

## Article

# Low-Noise Direct Denoising Sampling for DPM++ Samplers with Denoised Outputs

Zhigang Liang  and Jian Wang \* 

The College of Computer Science and Technology, Jilin University, No. 2699 Qianjin Street, Changchun 130012, China; liangzg22@mails.jlu.edu.cn

\* Correspondence: wangjian591@jlu.edu.cn

## Abstract

Diffusion models have been widely used in image generation and structured trajectory generation. DPM++ samplers improve few-step inference by recursively propagating current and historical denoised states, but their behavior in the low-noise regime remains less examined. This paper studies whether the recursive low-noise update is always necessary when the model-predicted denoised output has become relatively stable. Based on this analysis, we propose Low-Noise Direct Denoising Sampling (LNDDS), a training-free modification that replaces the recursive DPM++ update with the current denoised prediction during selected low-noise steps. LNDDS keeps the model architecture, training objective, noise schedule, number of sampling steps, and number of function evaluations unchanged. Experiments on MJHQ-30K image generation and HiLo-nuPlan traffic scenario generation show that the effect of LNDDS is domain-dependent. LNDDS does not provide consistent quality improvements for image generation under the evaluated setting, but it shows clear benefits for traffic scenario generation by improving trajectory quality, motion smoothness, and physical validity. Mechanism analysis further shows that low-noise direct updates affect sampling trajectories differently across domains. These results suggest that LNDDS is a task-dependent low-noise state-update strategy that is more suitable for structured trajectory generation than for general text-to-image generation.

**Keywords:** diffusion models; DPM++ samplers; denoised output; low-noise sampling; image generation; traffic scenario generation; autonomous driving

## 1. Introduction

Diffusion models have recently become an important paradigm in generative modeling and have demonstrated strong modeling capabilities in tasks such as image generation, video generation, trajectory prediction, and traffic scenario generation. The fundamental idea of diffusion models is to gradually perturb real data through a forward diffusion process and learn a progressive denoising mapping from the noise distribution to the data distribution during the reverse process. Existing studies have systematically formulated diffusion models from the perspectives of probabilistic modeling, continuous-time stochastic processes, and score matching, enabling diffusion models to evolve into a generative framework with stable training procedures and strong distribution modeling capabilities [1–3]. With the advancement of model architectures and training strategies, the application scope of diffusion models has expanded from early image generation tasks to high-resolution visual generation, video generation, multi-agent trajectory prediction, and autonomous driving traffic scenario generation [4–8]. For these tasks, generation quality depends not



Academic Editor: Andrey V. Savkin

Received: 29 July 2026

Revised: 23 September 2026

Accepted: 12 October 2026

Published: 13 October 2026

**Copyright:** © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

only on the modeling capability of the trained network but also strongly on the numerical behavior of the reverse sampling process [9–11]. In particular, for structured traffic scenario generation, errors accumulated during reverse sampling can directly affect trajectory continuity, multi-agent interaction rationality, and physical consistency [6,7].

Diffusion models typically require multiple iterative denoising steps to complete generation, making inference efficiency a long-standing limitation for practical deployment. To reduce sampling steps while maintaining generation quality, previous studies have proposed efficient samplers from perspectives such as non-Markovian sampling, pseudo numerical methods, high-order ODE solvers, and unified predictor-corrector frameworks [10–14]. These methods generally formulate the reverse diffusion process as a numerical solution problem of a continuous dynamical system or diffusion ODE, and improve few-step sampling performance through multi-step recursion, high-order approximation, or correction mechanisms. Among them, DPM-Solver++ (DPM++) [11] has become a widely used efficient sampler in practical diffusion-generation systems. It formulates reverse diffusion as the numerical integration of a diffusion ODE using model-predicted denoised outputs, while its multi-step variants reuse historical denoised predictions to improve stability and accuracy under few-step sampling settings. However, existing sampling methods mainly focus on numerical accuracy along the overall sampling trajectory, sampling step reduction, and global error control, while the internal update behavior of samplers at different noise stages remains insufficiently analyzed.

This paper focuses on the sampling behavior of DPM++ samplers in the low-noise regime. Existing samplers generally employ a unified recursive update strategy throughout the entire noise trajectory. However, in the low-noise stage, the current sample has already approached the data manifold, and the model-predicted denoised output gradually becomes stable. At this point, continuing to rely on historical sampling states from previous steps for recursive updates may introduce redundant dependence on residual noisy states. Since historical states are not completely clean, the remaining noise errors may continue to propagate during the later inference stages and affect the final generation results. In image generation tasks, such low-noise state changes may affect local appearance and fine-scale details, although such changes do not necessarily improve overall image quality. In traffic scenario generation tasks, they may manifest as trajectory oscillations, discontinuous motion patterns, unstable interactions, and reduced physical consistency. Based on this observation, this paper investigates the update mechanism in the low-noise stage, analyzes the relationship between model-predicted denoised outputs and DPM++ historical recursive terms, and proposes a Low-Noise Direct Denoising Sampling (LNDDS) method. When the noise level falls below a predefined threshold, the proposed method skips the recursive update of DPM++ and directly uses the model-predicted denoised output as the next sampling state, thereby reducing the propagation of residual noise through historical states in the low-noise regime.

## 2. Related Works

### 2.1. Diffusion Models and Applications

Diffusion models have become an important approach to generative modeling. DDPM establishes a probabilistic generation framework based on a forward noising process and a learned reverse denoising process [1]. Score-based Generative Modeling further provides a continuous-time formulation that unifies diffusion models through stochastic differential equations and probability flow ODEs [3]. These foundations have enabled diffusion models to be applied to multiple generation tasks, including high-resolution image synthesis [4], video generation [15], and structured multi-agent motion and traffic scenario generation [6,7].

In image generation, Latent Diffusion Models improve computational efficiency by performing the diffusion process in a learned latent space [4], while DiT improves model scalability by replacing the conventional U-Net backbone with a Transformer architecture [5]. Video Diffusion Models extend image-based diffusion architectures to jointly model spatial appearance and temporal dynamics, enabling the generation of temporally coherent video sequences [15]. For image generation, the generated object can generally be represented as  $x_0^{\text{img}} \in \mathbb{R}^{H \times W \times C}$ , and the model learns the conditional distribution  $p_\theta(x_0^{\text{img}} | c)$ , where  $c$  denotes a text prompt, class label, or other conditioning information.

Beyond visual-content generation, diffusion models have increasingly been applied to structured spatiotemporal tasks, particularly multi-agent trajectory prediction and autonomous-driving traffic scenario generation. MotionDiffuser models the joint distribution of multiple agents' future trajectories and supports controllable trajectory sampling through differentiable constraints [6]. Scenario Diffusion combines latent diffusion with map and scenario conditions to generate controllable agent poses, orientations, and trajectories for autonomous-driving simulation [7]. In these tasks, a generated traffic scenario can be represented as  $Y = \{s_i^t\}_{i=1:N, t=1:T_f}$ , where  $s_i^t = (x_i^t, y_i^t, \theta_i^t, v_i^t, \dots)$  denotes the motion state of the  $i$ -th agent at the  $t$ -th future timestep. Given historical trajectories  $H$  and map information  $M$ , the diffusion model learns the conditional distribution  $p_\theta(Y | H, M)$  to generate diverse future motions, controllable driving scenarios, and realistic multi-agent interactions [6,7].

Compared with image generation, traffic scenario generation imposes additional requirements on temporal continuity, agent-interaction rationality, map compliance, and physical consistency [6,7]. Across these application domains, generation quality depends not only on the network architecture and training objective but also on the numerical behavior of the reverse sampling process [10,11]. In contrast to existing task-oriented studies, this paper focuses on the inference-time behavior of DPM++(2M) in the low-noise regime and investigates how its recursive state-update rule affects the final generation quality.

## 2.2. Efficient Diffusion Samplers

Following this ODE perspective, DPM-Solver and DPM-Solver++ formulate diffusion sampling as a numerical solution problem of the diffusion ODE and improve few-step generation through high-order solvers [10,11]. DPM-Solver++ is designed for guided sampling and operates on model-predicted denoised outputs. It provides first-order, higher-order single-step, and higher-order multi-step variants, including DPM++(2S), DPM++(2M), and DPM++(3M). The single-step variants use intermediate model evaluations within each solver step, whereas the multi-step variants reuse denoised predictions from previous steps. This work focuses on the deterministic DPM++(2M) sampler, which is widely used for guided text-to-image generation and is adopted by Playground v2.5. UniPC further provides a unified predictor–corrector framework that improves the order of accuracy without additional model evaluations [14]. These methods have significantly improved diffusion inference efficiency and have become important components of modern diffusion-generation pipelines.

Despite their effectiveness, most existing efficient samplers mainly focus on global numerical accuracy, sampling step reduction, and quality-efficiency trade-offs along the entire denoising trajectory. The internal behavior of recursive samplers at different noise levels remains less explored. In particular, DPM++ samplers continue to propagate historical denoising states in the low-noise regime, where the model-predicted denoised output tends to become increasingly stable and close to the final clean sample. This motivates us to investigate whether such low-noise recursive updates introduce redundant dependence on residual noisy states.

### 3. Low-Noise Direct Denoising Sampling

This section presents the proposed Low-Noise Direct Denoising Sampling (LNDDS) strategy for DPM++ diffusion inference. We first revisit the denoised-output formulation and the recursive state update of DPM++ samplers. We then analyze the behavior of DPM++ updates in the low-noise regime and show that the recursive state carry may introduce redundant dependence on residual noisy states. Based on this analysis, we propose to replace the low-noise DPM++ update with the model-predicted denoised output when the noise level becomes sufficiently small.

#### 3.1. Problem Formulation

Let  $x_i \in \mathbb{R}^d$  denote the sampling state at noise level  $\sigma_i$ , where  $d$  is the state dimension. We use  $N$  sampling transitions, indexed by  $i = 0, \dots, N-1$ , to transform  $x_0$  into the final output  $x_N$ . The model is evaluated at the decreasing positive noise levels

$$\sigma_0 > \sigma_1 > \dots > \sigma_{N-1} > 0, \quad (1)$$

The last transition produces  $x_N$ ; when an implementation appends a terminal zero noise level, it is understood as the standard zero-noise-limiting update. This convention avoids evaluating logarithms at zero in the interior DPM-Solver++(2M) formulas below. Given a trained diffusion model, inference gradually transforms the initial noisy sample  $x_0$  into an estimate of the clean sample through these  $N$  updates. We define

$$D_\theta : \mathbb{R}^d \times \mathbb{R}_+ \times \mathcal{C} \rightarrow \mathbb{R}^d, \quad (2)$$

as the model-predicted denoising function, where  $\theta$  denotes the fixed parameters of the trained diffusion model and  $\mathcal{C}$  denotes the space of optional conditioning information. Given a noisy sample  $x_i$ , its noise level  $\sigma_i$ , and conditioning information  $c$ , the function  $D_\theta(x_i, \sigma_i; c)$  predicts the corresponding clean sample  $x_{\text{clean}}$ . Thus, the three arguments of Equation (2) are the current state, its noise level, and the condition, while the output has the same dimension as the state. For notational simplicity, the fixed conditioning variable  $c$  is omitted in the following derivations.

At sampling step  $i$ , we denote the model-predicted denoised output by

$$D_i := D_\theta(x_i, \sigma_i), \quad (3)$$

Thus,  $D_i \in \mathbb{R}^d$  is the model's current estimate of the clean sample at noise level  $\sigma_i$ , rather than the updated sampling state  $x_{i+1}$ . In this paper, we focus on samplers, such as DPM++, that explicitly use this model-predicted denoised output in their state-update rules.

We focus on the low-noise regime, where  $\sigma_i$  is close to zero. At this stage, the current sample has already approached the data manifold, and the model-predicted denoised output tends to become increasingly stable. However, DPM++ still performs recursive updates based on the current sampling state and historical denoising states. Since these states may contain residual errors, such recursive updates may continue to propagate state deviations into the final denoising stage.

#### 3.2. Denoised Outputs and DPM++ State Carry

DPM++ samplers update the sampling state through a recursive numerical solver. For an interior deterministic DPM-Solver++(2M) step with  $i \geq 1$  and  $\sigma_{i+1} > 0$ , the next state can be written in the denoised-output form

$$x_{i+1} = \alpha_i x_i + (1 - \alpha_i) \tilde{D}_i, \quad \alpha_i = \frac{\sigma_{i+1}}{\sigma_i}, \quad (4)$$

where  $\alpha_i \in (0, 1)$  is the fraction of the current noisy state carried to the next step and  $1 - \alpha_i$  is the weight assigned to the extrapolated denoised estimate  $\tilde{D}_i$ . For the second-order multi-step update, this estimate is

$$\begin{aligned}\tilde{D}_i &= \left(1 + \frac{1}{2r_i}\right)D_i - \frac{1}{2r_i}D_{i-1}, \\ r_i &= \frac{h_{i-1}}{h_i}, \\ h_i &= \log \frac{\sigma_i}{\sigma_{i+1}},\end{aligned}\tag{5}$$

Here,  $D_i = D_\theta(x_i, \sigma_i)$  denotes the current model-predicted denoised output, and  $D_{i-1}$  denotes the denoised output from the previous sampling step. The positive log-noise interval  $h_i$  is the step size from  $\sigma_i$  to  $\sigma_{i+1}$ , and  $r_i$  is the ratio of the preceding interval to the current interval. Consequently, the two coefficients in Equation (5) specify how the current and previous denoised predictions are combined. At the first transition, where  $D_{i-1}$  is unavailable, the standard first-order initialization is used; a terminal zero-noise transition is handled by the solver's standard limiting rule. Our analysis concerns the later low-noise multi-step updates for which Equation (5) is defined.

Equations (4) and (5) show that DPM++ does not only depend on the current denoised output, but also explicitly uses the current recursive state  $x_i$  and historical denoising information  $D_{i-1}$ . This stateful property is beneficial for high-order numerical accuracy, especially under few-step sampling settings. However, in the low-noise regime, when the denoised prediction has become relatively stable, continuing to propagate historical states may introduce redundant dependence on residual noisy states.

### 3.3. Low-Noise Error Propagation

We analyze how state deviations and denoising errors propagate through the DPM-Solver++(2M) recursion in the low-noise regime. Let  $\sigma_\tau > 0$  denote a predefined low-noise threshold. The first index inside this regime is

$$i^* = \min\{i : \sigma_i \leq \sigma_\tau\},\tag{6}$$

Thus, all updates with  $i \geq i^*$  use noise levels at or below the selected threshold. The state entering the low-noise regime is decomposed as

$$x_{i^*} = x_{i^*}^* + \delta_{i^*},\tag{7}$$

where  $x_{i^*}^*$  denotes a reference low-noise state and  $\delta_{i^*}$  denotes the residual state deviation accumulated before the low-noise boundary. This residual deviation may contain discretization errors, denoising prediction errors, and perturbations propagated from previous sampling steps.

To quantify its subsequent propagation, let  $x_i^*$  and  $D_i^*$  denote a reference state and its associated reference denoised output, respectively. We define the state and denoised-output deviations as

$$e_i = x_i - x_i^*, \quad \varepsilon_i = D_i - D_i^*,\tag{8}$$

In particular,  $e_{i^*} = \delta_{i^*}$ . For the DPM++ recursion analyzed next, the reference trajectory follows the same noise schedule and DPM-Solver++(2M) coefficients as the actual trajectory. Hence,  $e_i$  measures a perturbation between two trajectories governed by the same update rule, rather than the unknown error with respect to a ground-truth sample.

Subtracting the reference update from the actual update in Equations (4) and (5) gives the state-error recursion

$$e_{i+1} = \alpha_i e_i + (1 - \alpha_i) \left[ \left( 1 + \frac{1}{2r_i} \right) \varepsilon_i - \frac{1}{2r_i} \varepsilon_{i-1} \right], \quad (9)$$

Equation (9) identifies two error-propagation channels. First, the state-carry term  $\alpha_i e_i$  recursively transfers the current state deviation. Second, the multi-step term involving  $\varepsilon_{i-1}$  transfers the deviation of the previous denoised prediction across sampling steps. The term involving  $\varepsilon_i$  is the effect of the current denoised-output deviation. Equation (9) is an identity under the paired-trajectory construction above; the inequalities below arise only after norm bounds are applied.

We assume that the denoising map is locally Lipschitz around the reference trajectory:

$$\|\varepsilon_i\|_2 \leq L_i \|e_i\|_2 + b_i, \quad (10)$$

where  $L_i$  measures the local sensitivity of the denoising map to its input state, and  $b_i \geq 0$  accounts for any discrepancy between the model prediction at the reference state and the selected reference denoised output. If  $D_i^* = D_\theta(x_i^*, \sigma_i)$ , local Lipschitz continuity gives the bound with  $b_i = 0$ . Otherwise,  $b_i$  absorbs that reference or model-approximation mismatch. Separating  $L_i$  and  $b_i$  distinguishes amplification of an input-state perturbation from an additive discrepancy.

Applying the triangle inequality to Equation (9) gives

$$\|e_{i+1}\|_2 \leq A_i \|e_i\|_2 + B_i \|e_{i-1}\|_2 + c_i, \quad (11)$$

where

$$\begin{aligned} A_i &= |\alpha_i| + |1 - \alpha_i| \left| 1 + \frac{1}{2r_i} \right| L_i, \\ B_i &= |1 - \alpha_i| \left| \frac{1}{2r_i} \right| L_{i-1}, \\ c_i &= |1 - \alpha_i| \left[ \left| 1 + \frac{1}{2r_i} \right| b_i + \left| \frac{1}{2r_i} \right| b_{i-1} \right], \end{aligned} \quad (12)$$

All three coefficients are nonnegative. Here,  $A_i$  quantifies propagation through the state-carry and current denoising terms;  $B_i$  quantifies the additional dependence on the previous step; and  $c_i$  collects the additive discrepancies  $b_i$  and  $b_{i-1}$ . Therefore, a non-negligible  $B_i$  can continue to carry historical deviations into the final sampling steps.

For a multi-step interpretation, define

$$\mathbf{u}_i = \begin{bmatrix} \|e_i\|_2 \\ \|e_{i-1}\|_2 \end{bmatrix}, \quad \mathbf{M}_i = \begin{bmatrix} A_i & B_i \\ 1 & 0 \end{bmatrix}, \quad \mathbf{q}_i = \begin{bmatrix} c_i \\ 0 \end{bmatrix}, \quad (13)$$

Equation (11) can then be expressed component-wise as

$$\mathbf{u}_{i+1} \leq \mathbf{M}_i \mathbf{u}_i + \mathbf{q}_i, \quad (14)$$

where  $\leq$  denotes element-wise inequality. The augmented vector  $\mathbf{u}_i$  stores the current and previous error magnitudes,  $\mathbf{M}_i$  advances this two-step recursion, and  $\mathbf{q}_i$  injects the additive discrepancy. After  $m$  low-noise DPM++ updates, repeated application yields

$$\begin{aligned} \mathbf{u}_{i^*+m} &\leq \mathbf{M}_{i^*+m-1} \cdots \mathbf{M}_{i^*} \mathbf{u}_{i^*} \\ &\quad + \sum_{j=0}^{m-1} (\mathbf{M}_{i^*+m-1} \cdots \mathbf{M}_{i^*+j+1}) \mathbf{q}_{i^*+j}, \end{aligned} \quad (15)$$

where an empty matrix product is interpreted as the identity matrix. Equation (15) quantifies how the initial residual deviation  $\delta_{i^*}$  and subsequent denoising errors propagate through the remaining low-noise steps. In particular, the product of the matrices  $\mathbf{M}_i$  determines whether the initial deviation is attenuated or amplified, whereas each summand propagates the discrepancy introduced at step  $i^* + j$  through all later steps. Matrix products are ordered chronologically from right to left.

In contrast, LNDDS directly sets the next state to the current denoised prediction. To distinguish this trajectory from the DPM++ trajectory above, we use the superscript  $L$ :

$$x_{i+1}^L = D_\theta(x_i^L, \sigma_i), \quad (16)$$

Let  $x_{i+1}^{*,L} = D_i^*$ ,  $e_i^L = x_i^L - x_i^{*,L}$ , and  $\varepsilon_i^L = D_\theta(x_i^L, \sigma_i) - D_i^*$ . Relative to this rule-matched reference, the local error satisfies

$$\|e_{i+1}^L\|_2 = \|\varepsilon_i^L\|_2 \leq L_i \|e_i^L\|_2 + b_i, \quad (17)$$

Compared with Equation (11), the LNDDS update removes both the explicit state-carry coefficient  $\alpha_i$  and the historical error term associated with  $e_{i-1}$ . It therefore prevents previous denoised outputs from being explicitly reused after the low-noise switching point. Because the bounds use reference trajectories matched to their respective update rules, this comparison isolates propagation structure; it does not by itself prove lower error with respect to the data distribution.

We next analyze how the number of direct-denoising steps affects the propagation of low-noise errors. Recall that  $N$  is the total number of sampling transitions, and let  $K \in \{0, 1, \dots, N\}$  denote the number of final transitions using LNDDS. The baseline is  $K = 0$ . For  $K \geq 1$ , the switching index is

$$i_\tau = N - K, \quad (18)$$

Therefore, selecting  $K \geq 1$  under a fixed schedule is equivalent to selecting the low-noise threshold

$$\sigma_\tau = \sigma_{N-K}, \quad (19)$$

For comparisons across different total numbers of sampling steps, we additionally define the normalized direct-denoising ratio as

$$\rho = \frac{K}{N}, \quad (20)$$

Here,  $\rho \in [0, 1]$  is the fraction of sampling transitions replaced by direct denoising. Reporting both  $K$  and  $\rho$  avoids interpreting the same absolute  $K$  as the same range under different discretizations.

Repeatedly applying Equation (17) over the final  $K$  sampling steps gives

$$\begin{aligned} \|e_N^L\|_2 &\leq \left( \prod_{i=N-K}^{N-1} L_i \right) \|e_{N-K}^L\|_2 \\ &\quad + \sum_{j=N-K}^{N-1} \left( \prod_{\ell=j+1}^{N-1} L_\ell \right) b_j, \end{aligned} \quad (21)$$

where an empty product is interpreted as one. The first term in Equation (21) describes the propagation of the LNDDS-referenced deviation present at the switching point, while the

second term describes the accumulation of denoising approximation errors over the selected  $K$  steps. Each  $b_j$  is multiplied by the sensitivities of all subsequent direct-denoising steps.

For a more interpretable special case, suppose that

$$L_i \leq \bar{L}, \quad b_i \leq \bar{b}, \quad i = N - K, \dots, N - 1, \quad (22)$$

If  $0 \leq \bar{L} < 1$ , then

$$\|e_N^L\|_2 \leq \bar{L}^K \|e_{N-K}^L\|_2 + \bar{b} \frac{1 - \bar{L}^K}{1 - \bar{L}}, \quad (23)$$

Equation (23) shows that increasing  $K$  can attenuate the state deviation at the switching boundary when the denoising map is locally contractive. However, increasing  $K$  also includes earlier and potentially less stable denoising steps and accumulates more model-approximation errors. If some of the selected steps have  $L_i \geq 1$ , or if their corresponding  $b_i$  values are large, the contraction condition no longer holds and the final error may increase. The boundary term decays as  $\bar{L}^K$ , whereas the additive term approaches the error floor  $\bar{b}/(1 - \bar{L})$ . Thus, even in the contractive case, increasing  $K$  helps only when boundary-error reduction outweighs the additional accumulated discrepancy.

Consequently, the error bound is not necessarily monotonic with respect to  $K$ . A small  $K$  may not sufficiently remove the historical-state dependence of DPM-Solver++(2M), whereas an excessively large  $K$  may switch to direct denoising before the denoised prediction has become stable. The appropriate value of  $K$  therefore depends on the local stability of the denoising model, the noise schedule, the total number of sampling steps, and the structure of the generation task.

The comparison between Equation (11) and (21) do not imply that LNDDS always has a smaller global sampling error than DPM-Solver++(2M). Direct denoising is expected to be beneficial only when the selected low-noise denoising maps are sufficiently stable and the historical-error contribution  $B_i \|e_{i-1}\|_2$  is larger than the approximation error introduced by the direct updates. If multi-step corrections are required to recover fine-scale information, increasing  $K$  may instead degrade sample quality. This conditional result is consistent with the different behaviors observed in image generation and structured traffic trajectory generation.

The domain difference may also be interpreted from a frequency perspective, although the present experiments do not establish a causal frequency-domain mechanism. In images, low-noise corrections may continue to refine spatial high-frequency content such as edges and textures; removing them too early can reduce fidelity. In traffic trajectories, some temporal high-frequency variation instead corresponds to acceleration oscillation or jitter, so suppressing it can improve motion regularity. This interpretation further motivates selecting  $K$  separately for each domain.

Because the reference state  $x_i^*$ , the local sensitivity  $L_i$ , and the approximation error  $b_i$  are unavailable during practical sampling, we use observable quantities as empirical proxies. The denoised-prediction change is defined as

$$C_i = \frac{\|D_i - D_{i-1}\|_2}{\|D_i\|_2 + \epsilon}, \quad (24)$$

and the DPM++ update deviation is defined as

$$\Delta_i = \frac{\|x_{i+1}^{\text{DPM++}} - D_i\|_2}{\|D_i\|_2 + \epsilon}, \quad (25)$$

Here,  $\epsilon > 0$  is a fixed constant that prevents division by zero. The quantity  $C_i$  measures the relative change between consecutive denoised predictions, whereas  $\Delta_i$  measures the relative correction introduced by the DPM++ update, with  $x_{i+1}^{\text{DPM++}}$  computed by Equations (4)–(5). Both are dimensionless diagnostics; neither is a sample-quality metric or a guarantee that switching will improve the output.

In practice, we recommend selecting  $K$  on a held-out validation set. Given  $M$  validation samples, the average denoised-prediction change at step  $i$  is

$$\bar{C}_i = \frac{1}{M} \sum_{m=1}^M C_i^{(m)}, \quad (26)$$

where  $C_i^{(m)}$  is computed for validation sample  $m$ . Averaging yields a step-wise stability profile for the target domain. A candidate low-noise switching point can then be identified as

$$i_\tau^{\text{cand}} = \min\{i : \bar{C}_j \leq \tau_C, \quad \forall j = i, \dots, N-1\}, \quad (27)$$

where  $\tau_C > 0$  is determined using validation data. The universal quantifier requires stability at every subsequent step. If the set in Equation (27) is empty, we set  $K_{\text{cand}} = 0$  and retain the baseline. Otherwise, the maximum candidate range is

$$K_{\text{cand}} = N - i_\tau^{\text{cand}}, \quad (28)$$

Thus,  $K_{\text{cand}}$  is a screening bound, not a claim that the largest admissible  $K$  is optimal. The final value is selected from the candidate set using a domain-specific validation objective:

$$K^* = \arg \min_{K \in \mathcal{K}_{\text{cand}}} \mathcal{J}_{\text{val}}(K), \quad (29)$$

where  $\mathcal{K}_{\text{cand}} = \{K \in \mathcal{K} : K \leq K_{\text{cand}}\}$ , and  $\mathcal{K}$  is a prespecified finite set including  $K = 0$ . The scalar  $\mathcal{J}_{\text{val}}$  is the primary domain-specific validation objective; if metrics are aggregated, their weights must be fixed before test-set evaluation. Once selected,  $K^*$  is held fixed for testing.

This procedure should be interpreted as a practical selection guideline rather than a universally optimal rule. Image generation and structured trajectory generation may enter stable denoising regimes at different noise levels and may assign different importance to low-noise corrections. Therefore, the effective direct-denoising range must be determined separately for each domain. The step-wise and aggregated behaviors of  $C_i$ ,  $\Delta_i$ , and different values of  $K$  are evaluated in Section 4.

### 3.4. Low-Noise Direct Denoising Sampling

Based on the above analysis, we propose Low-Noise Direct Denoising Sampling (LNDDS). The core idea is simple: when the noise level is higher than a predefined threshold, we keep the standard DPM++ update; when the noise level falls below the threshold, we skip the recursive DPM++ update and directly use the model-predicted denoised output as the next sampling state. Formally, the proposed sampling rule is

$$x_{i+1} = \begin{cases} \mathcal{S}_{\text{DPM++}}(x_i, \sigma_i, \sigma_{i+1}; D_{i-1}), & \sigma_i > \sigma_\tau, \\ D_\theta(x_i, \sigma_i), & \sigma_i \leq \sigma_\tau, \end{cases} \quad (30)$$

where  $\mathcal{S}_{\text{DPM++}}$  denotes the standard DPM++ update and  $\sigma_\tau$  is the switching threshold. The semicolon separates the historical output  $D_{i-1}$  from the current state and noise levels;

the current output  $D_i$  is given by Equation (3). Equivalently, for the last  $K$  transitions, the first LNDDS update is at  $i = N - K$ .

The proposed strategy removes the historical-state dependency of DPM++ in the low-noise regime. In Equation (30), the low-noise update becomes a local denoising step that only depends on the current sample and noise level (and the fixed condition omitted from the notation). Therefore, it avoids the explicit recursive state-carry term  $\alpha_i x_i$  and the multi-step extrapolation term involving  $D_{i-1}$ . This reduces the repeated propagation of residual state deviations through recursive low-noise updates.

It is important to note that LNDDS is not an early stopping strategy. The sampling process still proceeds through the same noise schedule and uses the same number of denoising steps. The only difference is the state-update rule used after the noise level falls below  $\sigma_\tau$ . Therefore, LNDDS does not require any modification to the network architecture, training objective, or learned parameters. It only changes the sampling rule during inference and is compatible with diffusion samplers that provide a model-predicted denoised output. In image generation, this modification may change local appearance and fine-scale details, although such changes do not necessarily translate into better distribution-level quality. In traffic scenario generation, it can help reduce trajectory jitter and improve motion continuity under the evaluated setting.

## 4. Experiments

We evaluate LNDDS in two domains: text-to-image generation and multi-agent traffic scenario generation. The goal is not to change the model architecture or training objective, but to test the effect of replacing the low-noise DPM++ recursive state update with the current model-predicted denoised output during inference. The baseline and LNDDS use the same model weights, conditions, random seeds, initial noise, sigma schedule, number of sampling steps, and number of network evaluations. Therefore, the reported differences are caused only by the low-noise state-update rule.

In the following tables, LNDDS- $K$  denotes using direct denoising in the last  $K$  low-noise sampling steps, while the baseline, equivalently LNDDS-0, keeps the original DPM++ update for all sampling steps. We evaluate  $K \in \{2, 5, 7, 10\}$ . Under a fixed sigma schedule, selecting the last  $K$  sampling steps is equivalent to choosing  $\sigma_\tau$  as the largest noise level among those steps. Because the total number of sampling steps can vary, we interpret  $K$  together with the normalized switching ratio  $K/N$ , where  $N$  is the total number of inference steps.

### 4.1. Experimental Setup

#### 4.1.1. Image Generation

We use Playground v2.5 [16] as the text-to-image generation model. The model weights, text encoders, variational autoencoder, classifier-free guidance configuration, scheduler configuration, and sigma schedule are kept unchanged for all methods. The evaluation is conducted on MJHQ-30K [16], which contains 30,000 prompts and corresponding reference images across ten semantic categories. For each prompt, all methods use the same random seed and the same initial latent. The default image resolution is  $1024 \times 1024$ , the guidance scale is 3.0, and the number of inference steps is 30.

#### 4.1.2. Traffic Scenario Generation

For traffic scenario generation, we use HiLo, a diffusion-based multi-agent scenario generation model trained on approximately one million nuPlan [17] training samples. A single HiLo checkpoint is used for all experiments, and LNDDS is applied only during inference. The evaluation is performed on a held-out nuPlan test set containing 15,000

scenarios. For each scenario, all methods use the same historical trajectories, map inputs, agent masks, random seed, initial noise, sigma schedule, and number of denoising steps. Unless otherwise stated, the main traffic results use 20 inference steps.

The experimental protocols for both image generation and traffic scenario generation, including the models, datasets, evaluation sizes, inference steps, and compared methods, are summarized in Table 1.

**Table 1.** Summary of the experimental protocol.

| Domain  | Model           | Dataset  | Evaluation Size | Steps | Compared Methods                              |
|---------|-----------------|----------|-----------------|-------|-----------------------------------------------|
| Image   | Playground v2.5 | MJHQ-30K | 30,000 prompts  | 30    | Baseline, LNDDS-2, LNDDS-5, LNDDS-7, LNDDS-10 |
| Traffic | HiLo            | nuPlan   | 15,000 scenes   | 20    | Baseline, LNDDS-2, LNDDS-5, LNDDS-7, LNDDS-10 |

## 4.2. Evaluation Metrics

### 4.2.1. Image Metrics

We report five image generation metrics. FID [18] and CMMD [19] evaluate distributional consistency between generated and reference images, where lower values are better. ImageReward [20] (IR) estimates prompt-conditioned human preference, where higher values are better. CLIP Score [21] (CS) measures text–image semantic alignment, and Aesthetic Score [22] (AS) estimates visual appeal; higher values are better for both metrics.

### 4.2.2. Traffic Metrics

For trajectory accuracy, we report average displacement error (ADE) and final displacement error (FDE):

$$\text{ADE} = \frac{1}{AT} \sum_{a=1}^A \sum_{t=1}^T \|\hat{\mathbf{p}}_a^t - \mathbf{p}_a^t\|_2, \quad (31)$$

$$\text{FDE} = \frac{1}{A} \sum_{a=1}^A \|\hat{\mathbf{p}}_a^T - \mathbf{p}_a^T\|_2, \quad (32)$$

Here,  $A$  is the number of valid agents,  $T$  is the number of predicted time steps, and  $\hat{\mathbf{p}}_a^t$  and  $\mathbf{p}_a^t$  are the generated and recorded two-dimensional positions of agent  $a$  at time  $t$ . Equation (31) averages over all valid agent–time pairs, whereas Equation (32) evaluates only the last predicted time step. We use  $A$ , rather than  $N$ , to avoid conflict with the number of sampling transitions in Section 3. We also report mean jerk, a smoothness score, and vehicle collision rate. Lower values are better for all traffic metrics in the reported tables.

### 4.2.3. Low-Noise Mechanism Metrics

To inspect low-noise sampling, we record three dimensionless diagnostics of the intermediate states. The denoised-prediction change from Equation (24) is evaluated as

$$C_i = \frac{\|D_i - D_{i-1}\|_2}{\|D_i\|_2 + \epsilon}, \quad (33)$$

where a smaller value indicates that consecutive model-predicted denoised outputs are closer. The DPM++ update deviation from Equation (25) is evaluated as

$$\Delta_i = \frac{\|x_{i+1}^{\text{DPM++}} - D_i\|_2}{\|D_i\|_2 + \epsilon}, \quad (34)$$

which measures the relative distance between the original DPM++ next state and the direct-denoising state  $D_i$ . Finally, the cross-sampler state divergence is

$$E_i = \frac{\|x_i^L - x_i^B\|_2}{\|x_i^B\|_2 + \epsilon}, \quad (35)$$

where  $x_i^B$  and  $x_i^L$  are paired baseline and LNDDS states obtained using the same condition and initial noise. The constant  $\epsilon > 0$  is the same as in Equations (24) and (25). The metric  $E_i$  measures the accumulated relative state difference and is zero before the samplers diverge. None of  $C_i$ ,  $\Delta_i$ , and  $E_i$  directly measures perceptual quality, trajectory accuracy, or physical validity.

### 4.3. Main Results

#### 4.3.1. Image Generation on MJHQ-30K

Table 2 shows that the baseline obtains the best FID, CMMD, and IR on MJHQ-30K. With  $K = 2$ , CS and AS increase from 29.348 and 6.817 to 29.472 and 6.861, respectively, while FID, CMMD, and IR become worse. The LNDDS-5 setting obtains the best CS (29.648) and AS (6.921), but further degrades the three distribution- and preference-level metrics. Increasing  $K$  to 7 and 10 produces progressively larger degradation, with FID increasing to 19.107 and 30.390, respectively. These results reveal a non-monotonic, metric-dependent trend: a small or moderate direct-denoising range can slightly improve semantic alignment and aesthetics, whereas an excessively large  $K$  substantially degrades the overall image distribution.

**Table 2.** Image generation performance on MJHQ-30K. All methods use the same prompts, random seeds, initial latents, model weights, sampling steps, and number of network evaluations. Best results are shown in bold.

| Method   | FID ↓        | CMMD ↓        | IR ↑         | CS ↑          | AS ↑         |
|----------|--------------|---------------|--------------|---------------|--------------|
| Baseline | <b>7.179</b> | <b>0.1001</b> | <b>1.111</b> | 29.348        | 6.817        |
| LNDDS-2  | 8.872        | 0.1227        | 1.077        | 29.472        | 6.861        |
| LNDDS-5  | 11.407       | 0.1567        | 1.050        | <b>29.648</b> | <b>6.921</b> |
| LNDDS-7  | 19.107       | 0.2685        | 0.912        | 29.258        | 6.832        |
| LNDDS-10 | 30.390       | 0.4311        | 0.679        | 28.698        | 6.691        |

Note: ↓ indicates that lower values are better, while ↑ indicates that higher values are better. Bold values indicate the best performance for each metric.

#### 4.3.2. Traffic Scenario Generation on nuPlan

Table 3 summarizes the traffic generation results. As  $K$  increases from 2 to 5, 7, and 10, ADE decreases from 1.918 to 1.854, 1.752, and 1.588, while FDE decreases from 4.636 to 4.571, 4.383, and 4.089, respectively. Mean jerk and collision rate show a similar overall improvement. Smoothness is not strictly monotonic: LNDDS-2 increases the smoothness cost from 0.047 to 0.053, whereas LNDDS-5, LNDDS-7, and LNDDS-10 reduce it to 0.038, 0.040, and 0.028, respectively. Among the tested settings, LNDDS-10 obtains the best values for all reported traffic metrics, reducing mean jerk from 106.742 to 12.537 and collision rate from 0.407% to 0.255%. Under this HiLo-nuPlan setting, a wider low-noise direct-denoising range is generally associated with lower trajectory error, smoother motion, and fewer vehicle collisions. This conclusion is empirical and should not be generalized beyond the evaluated model, dataset, and sampling configuration without further experiments.

**Table 3.** Traffic scenario generation results on the held-out nuPlan test set using 20 inference steps. The same HiLo checkpoint, scene conditions, random seeds, initial noise, and sampling steps are used for all methods.

| Method   | ADE ↓        | FDE ↓        | Jerk ↓        | Smoothness ↓ | Collision (%) ↓ |
|----------|--------------|--------------|---------------|--------------|-----------------|
| Baseline | 1.954        | 4.663        | 106.742       | 0.047        | 0.407           |
| LNDDS-2  | 1.918        | 4.636        | 70.174        | 0.053        | 0.342           |
| LNDDS-5  | 1.854        | 4.571        | 15.333        | 0.038        | 0.366           |
| LNDDS-7  | 1.752        | 4.383        | 14.220        | 0.040        | 0.331           |
| LNDDS-10 | <b>1.588</b> | <b>4.089</b> | <b>12.537</b> | <b>0.028</b> | <b>0.255</b>    |

Note: ↓ indicates that lower values are better, while ↑ indicates that higher values are better. Bold values indicate the best performance for each metric.

#### 4.4. Sensitivity to the Number of Direct-Denoising Steps

Figure 1 visualizes the relative metric changes reported in Tables 2 and 3. The figure uses relative improvement instead of raw metric values because the image and traffic metrics have different numerical scales. In image generation, CS and AS initially improve and reach their best values at  $K = 5$ , whereas FID, CMMD, and IR deteriorate as  $K$  increases. The degradation becomes substantially larger at  $K = 7$  and  $K = 10$ , indicating that direct denoising is activated before all fine-scale image refinements have stabilized. In traffic generation, ADE, FDE, jerk, and collision rate generally improve as  $K$  increases, and  $K = 10$  produces the best overall results among the evaluated settings. Smoothness shows a small non-monotonic variation at  $K = 2$  and  $K = 7$ , but its best value is also obtained at  $K = 10$ . These different trends demonstrate that  $K$  is a domain-dependent hyperparameter rather than a universally transferable constant. In practice,  $K$  should be selected on a held-out validation set using the primary quality and validity metrics of the target domain.

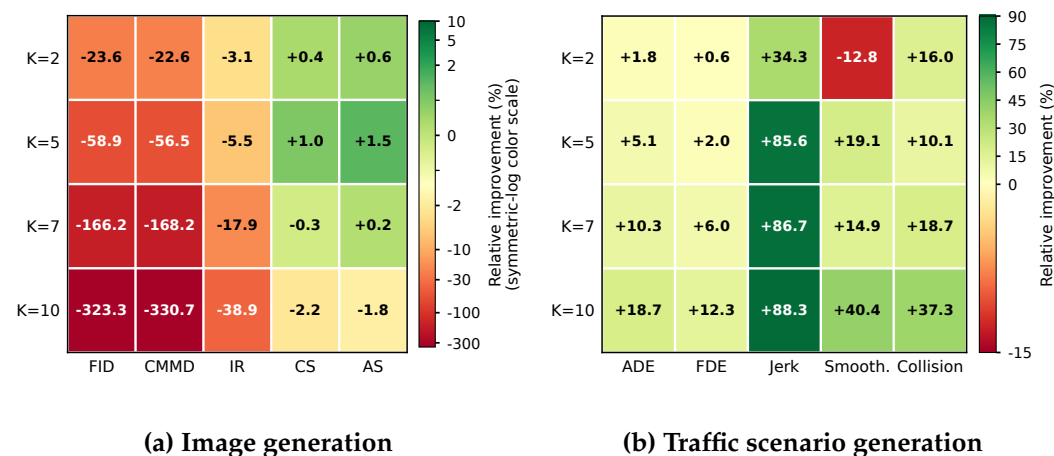


**Figure 1.** Relative improvement of LNDDS-2, LNDDS-5, LNDDS-7, and LNDDS-10 over the baseline in image generation and traffic scenario generation. Positive values indicate better performance after accounting for the preferred direction of each metric. Image metrics exhibit a non-monotonic and metric-dependent response to  $K$ , whereas the traffic metrics generally improve as the direct-denoising range increases.

#### 4.5. Low-Noise Mechanism Analysis

Table 4 reports the low-noise mechanism statistics. In the image domain, the averaged  $C_i$  and  $\Delta_i$  decrease from the baseline to LNDDS-2 and LNDDS-5, reaching their smallest values of 0.0056 and 0.0088 at  $K = 5$ . However, they increase to (0.0193, 0.0451) at  $K = 7$  and (0.0404, 0.0992) at  $K = 10$ . Meanwhile, the final state divergence increases from 0.0111 at  $K = 2$  to 0.0279, 0.1733, and 0.3918 as  $K$  increases to 5, 7, and 10. This rapid increase beyond  $K = 5$  is consistent with the degradation of FID, CMMD, and IR in Table 2. In the

traffic domain, all four LNDDS settings reduce the DPM++ update deviation relative to the baseline value of 1.3018. The deviation drops to 0.7842 at  $K = 2$  and reaches its minimum of 0.0074 at  $K = 5$ , before increasing moderately to 0.0158 and 0.0286 at  $K = 7$  and  $K = 10$ . The corresponding final state divergence grows with  $K$ , but this greater deviation remains associated with improved traffic metrics. These results support a limited interpretation: increasing  $K$  changes both the magnitude and the onset of the sampling-trajectory deviation, while whether this change is beneficial depends on the data domain.

Figure 2 provides a visual summary of the mechanism statistics in Table 4. In the image domain, the mechanism metrics initially decrease at  $K = 2$  and  $K = 5$ , but both the update deviation and final state divergence increase sharply at  $K = 7$  and  $K = 10$ . In the traffic domain, all evaluated values of  $K$  reduce the DPM++ update deviation relative to the baseline, while the final state divergence increases gradually with  $K$ . These observations are consistent with the domain-dependent trends reported in Tables 2 and 3.

**Table 4.** Summary of low-noise mechanism metrics. Values are averaged over the selected low-noise steps for each method.

| Domain  | Method   | $C_i \downarrow$ | $\Delta_i \downarrow$ | Final $E_i$ |
|---------|----------|------------------|-----------------------|-------------|
| Image   | Baseline | 0.0104           | 0.0208                | 0.0000      |
|         | LNDDS-2  | 0.0081           | 0.0162                | 0.0111      |
|         | LNDDS-5  | 0.0056           | 0.0088                | 0.0279      |
|         | LNDDS-7  | 0.0193           | 0.0451                | 0.1733      |
|         | LNDDS-10 | 0.0404           | 0.0992                | 0.3918      |
| Traffic | Baseline | 0.0088           | 1.3018                | 0.0000      |
|         | LNDDS-2  | 0.0061           | 0.7842                | 0.0053      |
|         | LNDDS-5  | 0.0019           | 0.0074                | 0.0127      |
|         | LNDDS-7  | 0.0032           | 0.0158                | 0.0374      |
|         | LNDDS-10 | 0.0058           | 0.0286                | 0.0753      |

Note:  $\downarrow$  indicates that lower values are better, while  $\uparrow$  indicates that higher values are better. Bold values indicate the best performance for each metric.

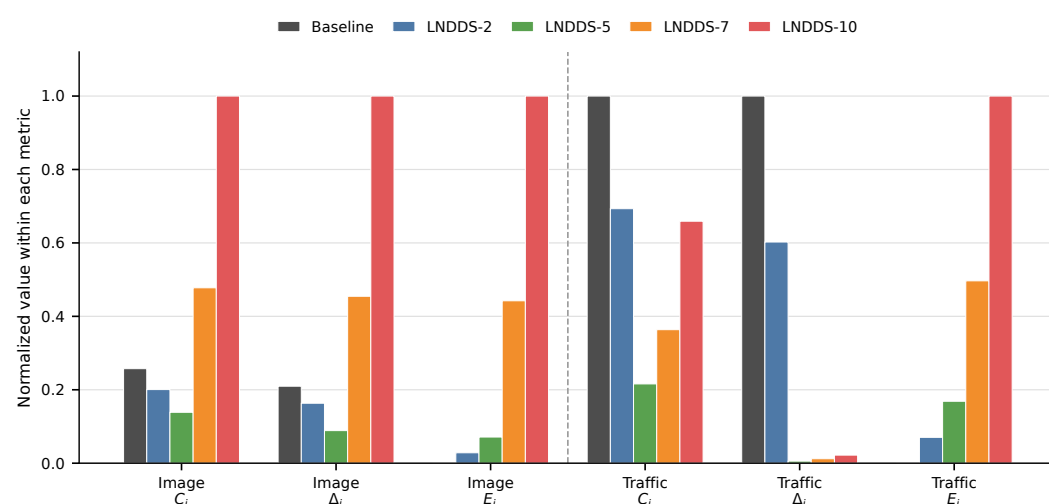


**Figure 2.** Summary visualization of the low-noise mechanism metrics. The metrics  $C_i$ ,  $\Delta_i$ , and final  $E_i$  describe denoised prediction stability, DPM++ update deviation, and cross-sampler state divergence, respectively. This figure complements Table 4 by showing the relative scale of the low-noise state changes.

#### 4.6. Robustness to the Number of Sampling Steps

We further evaluate the baseline and LNDDS using 20, 30, and 50 inference steps. This experiment checks whether the observed trend is tied to a single sampling discretization.

Table 5 shows a consistent domain-level difference. In image generation, LNDDS-2 remains closest to the corresponding baseline, while LNDDS-5 provides small improvements in CS and AS at the cost of worse FID, CMMD, and IR. The degradation generally becomes larger at  $K = 7$  and  $K = 10$ , especially when fewer inference steps are used. In traffic generation, all four LNDDS settings reduce ADE, FDE, jerk, and collision rate across 20, 30, and 50 inference steps. Smoothness is not strictly monotonic for every combination of  $N$  and  $K$ , but LNDDS-10 provides the strongest overall improvement. The different responses across  $K$  and  $N$  further indicate that the effective switching range depends on both the generation domain and the sampling discretization.

**Table 5.** Robustness to the number of inference steps.

| Steps | Method   | Image  |        |        |        |       | Traffic |       |         |              |                 |
|-------|----------|--------|--------|--------|--------|-------|---------|-------|---------|--------------|-----------------|
|       |          | IR ↑   | FID ↓  | CMMD ↓ | CS ↑   | AS ↑  | ADE ↓   | FDE ↓ | Jerk ↓  | Smoothness ↓ | Collision (%) ↓ |
| 20    | Baseline | 1.105  | 17.164 | 0.1018 | 29.328 | 6.814 | 1.913   | 4.573 | 106.833 | 0.047        | 0.391           |
|       | LNDDS-2  | 1.095  | 18.215 | 0.1182 | 29.465 | 6.853 | 1.880   | 4.545 | 70.271  | 0.033        | 0.379           |
|       | LNDDS-5  | 1.055  | 19.777 | 0.1418 | 29.658 | 6.899 | 1.811   | 4.478 | 15.415  | 0.038        | 0.342           |
|       | LNDDS-7  | 0.369  | 43.511 | 0.4539 | 27.794 | 6.243 | 1.715   | 4.294 | 14.322  | 0.043        | 0.315           |
|       | LNDDS-10 | −0.685 | 79.325 | 0.9236 | 24.981 | 5.248 | 1.551   | 3.999 | 12.666  | 0.029        | 0.250           |
| 30    | Baseline | 1.103  | 16.903 | 0.1009 | 29.363 | 6.805 | 1.905   | 4.555 | 108.990 | 0.047        | 0.392           |
|       | LNDDS-2  | 1.104  | 17.245 | 0.1078 | 29.463 | 6.843 | 1.887   | 4.543 | 75.311  | 0.052        | 0.387           |
|       | LNDDS-5  | 1.088  | 17.748 | 0.1197 | 29.600 | 6.876 | 1.841   | 4.504 | 24.765  | 0.040        | 0.355           |
|       | LNDDS-7  | 0.929  | 26.614 | 0.2436 | 29.252 | 6.801 | 1.792   | 4.443 | 20.082  | 0.043        | 0.342           |
|       | LNDDS-10 | 0.676  | 39.900 | 0.4287 | 28.715 | 6.674 | 1.710   | 4.331 | 13.039  | 0.033        | 0.305           |
| 50    | Baseline | 1.106  | 16.789 | 0.1013 | 29.376 | 6.810 | 1.902   | 4.552 | 109.873 | 0.046        | 0.391           |
|       | LNDDS-2  | 1.109  | 16.892 | 0.1062 | 29.436 | 6.835 | 1.896   | 4.551 | 83.742  | 0.052        | 0.385           |
|       | LNDDS-5  | 1.101  | 17.038 | 0.1117 | 29.503 | 6.848 | 1.861   | 4.523 | 44.533  | 0.043        | 0.363           |
|       | LNDDS-7  | 1.092  | 18.699 | 0.1294 | 29.545 | 6.882 | 1.843   | 4.502 | 32.373  | 0.051        | 0.363           |
|       | LNDDS-10 | 1.050  | 21.167 | 0.1534 | 29.657 | 6.907 | 1.797   | 4.454 | 14.109  | 0.037        | 0.338           |

Note: ↓ indicates that lower values are better, while ↑ indicates that higher values are better.

#### 4.7. Qualitative Comparison

In addition to the quantitative results, we provide qualitative comparisons for both image generation and traffic scenario generation. These examples are used to visually inspect how the low-noise direct update changes the generated outputs. They are not intended to replace the quantitative metrics, but rather to provide complementary evidence for the trends reported in Tables 2 and 3.

Figure 3 shows representative image generation examples. The comparison uses the same prompt, seed, and initial latent for all methods, so the visual differences are caused by the low-noise state-update rule. Consistent with Table 2, the qualitative results show that LNDDS may change local visual appearance and overall aesthetics, but this does not necessarily correspond to better distribution-level quality. In particular, using more direct denoising steps can produce larger visual deviations from the baseline, which is consistent with the degradation of FID, CMMD, and IR under LNDDS-10.

Figure 4 presents representative traffic trajectory comparisons. Since all methods use the same scene condition and initial noise, the differences mainly reflect the effect of replacing the recursive DPM++ update in the low-noise stage. Compared with the baseline, the LNDDS variants tend to produce smoother trajectories in the shown examples. This visual trend is consistent with the quantitative reductions in jerk, smoothness cost, and

collision rate reported in Table 3. However, these qualitative examples should be interpreted together with the aggregate metrics, because individual examples cannot fully characterize the overall behavior distribution.



**Figure 3.** Qualitative comparison of image generation results. Each row uses the same prompt, random seed, initial latent, model weights, and sampling steps. For readability, the columns show the reference image, the baseline sampler, and two representative settings, LNDDS-5 and LNDDS-10; the complete  $K \in \{2, 5, 7, 10\}$  comparison is reported quantitatively. The examples illustrate that LNDDS can change visual appearance in the low-noise stage, but the changes are not uniformly beneficial across image cases.

#### 4.8. Discussion

The completed experiments support three restrained conclusions. First, LNDDS does not consistently improve image generation quality on MJHQ-30K. The baseline remains better in FID, CMMD, and IR; LNDDS-2 produces only small changes, and LNDDS-5 obtains the best CS and AS, whereas LNDDS-7 and LNDDS-10 cause increasingly large degradation in most image metrics. Second, in the current HiLo-nuPlan setting, ADE, FDE, jerk, and collision rate generally improve as  $K$  increases, with LNDDS-10 producing the best overall traffic results among the tested settings. Third, the effect of increasing the number of direct low-noise steps is strongly domain-dependent. A wider switching range is beneficial for the evaluated structured trajectories but detrimental to the evaluated image distribution once  $K$  exceeds the narrow low-noise region.

Overall, LNDDS should be interpreted as a low-noise state-update strategy whose effect depends on the structure of the generation task. The current results do not support a broad claim that LNDDS is a universal image-quality improvement method. They instead indicate that replacing the recursive DPM++ update in the low-noise stage can be useful for structured trajectory generation, while its benefit for text-to-image generation is limited under the evaluated setting. Consequently,  $K$  should be selected using a held-out validation set and domain-specific quality metrics rather than treated as a universal constant.

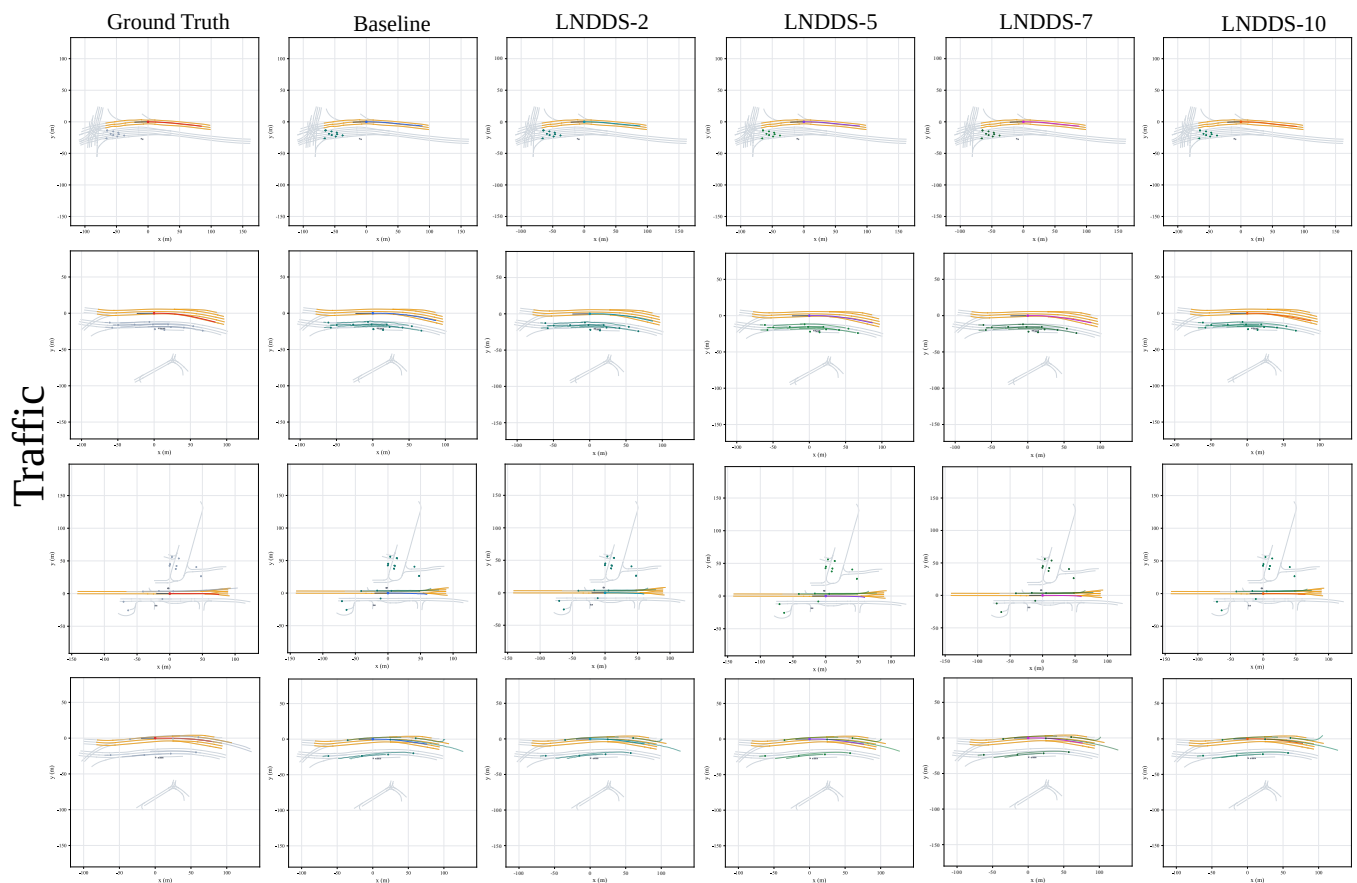


**Figure 4.** Qualitative Each example uses the same scene condition, map context, agent history, random seed, initial noise, and number of sampling steps. For readability, the columns show the recorded future trajectory, the baseline sampler, and two representative settings, LNDDS-5 and LNDDS-10; all four values of  $K$  are evaluated quantitatively. The gray and yellow lines represent lane markings, while the other colored lines represent vehicle trajectories generated under different values of  $K$ . The examples are selected to show typical changes in trajectory smoothness, interaction consistency, and collision tendency.

## 5. Conclusions and Limitations

This paper proposes Low-Noise Direct Denoising Sampling (LNDDS) for DPM++ samplers. The method replaces the recursive DPM++ update with the current model-predicted denoised output during the last  $K$  low-noise steps, while keeping the model architecture, model parameters, noise schedule, total number of sampling steps, and number of function evaluations unchanged. Therefore, the difference between LNDDS and the baseline is caused solely by the state-update rule in the low-noise regime. We evaluate  $K \in \{2, 5, 7, 10\}$  to investigate the sensitivity of the method to the low-noise switching range.

The experimental results demonstrate that the effect of  $K$  is strongly domain-dependent. On MJHQ-30K image generation, LNDDS-2 slightly improves CLIP Score and Aesthetic Score from 29.348 and 6.817 to 29.472 and 6.861, respectively, but degrades FID, CMMD, and ImageReward. LNDDS-5 achieves the best CLIP Score (29.648) and Aesthetic Score (6.921) among the evaluated settings, while further degrading the distribution- and preference-level metrics. Increasing  $K$  to 7 and 10 produces progressively larger degradation: FID increases from the baseline value of 7.179 to 19.107 and 30.390, respectively, while ImageReward decreases from 1.111 to 0.912 and 0.679. At  $K = 10$ , CLIP Score and Aesthetic Score also become worse than the baseline. These results reveal a non-monotonic and metric-dependent response to  $K$ . A small or moderate direct-denoising range may slightly

improve semantic alignment and visual aesthetics, whereas an excessively large  $K$  can remove low-noise refinement steps that remain important for image generation. Therefore, LNDDS should not be regarded as a general-purpose image-quality improvement method.

In contrast, LNDDS provides clearer benefits in the HiLo–nuPlan traffic scenario generation task. As  $K$  increases from 2 to 5, 7, and 10, ADE decreases from 1.918 to 1.854, 1.752, and 1.588, respectively, compared with 1.954 for the baseline. FDE similarly decreases from 4.636 to 4.571, 4.383, and 4.089, compared with the baseline value of 4.663. Mean jerk is substantially reduced from 106.742 for the baseline to 70.174, 15.333, 14.220, and 12.537 for LNDDS-2, LNDDS-5, LNDDS-7, and LNDDS-10, respectively. Although smoothness and collision rate do not improve strictly monotonically for every intermediate value of  $K$ , LNDDS-10 achieves the best overall traffic performance, reducing smoothness cost to 0.028 and collision rate to 0.255%. These results suggest that a wider low-noise direct-denoising range is generally beneficial for the evaluated structured trajectory generation task, where it can reduce trajectory jitter and improve physical consistency.

The mechanism analysis further explains the domain-dependent effect of  $K$ . In the image domain, the averaged denoised prediction change  $C_i$  decreases from 0.0104 for the baseline to 0.0081 and 0.0056 for LNDDS-2 and LNDDS-5, respectively. The corresponding DPM++ update deviation  $\Delta_i$  decreases from 0.0208 to 0.0162 and 0.0088. However, when  $K$  increases to 7 and 10,  $C_i$  increases to 0.0193 and 0.0404, while  $\Delta_i$  increases to 0.0451 and 0.0992. The final state divergence  $E_i$  also grows from 0.0111 at  $K = 2$  to 0.0279, 0.1733, and 0.3918 at  $K = 5$ ,  $K = 7$ , and  $K = 10$ , respectively. The sharp increase beyond  $K = 5$  is consistent with the degradation of the image-generation metrics.

In the traffic domain, all four LNDDS settings reduce the DPM++ update deviation relative to the baseline value of 1.3018. Specifically,  $\Delta_i$  decreases to 0.7842, 0.0074, 0.0158, and 0.0286 for  $K = 2$ ,  $K = 5$ ,  $K = 7$ , and  $K = 10$ , respectively. Although the final state divergence increases from 0.0053 at  $K = 2$  to 0.0127, 0.0374, and 0.0753 as  $K$  increases, this growing deviation remains associated with improved traffic-generation metrics. These observations show that a larger deviation from the baseline sampling trajectory is not necessarily harmful; its effect depends on whether the removed low-noise recursive updates contribute useful domain-specific refinement. Consequently, LNDDS should be interpreted as a task-dependent low-noise state-update strategy rather than a universally effective sampler improvement.

This work has several limitations. First, the image-generation experiments are limited to Playground v2.5 and MJHQ-30K. Although MJHQ-30K is well aligned with the high-aesthetic-quality text-to-image generation setting considered in this work, evaluation on a single image benchmark is insufficient to establish generality across different image distributions. In particular, the current experiments do not cover commonly used benchmarks such as MS-COCO and ImageNet, nor do they cover additional diffusion models, sampling schedules, and DPM++ variants. Therefore, the reported image-generation results should be interpreted as a controlled evaluation under the Playground v2.5–MJHQ-30K setting rather than evidence of universal superiority over existing solvers.

Second, although  $K \in \{2, 5, 7, 10\}$  provides a broader sensitivity analysis than using only two settings, these discrete values do not exhaustively characterize the optimal switching point. Evaluating every possible value of  $K$  requires generating and assessing a large number of samples and is computationally expensive. Third, the current switching strategy uses a fixed  $K$  for all samples and cannot adapt to sample-specific denoised prediction stability, DPM++ update deviation, or domain characteristics. Finally, the traffic experiments are conducted in an open-loop setting, and the effect of the generated scenarios on closed-loop autonomous-driving simulation remains to be evaluated.

Future work will investigate adaptive switching strategies that select  $K$  or the switching point according to observable low-noise statistics, such as the denoised prediction change  $C_i$  and the DPM++ update deviation  $\Delta_i$ . We will further evaluate LNDDS on MS-COCO, ImageNet, and other image-generation benchmarks to examine its behavior across different image distributions and task settings. Additional diffusion models, noise schedules, and DPM++ variants will also be considered to determine the scope and limitations of the proposed state-update strategy. Finally, we will extend the traffic evaluation to additional structured-generation datasets and closed-loop autonomous-driving simulation environments.

**Author Contributions:** Conceptualization, Z.L. and J.W.; methodology, J.W.; software, Z.L.; validation, Z.L.; investigation, Z.L., J.W.; data curation, Z.L.; writing—original draft preparation, Z.L.; writing—review and editing, Z.L. and J.W.; supervision, J.W.; project administration, J.W. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data is contained within the article. The original data presented in the study are openly available in MJHQ-30K dataset at <https://huggingface.co/datasets/playgroundai/MJHQ-30K/commit/d72535f77a46b6bd05d9d52f2c63d4de115b701f> (accessed on 28 July 2026). The implementation code is publicly available at [https://gitee.com/liang\\_zhi\\_gang/lndds.git](https://gitee.com/liang_zhi_gang/lndds.git) (accessed on 28 July 2026).

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Ho, J.; Jain, A.; Abbeel, P. Denoising Diffusion Probabilistic Models. In Proceedings of the Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020 (NeurIPS 2020), Virtual, 6–12 December 2020.
2. Nichol, A.Q.; Dhariwal, P. Improved Denoising Diffusion Probabilistic Models. In Proceedings of the 38th International Conference on Machine Learning (ICML 2021), Virtual Event, 18–24 July 2021.
3. Song, Y.; Sohl-Dickstein, J.; Kingma, D.P.; Kumar, A.; Ermon, S.; Poole, B. Score-Based Generative Modeling through Stochastic Differential Equations. In Proceedings of the 9th International Conference on Learning Representations (ICLR 2021), Virtual Event, Austria, 3–7 May 2021.
4. Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; Ommer, B. High-Resolution Image Synthesis with Latent Diffusion Models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2022), New Orleans, LA, USA, 18–24 June 2022. <https://doi.org/10.1109/CVPR52688.2022.01042>.
5. Peebles, W.; Xie, S. Scalable Diffusion Models with Transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV 2023), Paris, France, 1–6 October 2023. <https://doi.org/10.1109/ICCV51070.2023.00387>.
6. Jiang, C.M.; Cornman, A.; Park, C.; Sapp, B.; Zhou, Y.; Anguelov, D. MotionDiffuser: Controllable Multi-Agent Motion Prediction Using Diffusion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2023), Vancouver, BC, Canada, 17–24 June 2023. <https://doi.org/10.1109/CVPR52729.2023.00930>.
7. Pronovost, E.; Ganesina, M.R.; Hendy, N.; Wang, Z.; Morales, A.; Wang, K.; Roy, N. Scenario Diffusion: Controllable Driving Scenario Generation With Diffusion. In Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023 (NeurIPS 2023), New Orleans, LA, USA, 10–16 December 2023.
8. Feng, L.; Li, Q.; Peng, Z.; Tan, S.; Zhou, B. TrafficGen: Learning to Generate Diverse and Realistic Traffic Scenarios. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2023), London, UK, 29 May 2023–2 June 2023. <https://doi.org/10.1109/ICRA48891.2023.10160296>.
9. Karras, T.; Aittala, M.; Aila, T.; Laine, S. Elucidating the Design Space of Diffusion-Based Generative Models. In Proceedings of the Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022 (NeurIPS 2022), New Orleans, LA, USA, 28 November 2022–9 December 2022.
10. Lu, C.; Zhou, Y.; Bao, F.; Chen, J.; Li, C.; Zhu, J. DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps. In Proceedings of the Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022 (NeurIPS 2022), New Orleans, LA, USA, 28 November 2022–9 December 2022.

11. Lu, C.; Zhou, Y.; Bao, F.; Chen, J.; Li, C.; Zhu, J. DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models. *Mach. Intell. Res.* **2025**, *22*, 730–751. <https://doi.org/10.1007/S11633-025-1562-4>.
12. Song, J.; Meng, C.; Ermon, S. Denoising Diffusion Implicit Models. In Proceedings of the 9th International Conference on Learning Representations (ICLR 2021), Virtual Event, Austria, 3–7 May 2021.
13. Liu, L.; Ren, Y.; Lin, Z.; Zhao, Z. Pseudo Numerical Methods for Diffusion Models on Manifolds. In Proceedings of the The Tenth International Conference on Learning Representations (ICLR 2022), Virtual Event, 25–29 April 2022.
14. Zhao, W.; Bai, L.; Rao, Y.; Zhou, J.; Lu, J. UniPC: A Unified Predictor-Corrector Framework for Fast Sampling of Diffusion Models. In Proceedings of the Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023 (NeurIPS 2023), New Orleans, LA, USA, 10–16 December 2023.
15. Ho, J.; Salimans, T.; Gritsenko, A.; Chan, W.; Norouzi, M.; Fleet, D.J. Video Diffusion Models. In Proceedings of the Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022 (NeurIPS 2022), New Orleans, LA, USA, 28 November 2022–9 December 2022.
16. Playground v2.5: Three Insights Towards Enhancing Aesthetic Quality in Text-to-Image Generation. Available online: <https://huggingface.co/playgroundai/playground-v2.5-1024px-aesthetic> (accessed on 28 July 2026).
17. Karnchanachari, N.; Geromichalos, D.; Tan, K.S.; Li, N.; Eriksen, C.; Yaghoubi, S.; Mehdipour, N.; Bernasconi, G.; Fong, W.K.; Guo, Y.; et al. Towards Learning-Based Planning: The nuPlan Benchmark for Real-World Autonomous Driving. In Proceedings of the 2024 IEEE International Conference on Robotics and Automation (ICRA), Yokohama, Japan, 13–17 May 2024.
18. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In Proceedings of the Advances in Neural Information Processing Systems 30 (NeurIPS 2017), Long Beach, CA, USA, 4–9 December 2017.
19. Jayasumana, S.; Ramalingam, S.; Veit, A.; Glasner, D.; Chakrabarti, A.; Kumar, S. Rethinking FID: Towards a Better Evaluation Metric for Image Generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 17–21 June 2024.
20. Xu, J.; Liu, X.; Wu, Y.; Tong, Y.; Li, Q.; Ding, M.; Tang, J.; Dong, Y. ImageReward: Learning and Evaluating Human Preferences for Text-to-Image Generation. In Proceedings of the Advances in Neural Information Processing Systems 36 (NeurIPS 2023), New Orleans, LA, USA, 10–16 December 2023.
21. Radford, A.; Kim, J.W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. Learning Transferable Visual Models From Natural Language Supervision. In Proceedings of the 38th International Conference on Machine Learning, Virtual Event, 18–24 July 2021.
22. LAION-AI/aesthetic-predictor. A Linear Estimator on Top of CLIP to Predict the Aesthetic Quality of Pictures. Available online: <https://github.com/LAION-AI/aesthetic-predictor> (accessed on 28 July 2026).

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.