PTDiffusion: Free Lunch for Generating Optical Illusion Hidden Pictures with Phase-Transferred Diffusion Model

Supplementary Material

1. Preliminary background

1.1. Diffusion model background

Since the advent of Denoising Diffusion Probabilistic Model (DDPM), diffusion model has soon dominated research field of generative AI due to its advantages in training stability and sampling diversity as compared with GAN. Grounded in the theory of stochastic differential equations, diffusion model learns to iteratively denoise a noise-corrupted input signal (*e.g.*, an image or a video clip), ultimately generating clean data that follow the underlying target distribution. Diffusion model is conceptually composed of a forward diffusion process and a reverse denoising process. The forward diffusion process gradually adds noise to the data over a series of steps, transforming the data into a random Gaussian distribution, while the reverse denoising process learns to reverse the forward process by iteratively removing noise from the data, starting from pure noise and gradually reconstructing the original data. The model is trained to predict the noise added at each step of the forward process. By learning to denoise, the model can generate new data samples by starting from random noise and applying the reverse process.

Given the original data distribution $q(x_0)$, the forward diffusion process applies a T-step Markov chain to gradually add noise to the original data x_0 according to the conditional distribution $q(x_t|x_{t-1})$, which is defined as follows:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)\mathcal{I}), \tag{1}$$

where α_t follows a predefined schedule, $\alpha_t \in (0,1)$, $\alpha_t > \alpha_{t+1}$. Using the notation $\bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i$, we can derive the marginal distribution $q(x_t|x_0)$ that can be used to directly obtain x_t from x_0 in a single step for arbitrary time step t:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathcal{I}), \tag{2}$$

where $\sqrt{\bar{\alpha}_T} \approx 0$. With the forward diffusion process, the source data x_0 is transformed into x_T that follows an isotropic Gaussian distribution.

The reverse denoising process learns to conversely convert a Gaussian noise x_T to the manifold of the original data distribution $q(x_0)$ by gradually estimating and sampling from the posterior distribution $p(x_{t-1}|x_t)$. Since the posterior distribution $p(x_{t-1}|x_t)$ is mathematically intractable, we can derive the conditional posterior distribution $p(x_{t-1}|x_t, x_0)$ with the Bayes formula and some algebraic manipulation:

$$p(x_{t-1}|x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t\mathcal{I}), \tag{3}$$

$$\tilde{\mu}_t(x_t, x_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha}_t(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t, \tag{4}$$

$$\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t, \tag{5}$$

where $\beta_t = 1 - \alpha_t$. However, the conditional posterior distribution $p(x_{t-1}|x_t, x_0)$ cannot be directly used for sampling since x_0 is unavailable at inference time (x_0 is the target of the sampling process). Thus, DDPM tries to estimate the unknown x_0 given the x_t at each time step. Considering the reparameterization form of Eq. 2:

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, \tag{6}$$

in which ϵ_t denotes the randomly sampled Gaussian noise that maps x_0 to x_t in a single step according to Eq. 2. Given Eq. 6, we can represent x_0 using x_t and ϵ_t:

$$x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_t). \tag{7}$$

However, the Gaussian noise ϵ_t sampled in the forward diffusion process is also unknown for the reverse denoising process where only x_t is available. Consequently, DDPM builds a noise estimation network ϵ_θ that predicts the sampled Gaussian noise ϵ_t in Eq. 7 with x_t and time step t as input, which is realized by training ϵ_θ with the following noise regression loss:

$$L = \|\epsilon_t - \epsilon_\theta(x_t, t)\|_2, \tag{8}$$

where $t \sim$ Uniform($\{1, ..., T\}$), $\epsilon_t \sim \mathcal{N}(0, \mathcal{I})$, x_t is computed via Eq. 6. After model training, $y_\theta(x_t)$, the estimation of x_0 given x_t, can be obtained simply by replacing ϵ_t in Eq. 7 with the predicted noise $\epsilon_\theta(x_t, t)$:

$$y_\theta(x_t) = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t, t)). \tag{9}$$

Replacing the unknown x_0 in Eq. 3 with its predicted estimation $y_\theta(x_t)$ given by Eq. 9, we can sample x_{t-1} based on x_t from the approximate posterior distribution $\mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, y_\theta(x_t)), \tilde{\beta}_t\mathcal{I})$, and thus sample the ultimate x_0 step by step from the initial Gaussian noise x_T.

1.2. Conditional diffusion model

Taking the image generation task as an example, conditional diffusion model tackles conditional image synthesis by introducing additional condition c to the model to guide image generation (denoising) process. In this paradigm, the condition signal c together with x_t and time step t are taken as input to the noise estimation network ϵ_θ, such that ϵ_θ is trained to conditionally predict the added Gaussian noise in the forward diffusion process, as supervised by the randomly sampled ϵ_t in Eq. 6. The training loss given by Eq. 8 is correspondingly updated as:

$$L = \|\epsilon_t - \epsilon_\theta(x_t, t, c)\|_2, \tag{10}$$

where $t \sim \text{Uniform}(\{1, ..., T\})$, $\epsilon_t \sim \mathcal{N}(0, \mathcal{I})$, x_t is computed via Eq. 6. After model training, the reverse sampling process is applied to generate new images from random Gaussian noise x_T, based on the step-by-step denoising according to the conditional posterior distribution given by Eq. 3, in which the unknown x_0 is approximated by the linear combination of x_t and the conditional noise estimation, *i.e.*, the $y_\theta(x_t)$ (the approximate x_0 estimated by x_t) in Eq. 9 is updated as:

$$y_\theta(x_t, c) = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t, t, c)). \tag{11}$$

1.3. Denoising diffusion implicit model

Denoising diffusion implicit model (DDIM) is a variant of diffusion model that builds on the framework of DDPM but enables much more efficient sampling while maintaining high-quality generation results. DDIM can generate samples in significantly fewer steps compared with DDPM by modeling the reverse denoising process as a non-Markovian process and skipping the intermediate denoising steps.

DDIM is totally the same as DDPM in model training and only differs with DDPM in model inference, namely that DDIM can directly inherit the pre-trained DDPM model. To compute x_{t-1} from x_t in the reverse denoising (sampling) process, DDIM features a two-step deterministic denoising. In the first step, DDIM estimates an approximate x_0 based on x_t using Eq. 9. In the second step, DDIM computes x_{t-1} from the approximate x_0 using the forward diffusion in the form of Eq. 6:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}y_\theta(x_t) + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_{t-1}, \tag{12}$$

where $y_\theta(x_t)$ is given by Eq. 9. Considering that the ϵ_{t-1} in the above equation is the sampled Gaussian noise in the forward diffusion process, which is unknown in the reverse denoising process, we can replace ϵ_{t-1} with $\epsilon_\theta(x_{t-1}, t-1)$, the approximate ϵ_{t-1} estimated by the network ϵ_θ. Therefore, the Eq. 12 can be updated as:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}y_\theta(x_t) + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_\theta(x_{t-1}, t-1). \tag{13}$$

However, the $\epsilon_\theta(x_{t-1}, t-1)$ in the above equation is also unavailable since x_{t-1} is unknown (we only know x_t and want to compute x_{t-1}). Thus, we can further approximate $\epsilon_\theta(x_{t-1}, t-1)$ with $\epsilon_\theta(x_t, t)$ and arrive to the final DDIM sampling equation:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}y_\theta(x_t) + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_\theta(x_t, t). \tag{14}$$

Eq. 14 shows that the reverse sampling process of DDIM is totally deterministic, namely, each starting Gaussian noise x_T yields a unique sampling result x_0.

Note that the above derived two-step sampling process of $x_t \to x_0 \to x_{t-1}$ also applies for $x_t \to x_0 \to x_{t+1}$. That is, a clean image x_0 can be deterministically inverted into a Gaussian noise through the following inversion process:

$$x_{t+1} = \sqrt{\bar{\alpha}_{t+1}}y_\theta(x_t) + \sqrt{1-\bar{\alpha}_{t+1}}\epsilon_\theta(x_t, t). \tag{15}$$

The DDIM inversion given by Eq. 15 has wide applications in image editing and style transfer. For conditional image generation of DDIM, the $y_\theta(x_t)$ and $\epsilon_\theta(x_t, t)$ in Eq. 14 and Eq. 15 are updated to $y_\theta(x_t, c)$ and $\epsilon_\theta(x_t, t, c)$ respectively.

1.4. Latent diffusion model

Latent diffusion model (LDM) compresses images from high-dimensional pixel space into low-dimensional feature space via pre-trained autoencoder, and builds diffusion model based on the latent feature space, such that computational overhead for both training and inference can be dramatically lowered. The training of LDM is similar to Eq. 10 except that we use notation z to denote latent features:

$$L = \|\epsilon_t - \epsilon_\theta(z_t, t, c)\|_2, \tag{16}$$

where $\epsilon_t \sim \mathcal{N}(0, \mathcal{I})$, $z_t = \sqrt{\bar{\alpha}_t}z_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t$, $z_0 = E(x_0)$, E is the pre-trained image encoder. The reverse denoising process from $z_T \sim \mathcal{N}(0, \mathcal{I})$ to z_0 is the same as $x_T \sim \mathcal{N}(0, \mathcal{I})$ to x_0 in DDPM. After reverse denoising process, the denoised clean features z_0 is decoded by the pre-trained decoder D to yield the finally generated image x_0, *i.e.*, $x_0 = D(z_0)$. In LDM framework, the condition c could be the extracted image features that are concatenated with x_t as the input of ϵ_θ for image-to-image translation applications, and also could be the encoded textual features that are interacted with x_t with cross-attention layers inside ϵ_θ for text-to-image synthesis task.

2. More qualitative results

Below we showcase more qualitative results of our PTDiffusion as a supplement to the main text. In Fig. 1 and Fig. 2, we display more results of hidden content discernibility control realized by varying the async distance parameter d in our APTM. In Fig. 3 and Fig. 4, we display more results demonstrating the sampling diversity property of our method, namely generating diversified illusion pictures with fixed reference image and text prompt. Finally, we present more optical illusion hidden pictures generated by our method in Fig. 5 to Fig. 15.

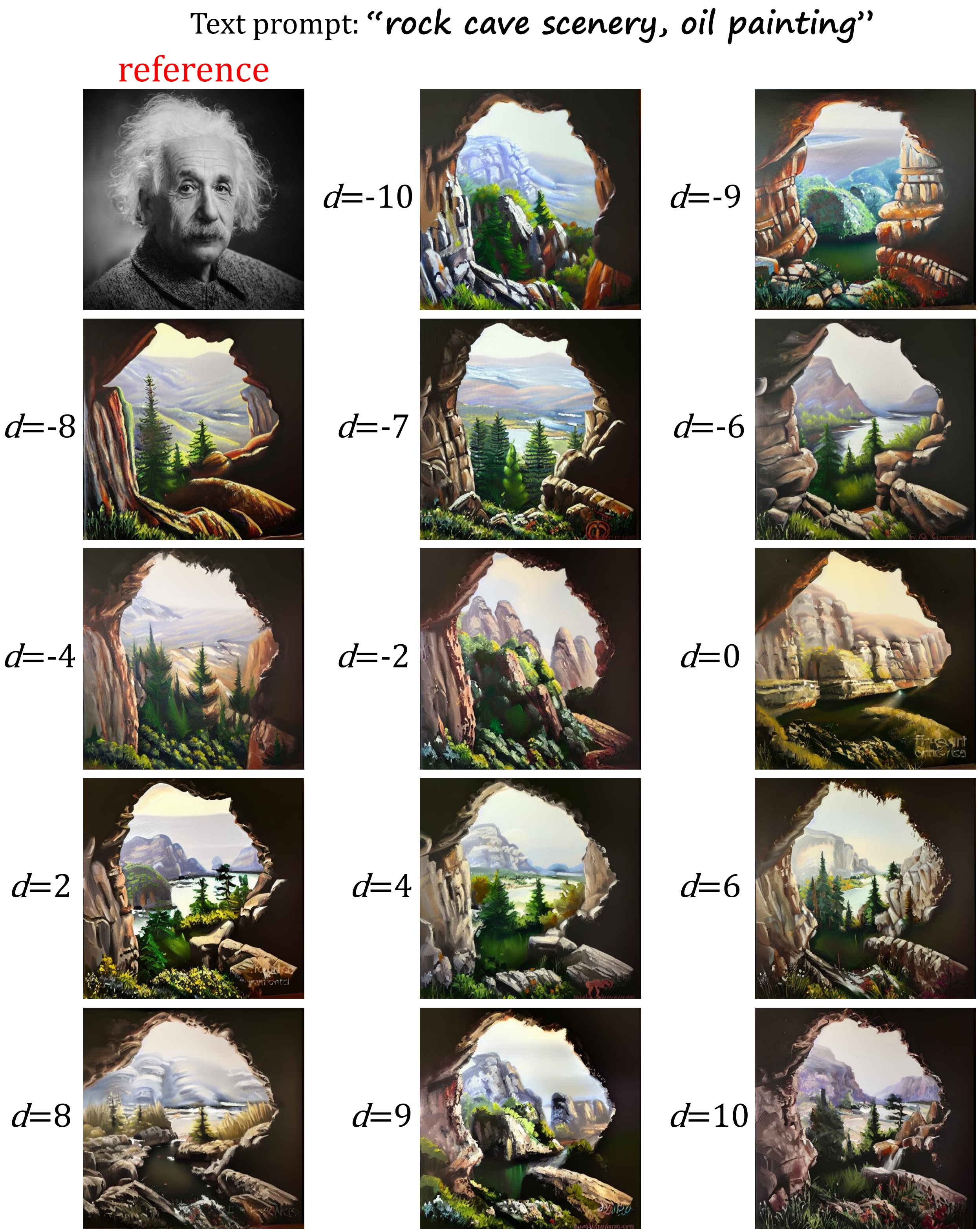

Figure 1. More results of hidden content discernibility control realized by varying the async distance parameter d in our method.

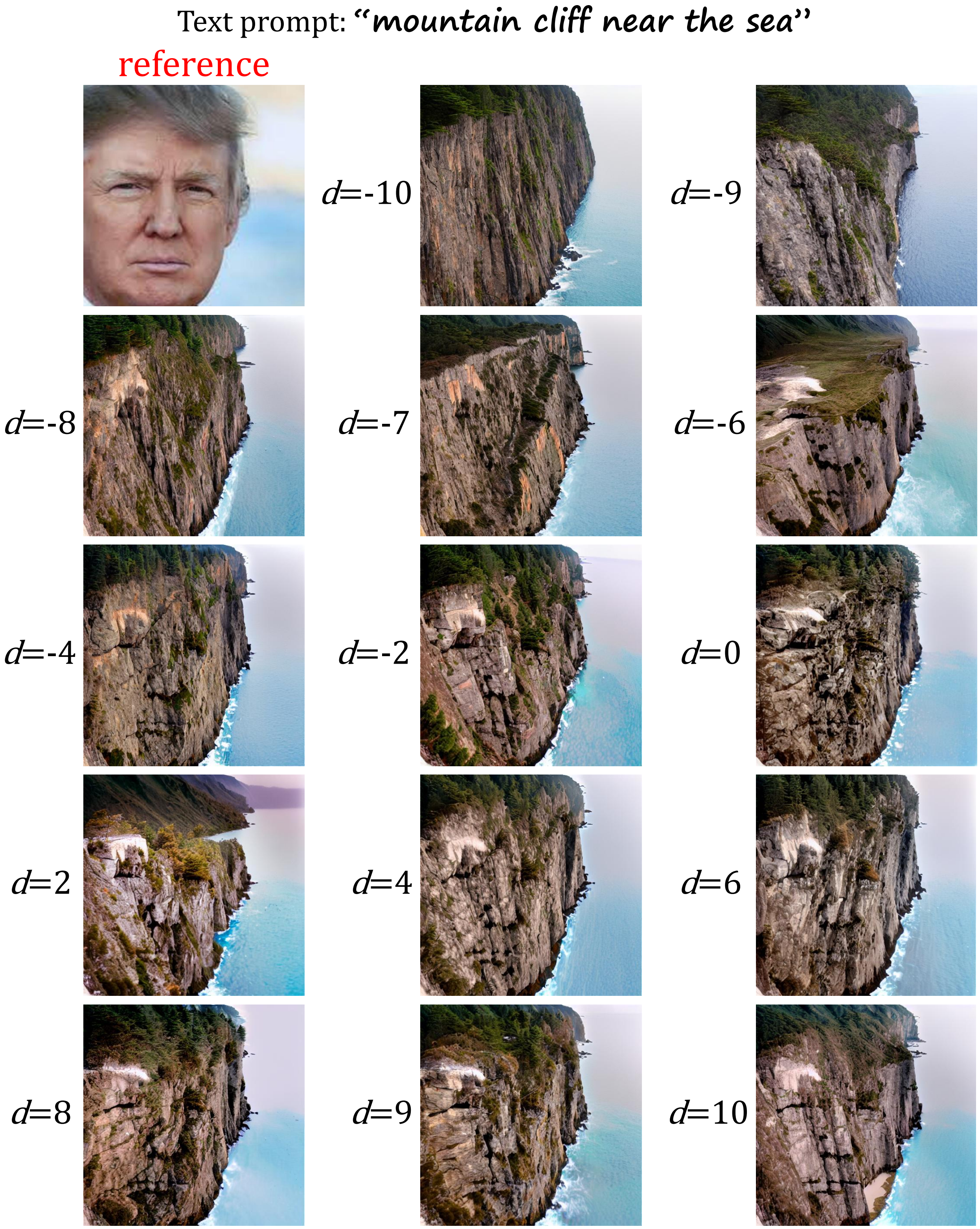

Figure 2. More results of hidden content discernibility control realized by varying the async distance parameter d in our method.

Figure 3. More examples of diversified sampling results of our method realized by varying the initial Gaussian noise $\tilde{z}_T$.

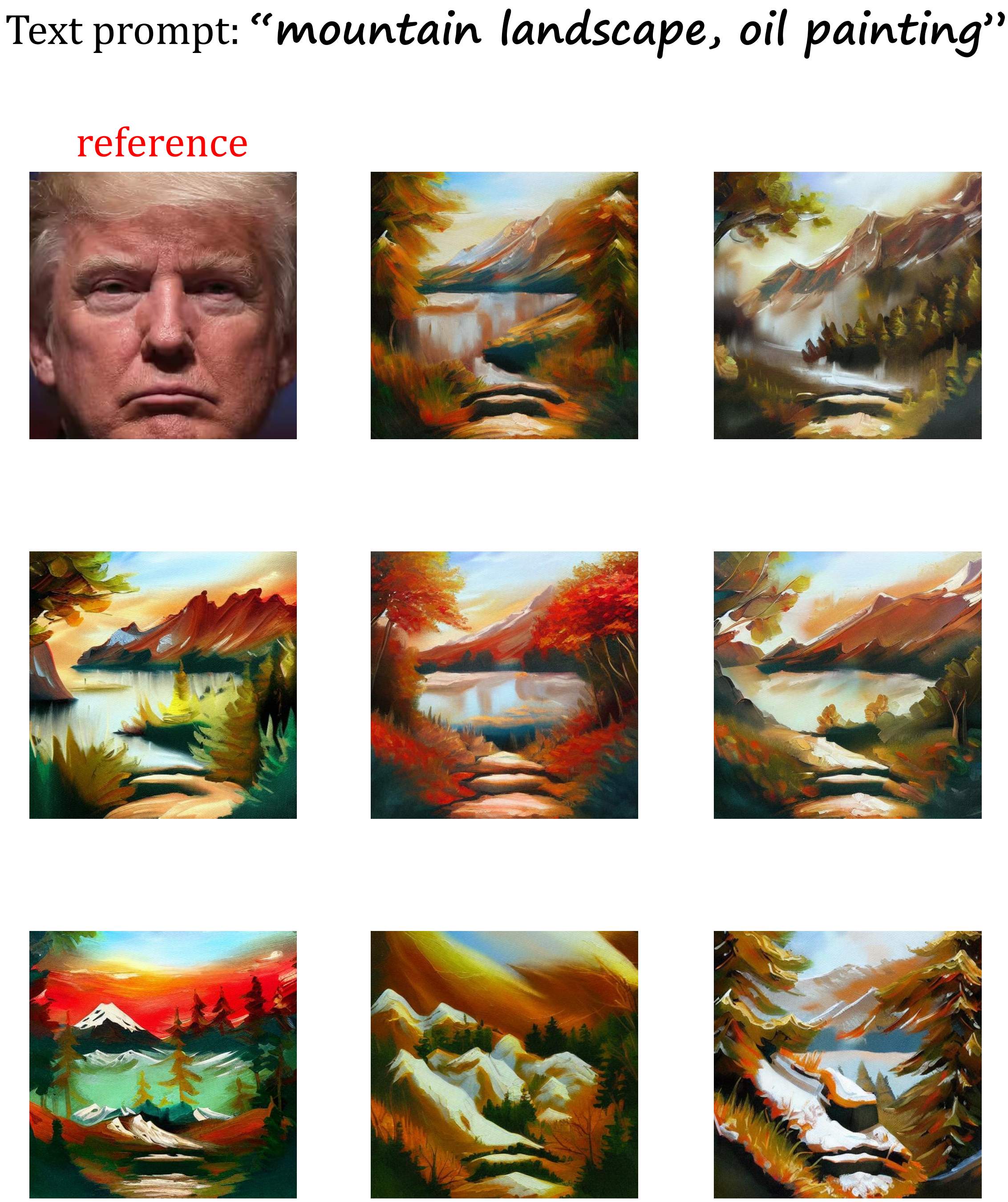

Figure 4. More examples of diversified sampling results of our method realized by varying the initial Gaussian noise $\tilde{z}_T$.

Figure 5. More qualitative results of our method.

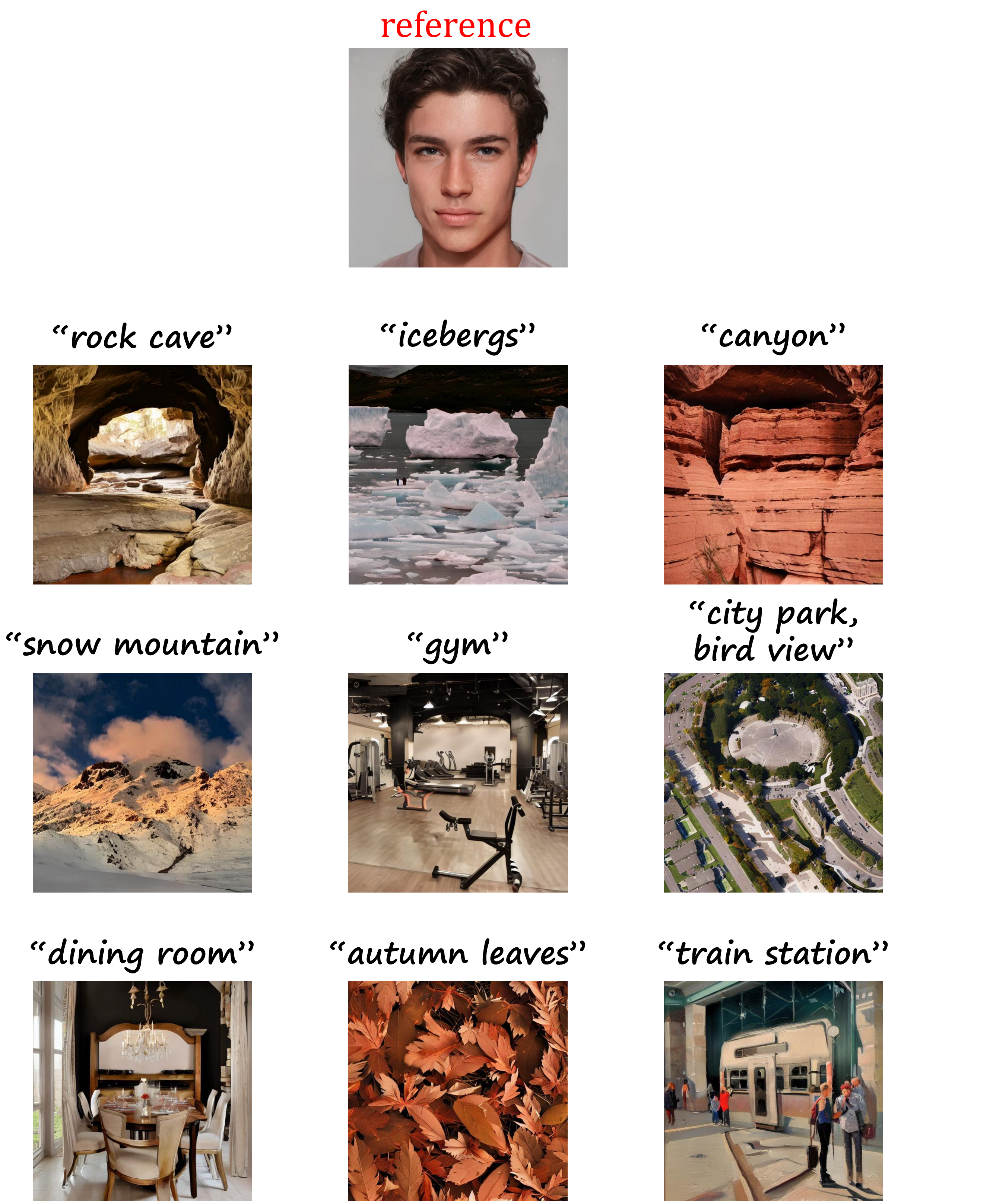

Figure 6. More qualitative results of our method.

Figure 7. More qualitative results of our method.

Figure 8. More qualitative results of our method.

Figure 9. More qualitative results of our method.

Figure 10. More qualitative results of our method.

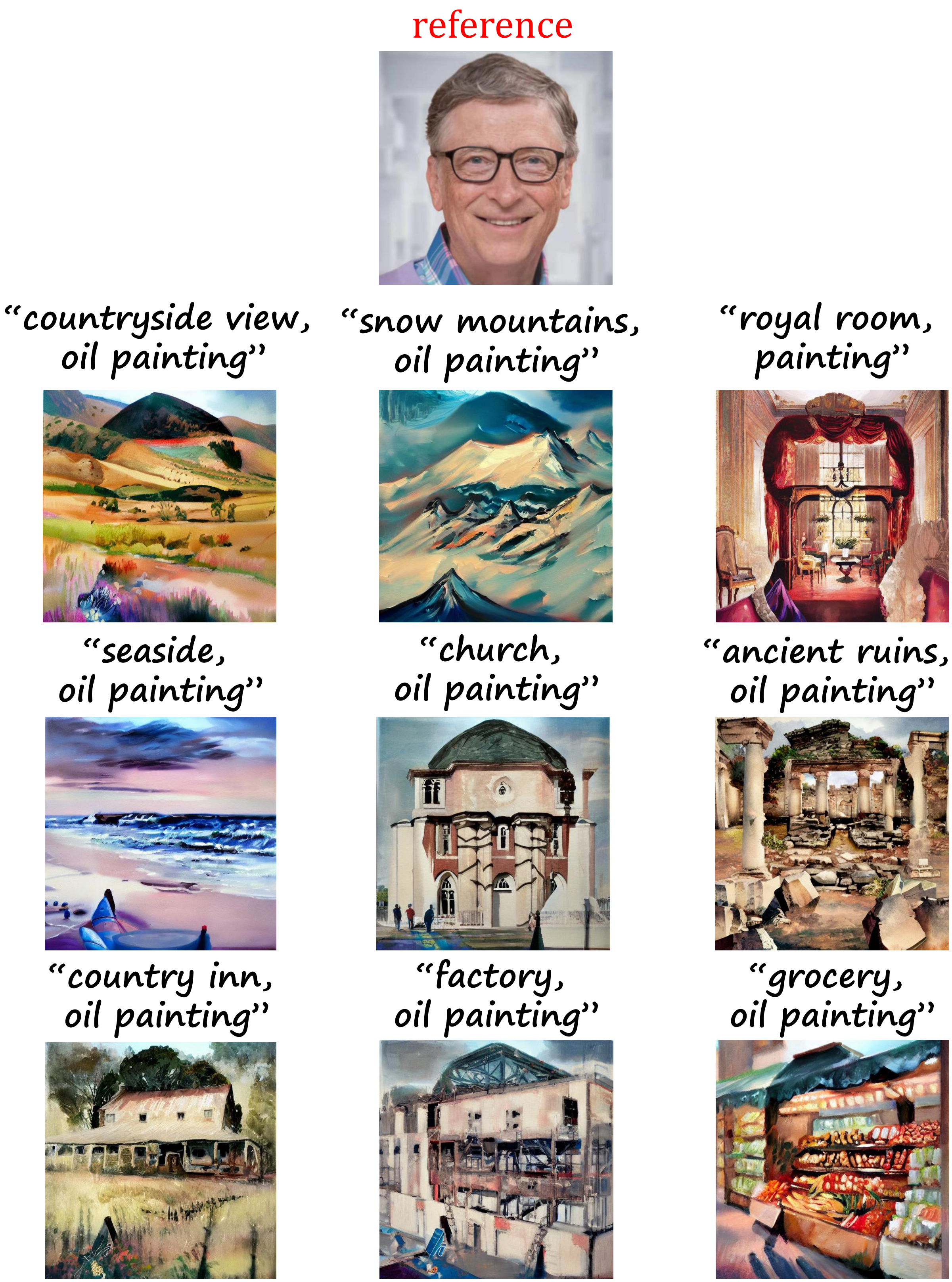

Figure 11. More qualitative results of our method.

Figure 12. More qualitative results of our method.

Figure 13. More qualitative results of our method.

Figure 14. More qualitative results of our method.

Figure 15. More qualitative results of our method.