

# Correcting Diffusion Generation through Resampling

Yujian Liu  
UC Santa Barbara  
yujianliu@ucsb.edu

Yang Zhang  
MIT-IBM Watson AI Lab  
yang.zhang2@ibm.com

Tommi Jaakkola  
MIT CSAIL  
tommi@csail.mit.edu

Shiyu Chang  
UC Santa Barbara  
chang87@ucsb.edu



Figure 1. Sample images generated by Stable Diffusion v2.1 with and without our particle filtering algorithm (PF). The left two columns illustrate the missing object errors, and the right column illustrates the unnatural distortions. Missing objects are highlighted in red.

## Abstract

*Despite diffusion models’ superior capabilities in modeling complex distributions, there are still non-trivial distributional discrepancies between generated and ground-truth images, which has resulted in several notable problems in image generation, including missing object errors in text-to-image generation and low image quality. Existing methods that attempt to address these problems mostly do not tend to address the fundamental cause behind these problems, which is the distributional discrepancies, and hence achieve sub-optimal results. In this paper, we propose a particle filtering framework that can effectively address both problems by explicitly reducing the distributional discrepancies. Specifically, our method relies on a set of external guidance, including a small set of real images and a pre-trained object detector, to gauge the distribution gap, and then design the resampling weight accordingly to correct the gap. Experiments show that our methods can effectively correct missing object errors and improve image quality in various image generation tasks. Notably, our method outper-*

*forms the existing strongest baseline by 5% in object occurrence and 1.0 in FID on MS-COCO. Our code is publicly available at [https://github.com/UCSB-NLP-Chang/diffusion\\_resampling.git](https://github.com/UCSB-NLP-Chang/diffusion_resampling.git).*

## 1. Introduction

Diffusion models have achieved impressive success in generating high-quality images in various image-generation applications [3, 6, 15, 20, 23, 25, 29, 31, 40, 43–45, 47, 48, 50, 51, 54–57, 61, 66, 67]. Such success can be ascribed to diffusion models’ superior capability in modeling complex data distributions and thus low distributional discrepancies between the real and generated images. However, the diffusion generation process still introduces distributional discrepancies in modeling data distributions, due to the limited representation power of the denoising network, the errors introduced in discretizing and numerically solving the continuous ODE or SDE denoising trajectories [32, 65], etc.

Such distributional discrepancies have led to several notable problems in image generation. For example, in text-to-

image generation, diffusion models suffer from the *missing object errors*, i.e., objects that are mentioned in the input text are sometimes not generated [18, 42, 62]. As shown in Figure 1 top-left, given the input description ‘*Cat on chair peering over top of table at glass of beverage*’, the stable diffusion model [48], a powerful text-to-image model, sometimes fails to generate the glass of beverage.

Another well-known problem in diffusion-based image generation is the *low image quality*, such as artifacts and unnatural distortions. In Figure 1 bottom-right, given the input description ‘*A man sitting on a bed with a dog while holding a cell phone*’, a sample image generated by stable diffusion contains a person with a missing left lower leg.

Since the fundamental cause behind these problems is the distribution gap between the generated and real images, an effective remedy to these problems should be to close the distribution gap. However, the existing research attempts to address these problems either do not aim to close the gap, or are not effective in doing so. For example, to address the missing object errors, some previous works designed cross-attention mechanism that controls how each token in the caption should be attended so that no object is ignored [4, 18, 35, 62], but they do not aim to reduce the distributional gap, so their effects on generating complete objects are limited [30], and their over-emphasis on improving object occurrence could lead to quality degradation. To address the low image quality issue, Kim et al. [32] introduces a discriminator to rectify the score function in the diffusion generation process, which is still subject to errors due to numerically solving the ODE/SDE denoising trajectories.

In this paper, we aim to develop a sampling-based framework that can effectively address both issues above by explicitly reducing the distributional discrepancies. We first present some findings of our initial exploration, which shows that sampling-based methods turn out to be a more effective way of modifying the output distribution of generated images and achieving desired generation properties than the more complicated baselines. Based on these findings, we design a particle-filtering framework [12, 16] that allows us to approximately sample from the ground-truth distribution. Specifically, our method relies on a set of external guidance, including a small set of real images and a pre-trained object detector, to gauge the distribution gap, and then design the resampling weight accordingly.

We evaluate our approach on text-to-image generation, unconditional and class-conditioned generation. On existing text prompt benchmarks [36, 62], our approach outperforms all the competitive specialized text-to-image methods in improving faithfulness to input text and image quality, as illustrated in Figure 1. Notably, our method outperforms the existing strongest baseline by 5% in object occurrence and 1.0 in FID on MS-COCO. On ImageNet-64, our method achieves a state-of-the-art FID of 1.02 for class-conditioned

generation, outperforming the strong baseline that uses discriminator guidance [32].

## 2. Related Works

**Faithful Text-to-image Generation.** Recent studies highlight faithfulness as a key challenge in text-to-image diffusion models [9–11, 18, 19, 34, 37, 41, 42, 62], including missing objects, mistakenly bound attributes, wrong locations, etc. To address this challenge, existing work modifies the diffusion generation process to separately focus on each aspect in the caption and later combine the outputs of each part [17, 38, 59]. For example, Wu et al. [62] and Feng et al. [18] modify the cross-attention between image and text to separately attend to each noun phrase in the text, and combine attention outputs. Several works leverage layout information to increase objects’ attention weights in specified image regions [4, 35, 62]. Karthik et al. [30] also employs the sample selection idea, but their selection is only performed at the final step, and does not aim to approach the true distribution, whereas our method reduces the distributional gap at each denoising step. Moreover, our method can extend beyond text-to-image generation.

**Particles in Diffusion Generation.** Particle filtering has been applied in diffusion generation to obtain samples from a target distribution [2, 58, 60]. However, these works focus on re-purposing an unconditional diffusion model as a conditional model, rather than reducing the gap with the ground-truth distribution. Corso et al. [13] proposes particle guidance to increase the diversity among generated samples. Their method modifies the score function with a joint-particles potential, which is orthogonal to our work.

**Diffusion Generation with Discriminator.** Discriminators have been used to improve diffusion generation. Xiao et al. [63] uses a GAN to approximate multimodal distributions in the denoising process to reduce the number of denoising steps. Kang et al. [27] trains diffusion models with the additional objective of fooling a discriminator to improve image quality when the number of denoising steps is small. Kim et al. [32] leverages a discriminator to generate samples closer to ground-truth distribution. However, their discriminator is used to modify the model score, whereas ours is used to resample images, which is less affected by the discretization errors of the sampling algorithm.

## 3. Methodology

### 3.1. Background and Notation

Throughout this section, we use upper-case letters,  $\mathbf{X}$ , to denote random vectors, and lower-case letters,  $x$ , to denote deterministic vectors. We use the colon notation,  $\mathbf{X}_{1:T}$ , to denote a set of variables ranging from  $\mathbf{X}_1$  to  $\mathbf{X}_T$ .

In this work, we will primarily focus on text-to-image diffusion models, which, given an input text description denoted as  $\mathbf{C}$ , generate images that satisfy the text description via a denoising process that generates a sequence of noisy images from  $\mathbf{X}_T$  down to  $\mathbf{X}_0$ , where  $\mathbf{X}_0$  represents the clean image, and  $\mathbf{X}_t$  represents the image corrupted with Gaussian noise. The noise is larger with a greater  $t$ , and with sufficiently large  $T$ ,  $\mathbf{X}_T$  approaches pure Gaussian noise. The denoising process can be formulated as follows:

$$q(\mathbf{X}_{0:T}|\mathbf{C}) = q(\mathbf{X}_T) \prod_{t=0}^{T-1} q(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C}), \quad (1)$$

where  $q(\mathbf{X}_T)$  follows the pure Gaussian distribution. Different denoising algorithms would induce different transitional probabilities  $q(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C})$ . Since our work does not rely on specific choices of denoising algorithms, we will leave  $q(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C})$  abstract throughout this section.

In the following, we will use  $p(\mathbf{X}_t|\mathbf{C})$  to denote the ground-truth distribution, *i.e.*, real images corrupted with Gaussian noise, and  $q(\mathbf{X}_t|\mathbf{C})$  as the distribution produced by the actual denoising network. Due to the learning capacity of the denoising network and approximation errors,  $q(\mathbf{X}_t|\mathbf{C})$  can differ from  $p(\mathbf{X}_t|\mathbf{C})$ .

### 3.2. Problem Formulation

The general goal of this paper is to reduce the gap between  $q(\mathbf{X}_t|\mathbf{C})$  and  $p(\mathbf{X}_t|\mathbf{C})$ , with the help of some external guidance, in order to address the following two prominent diffusion generation errors.

- **Missing Object Errors:** The generated images sometimes miss the objects mentioned in the text input. Formally, we introduce two bag-of-object variables. One is the *object mention* variable, denoted as  $\mathbf{O}_C$ , whose  $i$ -th element,  $O_{Ci}$ , equals one if object  $i$  is mentioned in the input text  $\mathbf{C}$  and zero otherwise. The other is the *object occurrence* variable, denoted as  $\mathbf{O}_X$ , whose  $i$ -th element,  $O_{Xi}$ , equals one if object  $i$  occurs in the image  $\mathbf{X}_0$  and zero otherwise. Then the missing object errors refer to the case where  $O_{Ci} = 1$  but  $O_{Xi} = 0$ .
- **Low Image Quality:** The generated images can sometimes suffer from unnatural distortion and texture. The low image quality problem can be ascribed as the general distribution gap between  $p(\mathbf{X}_0)$  and  $q(\mathbf{X}_0)$ .

We consider the following two types of external guidance to correct the generation errors.

- An **object detector** which can predict the probability that a certain object appears in the image, *i.e.*  $\hat{p}(O_{Xi} = 1|\mathbf{X}_0)$ , which provides useful information in correcting the missing object errors.
- A small set of **real images**,  $\mathcal{D}$ , either with or without text captions, which is useful in gauging the distribution gap.

In the following, we will develop different methods under different availability of external guidance.

### 3.3. Initial Exploration: A Naive Approach

As an initial exploration, we first focus on the subproblem that uses the object detector to correct the missing object errors. We will start with a naive sample selection approach, consisting of two steps. First, we use the diffusion model to generate  $K$  samples,  $\bigcup_{k=1}^K \{\mathbf{x}_0^{(k)}\}$ . Second, we simply select the image with the best object occurrence probability of the objects that are mentioned in the text, *i.e.*,

$$\max_{\mathbf{x}_0^{(k)}} \prod_{i: O_{Ci}=1} \hat{p}(O_{Xi} = 1|\mathbf{X}_0 = \mathbf{x}_0^{(k)}). \quad (2)$$

We dub the approach as **OBJECTSELECT**, whose implementation details are elaborated in Appendix C.

We run **OBJECTSELECT** on two benchmark datasets (GPT-Synthetic [62] and MS-COCO [36]) with complex text descriptions, together with four baselines that also use object detectors or other object occurrence feedback to improve the object occurrence of the generated images. The object occurrence ratios of all the methods are shown in Figure 4. As can be observed, **OBJECTSELECT** can significantly outperform the other baselines, some of which are much more complicated. More details of this experiment can be found in Section 4.1.

Admittedly, despite the effectiveness of **OBJECTSELECT**, this algorithm comes with obvious limitations. First, it only addresses the object occurrence issue and does not aim to approach the ground-truth conditional distribution  $p(\mathbf{X}_0|\mathbf{C})$ , so the selected images may compromise in quality. Second, the success of this approach relies on generating a sufficiently large number of samples to be able to find a few with good object occurrence, so the sample efficiency is low. However, this experiment does provide us with an important insight: Performing sampling on multiple generation paths of diffusion models turns out to be a direct and effective way of modifying the generation distribution of diffusion models. Similar observations are also made in previous works [30, 46, 52]. If **OBJECTSELECT** can already do so well, can we design an even more effective sampling-based algorithm that overcomes its limitations and can simultaneously correct for the missing object errors and improve image quality?

### 3.4. A Particle Filtering Framework

Particle filtering has long been an effective Monte-Carlo sampling framework for approximately generating random samples from a target distribution [12, 16], and has been successfully applied to diffusion models [58, 60]. Inspired by this, we will explore how to incorporate the external guidance into the particle filtering framework, to achieve a target distribution of  $p(\mathbf{X}_0|\mathbf{C})$ .

Our particle filtering framework obtains samples of  $\mathbf{X}_{0:T}$  in reverse order. It first generates  $K$  samples of  $\mathbf{X}_T$ , denoted as  $\{\mathbf{x}_T^{(k)}\}$ , conditional on which  $K$  samples of  $\mathbf{X}_{T-1}$  are

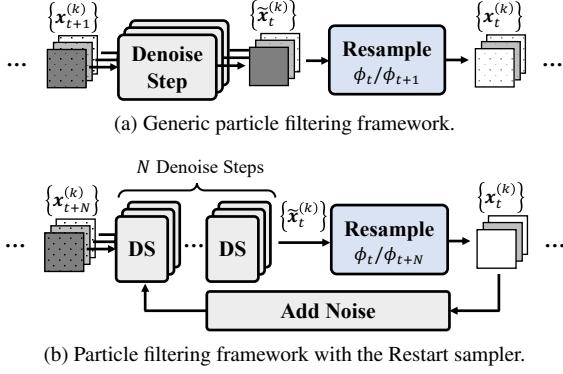


Figure 2. Illustration of our particle filtering framework.

derived. This process continues until  $K$  samples of  $\mathbf{X}_0$  are obtained. Specifically, samples of  $\mathbf{X}_t$ ,  $\{\mathbf{x}_t^{(k)}\}$ , are derived from samples of  $\mathbf{X}_{t+1}$ ,  $\{\mathbf{x}_{t+1}^{(k)}\}$ , via the following two steps.

**Step 1: Proposal.** For each sample  $\mathbf{x}_{t+1}^{(k)}$ , propose a sample of  $\mathbf{X}_t$  based on a proposal distribution  $r(\mathbf{X}_t|\mathbf{X}_{t+1} = \mathbf{x}_{t+1}^{(k)}, \mathbf{C})$ . This sample is denoted as  $\tilde{\mathbf{x}}_t^{(k)}$ .

**Step 2: Resampling.** Given all the sample pairs  $\bigcup_{k=1}^K \{(\mathbf{x}_{t+1}^{(k)}, \tilde{\mathbf{x}}_t^{(k)})\}$ , resample from this set  $K$  times (with replacement) with a probability proportional to a weight function,  $w(\mathbf{x}_{t+1}^{(k)}, \tilde{\mathbf{x}}_t^{(k)}|\mathbf{C})$ , and then only keep the latter in each sample pair. The resulting samples become  $\{\mathbf{x}_t^{(k)}\}$ .

Therefore, the key to the design of a particle filtering algorithm involves designing 1) the proposal distribution  $r(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C})$ , and 2) the resampling weight  $w(\mathbf{X}_{t+1}, \mathbf{X}_t|\mathbf{C})$ . Consistent with the design principles in Wu et al. [60], we adopt the following design:

$$\begin{aligned} r(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C}) &= q(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C}), \\ w(\mathbf{X}_{t+1}, \mathbf{X}_t|\mathbf{C}) &= \frac{\phi_t(\mathbf{X}_t|\mathbf{C})}{\phi_{t+1}(\mathbf{X}_{t+1}|\mathbf{C})}. \end{aligned} \quad (3)$$

The first line essentially means that the proposal step becomes the denoising step of the diffusion model. Denote the distribution that  $\{\mathbf{x}_t^{(k)}\}$  follows as  $v(\mathbf{X}_t|\mathbf{C})$ . Then it can be easily shown that, if we follow the design in Eq. (3),

$$v(\mathbf{X}_t|\mathbf{C}) = q(\mathbf{X}_t|\mathbf{C})\phi_t(\mathbf{X}_t|\mathbf{C}). \quad (4)$$

The proof is provided in Appendix A. In other words,  $\phi_t(\mathbf{X}_t|\mathbf{C})$  can be interpreted as a correction term that modifies the distribution of  $\mathbf{X}_t$ . If we could set

$$\phi_t(\mathbf{X}_t|\mathbf{C}) = \frac{p(\mathbf{X}_t|\mathbf{C})}{q(\mathbf{X}_t|\mathbf{C})}, \quad (5)$$

then we can ideally achieve  $v(\mathbf{X}_t|\mathbf{C}) = p(\mathbf{X}_t|\mathbf{C}), \forall t$ , which means the final generated image will follow the real image distributions. Figure 2a illustrates the overall process.

Now the question is how to compute the conditional likelihood ratio in Eq. (5). Sections 3.5 and 3.6 introduce two methods with different requirements on external guidance.

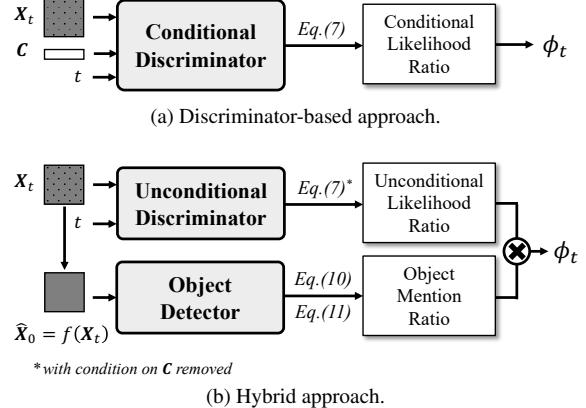


Figure 3. Calculation of the correction term  $\phi_t(\mathbf{X}_t|\mathbf{C})$ .

### 3.5. A Discriminator-Based Approach

Our first method, shown in Figure 3a, only utilizes a small set of real images with captions as the external guidance. It involves training a discriminator that discriminates between real and generated examples. Specifically, given a text description  $C$ , real samples of  $\mathbf{X}_t$  are obtained by corrupting real images that correspond to  $C$  with Gaussian noise; fake samples of  $\mathbf{X}_t$  are obtained by first generating images using the diffusion model with  $C$  as the text condition, and then corrupting the generated images with Gaussian noise.

Given the real and fake samples, we train a *conditional* discriminator  $d(\mathbf{X}_t|\mathbf{C}; t)$  by minimizing the canonical discrimination loss

$$\mathcal{L} = \mathbb{E}_{\mathbf{C}, t} [\mathbb{E}_{\mathbf{X}_t \sim p(\mathbf{X}_t|\mathbf{C})} [-\log d(\mathbf{X}_t|\mathbf{C}; t)] + \mathbb{E}_{\mathbf{X}_t \sim q(\mathbf{X}_t|\mathbf{C})} [-\log(1 - d(\mathbf{X}_t|\mathbf{C}; t))]]. \quad (6)$$

It has been shown [21] that the minimizer of (6), denoted as  $d^*(\mathbf{X}_t|\mathbf{C}; t)$ , can be used to compute the conditional likelihood ratio as:

$$\frac{p(\mathbf{X}_t|\mathbf{C})}{q(\mathbf{X}_t|\mathbf{C})} \approx \frac{d^*(\mathbf{X}_t|\mathbf{C}; t)}{1 - d^*(\mathbf{X}_t|\mathbf{C}; t)}. \quad (7)$$

Theoretically, the discriminator-based approach can correct any distributional discrepancies between real and generated images. However, due to the limitation in representation power and optimization schemes, the discriminator may fail to capture certain errors, such as the missing object errors. Next, we will introduce an alternative approach with a more fine-grained correction of different error types.

### 3.6. A Hybrid Approach

The hybrid approach (in Figure 3b) uses both the object detector and real image set as the external guidance to address the missing object errors and poor image quality respectively. Formally, according to the Bayes rule, we decom-

pose the conditional likelihood ratio in Eq. (5) as follows:

$$\begin{aligned} \frac{p(\mathbf{X}_t|\mathbf{C})}{q(\mathbf{X}_t|\mathbf{C})} &\stackrel{\textcircled{1}}{=} \frac{p(\mathbf{X}_t|\mathbf{C}, \mathbf{O}_C)}{q(\mathbf{X}_t|\mathbf{C}, \mathbf{O}_C)} \\ &= \frac{q(\mathbf{C}, \mathbf{O}_C)}{p(\mathbf{C}, \mathbf{O}_C)} \cdot \frac{p(\mathbf{X}_t)}{q(\mathbf{X}_t)} \cdot \frac{p(\mathbf{O}_C|\mathbf{X}_t)}{q(\mathbf{O}_C|\mathbf{X}_t)} \cdot \frac{p(\mathbf{C}|\mathbf{O}_C, \mathbf{X}_t)}{q(\mathbf{C}|\mathbf{O}_C, \mathbf{X}_t)}. \end{aligned} \quad (8)$$

Equality ① is due to the fact that the information in  $\mathbf{O}_C$ , by definition, comes entirely from  $\mathbf{C}$ . The right-hand side of Eq. (8) consists of four terms. The first term is about the prior distribution of text conditions, which should equal one because the text prior is not impacted by the denoising process. The second term addresses the general distributional discrepancies in images, which accounts for the *low image quality* issue. The third term accounts for the missing object errors. The fourth term accounts for any other inconsistencies between the image and text, such as inconsistent object characteristics, mispositioned objects, etc. Since our current focus is on correcting missing object errors and improving image quality, we will focus on estimating the second and third terms, which we will refer to as the *unconditional likelihood ratio* and *object mention ratio* respectively.

**Estimating the unconditional likelihood ratio.** The unconditional likelihood ratio can be estimated the same way as in Section 3.5, except that the discriminator should be replaced with an unconditional one without  $\mathbf{C}$ , hence denoted as  $d(\mathbf{X}_t; t)$ . Accordingly, we can replace the discriminator in Eq. (7) with  $d^*(\mathbf{X}_t; t)$  to compute the unconditional likelihood ratio. As a result, this hybrid approach does not require that the image set comes with captions.

**Estimating the object mention ratio.** Assuming conditional independence of different dimensions of  $\mathbf{O}_C$ , the object mention ratio can be further factorized as follows:

$$\begin{aligned} \frac{p(\mathbf{O}_C|\mathbf{X}_t)}{q(\mathbf{O}_C|\mathbf{X}_t)} &= \frac{\prod_i p(O_{Ci}|\mathbf{X}_t)}{\prod_i q(O_{Ci}|\mathbf{X}_t)} \\ &= \frac{\prod_{i:O_{Ci}=1} p(O_{Ci} = 1|\mathbf{X}_t)}{\prod_{i:O_{Ci}=1} q(O_{Ci} = 1|\mathbf{X}_t)} \cdot \frac{\prod_{i:O_{Ci}=0} p(O_{Ci} = 0|\mathbf{X}_t)}{\prod_{i:O_{Ci}=0} q(O_{Ci} = 0|\mathbf{X}_t)}. \end{aligned} \quad (9)$$

Essentially, Eq. (9) divides the objects into two groups, ones that are mentioned in the caption, and ones that are not. So the first term corrects for the *missing object errors* and the second term corrects for the *false generation* (i.e., generating objects that are not mentioned in the caption). Since the main concern of diffusion generation is the former, we will focus on computing the first term.

The numerator,  $p(O_{Ci} = 1|\mathbf{X}_t)$ , measures, pretending that the noisy image  $\mathbf{X}_t$  were corrupted from a real image and that the real image had a caption, what is the probability that the caption would mention object  $i$ . Since the caption almost always reflects what is in the image, this probability can be easily approximated by running the object detector on the predicted clean image from  $\mathbf{X}_t$ . Formally, denote  $f(\mathbf{X}_t)$  as the one-step prediction of the clean image by the

diffusion model, then

$$p(O_{Ci} = 1|\mathbf{X}_t) \approx \hat{p}(O_{Xi} = 1|f(\mathbf{X}_t)). \quad (10)$$

A more rigorous derivation is provided in Appendix B.

The denominator,  $q(O_{Ci} = 1|\mathbf{X}_t)$ , measures, pretending that the noisy image  $\mathbf{X}_t$  were generated by the (imperfect) diffusion model conditional on an input description  $\mathbf{C}$ , what is the probability that the input text had mentioned object  $i$ . Unlike the numerator, if an object  $i$  does not appear in the image, there is still a non-zero chance that the caption had mentioned the object  $i$ , considering the diffusion model may miss the object. As formally derived in Appendix B, this probability can be approximated as

$$\begin{aligned} q(O_{Ci} = 1|\mathbf{X}_t) &\approx \hat{p}(O_{Xi} = 1|f(\mathbf{X}_t)) \\ &\quad + \hat{p}(O_{Xi} = 0|f(\mathbf{X}_t)) \frac{(1 - \kappa_{it})\pi_{it}}{(1 - \kappa_{it})\pi_{it} + 1 - \pi_{it}}, \end{aligned} \quad (11)$$

where  $\pi_{it}$  is a hyper-parameter between 0 and 1; and

$$\kappa_{it} = \frac{\mathbb{E}_{\mathbf{X}_t, \mathbf{C} \sim q(\mathbf{X}_t, \mathbf{C})} [\mathbb{1}(\hat{O}_{Xit} = 1 \wedge O_{Ci} = 1)]}{\mathbb{E}_{\mathbf{C} \sim q(\mathbf{C})} [\mathbb{1}(O_{Ci} = 1)]} \quad (12)$$

denotes the percentage of occurrence of object  $i$  in the image predicted from  $\mathbf{X}_t$  when the text description mentions object  $i$ .  $\hat{O}_{Xit}$  denotes the (estimated) occurrence of object  $i$  in the clean image predicted from  $\mathbf{X}_t$ , i.e.,  $f(\mathbf{X}_t)$ , which is computed by passing  $f(\mathbf{X}_t)$  to the object detector and then see whether the output exceeds the threshold 0.5.  $\mathbb{1}(\cdot)$  denotes the indicator function.

To compute  $\kappa_{it}$ , we would need to run an initial generation round, where we feed a set of text descriptions to the diffusion model (*without* particle filtering) and generate a set of images. The numerator of Eq. (12) is computed by counting how many text-image pairs mention object  $i$  in the text and the corresponding  $\hat{O}_{Xit}$  equals one; the denominator is computed by counting how many text descriptions mention object  $i$ . Although the initial generation round introduces extra computation, it only needs to perform once.

**Summary.** To sum up, in the hybrid approach,  $\phi_t(\mathbf{X}_t|\mathbf{C})$  is computed as

$$\phi_t(\mathbf{X}_t|\mathbf{C}) = \frac{p(\mathbf{X}_t)}{q(\mathbf{X}_t)} \cdot \frac{\prod_{i:O_{Ci}=1} p(O_{Ci} = 1|\mathbf{X}_t)}{\prod_{i:O_{Ci}=1} q(O_{Ci} = 1|\mathbf{X}_t)}, \quad (13)$$

where the first term is computed by training an unconditional discriminator (similar to Eq. (7)), and the second term is computed from Eqs. (10) and (11) using the object detector. Algs. 1 and 2 in Appendix describe the particle filtering algorithm and the calculation of  $\phi_t(\mathbf{X}_t|\mathbf{C})$  respectively.

### 3.7. Generalization to Other Generation Settings

The proposed algorithm can be applied beyond text-to-image generation to generic conditional and unconditional generations by setting  $\mathbf{C}$  to other conditions or an empty set respectively. In this case, the discriminator-based approach will be used as the hybrid approach is no longer applicable.

## 4. Experiments

In this section, we will first demonstrate the effectiveness of our method on reducing the missing object errors and improving the image quality in text-to-image generation. We then evaluate our method on standard benchmarks of unconditional and class-conditioned generation. Finally, we will investigate various design choices in our framework.

### 4.1. Text-to-Image Generation

**Datasets.** We use two datasets: (1) GPT-Synthetic was originally introduced in Wu et al. [62] to evaluate text-to-image models’ ability in generating correct objects and their associated colors and positions. It contains 500 text descriptions generated by GPT-3 [7]. Each description contains 2 to 5 objects in MS-COCO [36], and objects are associated with random colors and spatial relations with other objects, e.g., ‘*The tie was placed to the right of the red backpack.*’ (2) We also evaluate on the validation set of MS-COCO. However, we focus on the subset of complex descriptions that contain at least four objects, which results in 261 captions (please see details in Appendix C).

**Baselines.** We compare with seven baselines: ① SD, which is the Stable Diffusion model [48] (v2.1-base) as is; ② D-GUIDANCE [32] that modifies the score function by adding an additional term  $\nabla \log \frac{p(\mathbf{X}_t|C)}{q(\mathbf{X}_t|C)}$ , where we use the hybrid approach in Section 3.6 to estimate  $\phi_t(\mathbf{X}_t|C) = \frac{p(\mathbf{X}_t|C)}{q(\mathbf{X}_t|C)}$ ; ③ SPATIAL-TEMPORAL [62] and ④ ATTEND-EXCITE [9] that modify each denoising step to ensure each object in the caption is attended (please refer to Section 2 for details); ⑤ OBJECTSELECT described in Section 3.3; ⑥ TIFASELECT and ⑦ REWARDSELECT, both in [30], which are similar to ⑤ but use TIFA score [26] and ImageReward [64] as the selection criteria respectively. We use Stable Diffusion v2.1-base for our method and baselines if possible (only ③ does not support it, for which we use v1.5). For methods that leverage an object detector, we use DETR [8] with ResNet-50 backbone [22]. For all sample selection methods (⑤-⑦ and ours), we select the best image based on each method’s sampling criterion for evaluation.

**Metrics.** We evaluate generated images using both objective and subjective metrics. For objective evaluation, we use **Object Occurrence** to measure the percentage of objects that occur in generated images over all objects mentioned in the caption. We calculate the percentage for each caption and take the average over all captions in the dataset. We use an object detector (DETR with ResNet-101 backbone) different from the one used during generation. In addition, we also calculate Fréchet inception distance (**FID**) [24] as a measure for image quality. Similar to Xu et al. [65], we calculate FID on 5,000 captions in the MS-COCO validation set that contain at least one object. For subjective evaluation, we recruit annotators on Amazon Mechanical Turk

to compare images generated by our method and baselines. Annotators are asked to evaluate two aspects: (1) **(Object)** Given an image, annotators need to identify all objects in it. (2) **(Quality)** Given two images for the same caption, annotators need to select the one that looks more real and natural. We randomly sample 100 captions<sup>1</sup> from each dataset for evaluation, and each image is rated by two annotators.

**Diffusion samplers.** All the methods are evaluated on the Restart sampler [65]. The only exceptions are SPATIAL-TEMPORAL and ATTEND-EXCITE, which use the same samplers as in their papers. The Restart sampler iterates between three steps: ① *Denoise* from  $t+N$  to  $t$  using the ODE trajectory; ② *Restart* by adding the noise back to the level  $t+N$ . ③ *Denoise* from  $t+N$  to  $t$  again, and either go back to step ② or proceed to the next iteration which denoises to  $t-N'$ . As shown in Figure 2b, our methods are combined with the Restart sampler by inserting the resampling module right before adding noise. We also experimented with the SDE sampler in EDM [29], but we found Restart sampler generally dominates EDM on both object occurrence and image quality. We thus only focus on Restart sampler in the main paper. EDM results are shown in Appendix C.

**Sampling configurations.** To have a fair comparison in terms of the computation cost, we evaluate all sample selection methods (including ours) under a comparable number of function evaluations (NFE). Specifically, we generate each image with a fixed NFE and report performance when  $K = 5, 10, 15$  images are generated for each caption. This ensures a fair comparison for methods with the same value of  $K$ . For non-selection methods (SD, D-GUIDANCE, SPATIAL-TEMPORAL, and ATTEND-EXCITE), we use the original sampling configuration in their papers and thus only report a single performance on each dataset. Appendix C details the computation cost for all methods.

**Results.** Figure 4 shows object occurrence and FID for all methods. The performance of sample selection methods is reported for three different values of  $K$ , which are indicated by the sizes of the points in the figure. A competitive algorithm should achieve a high object occurrence rate (right) and low FID (bottom), so the more the algorithm lies to the *bottom-right corner*, the more competitive the algorithm is.

There are two observations from Figure 4. First, the sampling-based methods (represented by circles) generally significantly outperform the non-sampling-based ones (represented by crosses). In particular, although SPATIAL-TEMPORAL and ATTEND-EXCITE improve object occurrence, the improvements are not as large as other sampling-based methods, and their computational costs (as shown in Appendix C) are also high compared to the sampling-

<sup>1</sup>63 % of captions in MS-COCO dataset contain “person” as an object. To evaluate on a more uniform distribution of objects, we randomly sample 25 captions that contain person and 75 captions that do not.

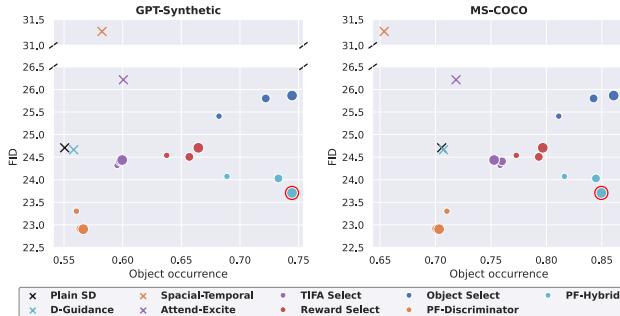


Figure 4. FID ( $\downarrow$ ) vs. Object occurrence ( $\uparrow$ ) for all methods. Ideal points should scatter at the bottom right corner. Object occurrence is measured on GPT-Synthetic (left) and MS-COCO (right), and FID is measured on MS-COCO.  $K = 5, 10, 15$  images are generated for sample selection methods, and the sizes of points indicate the value of  $K$  (larger  $K$  has larger points). The method that achieves the best combined performance is highlighted in red.

	GPT-Synthetic		MS-COCO	
	Object Occur ( $\uparrow$ )	Quality vs. PF-HYBRID ( $\uparrow$ )	Object Occur ( $\uparrow$ )	Quality vs. PF-HYBRID ( $\uparrow$ )
SD [48]	64.48	-6.00	59.87	-3.00
RWARDSELECT [30]	71.97	-3.00	63.25	-3.00
OBJECTSELECT	70.96	-11.00	67.35	-9.00
PF-DISCRIMINATOR	68.87	-8.00	59.98	-2.00
PF-HYBRID	75.79	-	68.13	-

Table 1. Human evaluation on object occurrence and image quality. ‘Quality’ is the win rate against PF-HYBRID (minus 50). Negative values indicate the method is worse than PF-HYBRID.

based methods. Second, our PF-HYBRID is the only algorithm that simultaneously achieves high object occurrence and low FID. OBJECT-SELECT achieves a high object occurrence, but the worst FID scores. The other proposed method, PF-DISCRIMINATOR, achieves a very low FID, but the object occurrence is significantly compromised, which verifies our claim that conditional discriminators tend to overlook object occurrence and focus only on image quality (Section 3.5). Notably, all three methods proposed in this paper, OBJECT-SELECT, PF-DISCRIMINATOR and -HYBRID lie at the frontier of the performance trade-off, significantly outperforming the existing baselines.

Table 1 shows the results for subjective evaluations, where object occurrence is computed the same way as the objective object occurrence. The quality score is computed as follows. Since PF-HYBRID is the most competitive algorithm, we perform a pairwise comparison between PF-HYBRID and each baseline. The baseline gets one point everytime a subject prefers the baseline over PF-HYBRID. Each comparison consists of 100 pairs so a score of 50 indicates a tie. We subtract all the scores by 50 so a negative score in each baseline indicates that PF-HYBRID is better. As can be observed, PF-HYBRID outperforms all the baselines in terms of object occurrence and subjective quality.

Figure 5 visualizes some generated images by our

method and baselines. As can be observed, the restart sampler can generate natural images but tends to miss objects mentioned in the text. OBJECTSELECT reduces the missing object errors but could lead to unnatural objects in the image (e.g., knife and fork). PF-HYBRID generates more complete objects and also improves image quality.

## 4.2. Unconditional & Class-conditioned Generation

**Experiment setup.** In addition to text-to-image, we evaluate two other image generation benchmarks, FFHQ [28] for unconditional generation and ImageNet-64 [14] for class-conditioned generation. In this case, object occurrence is not applicable so we will focus on evaluating image quality as measured by FID. We will only implement PF-DISCRIMINATOR because PF-HYBRID is not applicable (Sectoin 3.7). We use the pre-trained diffusion models in Karras et al. [29] and follow Kim et al. [32] to train the discriminator (details in Appendix D). For evaluation, we calculate FID on 50,000 generated images.

**Baselines and method variants.** We consider three baselines: ① The original Restart sampler; ② D-GUIDANCE using the discriminator to compute the correct term (following [32]); ③ D-SELECT, which is similar to OBJECTSELECT but uses the likelihood ratio  $\frac{p(\mathbf{X}_0|C)}{q(\mathbf{X}_0|C)}$  as the selection criterion (effectively an important sampling approach to restore the ground-truth distribution).

**Sampling configurations.** Similar to Section 4.1, we evaluate PF and D-SELECT for different values of  $K$  with a fixed NFE per image. For a fair comparison, we increase denoising steps for the original sampler and D-GUIDANCE to match the total NFE of the sampling-based methods.

**Results.** Figure 6 shows FID as a function of overall NFE. For PF and D-SELECT,  $K = 2, 4, 6$  images are generated.

There are two observations. First, all the methods generally improve as NFE increases, showing the effectiveness of increasing samples (for sampling-based methods) or increasing denoising steps (for non-sampling-based methods). Second, D-GUIDANCE generally performs competitively, especially at small NFEs, but our method consistently achieves the lowest FID across two datasets at large NFEs. Note that for our method, we only vary the number of particles as the only way to vary the NFE. As shown in Appendix E, our method would achieve an even larger performance gain at high NFEs if we vary the number of denoising steps and the number of particles simultaneously.

## 4.3. Ablation Study

We now investigate two important aspects of our method in text-to-image generation setting: the particle filtering algorithm and the resampling weight calculation. For each ablated version, we will evaluate on the object occurrence on GPT-Synthetic and MS-COCO, and FID on the latter.

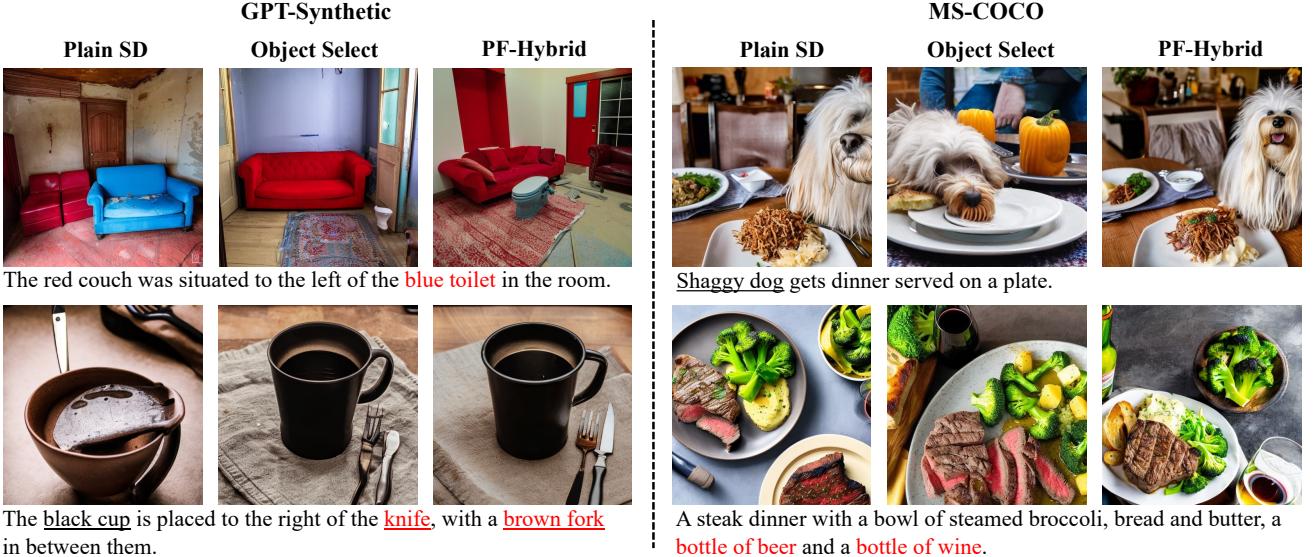


Figure 5. Visualization of generated samples. Missing objects are highlighted in red. Unnatural objects are highlighted with underline.

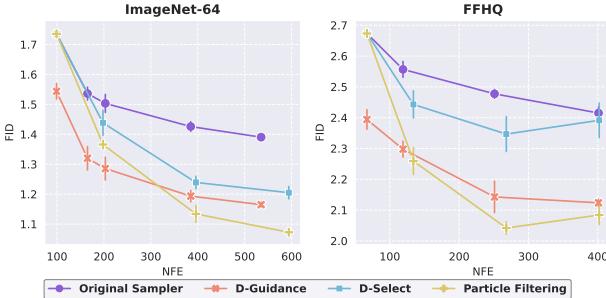


Figure 6. FID (average of 3 runs) on ImageNet-64 (left) and FFHQ (right). Error bars indicate standard deviations.

	Object Occurrence (%) ↑		FID ↓
	GPT-Syn	MS-COCO	
PF-HYBRID	72.96	83.84	24.03
- PF	67.16	80.49	24.18
- Discriminator	75.67	85.79	25.77
PF-DISCRIMINATOR	56.86	70.96	22.91
- PF	57.09	72.13	23.28
Object only	63.15	77.81	25.31

Table 2. Ablation study on the effects of particle filtering algorithm and particle weights design.

First, we investigate the effects of **removing particle filtering**. To do that, we compare PF-HYBRID and PF-DISCRIMINATOR with two approaches that do not involve particle filtering but only generate  $K$  images via the regular Restart sampler and select the best one with the largest value of  $\phi_0(\mathbf{X}_0|\mathcal{C})$ , calculated the same way as PF-HYBRID and PF-DISCRIMINATOR respectively. Table 2 shows the performance of these two methods (rows ‘–PF’). As can be observed, removing PF hurts the performance of PF-HYBRID on all metrics, demonstrating the importance of

the particle filtering process. Removing PF hurts FID for PF-DISCRIMINATOR but does not affect object occurrence, which indicates the conditional discriminator focuses on image quality instead of object occurrence.

Second, we investigate the design choices of the **resampling weight**. We explore two variants for estimating  $\phi_t(\mathbf{X}_t|\mathcal{C})$ . For PF-HYBRID, we remove its unconditional discriminator and only include the object mention ratio (Eq. (13)). For PF-DISCRIMINATOR, since we have observed that the discriminator tends to ignore missing objects, we introduce a variant to force the discriminator’s attention on missing objects by training the discriminator on the object detector’s output probability as the feature (instead of on the images). The corresponding results in Table 2 (rows ‘–Discriminator’ and ‘Object only’ respectively) show that these two variants manage to improve object occurrence, but at the cost of higher FIDs, which highlights the needs of balancing the two objectives.

We further study other aspects in class-conditioned generation, including when the resampling is performed (*i.e.*, before or after adding noise), the amount of noise being added, and the number of denoising steps per image. Notably, our PF method with Restart sampler achieves the state-of-the-art FID of 1.02 on ImageNet-64 when 4 images are generated with 165 NFE. The results of these experiments are presented in Appendix E.

## 5. Conclusion

In this paper, we proposed a sampling-based approach to correct the errors in diffusion generation and reduce the discrepancies between model-generated and real data distributions. Our method utilizes the particle filtering algorithm to approximately sample from the ground-truth distribution.

We further leverage a set of external guidance to estimate the resampling weights. Experiments show that our method effectively corrects the missing objects and low-quality errors in text-to-image generation. It also improves the image quality for unconditional and class-conditioned generation.

## 6. Acknowledgements

The work of Yujian Liu and Shiyu Chang was partially supported by National Science Foundation (NSF) Grant IIS-2207052 and IIS-2302730. TJ acknowledges support from the MIT-IBM Watson AI Lab.

## References

- [1] Brian. D. O. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12: 313–326, 1982. 14
- [2] Anonymous. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. In *Submitted to The Twelfth International Conference on Learning Representations*, 2023. 2
- [3] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1
- [4] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Qinsheng Zhang, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, Tero Karras, and Ming-Yu Liu. ediff-i: Text-to-image diffusion models with an ensemble of expert denoisers, 2023. 2
- [5] Dmitry Baranchuk, Andrey Voynov, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. In *International Conference on Learning Representations*, 2022. 15
- [6] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions, 2023. 1
- [7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, 2020. 6
- [8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers, 2020. 6
- [9] Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models, 2023. 2, 6, 16
- [10] Minghao Chen, Iro Laina, and Andrea Vedaldi. Training-free layout control with cross-attention guidance, 2023.
- [11] Jaemin Cho, Abhay Zala, and Mohit Bansal. Dall-eval: Probing the reasoning skills and social biases of text-to-image generation models, 2023. 2
- [12] Nicolas Chopin and Omiros Papaspiliopoulos. *An introduction to sequential Monte Carlo*. Springer, 2020. 2, 3
- [13] Gabriele Corso, Yilun Xu, Valentin de Bortoli, Regina Barzilay, and Tommi Jaakkola. Particle guidance: non-i.i.d. diverse sampling with diffusion models, 2023. 2
- [14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009. 7
- [15] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021. 1, 16
- [16] Arnaud Doucet, Nando De Freitas, and Neil Gordon. *Sequential Monte Carlo Methods in practice*. Springer, 2011. 2, 3
- [17] Yilun Du, Conor Durkan, Robin Strudel, Joshua B. Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc, 2023. 2
- [18] Weixi Feng, Xuehai He, Tsu-Jui Fu, Varun Jampani, Arjun Reddy Akula, Pradyumna Narayana, Sugato Basu, Xin Eric Wang, and William Yang Wang. Training-free structured diffusion guidance for compositional text-to-image synthesis. In *The Eleventh International Conference on Learning Representations*, 2023. 2
- [19] Weixi Feng, Wanrong Zhu, Tsu jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. Layoutgpt: Compositional visual planning and generation with large language models, 2023. 2
- [20] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion, 2022. 1
- [21] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. 4
- [22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 6
- [23] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control, 2022. 1
- [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. In *Advances in Neural Information Processing Systems*, 2017. 6
- [25] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. 1
- [26] Yushi Hu, Benlin Liu, Jungo Kasai, Yizhong Wang, Mari Ostendorf, Ranjay Krishna, and Noah A Smith. Tifa: Accurate and interpretable text-to-image faithfulness evaluation with question answering, 2023. 6
- [27] Junoh Kang, Jinyoung Choi, Sungik Choi, and Bohyung Han. Observation-guided diffusion probabilistic models, 2023. 2

- [28] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2019. 7
- [29] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, 2022. 1, 6, 7, 14, 16, 17
- [30] Shyamgopal Karthik, Karsten Roth, Massimiliano Mancini, and Zeynep Akata. If at first you don't succeed, try, try again: Faithful diffusion-based text-to-image generation by selection, 2023. 2, 3, 6, 7, 14, 16
- [31] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models, 2023. 1
- [32] Dongjun Kim, Yeongmin Kim, Se Jung Kwon, Wanmo Kang, and Il-Chul Moon. Refining generative process with discriminator guidance in score-based diffusion models, 2023. 1, 2, 6, 7, 14, 16
- [33] Mingi Kwon, Jaeseok Jeong, and Youngjung Uh. Diffusion models already have a semantic latent space. In *The Eleventh International Conference on Learning Representations*, 2023. 15
- [34] Kimin Lee, Hao Liu, Moonkyung Ryu, Olivia Watkins, Yuqing Du, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, and Shixiang Shane Gu. Aligning text-to-image models using human feedback, 2023. 2
- [35] Long Lian, Boyi Li, Adam Yala, and Trevor Darrell. Llm-grounded diffusion: Enhancing prompt understanding of text-to-image diffusion models with large language models, 2023. 2
- [36] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 2, 3, 6
- [37] Luping Liu, Zijian Zhang, Yi Ren, Rongjie Huang, Xiang Yin, and Zhou Zhao. Detector guidance for multi-object text-to-image generation, 2023. 2
- [38] Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B. Tenenbaum. Compositional visual generation with composable diffusion models, 2023. 2
- [39] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps, 2022. 14
- [40] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models, 2022. 1
- [41] Wan-Duo Kurt Ma, J. P. Lewis, Avisek Lahiri, Thomas Leung, and W. Bastiaan Kleijn. Directed diffusion: Direct control of object placement through attention guidance, 2023. 2
- [42] Gary Marcus, Ernest Davis, and Scott Aaronson. A very preliminary analysis of dall-e 2, 2022. 2
- [43] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jianjun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations, 2022. 1
- [44] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, Ying Shan, and Xiaohu Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models, 2023.
- [45] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models, 2022. 1
- [46] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021. 3
- [47] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022. 1
- [48] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1, 2, 6, 7, 14, 16
- [49] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. 15
- [50] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation, 2023. 1
- [51] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022. 1
- [52] Dvir Samuel, Rami Ben-Ari, Simon Raviv, Nir Darshan, and Gal Chechik. Generating images of rare concepts using pre-trained diffusion models, 2023. 3
- [53] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. Laion-5b: An open large-scale dataset for training next generation image-text models, 2022. 14
- [54] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2256–2265, 2015. 1
- [55] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.
- [56] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. 14
- [57] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models, 2023. 1
- [58] Brian L. Trippe, Jason Yim, Doug Tischer, David Baker, Tamara Broderick, Regina Barzilay, and Tommi S. Jaakkola.

- Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem. In *The Eleventh International Conference on Learning Representations*, 2023. 2, 3
- [59] Ruichen Wang, Zekang Chen, Chen Chen, Jian Ma, Haonan Lu, and Xiaodong Lin. Compositional text-to-image synthesis with attention map control of diffusion models, 2023. 2
  - [60] Luhuan Wu, Brian L. Trippe, Christian A. Naesseth, David M. Blei, and John P. Cunningham. Practical and asymptotically exact conditional sampling in diffusion models, 2023. 2, 3, 4
  - [61] Qiucheng Wu, Yujian Liu, Handong Zhao, Ajinkya Kale, Trung Bui, Tong Yu, Zhe Lin, Yang Zhang, and Shiyu Chang. Uncovering the disentanglement capability in text-to-image diffusion models, 2022. 1
  - [62] Qiucheng Wu, Yujian Liu, Handong Zhao, Trung Bui, Zhe Lin, Yang Zhang, and Shiyu Chang. Harnessing the spatial-temporal attention of diffusion models for high-fidelity text-to-image synthesis, 2023. 2, 3, 6, 16
  - [63] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans, 2022. 2
  - [64] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imageward: Learning and evaluating human preferences for text-to-image generation, 2023. 6
  - [65] Yilun Xu, Mingyang Deng, Xiang Cheng, Yonglong Tian, Ziming Liu, and Tommi Jaakkola. Restart sampling for improving generative processes, 2023. 1, 6, 14, 15
  - [66] Guanhua Zhang, Jiabao Ji, Yang Zhang, Mo Yu, Tommi Jaakkola, and Shiyu Chang. Towards coherent image inpainting using denoising diffusion implicit models, 2023. 1
  - [67] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023. 1

## A. Correctness of the Particle Filtering Framework

We now prove the correctness of our resampling weight design in the particle filtering algorithm. Denote the distribution that  $\{\mathbf{x}_t^{(k)}\}$  follows as  $v(\mathbf{X}_t|\mathbf{C})$ , based on the particle filtering process, it is easy to show the probability satisfies the following recursive relationship:

$$v(\mathbf{X}_t|\mathbf{C}) \propto \int v(\mathbf{X}_{t+1}|\mathbf{C})r(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C})w(\mathbf{X}_t, \mathbf{X}_{t+1}|\mathbf{C}) d\mathbf{X}_{t+1}, \quad (14)$$

where  $r(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C})$  is the proposal distribution and  $w(\mathbf{X}_t, \mathbf{X}_{t+1}|\mathbf{C})$  is the resampling weight.

Now consider  $r(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C}) = q(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C})$ , i.e., the diffusion model is used for proposal, and  $w(\mathbf{X}_t, \mathbf{X}_{t+1}|\mathbf{C}) = \frac{\phi(\mathbf{X}_t|\mathbf{C})}{\phi(\mathbf{X}_{t+1}|\mathbf{C})}$ . Suppose  $v(\mathbf{X}_{t+1}|\mathbf{C}) = q(\mathbf{X}_{t+1}|\mathbf{C})\phi(\mathbf{X}_{t+1}|\mathbf{C})$ , using the recursive relationship in Eq. (14), it can be shown that

$$\begin{aligned} v(\mathbf{X}_t|\mathbf{C}) &\propto \\ &\int v(\mathbf{X}_{t+1}|\mathbf{C})q(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C})w(\mathbf{X}_t, \mathbf{X}_{t+1}|\mathbf{C}) d\mathbf{X}_{t+1} \\ &= \int q(\mathbf{X}_{t+1}|\mathbf{C})\phi(\mathbf{X}_{t+1}|\mathbf{C})q(\mathbf{X}_t|\mathbf{X}_{t+1}, \mathbf{C})\frac{\phi(\mathbf{X}_t|\mathbf{C})}{\phi(\mathbf{X}_{t+1}|\mathbf{C})} d\mathbf{X}_{t+1} \\ &= \int q(\mathbf{X}_t, \mathbf{X}_{t+1}|\mathbf{C})\phi(\mathbf{X}_t|\mathbf{C}) d\mathbf{X}_{t+1} \\ &= q(\mathbf{X}_t|\mathbf{C})\phi(\mathbf{X}_t|\mathbf{C}). \end{aligned} \quad (15)$$

At time step  $T$ , let  $\phi(\mathbf{X}_T|\mathbf{C}) = 1$ , and thus  $v(\mathbf{X}_T|\mathbf{C}) = q(\mathbf{X}_T|\mathbf{C})\phi(\mathbf{X}_T|\mathbf{C})$ . Therefore, by mathematical induction,  $v(\mathbf{X}_t|\mathbf{C}) = q(\mathbf{X}_t|\mathbf{C})\phi(\mathbf{X}_t|\mathbf{C})$  for all  $t = 0, \dots, T$ .

Algorithm 1 describes the overall procedure of our particle filtering framework.

## B. Resampling Weight Calculation

In this section, we provide the detailed process for calculating resampling weights, and more specifically, the correction term  $\phi(\mathbf{X}_t|\mathbf{C})$ . Algorithm 2 describes the overall procedure. We will mainly focus on the hybrid approach. For the discriminator-based approach, please refer to Section 3.5.

Recall that in the hybrid approach,  $\phi(\mathbf{X}_t|\mathbf{C})$  consists of two terms: unconditional likelihood ratio  $\frac{p(\mathbf{X}_t)}{q(\mathbf{X}_t)}$  and object mention ratio  $\frac{p(O_{Ci}|\mathbf{X}_t)}{q(O_{Ci}|\mathbf{X}_t)}$ . The unconditional likelihood ratio can be estimated with an unconditional discriminator as in Eq.(7). We will next elaborate on how to estimate the object mention ratio.

**Estimating object mention ratio.** As shown in Eq. (9), the object mention ratio can be decomposed as

$$\frac{\prod_{i:O_{Ci}=1} p(O_{Ci}=1|\mathbf{X}_t)}{\prod_{i:O_{Ci}=1} q(O_{Ci}=1|\mathbf{X}_t)} \cdot \frac{\prod_{i:O_{Ci}=0} p(O_{Ci}=0|\mathbf{X}_t)}{\prod_{i:O_{Ci}=0} q(O_{Ci}=0|\mathbf{X}_t)},$$

where we focus on the first term that corresponds to the missing object errors.

To compute  $p(O_{Ci}=1|\mathbf{X}_t)$ , notice that under  $p$  distribution,  $O_{Ci} = O_{Xi}$ , because, by definition, a text caption is constructed to mention the objects that appear in the corresponding real image. Therefore,

$$\begin{aligned} p(O_{Ci}=1|\mathbf{X}_t) &= p(O_{Xi}=1|\mathbf{X}_t) \\ &\stackrel{\textcircled{1}}{=} \mathbb{E}_{\mathbf{X}_0 \sim p(\mathbf{X}_0|\mathbf{X}_t)} [p(O_{Xi}=1|\mathbf{X}_0)] \quad (16) \\ &\stackrel{\textcircled{2}}{\approx} \hat{p}(O_{Xi}=1|f(\mathbf{X}_t)), \end{aligned}$$

where equality  $\textcircled{1}$  is by simple chain rule and the fact that  $p(O_{Xi}|\mathbf{X}_0, \mathbf{X}_t) = p(O_{Xi}|\mathbf{X}_0)$  because  $O_{Xi}$  is defined as the object occurrence in the clean image  $\mathbf{X}_0$ , and hence is conditionally independent of  $\mathbf{X}_t$ . For  $\textcircled{2}$ , we make two approximations. First, we replace drawing a sample of  $\mathbf{X}_0$  from  $p(\mathbf{X}_0|\mathbf{X}_t)$  with the minimum mean squared error (MMSE) estimate of  $\mathbf{X}_0$  from  $\mathbf{X}_t$ , denoted as  $f(\mathbf{X}_t)$ , which can be conveniently and efficiently predicted by the denoising network via one-step generation of  $\mathbf{X}_0$  from  $\mathbf{X}_t$ . Second, we approximate the true object occurrence probability  $p(O_{Xi}|\mathbf{X}_0)$  with the one estimated by the object detector  $\hat{p}(O_{Xi}|\mathbf{X}_0)$ .

On the other hand,  $q(O_{Ci}=1|\mathbf{X}_t)$  can be most intuitively computed by training a neural network to predict object mentions in the input text from the generated samples. However, this approach is inefficient and would need to retrain the network everytime the denoising scheme changes. We would like to derive a more efficient alternative taking advantage of the object detector. Formally (notice that it is no longer true that  $O_{Ci} = O_{Xi}$ ),

$$\begin{aligned} q(O_{Ci}=1|\mathbf{X}_t) &= \mathbb{E}_{O_{Xit} \sim q(O_{Xit}|\mathbf{X}_t)} [q(O_{Ci}=1|O_{Xit}, \mathbf{X}_t)] \\ &= q(O_{Xit}=0|\mathbf{X}_t)q(O_{Ci}=1|O_{Xit}=0, \mathbf{X}_t) + \\ &\quad q(O_{Xit}=1|\mathbf{X}_t)q(O_{Ci}=1|O_{Xit}=1, \mathbf{X}_t), \end{aligned} \quad (17)$$

where  $O_{Xit}$  denotes the occurrence of object  $i$  in the clean image predicted from  $\mathbf{X}_t$ , i.e.,  $f(\mathbf{X}_t)$ . Similar to Eq. (16), the probability  $q(O_{Xit}|\mathbf{X}_t)$  can be estimated by  $\hat{p}(O_{Xi}|f(\mathbf{X}_t))$ . Therefore, what makes Eq. (17) different is the additional term  $q(O_{Xit}=0|\mathbf{X}_t)q(O_{Ci}=1|O_{Xit}=0, \mathbf{X}_t)$ , which corresponds to the case where even if the object detector fails to detect the object  $i$ , there is still chance that the caption used to generate the image contains object  $i$ , considering the imperfect diffusion model that may miss objects mentioned in the caption.

To calculate  $q(O_{Ci}=1|O_{Xit}, \mathbf{X}_t)$ , we further make an assumption that object occurrence  $O_{Xit}$  is a sufficient statistic to predict the object mention of the same object. Hence  $q(O_{Ci}=1|O_{Xit}, \mathbf{X}_t) = q(O_{Ci}=1|O_{Xit})$ . According to the Bayes rule,

$$q(O_{Ci}=1|O_{Xit}) = \frac{q(O_{Xit}|O_{Ci}=1)q(O_{Ci}=1)}{\sum_{O_{Ci} \in \{0,1\}} q(O_{Xit}|O_{Ci})q(O_{Ci})}. \quad (18)$$

---

**Algorithm 1** Particle Filtering Framework for Correcting Diffusion Generation

---

1: **Input:**  
 2: - Diffusion model with denosing distribution  $q(\mathbf{X}_t | \mathbf{X}_{t+1}, \mathbf{C})$ , condition signal  $\mathbf{c}$ , number of particles  $K$ .  
 3: - Note: operations involving index  $k$  are performed for  $k \in \{1, \dots, K\}$ .  
 4:  
 5: **Initialization:**  
 6: Sample  $\mathbf{x}_T^{(k)} \sim q(\mathbf{X}_T)$ , set  $\phi(\mathbf{x}_T^{(k)} | \mathbf{c}) = 1$   
 7: **for**  $t = T - 1$  to 0 **do** ▷ Iterate over time steps  
 8:      $\tilde{\mathbf{x}}_t^{(k)} \sim q(\mathbf{X}_t | \mathbf{x}_{t+1}^{(k)}, \mathbf{c})$  ▷ Proposal  
 9:      $\phi(\tilde{\mathbf{x}}_t^{(k)} | \mathbf{c}) = \text{CALCCORRECTION}(\tilde{\mathbf{x}}_t^{(k)}, \mathbf{c}, t)$  ▷ Calculate correction term in Eq. (5)  
 10:     $w(\mathbf{x}_{t+1}^{(k)}, \tilde{\mathbf{x}}_t^{(k)} | \mathbf{c}) = \frac{\phi(\tilde{\mathbf{x}}_t^{(k)} | \mathbf{c})}{\phi(\mathbf{x}_{t+1}^{(k)} | \mathbf{c})}$  ▷ Calculate resampling weight  
 11:     $\mathbf{x}_t^{(k)} \sim \text{MULTINOMIAL}(\{\tilde{\mathbf{x}}_t^{(k)}\}_{k=1}^K; \{w(\mathbf{x}_{t+1}^{(k)}, \tilde{\mathbf{x}}_t^{(k)} | \mathbf{c})\}_{k=1}^K)$  ▷ Resampling  
 12: **end for**  
 13:  
 14: **Output:**  
 15:  $\{\mathbf{x}_0^{(k)}\}_{k=1}^K$  that approximately follow the ground-truth distribution  $p(\mathbf{X}_0 | \mathbf{c})$ .

---

**Algorithm 2** CALCCORRECTION

---

1: **Input:**  
 2: - Sample  $\tilde{\mathbf{x}}_t$ , conditional signal  $\mathbf{c}$ , time step  $t$ .  
 3: - Conditional discriminator  $d(\mathbf{X}_t | \mathbf{C}; t)$ , unconditional discriminator  $d(\mathbf{X}_t; t)$ , object detector  $\hat{p}(O_{Xi} = 1 | \mathbf{X}_0)$ , statistics  $\kappa_{it}$  estimated from historically generated images, hyper-parameter  $\pi_{it}$ .  
 4:  
 5: **if** use discriminator-based approach **then** ▷ Section 3.5  
 6:      $\phi(\tilde{\mathbf{x}}_t | \mathbf{c}; t) = \frac{d(\tilde{\mathbf{x}}_t | \mathbf{c}; t)}{1 - d(\tilde{\mathbf{x}}_t | \mathbf{c}; t)}$  ▷ Eq. (7)  
 7: **else if** use hybrid approach **then** ▷ Section 3.6  
 8:      $\frac{p(\tilde{\mathbf{x}}_t)}{q(\tilde{\mathbf{x}}_t)} = \frac{d(\tilde{\mathbf{x}}_t; t)}{1 - d(\tilde{\mathbf{x}}_t; t)}$  ▷ Unconditional likelihood ratio  
 9:      $p(O_{Ci} = 1 | \tilde{\mathbf{x}}_t) \approx \hat{p}(O_{Xi} = 1 | f(\tilde{\mathbf{x}}_t))$  ▷ Eq. (10)  
 10:     $q(O_{Ci} = 1 | \tilde{\mathbf{x}}_t) \approx \hat{p}(O_{Xi} = 1 | f(\tilde{\mathbf{x}}_t)) + \hat{p}(O_{Xi} = 0 | f(\tilde{\mathbf{x}}_t)) \frac{(1 - \kappa_{it})\pi_{it}}{(1 - \kappa_{it})\pi_{it} + 1 - \pi_{it}}$  ▷ Eq. (11)  
 11:     $\phi_t(\tilde{\mathbf{x}}_t | \mathbf{c}) = \frac{p(\tilde{\mathbf{x}}_t)}{q(\tilde{\mathbf{x}}_t)} \cdot \frac{\prod_{i: O_{Ci}=1} p(O_{Ci} = 1 | \tilde{\mathbf{x}}_t)}{\prod_{i: O_{Ci}=1} q(O_{Ci} = 1 | \tilde{\mathbf{x}}_t)}$  ▷ Eq. (13)  
 12: **end if**  
 13:  
 14: **Output:**  
 15: - Correction term  $\phi_t(\tilde{\mathbf{x}}_t | \mathbf{c})$ .

---

Both  $q(O_{Xit} | O_{Ci} = 1)$  and  $q(O_{Xit} | O_{Ci} = 0)$  can be estimated from the samples of distribution  $q(\mathbf{X}_t | \mathbf{C})$ . Specifically, consider a different set of  $H$  samples  $\{\mathbf{x}_t^{(h)}\}$  that are generated without particle filtering process. For each sample  $\mathbf{x}_t^{(h)}$ , we compute the object occurrence estimate  $\hat{O}_{Xit}$  by feeding the predicted clean image,  $f(\mathbf{x}_t^{(h)})$ , to the object detector, and thresholding the output probability of the object detector with 0.5. Then  $q(O_{Xit} = 1 | O_{Ci} = 1)$  can be estimated by calculating the percentage of samples whose  $\hat{O}_{Xit} = 1$  among the samples whose corresponding input caption mentions object  $i$ , as in Eq. (12).  $q(O_{Xit} = 1 | O_{Ci} = 0)$  can be calculated similarly from sam-

ples whose input caption does not mention object  $i$ . In practice, since the diffusion model seldom generates objects that are not mentioned in the text, we empirically find that  $q(O_{Xit} = 1 | O_{Ci} = 0)$  is very close to zero. Hence, to reduce computation complexity, we will set this term to zero, and  $q(O_{Xit} = 0 | O_{Ci} = 0)$  to one. Finally,  $q(O_{Ci} = 0)$  and  $q(O_{Ci} = 1)$  controls how much weight we would like to place on each case, and we will treat them as hyperparameters and denote  $q(O_{Ci} = 1)$  as  $\pi_{it}$ . Note that although  $\pi_{it}$  can be dependent on both  $i$  and  $t$ , for simplicity, we set  $\pi_{it}$  to be the same for all  $i$  and  $t$ .

Denote  $q(O_{Xit} = 1 | O_{Ci} = 1)$  as  $\kappa_{it}$ , with the above as-

sumptions,

$$\begin{aligned} q(O_{Ci} = 1 | O_{Xit} = 1) &= 1 \\ q(O_{Ci} = 1 | O_{Xit} = 0) &= \frac{(1 - \kappa_{it})\pi_{it}}{(1 - \kappa_{it})\pi_{it} + 1 - \pi_{it}}. \end{aligned} \quad (19)$$

Plug Eq. (19) back to Eq. (17), we can derive Eq. (11) in the main paper.

## C. Experiments on Text-to-image Generation

### C.1. Reparameterize Stable Diffusion for Reverse-time SDE

Following Karras et al. [29] and Xu et al. [65], we use the reverse-time SDE [1] for generation:

$$d\mathbf{x} = -2\dot{\sigma}(t)\sigma(t)\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma(t)) dt + \sqrt{2\dot{\sigma}(t)\sigma(t)} d\omega_t, \quad (20)$$

where  $\omega_t$  is the standard Wiener process,  $\sigma(t) = t$  is the noise level at time  $t$ , and  $p(\mathbf{x}; \sigma(t))$  is the distribution obtained by adding Gaussian noise of standard deviation  $\sigma(t)$  to the input data.

To use stable diffusion [48] in Eq. (20), the key is to reparameterize the model to estimate the score  $\nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma(t))$ . Following Karras et al. [29], we estimate the score as follows:

$$\begin{aligned} \nabla_{\mathbf{x}} \log p(\mathbf{x}; \sigma) &= \frac{D(\mathbf{x}; \sigma) - \mathbf{x}}{\sigma^2}, \\ D(\mathbf{x}; \sigma) &= c_{\text{skip}}(\sigma)\mathbf{x} + c_{\text{out}}(\sigma)F(c_{\text{in}}(\sigma)\mathbf{x}; c_{\text{noise}}(\sigma)), \end{aligned} \quad (21)$$

where  $F(\cdot)$  is the denoising network trained in stable diffusion,  $c_{\text{skip}}(\sigma) = 1$ ,  $c_{\text{out}}(\sigma) = -\sigma$ ,  $c_{\text{in}}(\sigma) = 1/\sqrt{\sigma^2 + 1}$ ,  $c_{\text{noise}}(\sigma) = 999\sigma_{\text{train}}^{-1}(\sigma)$ ,  $\sigma_{\text{train}}(t)$  is the noise schedule used when training the denoising network, and  $\sigma_{\text{train}}^{-1}(\sigma)$  denotes the inverse of  $\sigma_{\text{train}}(t)$ . In particular, the forward process of Stable Diffusion v2.1-base can be considered as a discrete version of the VP SDE [56]:

$$d\mathbf{x} = f(t)\mathbf{x} dt + g(t) d\omega_t, \quad (22)$$

where

$$\begin{aligned} f(t) &= -\frac{1}{2}\beta(t), \quad g(t) = \sqrt{\beta(t)}, \\ \beta(t) &= ((\beta_{\max}^{0.5} - \beta_{\min}^{0.5})t + \beta_{\min}^{0.5})^2, \end{aligned} \quad (23)$$

with  $\beta_{\min} = 0.85$ , and  $\beta_{\max} = 12$ . We follow Karras et al. [29] (Eqs. (152) - (163)) to derive  $\sigma_{\text{train}}(t)$  by plugging in the scaled linear schedule  $\beta(t) = ((\beta_{\max}^{0.5} - \beta_{\min}^{0.5})t + \beta_{\min}^{0.5})^2$  used in Stable Diffusion v2.1-base, which results in:

$$\sigma_{\text{train}}(t) = \sqrt{e^{\frac{1}{3}\beta_d^2 t^3 + \beta_d \beta_{\min}^{0.5} t^2 + \beta_{\min} t} - 1}, \quad (24)$$

where  $\beta_d = \beta_{\max}^{0.5} - \beta_{\min}^{0.5}$ .

Finally, although Stable Diffusion is trained with discrete time steps, we follow Lu et al. [39] to directly feed the continuous time value  $c_{\text{noise}}(\sigma)$  to the model.

### C.2. Implementation Details for Baselines

**Identify objects in the caption.** For methods that require the identification of objects in the caption, we use the following two steps:

1. Extract all noun phrases in the caption using spaCy.<sup>2</sup> Filter out noun phrases that are stop words, e.g., “which” and “who”.
2. Check if extracted noun phrases match one of the object categories in MS-COCO, and only keep the phrases that belong to MS-COCO objects. The match is determined by whether the Levenshtein Distance between the noun phrase and object category name is smaller than 0.1 of the length of the noun phrase and object category name.

The same process is also used to filter out 261 captions in MS-COCO that contain at least four objects (to measure object occurrence), and 5,000 captions in MS-COCO that contain at least one object (to measure FID).

**Sample selection criteria.** For OBJECTSELECT, we use Eq. (2) to select the best image from generated images. Specifically, since the object detector outputs multiple proposal regions, where each proposal region comes with a predicted probability for object  $i$ , we take the maximum probability for object  $i$  over all proposal regions as  $\hat{p}(O_{Xi} = 1 | \mathbf{x}_0)$ . For REWARDSELECT and TIFASELECT, we use the official implementation [30] to select images, except that we only include object-related questions for TIFASELECT since our focus is on object occurrence.

For other baselines, we use the official implementation to generate images. For SPATIAL-TEMPORAL, we only optimize for five rounds since we found further optimization does not improve performance. For D-GUIDANCE, we only apply guidance when  $\sigma(t) < 5$  since we found at large noise levels, the object detector barely detects any object, which results in a noisy gradient. All methods are evaluated on both Restart sampler [65] and EDM sampler [29]. The only exceptions are SPATIAL-TEMPORAL and ATTEND-EXCITE, which use the original sampler in their papers.

### C.3. Discriminator Training

We follow Kim et al. [32] to train both unconditional and conditional discriminators. Table 3 shows the hyperparameters for discriminator training.

**Training data and objective.** We use LAION Aesthetics [53] to train discriminators. We randomly sample 1M images with aesthetics score higher than 6 as real images. For fake images, we generate 1M images with 49 NFE using EDM sampler. For conditional discriminator, the captions used to generate fake images are the same with the captions of the real images. For unconditional discriminator, we use a different set of 1M captions to generate fake images.

---

<sup>2</sup><https://spacy.io>.

	Unconditional	Conditional
# real images	1,000,000	1,000,000
# fake images	1,000,000	1,000,000
Same captions for real & fake images?	✗	✓
# Epochs	2	2
Batch size	128	128
Learning rate	$10^{-4}$	$10^{-4}$
Time sampling	Uniform	Uniform
Total GPU hours	22	22
GPUs	A6000	A6000

Table 3. Configurations of discriminator training for text-to-image generation.

The discriminators are trained with the canonical discrimination loss in Eq. (6). Specifically, given a clean image, we uniformly sample time  $t \in [10^{-5}, 1]$  and follow Eq. (22) to get the corresponding noisy image. The discriminator is then trained to discriminate noisy version of real and fake images.

**Model architecture.** We use the hidden representations of the middle block of the U-Net [49] to predict whether an image is real or fake, since previous works have found that these representations capture semantic information in the input image [5, 33]. Specifically, we initialize the discriminator with the U-Net in Stable Diffusion v2.1-base and only keep its down and middle blocks. We add a linear prediction layer on top the representations of the middle block. During training, we freeze the down block and only fine-tune the middle block and the prediction layer.

#### C.4. Sampling Configurations

For all sample selection methods (including ours), we generate each image with a fixed NFE and report performance when  $K = 5, 10, 15$  images are generated.

Table 4 shows the sampling configurations for both samplers. For Restart sampler,  $N_{\text{main}}$  denotes the number of steps in the main backward process,  $N_{\text{Restart},i}, K_i, t_{\min,i}, t_{\max,i}$  is the number of steps, number of repetitions, minimal, and maximum time points in the  $i$ -th restart interval, and  $l$  is the number of restart intervals. Following Xu et al. [65], we use Euler method for the main backward process and Heun method for the restart backward process. Please refer to the original paper for detailed explanations. For our PF methods, the resampling is always performed before adding noise, *i.e.*, at time  $t_{\min,i}$ , thus resampling is performed six times in total. For EDM sampler,  $N$  denotes the number of denoising steps,  $n_i$  denotes the index of steps when resampling is performed (for PF methods), and  $m$  denotes the number of resampling steps. Resampling is only performed four times because we empirically find performing resampling at each denoising step does not improve performance significantly compared to resampling only at a subset of steps.

Table 5 shows the comparison of computation cost for all

Restart configuration	
$N_{\text{main}}$	$\{(N_{\text{Restart},i}, K_i, t_{\min,i}, t_{\max,i})\}_{i=1}^l$
$30, \{(4, 1, 1.09, 1.92), (4, 2, 0.59, 1.09), (4, 2, 0.30, 0.59), (4, 1, 0.06, 0.30)\}$	
EDM configuration: $N, \{n_i\}_{i=1}^m$	
25, {10, 13, 16, 19}	

Table 4. Sampling configurations for text-to-image generation.

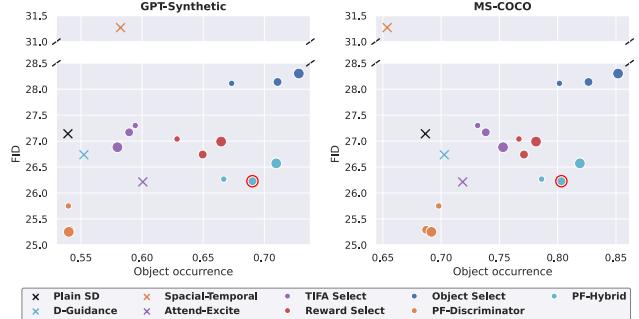


Figure 7. FID ( $\downarrow$ ) vs. Object occurrence ( $\uparrow$ ) for all methods evaluated with EDM sampler. Ideal points should scatter at the bottom right corner. Object occurrence is measured on GPT-Synthetic (left) and MS-COCO (right), and FID is measured on MS-COCO.  $K = 5, 10, 15$  images are generated for sample selection methods, and the sizes of points indicate the value of  $K$  (larger  $K$  has larger points). The method that achieves the best combined performance is highlighted in red.

methods. The reported cost for sample selection methods is when  $K = 5$  images are generated for each caption, since their performance already exceeds other baselines at  $K = 5$ . Based on the table, all methods are significantly faster than SPATIAL-TEMPORAL, and sample selection methods have similar runtime with ATTEND-EXCITE on EDM sampler and D-GUIDANCE on Restart sampler. Moreover, the sample selection methods can be run in parallel much easier than other methods, so their runtime can potentially be further reduced.

#### C.5. Additional Results

Figure 7 shows the results when EDM is used as the underlying sampler. There are two observations from the figure. First, the overall trend of all methods aligns with Figure 4 when Restart is used as the sampler. In particular, the sampling-based methods generally outperform the non-sampling-based ones. And the three methods proposed in this paper, OBJECTSELECT, PF-DISCRIMINATOR, and PF-HYBRID lie at the frontier of the performance trade-off, significantly outperform other methods. Furthermore, PF-HYBRID is the only method that simultaneously achieves a high object occurrence and a low FID. Second, the performance of EDM sampler is generally worse than that of Restart sampler on both object occurrence and FID, which

	EDM Sampler			Restart Sampler		
	NFE	Require Gradient?	Runtime (s)	NFE	Require Gradient?	Runtime (s)
SD [48]	49	✗	7	66	✗	9
D-GUIDANCE [32]	67	✓	20	106	✓	41
SPATIAL-TEMPORAL* [62]	250	✓	110	—	—	—
ATTEND-EXCITE* [9]	50	✓	29	—	—	—
TIFASELECT [30]	245	✗	39	330	✗	50
REWARDSELECT [30]	245	✗	32	330	✗	43
OBJECTSELECT	245	✗	32	330	✗	43
PF-DISCRIMINATOR	245	✗	33	330	✗	44
PF-HYBRID	265	✗	39	360	✗	53

Table 5. Computation cost for all methods. Runtime is measured on a single NVIDIA V100 GPU. \*: the two baselines use the original samplers in their papers.

	ImageNet-64	FFHQ
<b>Training Configurations</b>		
# real images	1,200,000	60,000
# fake images	1,200,000	60,000
# Epochs	50	50
Batch size	1024	256
Learning rate	$5 \times 10^{-4}$	$3 \times 10^{-4}$
Time sampling	Importance	Uniform
Total GPU hours	70	4
GPUs	A6000	A6000
<b>Shallow U-Net Architecture</b>		
Input shape	(B, 8, 8, 512)	(B, 8, 8, 512)
Class condition	✓	✗
# Resnet blocks	4	4
# Attention blocks	3	3

Table 6. Configurations of discriminator training for class-conditioned (ImageNet-64) and unconditional (FFHQ) generation.

shows the superiority of the restart process.

## D. Experiments on Unconditional and Class-conditioned Generation

### D.1. Discriminator Training

We follow the official implementation [32] to train discriminators on ImageNet-64 and FFHQ since the original paper did not experiment on the dataset or did not release the trained discriminator. Table 6 shows the training configurations. Following Kim et al. [32], the discriminators use features extracted by the frozen pre-trained classifier in Dhariwal and Nichol [15], and a shallow U-Net is added to discriminate images. During training, the pre-trained classifier is frozen and only the shallow U-Net is trained. Please refer to Kim et al. [32] for more details.

### D.2. Sampling Configurations

In order to have a fair comparison in terms of computation cost, we compare all methods under the same total NFE. For PF and D-SELECT, we generate each image with a fixed NFE and vary the number of particles in generation, *i.e.*, choose value of  $K$  from  $\{2, 4, 6\}$ . For the original sampler and D-GUIDANCE, we increase the number of steps or restart iterations to match the total NFE of sampling-based methods. All methods are evaluated on both Restart and EDM samplers. Table 7 and 8 show the sampling configurations for each sampler.

For EDM sampler, we use the same hyper-parameters as the original paper, except the  $S_{\text{churn}}$  (which controls the amount of noise added in each denoising step) on FFHQ, since we find adding small amount of noise ( $S_{\text{churn}} > 0$ ) outperforms the original setting where no noise is added ( $S_{\text{churn}} = 0$ ). Additionally, for time steps after resampling is performed, we increase  $S_{\text{churn}}$  to  $(\sqrt{2} - 1)N$  ( $N$  is the number of denoising steps), which is the maximum allowed value in Karras et al. [29]. Since we find a large added noise after resampling can improve performance (please see Section E). For Restart sampler, we use the same configurations as the original paper on ImageNet-64 and slightly modify them to adapt to FFHQ since the original paper did not experiment on FFHQ.

### D.3. Additional Results

Figure 8 shows FID as a function of total NFE when EDM is used as the sampler. Additionally, we also plot PF with Restart sampler for reference. There are three observations from the figure. First, FIDs of all methods generally decrease as NFE increases, and D-GUIDANCE, D-SELECT, and PF all outperform the original sampler. Second, our PF with Restart sampler achieves the lowest FID on both datasets. Third, the performance gain of our method when combined with EDM sampler is not as large as that when combined with Restart sampler. The difference can be ascribed to the fact that resampling is performed at

NFE per Image	Resampling Steps: $\{n_i\}_{i=1}^m$	$S_{\text{churn}}$
<b>ImageNet-64</b>		
127	{31, 35, 39, 43, 47, 51}	10
255	—	20
511	—	40
767	—	60
<b>FFHQ</b>		
63	{15, 19, 23, 27}	1.25
127	—	2.5
255	—	5.0
399	—	7.8125

Table 7. Sampling configurations for EDM sampler on standard benchmarks. Two highlighted rows are used in PF and D-SELECT to generate one image. The resampling steps are only applicable to our PF methods.  $S_{\text{churn}}$  controls the amount of noise added in each denoising step (please refer to Karras et al. [29] for details).

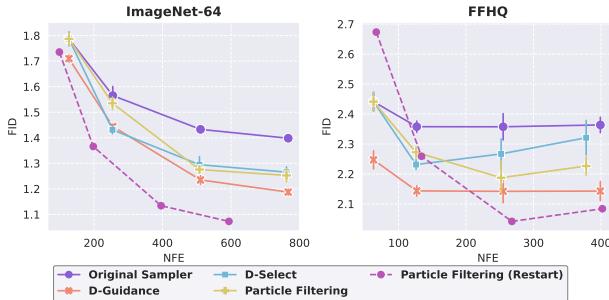


Figure 8. FID (average of 3 runs) on ImageNet-64 (left) and FFHQ (right) when evaluated with EDM sampler. The PF with Restart sampler (dashed line) is added for reference. Error bars indicate standard deviations.

smaller time steps for Restart sampler, which better selects a promising population among the particles since the discriminator can better distinguish real and model-generated images at smaller time steps. Additionally, the added noise after resampling is larger for Restart, which also benefits the exploration around the promising particles (see Section E for the impact of amount of added noise).

## E. Additional Ablation Study

In this section, we present additional ablation studies that investigate three design choices in our particle filtering framework. Particularly, we will study the impacts of when the resampling is inserted (*i.e.*, before or after adding noise), the amount of noise added, and the NFE used to generate a single image. We will study their impacts in class-conditioned generation on ImageNet-64 and when Restart sampler is used.

We start by studying **when is a proper time to insert resampling** during the generation. Particularly, we compare two time points to insert resampling. ① Resample be-

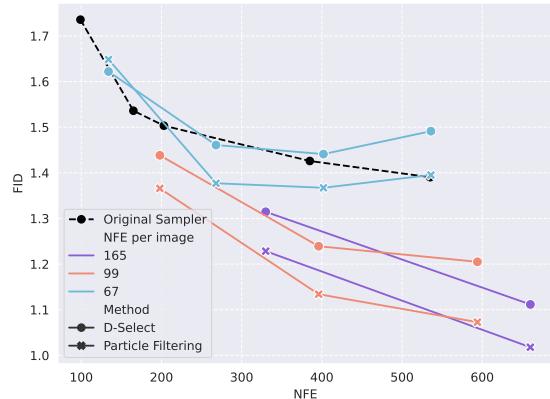


Figure 9. FID on ImageNet-64 when D-SELECT and our PF method are applied to different NFEs per image.

fore adding noise; ② Resample after adding noise. Table 9 shows the FID for both cases. It can be observed that resampling before adding noise significantly outperforms the counterpart. There could be two potential reasons for the performance difference. First, resampling after adding noise means resampling weights are estimated at higher noise scales, which is less accurate. Second, compared to ② that adopts a selection-after-exploration procedure, ① selects particles before exploration, which allows more particles to be sampled around the promising particles with large weights.

Following the above intuition, we further explore the impact of the **amount of noise added**. Specifically, we change the restart configuration such that the variance of the added noise at time  $t$  is reduced to  $t^2$ , while keeping all other configurations the same. Table 10 shows the FID when the original and reduced noise is added. The results indicate the importance of a large added noise.

Finally, we explore the impact of the **NFE for each image**. Instead of fixing an NFE and only varying the number of particles, we now evaluate our method when it is applied to different NFEs. Figure 9 shows the FID when each image is generated with NFE = 67, 99, 165. For each NFE, we further report the performance when  $K = 2, 4, 6$  images are generated (except for NFE = 165 due to the large cost). There are two observations from the figure. First, PF consistently outperforms D-SELECT across all NFEs, again demonstrating its advantages. Second, different NFEs present varying performance trends. When the number of denoising steps is small (NFE = 67), FID reaches a plateau at  $K = 4$  particles, and further increasing the value of  $K$  does not improve performance due to the large discretization error. On the contrary, FID continues decreasing as  $K$  increases when the discretization error is small (NFE = 99, 165). Particularly, generating  $K = 4$  particles

NFE per Image		Restart Configuration $N_{\text{main}}, \{(N_{\text{Restart},i}, K_i, t_{\min,i}, t_{\max,i})\}_{i=1}^l$
<b>ImageNet-64</b>		
67		18, $\{(5, 1, 19.35, 40.79), (5, 1, 1.09, 1.92), (5, 1, 0.59, 1.09), (5, 1, 0.06, 0.30)\}$
99		18, $\{(3, 1, 19.35, 40.79), (4, 1, 1.09, 1.92), (4, 4, 0.59, 1.09), (4, 1, 0.30, 0.59), (4, 4, 0.06, 0.30)\}$
165		18, $\{(3, 1, 19.35, 40.79), (4, 1, 1.09, 1.92), (4, 5, 0.59, 1.09), (4, 5, 0.30, 0.59), (4, 10, 0.06, 0.30)\}$
203		36, $\{(4, 1, 19.35, 40.79), (4, 1, 1.09, 1.92), (4, 5, 0.59, 1.09), (4, 5, 0.30, 0.59), (6, 6, 0.06, 0.30)\}$
385		36, $\{(3, 1, 19.35, 40.79), (6, 1, 1.09, 1.92), (6, 5, 0.59, 1.09), (6, 5, 0.30, 0.59), (6, 20, 0.06, 0.30)\}$
535		36, $\{(6, 1, 19.35, 40.79), (6, 1, 1.09, 1.92), (7, 6, 0.59, 1.09), (7, 6, 0.30, 0.59), (7, 25, 0.06, 0.30)\}$
<b>FFHQ</b>		
67		18, $\{(5, 1, 1.09, 1.92), (5, 2, 0.59, 1.09), (5, 1, 0.06, 0.30)\}$
119		18, $\{(8, 1, 19.35, 40.79), (8, 2, 1.09, 1.92), (8, 2, 0.59, 1.09), (8, 1, 0.06, 0.30)\}$
251		36, $\{(11, 1, 19.35, 40.79), (11, 3, 1.09, 1.92), (11, 3, 0.59, 1.09), (11, 2, 0.06, 0.30)\}$
401		48, $\{(18, 1, 19.35, 40.79), (18, 3, 1.09, 1.92), (18, 3, 0.59, 1.09), (18, 2, 0.06, 0.30)\}$

Table 8. Sampling configurations for Restart sampler on standard benchmarks. Two highlighted rows are used in PF and D-SELECT to generate one image.

	Before	After
$K = 2$	1.37	1.53
$K = 4$	1.11	1.31
$K = 6$	1.07	1.23

Table 9. FID on ImageNet-64 when resampling is inserted before or after adding noise.

	Original noise	Reduced noise
$K = 2$	1.37	1.54
$K = 4$	1.11	1.37
$K = 6$	1.07	1.31

Table 10. FID on ImageNet-64 with original and reduced noise.

when  $\text{NFE} = 165$  achieves the state-of-the-art FID of 1.02 on ImageNet-64.

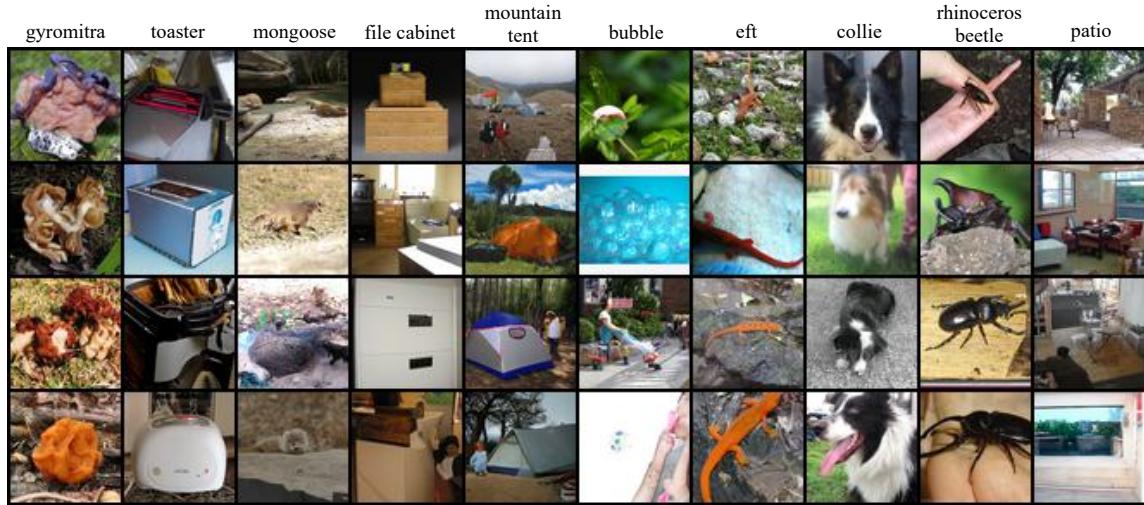
## F. Generated Samples

In this section, we present more generated samples for our method and baselines. Figures 10 and 11 show the uncropped samples on ImageNet-64 and FFHQ respectively. As can be observed, both D-GUIDANCE and PF generate images with higher quality than the original sampler. Compared to D-GUIDANCE, our method generates images with closer and more detailed views (*e.g.*, class “mongoose”).

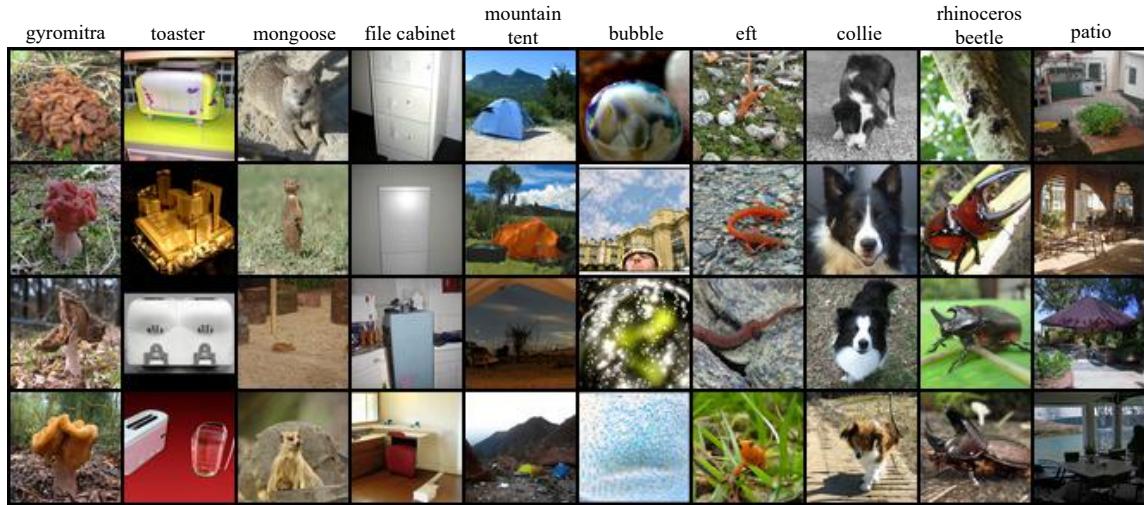
## G. Details of Subjective Evaluation

We recruit human annotators to evaluate both object occurrence and image quality of generated images. For object occurrence, annotators are provided with an image and a list of objects, and they are asked to identify whether each object occurs in the image or not. For image quality, annotators are given two images generated based on the same

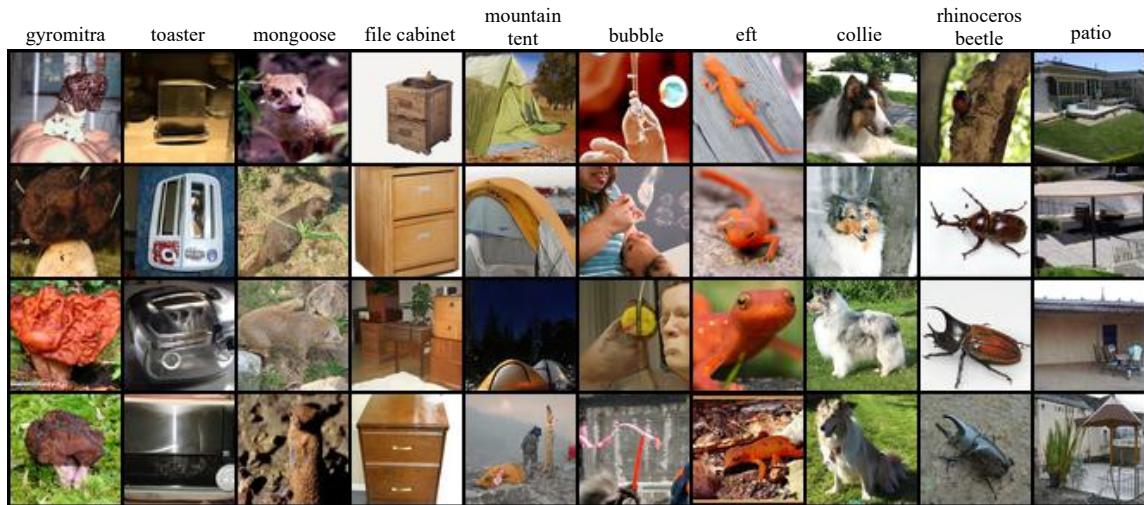
caption, where one image is generated by PF-HYBRID and the other image by a baseline. The caption is not revealed to annotators in order to make the evaluation focus more on image quality instead of image and text alignment. The annotation interfaces for object occurrence and image quality are shown in Tables 11 and 12 respectively.



Original sampler NFE: 535 FID: 1.39



D-guidance NFE: 535 FID: 1.16



Particle filtering NFE: 594 FID: **1.07**

Figure 10. Uncurated samples on ImageNet-64 for our method and baselines. All methods use Restart sampler.



Original sampler NFE: 251 FID: 2.47



D-guidance NFE: 251 FID: 2.08



Particle filtering NFE: 268 FID: **2.02**

Figure 11. Uncurated samples on FFHQ for our method and baselines. All methods use Restart sampler.

---

**Instructions:**

You will be given an image generated by some AI algorithm, your task is to **identify all objects** that appear in the image.

Notes:

- You should **ignore the characteristics** of the object such as color and count when deciding whether it appears or not. E.g., if the object is “a red apple,” you should still select it even if the image shows a green apple.
- Only select objects that you are **confident** about. E.g., the “knife” in the image below is not clear, so you should not select it.

**Example:**

Do the following objects appear in the image?

- A black blender: Yes
- bottle: No
- glass: Yes
- cutting board: Yes
- knife: No

---

**Task:**

Please identify **all objects** appear in the image



- 
- white kitchen sink
  - white coffee cups
  - wine glasses
  - knives
  - None of the above

---

Table 11. Instructions and an example task for the subjective evaluation on object occurrence.

---

**Instructions:**

In this task, you will see two images that are generated based on a text caption by different AI algorithms. You will be asked to evaluate **which image looks more real and natural**.

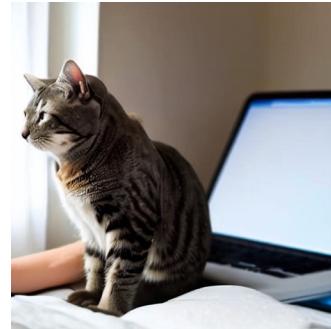
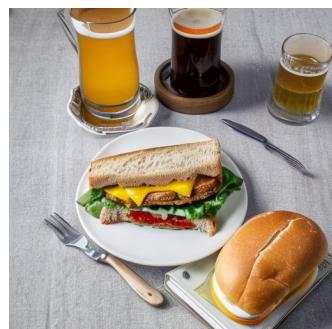
**Example:****(A)****(B)**

Image (B) generates a natural cat and laptop, whereas the laptop in image (A) is distorted, and the human body is also unnatural. Therefore, **image (B) is better**.

---

**Task:**

Which image looks more natural and like a real photo?

**(A)****(B)**

(A) is better    (B) is better

---

Table 12. Instructions and an example task for the subjective evaluation on image quality.