-PnP converges at a local minimum and provides a point estimate, whereas CG and LGD are sampling algorithms based on the reverse diffusion process.
- **Efficiency** D-PnP may take many iterations (*e.g.*, DreamFusion (Poole et al., 2022) uses 15,000 iterations for each result) whereas CG and LGD are much faster since they can inherit the developments of accelerated diffusion model sampling (*e.g.*, 10 to 1000 iterations), such as DDIM (Song et al., 2021a).
- **Domain** CG and LGD are restricted to the domain of the diffusion models, whereas D-PnP can be used on other domains (such as 3D from 2D diffusion models).

Therefore, we believe that LGD is a promising paradigm for conditional generation as it is more efficient than D-PnP and more flexible than CG.

6. Experiments

We evaluate LGD-MC on various synthetic and real-world experiments, such as a one-dimensional mixture of Gaussian (Sec. 6.1), image super-resolution (Sec. 6.2), controllable motion synthesis (Sec. 6.3). We also compare LGD-MC against CG and D-PnP on the same conditional image generation task (Sec. 6.4), where LGD-MC achieves superior results compared with D-PnP. We include additional experimental details in App. B.

Table 1. Image super-resolution experiments from 64×64 to 256×256, where the images are downsampled by a bicubic filter. The top two rows are inverse problem solvers not applicable to our general case. ($\star$) Degraded performance for DPS at 100 steps is documented in Figure 11, page 20 of Chung et al. (2022).

Method	FID ↓	ResNet50 Accuracy ↑	# Steps
DDRM	21.3	63.2%	100
ΠGDM	3.6	72.1%	100
DPS*	74.11	25.84%	100
LGD-MC ($n=1$)	5.42	71.98%	100
LGD-MC ($n=10$)	5.32	72.07%	100
LGD-MC ($n=100$)	5.21	72.03%	100

6.1. Mixture of Gaussian

Here, we use the mixture of Gaussian example in Sec. 4.2, where our goal is to sample from one of the Gaussians using the diffusion model for the mixture of two Gaussians and the logistic loss function. Three types of guidance terms are considered: Oracle, which uses the ground truth score $\nabla_{\mathbf{x}_t} \log p_t^{(\ell)}(\mathbf{y}|\mathbf{x}_t)$; LGD-DPS, which uses the DPS algorithm in Eq. 8 to guide the sampling; and LGD-MC, which uses the Monte Carlo approach in Eq. 9.

With the same underlying DDIM sampler, LGD-MC has much better histograms and KL divergence than LGD-DPS, as shown in Fig. 6. This suggests that our LGD-MC approximation is more accurate.

6.2. Image Super-resolution

Here, we evaluate LGD-MC over bicubic super-resolution, a linear inverse problem. The linear least squares loss is defined as $\ell_{\mathbf{y}}(\mathbf{x}_0) = \lambda\|\mathbf{y} - \boldsymbol{H}\mathbf{x}_0\|_2^2$ where $\boldsymbol{H}$ is bicubic downsampling convolution and λ is a hyperparameter. In this case, we do not have to resort to sampling-based methods as closed-form solutions to the integral in Eq. 7 exist (*i.e.*, ΠGDM, Anonymous (2022)). Nevertheless, we may use super-resolution as a sanity check to gauge the effectiveness of LGD-MC in practice. We show the results of several plug-and-play methods using diffusion models in Tab. 1, where the number of timesteps is 100. While the performance of LGD-MC is slightly worse than the state-of-the-art ΠGDM (which uses stronger assumptions) in terms of FID, it is comparable in terms of classifier accuracy and is significantly better than DDRM (Kawar et al., 2022a) and DPS (Chung et al., 2022).

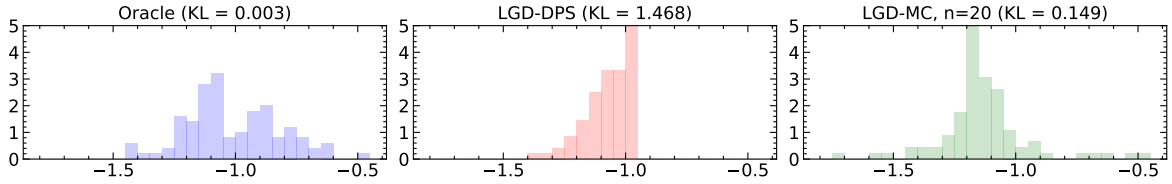

Figure 6. Histogram of samples from the three methods using the oracle, DPS, and LGD-MC, respectively, on the example in Fig. 4. Our LGD-MC obtains a 10× smaller KL value compared to LGD-DPS.

Table 2. Results on controllable motion synthesis for path following.

Prompts (abbr.)	(*i*) "backwards"		(*ii*) "balance beam"		(*iii*) "jogging"		(*iv*) "object above head"	
Metrics (↓)	*Objective*	*Embedding*	*Objective*	*Embedding*	*Objective*	*Embedding*	*Objective*	*Embedding*
Baseline	11.583	6.405	1.024	**3.756**	5.831	5.017	7.080	3.397
LGD-DPS	2.168	5.891	**0.388**	3.784	3.521	**5.090**	2.954	3.545
LGD-MC ($n = 10$)	2.150	5.887	0.395	3.778	3.505	5.093	2.857	**3.515**
LGD-MC ($n = 100$)	**2.126**	**5.886**	0.402	3.797	**3.458**	5.100	**2.800**	3.531

Table 3. Results on obstacle avoidance.

Prompts (abbr.)	(*v*) "walking"		(*vi*) "backwards"	
Metrics (↓)	*Obj.*	*Emb.*	*Obj.*	*Emb.*
Baseline	43.736	2.759	42.933	6.519
LGD-DPS	7.040	4.405	5.307	**6.240**
LGD-MC ($n = 10$)	3.902	3.187	**3.921**	6.339
LGD-MC ($n = 100$)	**3.666**	**3.100**	4.160	6.633

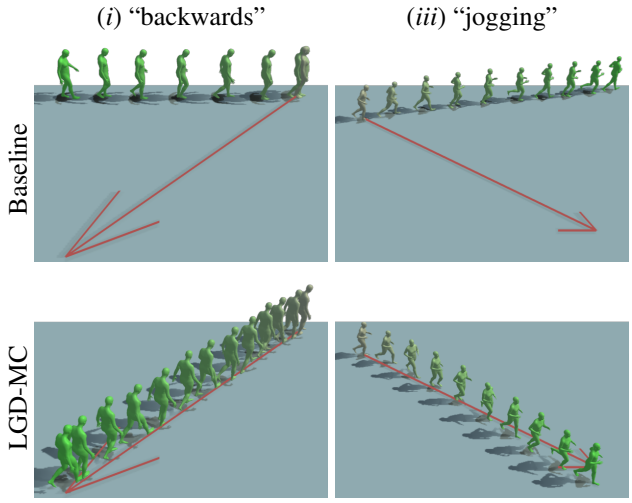

Figure 7. Random samples for prompt (*i*) "backwards", (*iii*) "jogging" for the path following case. The red arrows indicate the target location from starting position. Thanks to loss guidance, our method can better control the synthesized motions. More results are in Fig. 9, App. B.3.

6.3. Controllable Motion Synthesis for Human Motion Diffusion Models

Next, we consider a controllable motion synthesis task for text-conditioned human motion diffusion models (Tevet et al., 2022), where losses can be used to provide more fine-grained control to certain behaviors of the generated avatars, such as root motion (*i.e.*, the sequence of locations of the avatar). For a generated motion sequence $\mathbf{x}_0$ and its root motion (denoted as $\text{root}(\mathbf{x}_0)$), we consider two types of losses. The first (path following) encourages the avatar to follow a particular path $\mathbf{y}$:

$$\ell_{\mathbf{y}}^{(1)}(\mathbf{x}_0) = \|\mathbf{y} - \text{root}(\mathbf{x}_0)\|_2^2, \tag{14}$$

whereas the second (obstacle avoidance) encourages the avatar to reach a target destination $\mathbf{y}_\text{t}$ while avoiding an obstacle $\mathbf{y}_\text{obs}$ that blocks the most direct path:

$$\ell_{\mathbf{y}}^{(2)}(\mathbf{x}_0) = \|\mathbf{y}_\text{t} - \text{target}(\text{root}(\mathbf{x}_0))\|_2^2 + \text{collision}(\text{root}(\mathbf{x}_0) - \mathbf{y}_\text{obs}), \tag{15}$$

where $\text{target}(\cdot)$ evaluates the final destination of $\mathbf{x}_0$ and $\text{collision}(\cdot)$ is a sigmoid-like function that assigns near-zero values if $\mathbf{x}_0$ does not become close or collide with $\mathbf{y}_\text{obs}$ and very large values otherwise. We include additional details about the loss function in App. B.3.2.

Our objective is to generate motions that are not only based on text but also follow the desired conditions (*i.e.*, low loss values). To this end, we evaluate performance based on two metrics: *Objective*, which is the value of the desired loss function $\ell_\mathbf{y}$, divided by the number of frames; and *Embedding*, which measures the Euclidean distance between the embeddings of the generated motions and the provided text prompt (Guo et al., 2022). We show the validity of the *Embedding* metric in App. B.3.1. The desired motions should

have low loss values in the *Objective* metric and reasonable loss values in the *Embedding* metric. Four methods are evaluated: Baseline, which is the motion diffusion model that performs text-to-motion without considering the loss function; LGD-DPS based on Eq. 8; and LGD-MC based on Eq. 9 (with $n = 10$ and $n = 100$).

Path following. For loss $\ell_{\mathbf{y}}^{(1)}$, we synthesize motions based on four text prompts that specify root movements as well as diverse poses of the avatar: (*i*) "the person is walking backwards", (*ii*) "a person walking around in a balance beam", (*iii*) "the person is jogging", and (*iv*) "a person walks while holding an object above their head", where the desired path is a straight line in three different specific directions.

We list the results on these four cases in Tab. 2, where each result is the average of 3 different directions and 10 random runs for each direction. We show some examples in Fig. 7. All the methods have similar embedding losses, which means that the synthesized motions are following the text prompts. The *baseline* method has a very high objective loss since the baseline motion diffusion model does not control the root motions. LGD-DPS has much lower objective loss values but is still bested by LGD-MC in 3 out of 4 cases. Similarly, LGD-MC performs better with a larger n in most scenarios.

Obstacle avoidance. For loss $\ell_{\mathbf{y}}^{(2)}$, we design the target $\mathbf{y}_{\mathrm{t}}$ such that the shortest path would collide with the obstacle $\mathbf{y}_{\mathrm{obs}}$, and thus the avatar has to go around it to reach the target (see Fig. 10 in App. B.3.2 for an illustration). The collision loss is close to a step function, and its gradients have very small magnitudes for most values; this makes it difficult for DPS since it is based on the MMSE point estimate. However, this issue can be alleviated by our Monte Carlo approach since samples that avoid the obstacle will be weighted higher during backpropagation, thus making it easier for the updates to find solutions that avoid the obstacle.

Tab. 3 shows the quantitative results of this experiment, where we consider two prompts: (*v*) "a person walks", and (*vi*) "a person walking backwards slowly". Here, we observe an even more significant improvement over the *Objective* metric, indicating that LGD-MC is better at avoiding the obstacles than DPS (qualitative results in Fig. 2).

6.4. Sampling Conditional Images from Unconditional Diffusion Models

Finally, we perform a direct comparison with different conditional generation paradigms, such as classifier guidance (CG) and diffusion as plug-and-play priors (D-PnP). We employ losses that evaluate the *alignment* between the generated samples and specific conditions, such as the out-of-shelf CLIP loss for text-conditioned samples and classifiers for label-conditioned samples. Here, we generate text-conditioned and label-conditioned samples from unconditional 256×256 ImageNet diffusion models (see Fig. 1 for some examples). To quantitatively evaluate LGD and other algorithms, we generate $10,000$ samples and measure sample quality with FID (Heusel et al., 2017) and the *alignment* loss.

Table 4. Results on generating label-conditioned images with unconditional models.

Method	FID (↓)	Loss (↓)
Baseline	**37.28**	281.84
CG	44.64	**25.47**
D-PnP	317.21	221.75
LGD-DPS	48.24	31.95
LGD-MC ($n = 1$)	39.87	33.73
LGD-MC ($n = 5$)	38.65	32.80

Tab. 4 indicates that the LGD method offers improved controllability in sampling, albeit with a slight decrease in sample fidelity compared to the baseline unconditional sampling algorithm. On the other hand, the classifier guidance (CG) and LGD-DPS methods achieve lower classification loss at the expense of FID score. Interestingly, we find CG prioritizes control over fidelity on unconditional models, even though it has been shown to improve image quality for conditional models (Dhariwal & Nichol, 2021). Notably, our findings show that the LGD-MC method maintains a better FID score than other methods, while also achieving improved alignment, as compared to CG and LGD-DPS. The results also reveal that the D-PnP method struggles to generate high-quality samples when applied to the ImageNet dataset[2] and suggests that LGD would be a more reasonable paradigm for plug-and-play controllable generation.

7. Conclusions

In this paper, we study *Loss-Guided Diffusion* (LGD), a paradigm for guided conditional generation with diffusion models in a plug-and-play fashion. The resulting guidance term involves an intractable expectation that has to be approximated by losses evaluated on samples. We show that LGD-MC, a Monte Carlo approach using multiple samples, has better approximation capabilities than DPS, an approach that only uses one sample. This improvement in approximation also translates into improved empirical performance in various applications.

[2]Similar observations have been made in the community regarding D-PnP and ImageNet, see https://github.com/AlexGraikos/diffusion_priors/issues/1.

Limitations We note that while our current LGD-MC approach achieves reasonable empirical results, the Gaussian assumption made on $q(\mathbf{x}_0|\mathbf{x}_t)$ is still very inaccurate, which may limit the accuracy of the approximation and the downstream empirical performance. It would be interesting to investigate better principles for selecting $q(\mathbf{x}_0|\mathbf{x}_t)$, as well as methods such as annealed importance sampling with reduced variance. Our method also performs worse when the desired conditions cannot be covered by the prior distribution $p_0(\mathbf{x}_0)$, which is an issue shared with all plug-and-play generative modeling approaches. These are all interesting avenues for future work.

References

Anonymous. Pseudoinverse-guided diffusion models for inverse problems. 2022.

Balaji, Y., Nah, S., Huang, X., Vahdat, A., Song, J., Kreis, K., Aittala, M., Aila, T., Laine, S., Catanzaro, B., et al. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. arXiv preprint arXiv:2211.01324, 2022.

Bautista, M. A., Guo, P., Abnar, S., Talbott, W., Toshev, A., Chen, Z., Dinh, L., Zhai, S., Goh, H., Ulbricht, D., et al. Gaudi: A neural architect for immersive 3d scene generation. arXiv preprint arXiv:2207.13751, 2022.

Biloš, M., Rasul, K., Schneider, A., Nevmyvaka, Y., and Günnemann, S. Modeling temporal data as continuous functions with process diffusion. arXiv preprint arXiv:2211.02590, 2022.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., Brynjolfsson, E., Buch, S., Card, D., Castellon, R., Chatterji, N., Chen, A., Creel, K., Davis, J. Q., Demszky, D., Donahue, C., Doumbouya, M., Durmus, E., Ermon, S., Etchemendy, J., Ethayarajh, K., Fei-Fei, L., Finn, C., Gale, T., Gillespie, L., Goel, K., Goodman, N., Grossman, S., Guha, N., Hashimoto, T., Henderson, P., Hewitt, J., Ho, D. E., Hong, J., Hsu, K., Huang, J., Icard, T., Jain, S., Jurafsky, D., Kalluri, P., Karamcheti, S., Keeling, G., Khani, F., Khattab, O., Kohd, P. W., Krass, M., Krishna, R., Kuditipudi, R., Kumar, A., Ladhak, F., Lee, M., Lee, T., Leskovec, J., Levent, I., Li, X. L., Li, X., Ma, T., Malik, A., Manning, C. D., Mirchandani, S., Mitchell, E., Munyikwa, Z., Nair, S., Narayan, A., Narayanan, D., Newman, B., Nie, A., Niebles, J. C., Nilforoshan, H., Nyarko, J., Ogut, G., Orr, L., Papadimitriou, I., Park, J. S., Piech, C., Portelance, E., Potts, C., Raghunathan, A., Reich, R., Ren, H., Rong, F., Roohani, Y., Ruiz, C., Ryan, J., Ré, C., Sadigh, D., Sagawa, S., Santhanam, K., Shih, A., Srinivasan, K., Tamkin, A., Taori, R., Thomas, A. W., Tramèr, F., Wang, R. E., Wang, W., Wu, B., Wu, J., Wu, Y., Xie, S. M., Yasunaga, M., You, J., Zaharia, M., Zhang, M., Zhang, T., Zhang, X., Zhang, Y., Zheng, L., Zhou, K., and Liang, P. On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258, August 2021.

Choi, J., Kim, S., Jeong, Y., Gwon, Y., and Yoon, S. ILVR: Conditioning method for denoising diffusion probabilistic models. arXiv preprint arXiv:2108.02938, August 2021.

Chung, H. and Ye, J. C. Score-based diffusion models for accelerated mri. Medical Image Analysis, pp. 102479, 2022.

Chung, H., Kim, J., Mccann, M. T., Klasky, M. L., and Ye, J. C. Diffusion posterior sampling for general noisy inverse problems. arXiv preprint arXiv:2209.14687, 2022.

Crowson, K., Biderman, S., Kornis, D., Stander, D., Hallahan, E., Castricato, L., and Raff, E. Vqgan-clip: Open domain image generation and editing with natural language guidance. In Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVII, pp. 88–105. Springer, 2022.

Dhariwal, P. and Nichol, A. Diffusion models beat GANs on image synthesis. arXiv preprint arXiv:2105.05233, May 2021.

Dockhorn, T., Vahdat, A., and Kreis, K. GENIE: Higher-Order Denoising Diffusion Solvers. In Advances in Neural Information Processing Systems, 2022.

Efron, B. Tweedie's formula and selection bias. Journal of the American Statistical Association, 106(496):1602–1614, 2011.

Graikos, A., Malkin, N., Jojic, N., and Samaras, D. Diffusion models as plug-and-play priors. arXiv preprint arXiv:2206.09012, 2022.

Grathwohl, W., Wang, K.-C., Jacobsen, J.-H., Duvenaud, D., Norouzi, M., and Swersky, K. Your classifier is secretly an energy based model and you should treat it like one. arXiv preprint arXiv:1912.03263, 2019.

Grenander, U. and Miller, M. I. Representations of knowledge in complex systems. Journal of the Royal Statistical Society: Series B (Methodological), 56(4): 549–581, 1994.

Guo, C., Zou, S., Zuo, X., Wang, S., Ji, W., Li, X., and Cheng, L. Generating diverse and natural 3d human motions from text. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5152–5161, 2022.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two Time-Scale update rule converge to a local nash equilibrium. arXiv preprint arXiv:1706.08500, June 2017.

Ho, J. and Salimans, T. Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598, 2022.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. arXiv preprint arXiv:2006.11239, June 2020.

Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., and Fleet, D. J. Video diffusion models. arXiv preprint arXiv:2204.03458, 2022.

Jalal, A., Arvinte, M., Daras, G., Price, E., Dimakis, A. G., and Tamir, J. I. Robust compressed sensing MRI with deep generative priors. arXiv preprint arXiv:2108.01368, August 2021.

Karchev, K., Montel, N. A., Coogan, A., and Weniger, C. Strong-lensing source reconstruction with denoising diffusion restoration models. arXiv preprint arXiv:2211.04365, 2022.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. arXiv preprint arXiv:2206.00364, 2022.

Kawar, B., Vaksman, G., and Elad, M. SNIPS: Solving noisy inverse problems stochastically. arXiv preprint arXiv:2105.14951, May 2021.

Kawar, B., Elad, M., Ermon, S., and Song, J. Denoising diffusion restoration models. arXiv preprint arXiv:2201.11793, 2022a.

Kawar, B., Song, J., Ermon, S., and Elad, M. Jpeg artifact correction using denoising diffusion restoration models. arXiv preprint arXiv:2209.11888, 2022b.

Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. Diffwave: A versatile diffusion model for audio synthesis. arXiv preprint arXiv:2009.09761, 2020.

Li, X. L., Thickstun, J., Gulrajani, I., Liang, P., and Hashimoto, T. B. Diffusion-lm improves controllable text generation. arXiv preprint arXiv:2205.14217, 2022.

Lin, C.-H., Gao, J., Tang, L., Takikawa, T., Zeng, X., Huang, X., Kreis, K., Fidler, S., Liu, M.-Y., and Lin, T.-Y. Magic3d: High-resolution text-to-3d content creation. arXiv preprint arXiv:2211.10440, 2022.

Liu, L., Ren, Y., Lin, Z., and Zhao, Z. Pseudo numerical methods for diffusion models on manifolds. arXiv preprint arXiv:2202.09778, 2022.

Lovelace, J., Kishore, V., Wan, C., Shekhtman, E., and Weinberger, K. Latent diffusion for language generation. arXiv preprint arXiv:2212.09462, 2022.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. arXiv preprint arXiv:2206.00927, 2022.

Meng, C., Gao, R., Kingma, D. P., Ermon, S., Ho, J., and Salimans, T. On distillation of guided diffusion models. arXiv preprint arXiv:2210.03142, 2022.

Poole, B., Jain, A., Barron, J. T., and Mildenhall, B. Dreamfusion: Text-to-3d using 2d diffusion. arXiv preprint arXiv:2209.14988, 2022.

Qiao, Z., Nie, W., Vahdat, A., Miller III, T. F., and Anandkumar, A. Dynamic-backbone protein-ligand structure prediction with multiscale generative diffusion models. arXiv preprint arXiv:2209.15171, 2022.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. arXiv preprint arXiv:2103.00020, February 2021.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10684–10695, 2022.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., et al. Photorealistic text-to-image diffusion models with deep language understanding. arXiv preprint arXiv:2205.11487, 2022.

Saito, K., Murata, N., Uesaka, T., Lai, C.-H., Takida, Y., Fukui, T., and Mitsufuji, Y. Unsupervised vocal dereverberation with diffusion-based generative models. arXiv preprint arXiv:2211.04124, 2022.

Salimans, T. and Ho, J. Progressive distillation for fast sampling of diffusion models. arXiv preprint arXiv:2202.00512, February 2022.

Santurkar, S., Ilyas, A., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Image synthesis with a single (robust) classifier. Advances in Neural Information Processing Systems, 32, 2019.

Sawata, R., Murata, N., Takida, Y., Uesaka, T., Shibuya, T., Takahashi, S., and Mitsufuji, Y. A versatile diffusion-based generative refiner for speech enhancement. arXiv preprint arXiv:2210.17287, 2022.

Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al. Laion-5b: An open large-scale dataset for training next generation image-text models. arXiv preprint arXiv:2210.08402, 2022.

Shue, J. R., Chan, E. R., Po, R., Ankner, Z., Wu, J., and Wetzstein, G. 3d neural field generation using triplane diffusion. arXiv preprint arXiv:2211.16677, 2022.

Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. arXiv preprint arXiv:1503.03585, March 2015.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In International Conference on Learning Representations, 2021a.

Song, Y., Shen, L., Xing, L., and Ermon, S. Solving inverse problems in medical imaging with score-based generative models. arXiv preprint arXiv:2111.08005, 2021b.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In International Conference on Learning Representations, 2021c.

Tashiro, Y., Song, J., Song, Y., and Ermon, S. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. Advances in Neural Information Processing Systems, 34:24804–24816, 2021.

Tevet, G., Raab, S., Gordon, B., Shafir, Y., Cohen-Or, D., and Bermano, A. H. Human motion diffusion model. arXiv preprint arXiv:2209.14916, 2022.

Venkatakrishnan, S. V., Bouman, C. A., and Wohlberg, B. Plug-and-play priors for model based reconstruction. In 2013 IEEE Global Conference on Signal and Information Processing, pp. 945–948. IEEE, 2013.

Vincent, P. A connection between score matching and denoising autoencoders. Neural computation, 23(7):1661–1674, July 2011.

Wu, K. E., Yang, K. K., Berg, R. v. d., Zou, J. Y., Lu, A. X., and Amini, A. P. Protein structure generation via folding diffusion. arXiv preprint arXiv:2209.15611, 2022.

Xiao, Z., Kreis, K., and Vahdat, A. Tackling the generative learning trilemma with denoising diffusion GANs. In International Conference on Learning Representations (ICLR), 2022.

Xu, D., Jiang, Y., Wang, P., Fan, Z., Wang, Y., and Wang, Z. Neurallift-360: Lifting an in-the-wild 2d photo to a 3d object with 360° views. arXiv e-prints, pp. arXiv–2211, 2022a.

Xu, M., Yu, L., Song, Y., Shi, C., Ermon, S., and Tang, J. Geodiff: A geometric diffusion model for molecular conformation generation. arXiv preprint arXiv:2203.02923, 2022b.

Zhang, Q. and Chen, Y. Fast sampling of diffusion models with exponential integrator. arXiv preprint arXiv:2204.13902, April 2022.

Zhang, Q., Tao, M., and Chen, Y. gddim: Generalized denoising diffusion implicit models. arXiv preprint arXiv:2206.05564, June 2022.

Zheng, H., Nie, W., Vahdat, A., Azizzadenesheli, K., and Anandkumar, A. Fast sampling of diffusion models via operator learning. arXiv preprint arXiv:2211.13449, 2022.

A. Additional method details

A.1. Preliminaries on Diffusion Models

The family of distributions $p_t(\mathbf{x}_t)$, $t \in [0, T]$ are tied via the following stochastic differential equations (SDE) (Grenander & Miller, 1994; Karras et al., 2022; Zhang et al., 2022):

$$\begin{aligned} \mathrm{d}\mathbf{x} = &\underbrace{-\dot{\sigma}_t \sigma_t \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) \mathrm{d}t}_{\text{Probabilistic ODE}} \\ &\underbrace{-\beta_t \sigma_t^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) \mathrm{d}t + \sqrt{2\beta_t}\sigma_t \mathrm{d}\omega_t}_{\text{Langevin process}}, \end{aligned} \tag{16}$$

where $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ is the score function, ω_t is the standard Wiener process, and β_t is a hyperparameter that controls the stochasticity of the process. Different forms of SDEs in the literature, such as variance preserving and variance exploding ones, can be described with certain choices of σ_t and β_t, up to a time-dependent scaling factor over $\mathbf{x}$.

A.2. Comparing Different Paradigms for Conditional Generation

Table 5. Comparison of existing paradigms, such as Classifier(-free) Guidance (CG), Diffusion models as plug-and-play priors (D-PnP), and the proposed Loss-Guided Diffusion (LGD). See Sec. 5 for detailed explanations.

Paradigm	Plug-and-Play	Solution	Speed	Domain	Examples
CG	No	Samples	Fast	Restricted	Dhariwal & Nichol (2021); Ho & Salimans (2022)
D-PnP	Yes	Point estimate	Slow	Unrestricted	Graikos et al. (2022); Xu et al. (2022a)
LGD	Yes	Samples	Fast	Restricted	Ho et al. (2022); Chung et al. (2022)

A.3. Proof of Proposition 4.1

Proposition A.1. *Assume that $p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0)$ is bounded for all $\mathbf{x}_0$ and $\mathbf{y}$, and given $t \in (0, T]$, $p(\mathbf{x}_0|\mathbf{x}_t)$ and $q(\mathbf{x}_0|\mathbf{x}_t)$ are supported on $\mathcal{X}$. Then $\forall \mathbf{x}_t \in \mathcal{X}$, there exists a constant c such that:*

$$\left|\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}[p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0)] - \mathbb{E}_{q(\mathbf{x}_0|\mathbf{x}_t)}[p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0)]\right| \tag{12}$$

$$\leq c \cdot TV(p(\mathbf{x}_0|\mathbf{x}_t), q(\mathbf{x}_0|\mathbf{x}_t)), \tag{13}$$

where $TV(p, q)$ is the total variation between p and q.

Proof. Let $M = \max_{\mathbf{x}_0} p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0)$. Then,

$$\left|\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}[p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0)] - \mathbb{E}_{q(\mathbf{x}_0|\mathbf{x}_t)}[p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0)]\right| = \left|\int \left(p(\mathbf{x}_0|\mathbf{x}_t) - q(\mathbf{x}_0|\mathbf{x}_t)\right) p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0) \mathrm{d}\mathbf{x}_0\right| \tag{17}$$

$$\leq \left|\max_{\mathbf{x}_0} p_0^{(\ell)}(\mathbf{y}|\mathbf{x}_0)\right| \left(\int \left|\frac{p(\mathbf{x}_0|\mathbf{x}_t)}{q(\mathbf{x}_0|\mathbf{x}_t)} - 1\right| q(\mathbf{x}_0|\mathbf{x}_t) \mathrm{d}\mathbf{x}_0\right) = 2M \times \mathrm{TV}(p(\mathbf{x}_0|\mathbf{x}_t), q(\mathbf{x}_0|\mathbf{x}_t)), \tag{18}$$

so we can use the constant $2M$ to complete the proof. □

A.4. Ablation on the Number of Monte Carlo Samples

In Fig. 8, we show the estimation results of LGD-MC when $n \in \{1, 5, 10, 20\}$. The results show that better estimation is achieved when n is increased.

A.5. Computational Efficiency

To support our claims in Sec. 4.3 on compute and memory, we evaluate the memory consumption and average time to perform one iteration over image super-resolution on the same NVIDIA RTX 3090 GPU. Results in Tab. 6 suggest that the computational resources needed for LGD-MC grow very slowly as n grows.

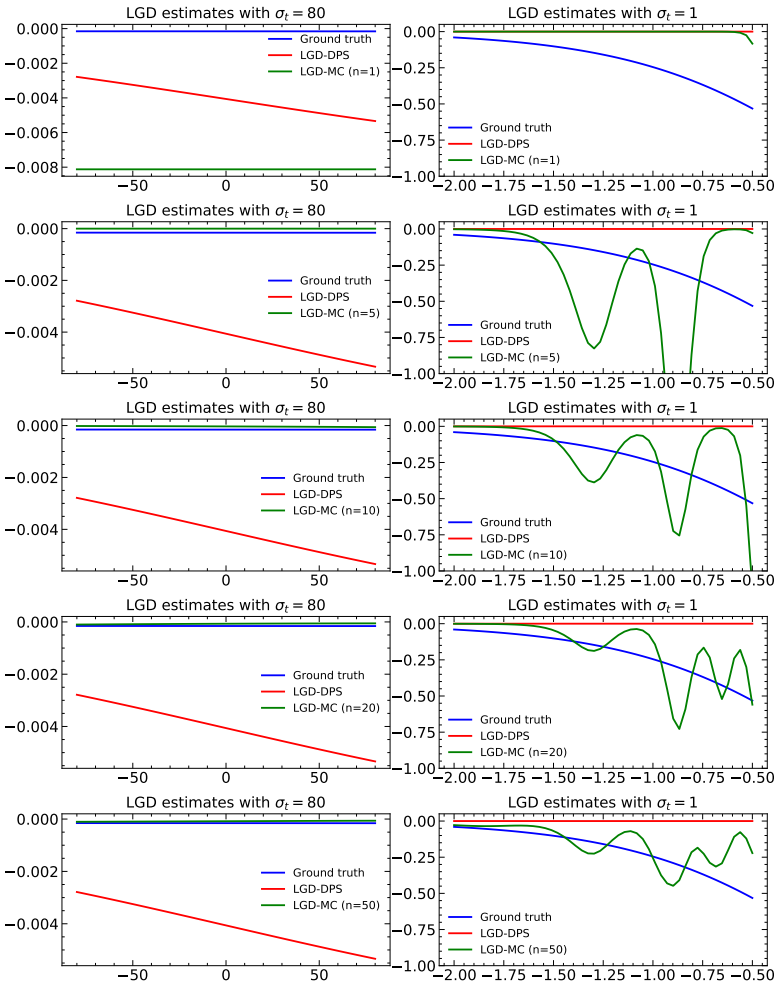

Figure 8. For the LGD problem in Sec. 4.2, the ground truth score, the score estimated by DPS and the score estimated by Monte Carlo, with σ_t being 1 (right) or 80 (left), respectively. From top to bottom: $n = 1, 5, 10, 20$. We note that LGD-MC performs as bad as DPS when $n = 1$, and its performance becomes better as n increases.

B. Additional Experimental Details

B.1. Mixture of Gaussian

For all three cases, we draw 100 samples using the same DDIM algorithm with 20 timesteps (Song et al., 2021a), and plot the histogram of the samples in Fig. 6. Additionally, we compute the mean μ and standard deviation σ from these samples, and measure the KL divergence between the ground truth $p(\mathbf{x}_0|\mathbf{y} = 0) = \mathcal{N}(-1, 0.2^2)$ and the Gaussian produced from generated statistics $\mathcal{N}(\mu, \sigma^2)$. The results in Fig. 6 show that while samples from LGD-MC are not perfect, they are still much closer to that of LGD-DPS. Thus, we believe the LGD-MC guidance terms are generally more accurate than LGD-DPS.

B.2. Image Super-resolution

We follow the approach in Anonymous (2022), where we use the DDIM sampler (Song et al., 2021a) with $\eta = 1.0$ (*i.e.*, the DDPM sampler) for 100 steps over the 50k ImageNet validation set. The unconditional model is used. Specifically, we use the following loss function:

$$\ell_{\mathbf{y}}(\mathbf{x}_0) = \frac{\|\mathbf{y} - \boldsymbol{H}\mathbf{x}_0\|_2^2}{2s_t^2}, \tag{19}$$

Table 6. Computational resources spent with different n on the image super-resolution task.

Resources	Peak memory	Wall-clock time / iter
LGD-DPS	16.51 GB	0.380 s
LGD-MC ($n = 1$)	16.51 GB	0.387 s
LGD-MC ($n = 10$)	16.53 GB	0.390 s
LGD-MC ($n = 100$)	16.67 GB	0.410 s

where $s_t = 0.25$. To prevent large gradient values towards the end of the sampling process, we multiply the LGD approximations by s_t^2. FID and ResNet-50 classifier accuracy are used as metrics.

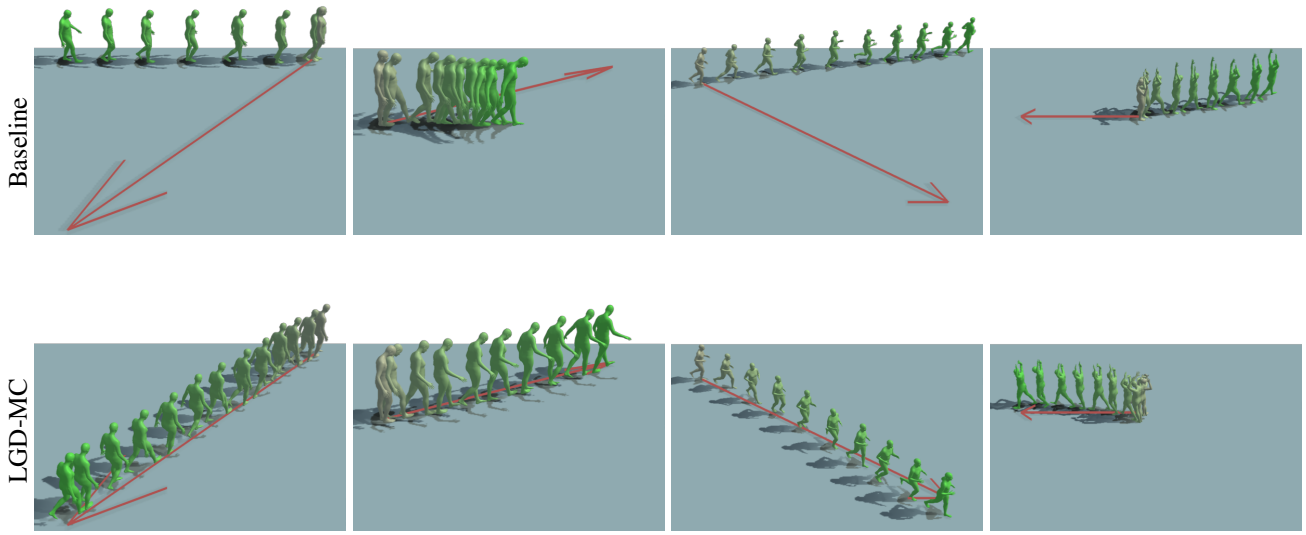

Figure 9. Random samples for (*i*) "backwards", (*ii*) "balance beam", (*iii*) "jogging" and (*iv*) "object above head" for the path following case. The Baseline method fails to produce satisfactory trajectories due to a lack of training data and path-following capabilities. LGD, which utilizes loss guidance, is able to generate more controlled motion trajectories with a small additional computational overhead.

B.3. Controllable Motion Synthesis

We save the videos of the motion synthesis experiments in `lgd_motion_video.pptx`.

For all experiments, we use the same motion diffusion model (Tevet et al., 2022) and the 100-step DDIM sampler, with $r_t = \sigma_t / \sqrt{1 + \sigma_t^2}$. We compute the root motion from the samples of the motion diffusion model, where the (x, y, z) location of each joint can be computed. The reported *Objective* loss is the actual loss $\ell_{\mathbf{y}}(\mathbf{x}_0)$ divided by the number of frames. For path following, we consider three different directions for each prompt, and each direction has 10 random seeds, the metrics are then averaged together over the 30 synthesized motions. For obstacle avoidance, we consider two directions for each prompt. We show additional path-following results in Fig. 9.

B.3.1. Validity of the Embedding Metric

In Tab. 7, we show the confusion matrix between different text embeddings and motion embeddings from the conditioned text in the path-following experiment. The results show that the embeddings from the text-generated motion are indeed closer to the corresponding text embedding than others. This suggests that the embedding metric is a valid indicator of how close the generated motion follows the text prompt.

Table 7. Confusion matrix between the text embeddings and motion embeddings from the conditioned text.

Text \ Motion	(*i*)	(*ii*)	(*iii*)	(*iv*)
(*i*)	**3.37**	9.69	9.87	3.94
(*ii*)	8.79	**5.50**	11.24	8.67
(*iii*)	9.83	9.51	**4.35**	10.13
(*iv*)	4.85	10.89	10.26	**2.19**

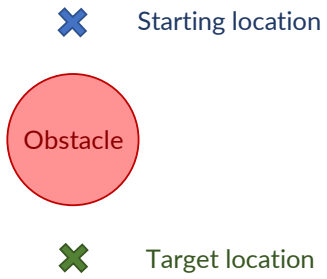

Figure 10. High-level illustration of the obstacle avoidance task. It is easier to walk in a straight line to reach the target location, but this would incur higher penalties since it crosses the obstacle. To reduce the loss, the avatar should walk around the obstacle.

B.3.2. Obstacle avoidance

Fig. 10 shows an illustration of the obstacle avoidance task being considered.

For the collision loss, we use the following function:

$$\text{collision}(\text{root}(\mathbf{x}_0) - \mathbf{y}_{\text{obs}}) = \sum_i \text{sigmoid}(-(\|\text{root}(\mathbf{x}_0^{(i)}) - \mathbf{y}_{\text{obs}}\|_2^2 - 1.0) \times 50) \times 100, \tag{20}$$

which sums over all the individual timesteps i. The loss is close to zero if the distance of the root location is at least 1.0 from $\mathbf{y}_{\text{obs}}$; otherwise, it will be close to 100. This incurs a heavy penalty for violating the collision constraints.

B.4. Conditional Image Generation

In this experiment, we use the pre-trained unconditioned 256×256 ImageNet diffusion model [3] from OpenAI guided diffusion models (Dhariwal & Nichol, 2021). For the qualitative text-to-image experiment, we employ the matching score of OpenAI VIT-Large CLIP (Radford et al., 2021) to guide conditional generation. We adopt data augmentation tricks suggested in Crowson et al. (2022) to achieve better visual quality. For the quantitative result in Tab. 4, we adopt the pre-trained classifier from OpenAI guided diffusion models (Dhariwal & Nichol, 2021); some images are shown in Fig. 12. We note the classifier is originally trained to classify noised images in an amortized manner, which can directly be used in the classifier-guidance (CG) setting. For D-PnP and LGD, we use the same classifier model but with $t = 10^{-3}$. We note that better-trained, more robust classifiers can be also used and are believed to improve D-PnP and LGD performance (Santurkar et al., 2019; Grathwohl et al., 2019). For D-PnP, we use the official codebase[4] with our ImageNet diffusion model. The unsatisfying sample quality of D-PnP (see Fig. 11) is not a new finding and how to improve it in complex datasets such as ImageNet remains an open problem [5]. It is likely that this is due to the complexity of the data distribution of ImageNet, and that D-PnP may perform better in simpler scenarios, such as generating MNIST digits or faces conditioned on specific styles (Graikos et al., 2022). Regarding sampling for Baseline and LGD, we adopt the stochastic Heun method (Karras et al., 2022) with 71 NFE.

[3] https://openaipublic.blob.core.windows.net/diffusion/jul-2021/256x256_diffusion_uncond.pt
[4] https://github.com/AlexGraikos/diffusion_priors/blob/main/ffhq/infer_with_attributes.ipynb
[5] https://github.com/AlexGraikos/diffusion_priors/issues/1

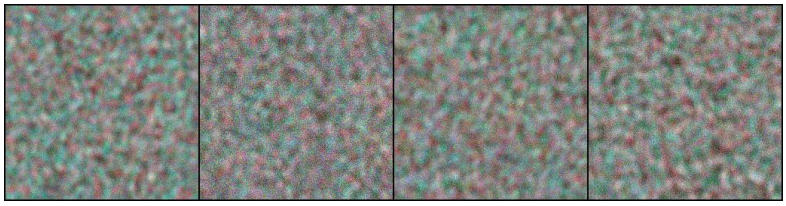

Figure 11. Samples from D-PnP on unconditional 256×256 diffusion models. D-PnP fails to generate realistic images.

Baseline | Classifier Guidance (CG) | LGD-DPS | LGD-MC (n=5)

Figure 12. Generated ImageNet samples from various methods for *'tench, Tinca tinca'*, *'jeep, landrover'*.