BEYOND THE HISTORY SPAN: STATE-GUIDED SPAN EXPANSION FOR EFFICIENT DIFFUSION INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

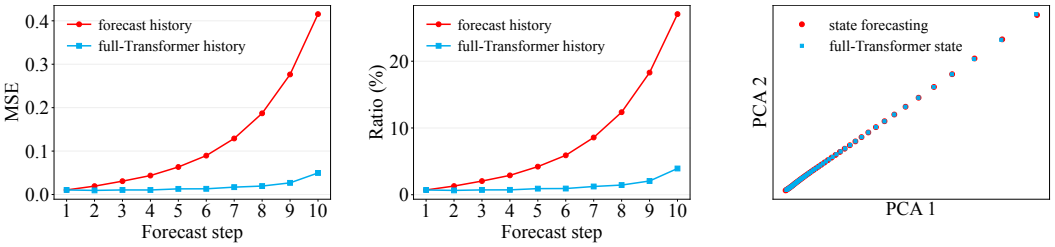
Diffusion Transformers require repeated model evaluations across many sampling steps, resulting in substantial computational overhead. A recent line of methods reduces this cost by replacing selected model evaluations with linear forecasts. These methods represent the Transformer velocity at each skipped step as a linear combination of historical velocities. However, forecasting errors grow over consecutive forecast steps because such forecasts are confined to the linear span of historical velocities and cannot represent velocity components outside this span. To overcome this limitation, we propose state-guided span expansion (SGSE), which expands the historical velocity span using a complementary direction derived from the sampling state. Specifically, we forecast a sampling state and map it through the inverse sampler update to obtain a state-implied velocity. The out-of-span component of this state-implied velocity is directionally similar to the corresponding component of the full-Transformer velocity, providing complementary information unavailable from historical velocities alone. SGSE incorporates this complementary direction into velocity forecasting, expanding the representational space beyond the historical span and thus reducing forecasting error. Experiments on image and video generation demonstrate that SGSE consistently outperforms existing forecasting-based acceleration methods.

1 INTRODUCTION

Transformer-based diffusion models have demonstrated effectiveness in image and video generation (Peebles & Xie, 2023; Chen et al., 2024). However, their practical use is hindered by the substantial computational cost of inference, as the Transformer requires repeated evaluations over a large number of sampling steps. A class of training-free caching techniques has been proposed to reduce this cost by exploiting the similarity of intermediate representations across adjacent timesteps. Earlier methods directly reuse previously computed intermediate representations (Ma et al., 2024; Selvaraju et al., 2024; Liu et al., 2025a; Ma et al., 2025), whereas more recent methods forecast the representation at the current timestep from those obtained at preceding timesteps (Liu et al., 2025b; Son et al., 2026). This forecasting paradigm also applies to flow-matching models, where each Transformer evaluation outputs a velocity field (Lipman et al., 2023; Liu et al., 2023a). In this formulation, performing full Transformer evaluations only at selected timesteps to compute the velocity, and using lightweight velocity forecasting at the remaining timesteps can substantially accelerate inference.

Existing velocity-forecasting methods typically represent the target velocity as a linear combination of velocities in the history, with the combination coefficients learned through offline calibration (Shen et al., 2026). The resulting forecasted velocity is then appended to the velocity history and reused in subsequent forecasts. However, this recursive reuse leads to a progressive increase in forecasting error over consecutive forecasting steps. As shown in Fig. 1(a), the forecasting error grows when forecasted velocities are recursively reused, whereas forecasts based on velocities obtained from full Transformer evaluations remain substantially more accurate. We attribute this behavior to the limited representational capacity of the velocity-history span. Since each forecasted velocity is a linear combination of the existing historical velocities, appending it to the history leaves the velocity-history span unchanged. Consequently, any component of the full-Transformer velocity lying outside this span cannot be recovered through such linear forecasting. As shown in Fig. 1(b), the relative out-of-span component of the full-Transformer velocity increases over consecutive forecasting steps, closely following the growth in forecasting error observed in Fig. 1(a). These observations

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Velocity forecast error (b) Out-of-span energy (c) One-step state forecasting PCA

Figure 1: Linear forecasting analysis on FLUX.2 (Black Forest Labs, 2025). (a–b) After 40 full-Transformer evaluations, the next 10 timesteps are consecutively forecast. The forecast history denotes forecasting from the first 40 full-Transformer velocities together with previously forecasted velocities, whereas the full-Transformer history uses all preceding full-Transformer velocities. (a) MSE between the forecasted velocity and the full-Transformer velocity. (b) Ratio of the squared L_2 norm of the full-Transformer velocity component orthogonal to the velocity-history span to that of the full-Transformer velocity. (c) PCA visualization of one-step state forecasting over 45 consecutive timesteps starting from timestep 5, where each forecast uses the full-Transformer history.

indicate that linear combinations of historical velocities are insufficient to accurately approximate the full-Transformer velocity.

To overcome this limitation, we introduce the sampling state as an additional source of information for velocity forecasting. Specifically, we perform linear forecasting of the state and extract from it a complementary direction that expands the velocity-history span. As shown in Fig. 1(c), the one-step forecasted states are close to the corresponding full-Transformer states in principal component analysis (PCA) space. This indicates that the next state can be reliably estimated from recent state history. However, recursive state forecasting is more susceptible to error accumulation and is therefore unsuitable as a replacement for velocity forecasting. Importantly, the one-step forecasted state still provides useful geometric information. Mapping the forecasted state through the inverse sampler update produces a state-implied velocity whose out-of-span component shows directional similarity to the component needed to expand the velocity-history span. Based on this observation, we propose state-guided span expansion (SGSE). SGSE uses the out-of-span component of the state-implied velocity as an additional direction to expand the velocity-history span and calibrates its sign and magnitude with a timestep-specific coefficient. We show that this state-guided direction effectively expands the history span and reduces both the theoretical minimum achievable approximation error and the actual forecasting error. Experiments on image and video generation show that SGSE more closely matches full-Transformer outputs than existing forecasting-based methods at comparable computational costs. We summarize our contributions as follows:

- We identify the limited representational capacity of the historical velocity span as a key source of error accumulation in consecutive velocity forecasting, and show that expanding this span can lower the theoretical minimum forecasting error.
- We propose state-guided span expansion (SGSE), which expands the historical velocity span with a complementary direction inferred from the sampling state, enabling more accurate velocity forecasting.
- Experiments on image and video generation demonstrate that SGSE consistently outperforms existing forecasting-based acceleration methods in generation quality across different acceleration ratios.

2 Related Work

2.1 Timestep Reduction

Diffusion models incur substantial inference cost due to their iterative sampling (Ho et al., 2020). One line of work reduces this cost by training models for few-step generation. Progressive distillation (Salimans & Ho, 2022) iteratively distills a multi-step sampler into student models with fewer

sampling steps. Consistency models (Song et al., 2023) learn a direct mapping from noisy inputs to clean samples, enabling few-step generation. Distribution matching distillation (Yin et al., 2024) further distills pretrained diffusion models into one-step generators by matching teacher and student distributions. Recent methods have also extended few-step generation to flow-based models, leveraging their continuous transport formulation to enable few-step generation (Liu et al., 2024; Dao et al., 2025). Although these methods reduce the number of sampling steps, they typically require an additional distillation stage incurring extra training computation.

Another line of work accelerates sampling without retraining by designing more efficient sampling trajectories or numerical solvers to reduce the number of model evaluations. DDIM (Song et al., 2021) introduces a non-Markovian sampling process that allows intermediate timesteps to be skipped during inference. DPM-Solver (Lu et al., 2022) employs high-order ordinary differential equation (ODE) solvers to more accurately approximate the diffusion sampling trajectory with fewer model evaluations. UniPC (Zhao et al., 2023) further introduces a unified predictor–corrector framework to improve sampling accuracy at low step counts. However, reducing sampling steps makes the generative dynamics harder to approximate, degrading generation quality.

2.2 Feature Caching

Feature caching accelerates inference by replacing selected Transformer evaluations with cached approximations while preserving the original sequence of sampling timesteps. Early methods directly reuse features from nearby timesteps (Ma et al., 2024; Selvaraju et al., 2024; Wimbauer et al., 2024). Since a fixed caching strategy is not equally effective across timesteps and samples, later methods design adaptive cache-refresh mechanisms based on the characteristics of intermediate representations. TeaCache (Liu et al., 2025a) uses differences between current and cached input features to estimate cache error and decide whether to reuse the cached output. AdaCache (Kahatapitiya et al., 2025) dynamically adjusts the cache interval based on representation changes. MagCache (Ma et al., 2025) uses the magnitude ratio of successive residual outputs to estimate cache error and adaptively determine when cached residuals can be reused. At higher acceleration ratios, the intervals between full-Transformer evaluations become larger, making reused cached representations less accurate.

More recent methods improve approximation accuracy by forecasting features or model outputs rather than directly reusing cached representations. TaylorSeer (Liu et al., 2025b) applies Taylor expansion to cached features to extrapolate subsequent features. L2P (Shen et al., 2026) formulates forecasting as a linear combination of historical representations and calibrates timestep-specific coefficients from a small set of full-Transformer trajectories. MeanCache (Gao et al., 2026) estimates interval-average Transformer velocities using cached Jacobian–vector products and determines cache refreshes based on trajectory stability. VDE (Tan et al., 2026) decomposes the velocity into structured components and estimates each component based on its temporal evolution.

3 Method

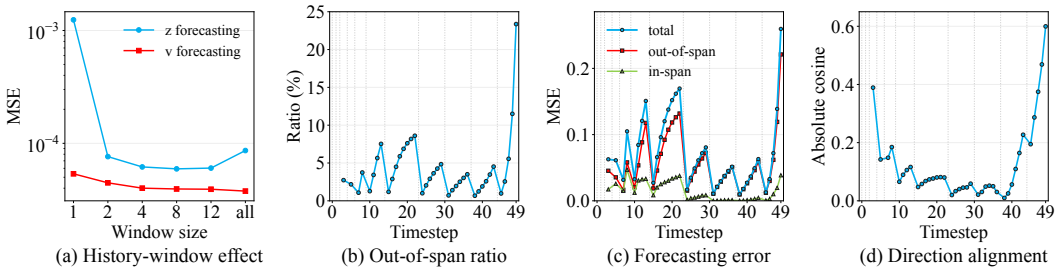
3.1 Preliminaries

Flow matching. Flow-matching models formulate the diffusion process as a continuous-time transport in which the sample state evolves under a time-dependent velocity field (Lipman et al., 2023; Liu et al., 2023a). At sampling step i, let z_i denote the current state and let t_i denote the corresponding time. The velocity obtained from an evaluation is defined as

$$v_i^* = \mathcal{F}(z_i, t_i, y), \tag{1}$$

where $\mathcal{F}(\cdot)$ denotes the generative model and y is the conditioning input. A numerical solver (Karras et al., 2022; Zhao et al., 2023) then advances z_i to z_{i+1} using $v_i^\star$. Since diffusion models typically adopt Transformers as their backbone, repeated evaluations constitute a major inference cost.

Linear forecasting. To reduce model evaluations during diffusion generation, full-Transformer evaluations are performed only at selected timesteps, while the velocity at the remaining timesteps is forecast from historical velocities. At timestep i, the current velocity is forecast as a linear com-



(a) History-window effect (b) Out-of-span ratio (c) Forecasting error (d) Direction alignment

Figure 2: Linear forecasting analysis on FLUX.2 (Black Forest Labs, 2025). Statistics are averaged over 50 PartiPrompts (Yu et al., 2022) samples. All evaluations follow the full-Transformer schedule with 11 evaluations described in Appendix A and Table 4. (a) Next-state MSE of velocity and state forecasting under different history-window sizes, averaged over all forecasting timesteps. (b) Ratio of the full-Transformer velocity energy outside the velocity-history span. (c) Velocity forecast MSE decomposed into total, in-span, and out-of-span errors. (d) Absolute cosine similarity between the missing velocity component and the out-of-span direction derived from the state-implied velocity.

bination of the previous m cached velocities:

$$\hat{v}_i = \sum_{j=i-m}^{i-1} \alpha_{i,j} v_j, \tag{2}$$

where m denotes the history-window size, v_j denotes the cached velocity at timestep j, and $\alpha_{i,j}$ is the timestep-specific velocity forecasting coefficient. The forecasted velocity $\hat{v}_i$ is then used by the numerical solver to update the sampling state. In the same manner, we forecast the next state from the most recent m states:

$$\hat{z}_{i+1} = \sum_{j=i-m+1}^{i} \beta_{i,j} z_j, \tag{3}$$

where $\beta_{i,j}$ is the timestep-specific state forecasting coefficient. We further investigate the effect of history-window size m on linear forecasting. For a unified comparison, we evaluate both velocity and state forecasting using the timestep-averaged MSE between the predicted next state and the corresponding state on the full-Transformer trajectory. Specifically, velocity forecasting updates the current state using $\hat{v}_i$ to obtain the next state, whereas state forecasting directly predicts $\hat{z}_{i+1}$. Both formulations use full-Transformer histories for forecasting. As shown in Fig. 2(a), the error reduction becomes limited once m exceeds four. This suggests that the most recent history contributes most to prediction, while further enlarging the history window provides little additional benefit. We therefore set the history-window size to four for both formulations in the following analysis and experiments. When fewer than four historical velocities are available, we use all available velocities.

3.2 Motivation

We examine the representational limitation of history-based linear velocity forecasting. At each forecasting timestep i, we construct the velocity-history span from the four most recent velocities:

$$\mathcal{H}_i = \text{span}\left\{v_{i-4}, v_{i-3}, v_{i-2}, v_{i-1}\right\}, \tag{4}$$

where v_j denotes the cached velocity at timestep j, flattened into a column vector. We refer to the velocity $v_i^\star$ obtained by a full-Transformer evaluation at the current state z_i in Eq. (1) as the reference velocity. Let P_i denote the orthogonal projection matrix onto $\mathcal{H}_i$, and let $(I - P_i)$ denote the orthogonal projection matrix onto $\mathcal{H}_i^\perp$, where I is the identity matrix. Since $\hat{v}_i \in \mathcal{H}_i$, the velocity forecasting error can be decomposed as

$$v_i^* - \hat{v}_i = (P_i v_i^* - \hat{v}_i) + (I - P_i)\, v_i^*. \tag{5}$$

The squared forecasting error therefore decomposes as

$$\|v_i^* - \hat{v}_i\|_2^2 = \|P_i v_i^* - \hat{v}_i\|_2^2 + \|(I - P_i)\, v_i^*\|_2^2. \tag{6}$$

The first term is the in-span forecasting error, while the second is the out-of-span error that cannot be represented by any linear combination of the historical velocities. We define the minimum achievable approximation error under the current velocity-history span as

$$\min_{u \in \mathcal{H}_i} \|v_i^* - u\|_2^2 = \|(I - P_i)v_i^*\|_2^2 . \tag{7}$$

Therefore, the component of the reference velocity missing from the history span is defined as

$$g_i = (I - P_i)v_i^*, \tag{8}$$

and its out-of-span energy ratio as

$$R_i = \frac{\|g_i\|_2^2}{\|v_i^*\|_2^2}. \tag{9}$$

As shown in Fig. 2(b), the out-of-span ratio of the reference velocity is nonzero at multiple forecasting timesteps. Fig. 2(c) further shows that the out-of-span error contributes substantially to the total forecasting error at several high-error timesteps, indicating that the history-span limitation is an important source of velocity forecasting error.

We further investigate whether the forecasted state provides out-of-span information related to the missing component. By Eq. (3), we obtain the forecasted next state $\hat{z}_{i+1}$ and map it through the inverse relation of the sampler update to obtain a state-implied velocity:

$$\tilde{v}_i = \mathcal{I}_i\left(\hat{z}_{i+1}, z_i\right), \tag{10}$$

where $\mathcal{I}_i(\cdot,\cdot)$ denotes the sampler-specific inverse mapping from a target next state to the corresponding current velocity. We then extract its out-of-span component:

$$q_i = (I - P_i)\tilde{v}_i. \tag{11}$$

For nonzero g_i and q_i, we define their direction alignment as

$$A_i = \frac{|\langle g_i, q_i \rangle|}{\|g_i\|_2 \, \|q_i\|_2}. \tag{12}$$

We use the absolute cosine because q_i and $-q_i$ span the same one-dimensional subspace and thus provide the same representational capacity. As shown in Fig. 2(d), the out-of-span direction derived from the state-implied velocity exhibits directional similarity to the missing velocity component at multiple forecasting timesteps, including several with pronounced out-of-span error. This suggests that the state-implied velocity provides complementary out-of-span information relevant to the missing component.

3.3 State-Guided Span Expansion

Motivated by these observations, we propose state-guided span expansion (SGSE), which expands the velocity-history span with the out-of-span direction derived from the state-implied velocity.

Overview. At timestep i, velocity and state forecasting yield $\hat{v}_i$ and $\hat{z}_{i+1}$, respectively. We arrange the four most recent historical velocities into the history matrix

$$V_i = \left[v_{i-4}, v_{i-3}, v_{i-2}, v_{i-1}\right]^\top . \tag{13}$$

The row space of V_i corresponds to the velocity-history span $\mathcal{H}_i$. Using the Moore-Penrose pseudoinverse, the orthogonal projection matrix onto $\mathcal{H}_i$ is

$$P_i = V_i^\top \left(V_i V_i^\top\right)^\dagger V_i, \tag{14}$$

where $(\cdot)^\dagger$ denotes the pseudoinverse. Following Eq. (10), we map the next-state forecast $\hat{z}_{i+1}$ through the inverse sampler update to obtain the state-implied velocity $\tilde{v}_i$. We then extract its out-of-span component according to Eq. (11), yielding the state-guided direction q_i. To adjust its sign and magnitude across timesteps, we introduce a calibrated timestep-specific scaling coefficient γ_i and obtain the final velocity forecast as

$$\bar{v}_i = \hat{v}_i + \gamma_i q_i. \tag{15}$$

This velocity is then used by the numerical solver to update the sampling state and cached for subsequent forecasting.

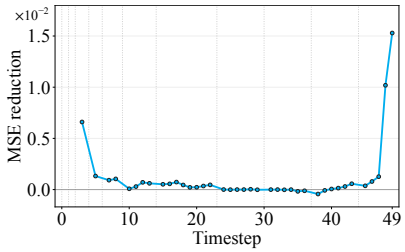


Figure 3: Velocity forecasting MSE reduction achieved by SGSE on FLUX.2 (Black Forest Labs, 2025), averaged over 50 PartiPrompts (Yu et al., 2022) samples. At each forecasting timestep, the reduction is computed as the MSE between the forecasted velocity $\hat{v}_i$ and the reference velocity $v_i^\star$ minus that between the final forecasted velocity $\bar{v}_i$ and $v_i^\star$.

Expansion of the representable space. Since $\hat{v}_i \in \mathcal{H}_i$ and $q_i \in \mathcal{H}_i^\perp$, the state-guided direction expands the representable space of historical velocities from $\mathcal{H}_i$ to

$$\widetilde{\mathcal{H}}_i = \mathcal{H}_i \oplus \mathrm{span}\{q_i\}, \tag{16}$$

where $\oplus$ denotes the orthogonal direct sum. Let $\widetilde{P}_i$ denote the orthogonal projection matrix onto $\widetilde{\mathcal{H}}_i$. For $q_i \neq 0$, the projection of the reference velocity $v_i^\star$ onto the expanded space is

$$\widetilde{P}_i v_i^* = P_i v_i^* + \frac{\langle g_i, q_i \rangle}{\|q_i\|_2^2} q_i, \tag{17}$$

where g_i is the missing velocity component defined in Eq. (8). The first term lies in the original velocity-history span, while the second is the projection of g_i onto the state-guided direction q_i. Since the two terms lie in $\mathcal{H}_i$ and $\mathcal{H}_i^\perp$, they are orthogonal. For nonzero g_i and q_i, using the definition of A_i in Eq. (12) gives

$$\left\| \widetilde{P}_i v_i^* \right\|_2^2 = \|P_i v_i^*\|_2^2 + \frac{|\langle g_i, q_i \rangle|^2}{\|q_i\|_2^2} = \|P_i v_i^*\|_2^2 + \|g_i\|_2^2 A_i^2. \tag{18}$$

As shown in Fig. 2(d), A_i is nonzero at most forecasting timesteps, allowing SGSE to expand the history span toward the missing velocity component throughout most of the sampling process.

Error reduction from span expansion. By the optimality of orthogonal projection, the minimum achievable approximation error in the expanded space is

$$\min_{u \in \widetilde{\mathcal{H}}_i} \|v_i^* - u\|_2^2 = \|v_i^*\|_2^2 - \left\| \widetilde{P}_i v_i^* \right\|_2^2 = \|g_i\|_2^2 \left(1 - A_i^2\right). \tag{19}$$

Compared with the minimum error $\|g_i\|_2^2$ in the original velocity-history span, the expanded span reduces the minimum achievable error by $\|g_i\|_2^2 A_i^2$. Thus, the potential representational gain is jointly determined by the magnitude of the missing velocity component and its alignment with the state-guided direction. For the final forecast $\bar{v}_i$, the forecasting error can be decomposed as

$$\|v_i^* - \bar{v}_i\|_2^2 = \|P_i v_i^* - \hat{v}_i\|_2^2 + \|g_i - \gamma_i q_i\|_2^2. \tag{20}$$

We further examine whether the representational gain leads to lower forecasting error. As shown in Fig. 3, SGSE reduces MSE most at early and late timesteps, where the directional similarity in Fig. 2(d) is also stronger. These results are consistent with the representational gain predicted by our analysis.

4 Experiments

4.1 Implementation Details

Models, datasets, and metrics. We evaluate SGSE on text-to-image and text-to-video generation using FLUX.2 Klein Base 4B (Black Forest Labs, 2025) and Wan2.1-T2V-1.3B (Wan et al., 2025),

Table 1: Quantitative comparisons on text-to-image generation using FLUX.2 Klein Base 4B (Black Forest Labs, 2025) on DrawBench (Saharia et al., 2022). **Bold** and underlined denote the best and second-best results under similar computational costs, respectively. FT steps denote the number of FT evaluations. Our FT schedules are detailed in Appendix A. * denotes methods without an official configuration at a comparable FT budget, for which we use our aligned schedule.

Methods	FT steps	SSIM ↑	PSNR ↑	LPIPS ↓	CLIPScore ↑
FT	50	1.0000	∞	0.0000	0.8252
TaylorSeer (N3O2) (Liu et al., 2025b)	18	0.8556	21.3913	0.1279	0.8250
VDE (medium) (Tan et al., 2026)	18	0.8455	21.3583	0.1160	0.8248
MeanCache* (Gao et al., 2026)	18	0.8638	21.9009	0.1189	0.8251
L2P* (Shen et al., 2026)	18	0.8794	22.7419	0.1024	0.8242
SGSE (Ours)	18	**0.8869**	**23.1729**	**0.0959**	**0.8252**
TaylorSeer (N6O2) (Liu et al., 2025b)	10	0.7078	16.0817	0.2799	0.8199
VDE* (Tan et al., 2026)	11	0.7179	18.1165	0.1953	0.8218
MeanCache ($B = 10$) (Gao et al., 2026)	10	0.7567	17.9781	0.2392	0.8201
L2P ($N = 5$) (Shen et al., 2026)	11	0.8228	20.7576	0.1480	0.8209
SGSE (Ours)	11	**0.8358**	**21.0309**	**0.1432**	**0.8228**
TaylorSeer (N7O2) (Liu et al., 2025b)	9	0.6820	15.4630	0.3065	0.8165
VDE* (Tan et al., 2026)	9	0.6696	16.4238	0.2591	0.8168
MeanCache* (Gao et al., 2026)	9	0.7305	17.7618	0.2590	0.8155
L2P ($N = 7$) (Shen et al., 2026)	9	0.7674	18.3860	**0.2152**	0.8169
SGSE (Ours)	9	**0.7754**	**18.5176**	0.2169	**0.8170**

Table 2: Quantitative comparisons on text-to-video generation using Wan2.1-T2V-1.3B (Wan et al., 2025) on VBench (Huang et al., 2024). **Bold** and underlined denote the best and second-best results under similar computational costs, respectively. FT steps denote the number of FT evaluations. Our FT schedules are detailed in Appendix A. * denotes methods without an official configuration at a comparable FT budget, for which we use our aligned schedule.

Methods	FT steps	SSIM↑	PSNR↑	LPIPS↓	VBench↑
FT	50	1.0000	∞	0.0000	84.0130
TaylorSeer (N3O1) (Liu et al., 2025b)	17	0.4621	12.9481	0.4888	82.7288
VDE (medium) (Tan et al., 2026)	18	0.6953	18.8560	0.2048	83.3056
L2P* (Shen et al., 2026)	18	0.7526	20.3725	0.1942	**83.8872**
SGSE (Ours)	18	**0.7600**	**20.5822**	**0.1858**	83.8665
TaylorSeer (N5O1) (Liu et al., 2025b)	10	0.4064	11.7416	0.5842	80.3218
VDE* (Tan et al., 2026)	11	0.5038	15.3503	0.3735	78.9651
L2P (N=5) (Shen et al., 2026)	11	0.6303	17.5107	0.3333	**83.0662**
SGSE (Ours)	11	**0.6354**	**17.6008**	**0.3292**	82.9168

respectively. For FLUX.2, we generate 1024×1024 images for the DrawBench prompts (Saharia et al., 2022). For Wan2.1, we generate 81-frame 832×480 videos for the VBench prompts (Huang et al., 2024). We measure fidelity to full-Transformer (FT) outputs using PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018), and report CLIPScore (Hessel et al., 2021) for image generation and VBench (Huang et al., 2024) for video generation. Detailed sampling configurations are provided in Appendix A.

Comparison methods. We compare SGSE with TaylorSeer (Liu et al., 2025b), VDE (Tan et al., 2026), and L2P (Shen et al., 2026) on both models. MeanCache (Gao et al., 2026) is included only for FLUX.2, as its official JVP implementation cannot be directly applied to Wan2.1. We reproduce the results of comparison methods using the official code. We use official configurations at comparable FT budgets when available. Otherwise, we use the corresponding SGSE FT schedule.

Calibration. Keeping the pretrained models frozen, we calibrate the velocity-forecasting coefficients $\alpha_{i,j}$, state-forecasting coefficients $\beta_{i,j}$, and SGSE scaling coefficient γ_i offline and fix them

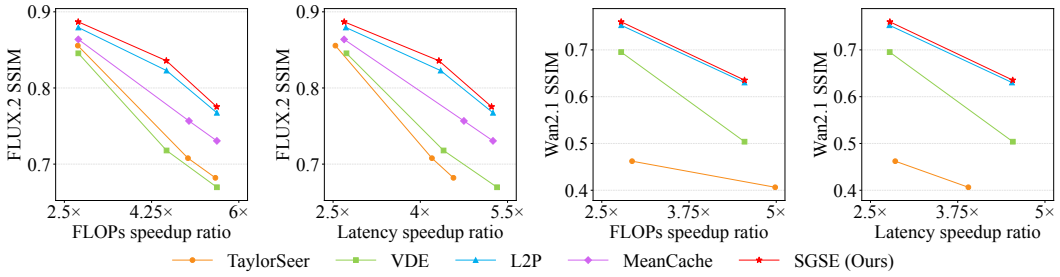


Figure 4: Quality-efficiency trade-offs on FLUX.2 and Wan2.1. We report SSIM against latency and FLOPs speedup ratios under different acceleration budgets. Speedup measurement configurations are provided in Appendix A.

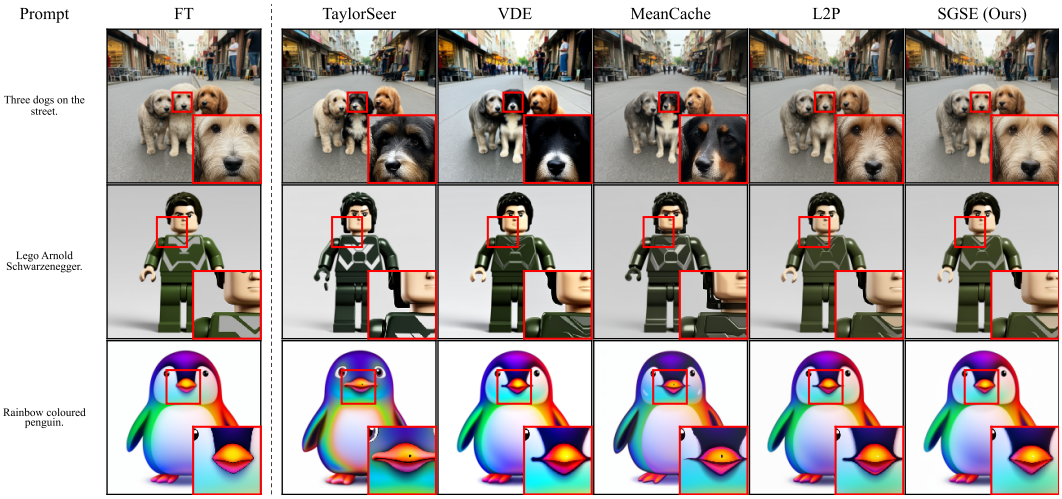


Figure 5: Qualitative comparisons on DrawBench (Saharia et al., 2022) using FLUX.2 Klein Base 4B (Black Forest Labs, 2025). Accelerated methods use 10/11 FT evaluations.

during inference. Calibration uses 50 PartiPrompts samples (Yu et al., 2022) for FLUX.2 and 50 FETV prompts (Liu et al., 2023b) that do not overlap with VBench for Wan2.1. L2P is calibrated on the same sets. Calibration objectives and optimization details are provided in Appendix B.

4.2 Quantitative Results

Generation quality. Tables 1 and 2 present quantitative comparisons on image and video generation, respectively. Across all evaluated FT budgets, SGSE achieves the highest SSIM and PSNR on both FLUX.2 and Wan2.1, demonstrating consistently improved fidelity to FT inference results. On FLUX.2, SGSE also attains the highest CLIPScore at all budgets and the lowest or second-lowest LPIPS. On Wan2.1, SGSE achieves the lowest LPIPS at both budgets while retaining VBench scores close to FT results. These results show that the fidelity improvements of SGSE persist across different acceleration levels and generation modalities while preserving competitive semantic quality.

Quality-efficiency trade-off. We further evaluate the quality-efficiency trade-off under different acceleration budgets. As shown in Fig. 4, SGSE consistently achieves higher fidelity at comparable speedups across all evaluated operating points on both FLUX.2 and Wan2.1. These results show that the gains of SGSE are not specific to a particular FT schedule, but persist as the number of FT evaluations decreases. The advantage is observed for both FLOPs and latency, indicating that state forecasting, projection, and scaling incur only limited computational overhead. Consequently, SGSE provides a more favorable quality-efficiency trade-off over different acceleration levels.

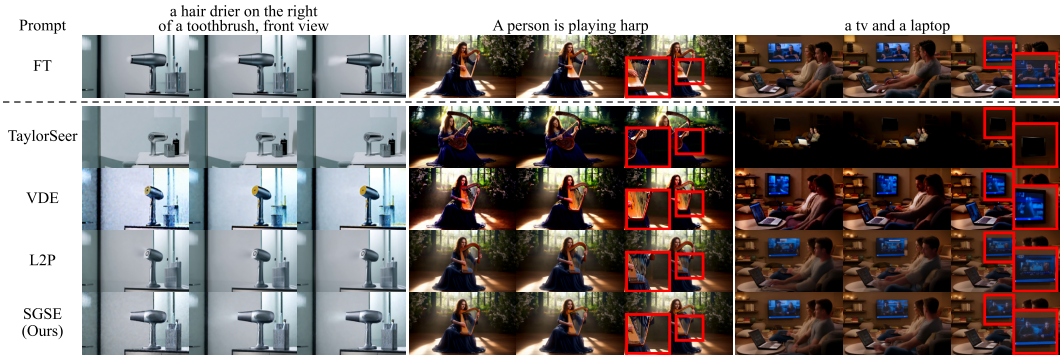


Figure 6: Qualitative comparisons on VBench (Huang et al., 2024) using Wan2.1-T2V-1.3B (Wan et al., 2025). Accelerated methods use 10/11 FT evaluations. Each video shows the first, middle, and last frames.

Table 3: Ablation of the key components in SGSE on FLUX.2 under the 11-FT schedule.

Methods	SSIM ↑	PSNR ↑	LPIPS ↓	CLIPScore ↑
w/o state guidance	0.8265	20.8545	0.1469	0.8218
w/o orthogonal projection	0.8275	20.8634	0.1460	0.8222
w/o learned scaling	0.7693	18.5686	0.2178	0.8142
SGSE	**0.8358**	**21.0309**	**0.1432**	**0.8228**

4.3 Qualitative Results

We show in Figs. 5 and 6 qualitative comparisons under comparable 10/11-FT budgets. Compared with existing methods, SGSE produces outputs that are visually closer to the FT and better preserve object structure, color consistency, and prompt-specific details. For example, in the first row of Fig. 5, SGSE more faithfully preserves the shape and fur color of the dog. In the first column of Fig. 6, only SGSE preserves the shape of the hair drier and the scene composition of the FT results. These qualitative results suggest that SGSE better preserves visual quality under accelerated inference. Results under other FT schedules are provided in Appendix C.

4.4 Ablation Analysis

Table 3 analyzes the key components of SGSE. For w/o state guidance, we remove the state-guided term $\gamma_i q_i$ from Eq. (15), leaving only velocity forecasting. The performance drop shows that the state-guided direction provides useful out-of-span information beyond velocity forecasting alone. For w/o orthogonal projection, we remove the projection $(I - P_i)$ in Eq. (11) and directly use the state-implied velocity $\tilde{v}_i$ as the span-expansion direction. The performance degradation without orthogonal projection shows that isolating the complementary out-of-span component is important for effective span expansion. For w/o learned scaling, we fix $\gamma_i = 1$ in Eq. (15), which removes the ability to adjust the sign and magnitude of the state-guided direction. The large degradation shows that span expansion provides additional representational capacity, whereas the calibrated scaling coefficient is required to effectively realize this capacity in the final velocity forecast. These ablations show that the above components jointly contribute to the effectiveness of SGSE. More ablation analyses are provided in Appendix D.

5 Conclusion

We identify the history-span limitation as an important source of error in linear velocity forecasting and propose state-guided span expansion (SGSE) to address it. SGSE expands the velocity-history span with a state-guided direction related to the missing velocity component, yielding a quantifiable reduction in the minimum achievable approximation error. Experiments on image and video generation show that SGSE more closely matches FT outputs across multiple acceleration budgets.

AI USE STATEMENT

Generative AI tools were used to assist with translation, language editing, and polishing of author-written manuscript text, as well as with literature retrieval and discovery, manuscript organization, and LaTeX formatting. They were also used to draft parts of the appendix, which were subsequently reviewed and revised by the authors. For mathematical derivations, generative AI was used to suggest proof strategies, check notation, identify errors, supplement intermediate steps, and help present author-developed derivations in manuscript form. Generative AI additionally assisted with the design of ablation studies and with code implementation and debugging. The core research idea and proposed method were independently developed by the authors. All AI-assisted mathematical arguments, code, and manuscript content were manually reviewed and verified by the authors. The authors take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Anonymous source code and experiment configurations are provided in the supplementary material. The code will be made publicly available after the anonymous review period.

REFERENCES

Black Forest Labs. FLUX.2: Frontier visual intelligence. `https://bfl.ai/blog/flux-2`, 2025.

Shoufa Chen, Mengmeng Xu, Jiawei Ren, Yuren Cong, Sen He, Yanping Xie, Animesh Sinha, Ping Luo, Tao Xiang, and Juan-Manuel Perez-Rua. GenTron: Diffusion transformers for image and video generation. In *CVPR*, 2024.

Quan Dao, Hao Phung, Trung Tuan Dao, Dimitris N. Metaxas, and Anh Tran. Self-corrected flow distillation for consistent one-step and few-step image generation. In *AAAI*, 2025.

Huanlin Gao, Ping Chen, Fuyuan Shi, Ruijia Wu, Yantao Li, Qiang Hui, Yuren You, Ting Lu, Chao Tan, Shaoan Zhao, Zhaoxiang Liu, Fang Zhao, Kai Wang, and Shiguo Lian. MeanCache: From instantaneous to average velocity for accelerating flow matching inference. In *ICLR*, 2026.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In *EMNLP*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *CVPR*, 2024.

Kumara Kahatapitiya, Haozhe Liu, Sen He, Ding Liu, Menglin Jia, Chenyang Zhang, Michael S. Ryoo, and Tian Xie. Adaptive caching for faster video generation with diffusion transformers. In *ICCV*, 2025.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*, 2023.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It's time to cache for video diffusion model. In *CVPR*, 2025a.

Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From reusing to forecasting: Accelerating diffusion models with TaylorSeers. In *ICCV*, 2025b.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023a.

Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and Qiang Liu. InstaFlow: One step is enough for high-quality diffusion-based text-to-image generation. In *ICLR*, 2024.

Yuanxin Liu, Lei Li, Shuhuai Ren, Rundong Gao, Shicheng Li, Sishuo Chen, Xu Sun, and Lu Hou. FETV: A benchmark for fine-grained evaluation of open-domain text-to-video generation. In *NeurIPS*, 2023b.

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. In *NeurIPS*, 2022.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. DeepCache: Accelerating diffusion models for free. In *CVPR*, 2024.

Zehong Ma, Longhui Wei, Feng Wang, Shiliang Zhang, and Qi Tian. MagCache: Fast video generation with magnitude-aware cache. In *NeurIPS*, 2025.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. In *NeurIPS*, 2022.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *ICLR*, 2022.

Pratheba Selvaraju, Tianyu Ding, Tianyi Chen, Ilya Zharkov, and Luming Liang. FORA: Fast-forward caching in diffusion transformer acceleration. *arXiv preprint*, 2024.

Zhirong Shen, Rui Huang, Jiacheng Liu, Chang Zou, Peiliang Cai, Shikang Zheng, Zhengyi Shi, Liang Feng, and Linfeng Zhang. Beyond fixed formulas: Data-driven linear predictor for efficient diffusion models. In *CVPR*, 2026.

Byunggwan Son, Jeimin Jeon, Jeongwoo Choi, and Bumsub Ham. Relational feature caching for accelerating diffusion transformers. In *ICLR*, 2026.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *ICML*, 2023.

Junwen Tan, Jinglin Liang, Hongyuan Chen, and Shuangping Huang. VDE: Training-free accelerating rectified flow model via velocity decomposition and estimation. In *CVPR*, 2026.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint*, 2025.

Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE TIP*, 2004.

Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, Christian Rupprecht, Daniel Cremers, Peter Vajda, and Jialiang Wang. Cache me if you can: Accelerating diffusion models through block caching. In *CVPR*, 2024.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *CVPR*, 2024.

Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. Scaling autoregressive models for content-rich text-to-image generation. *TMLR*, 2022.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. In *NeurIPS*, 2023.

Table 4: FT evaluation schedules used by SGSE. All settings retain the original 50 sampling steps. Listed timesteps use FT evaluations, while the remaining timesteps use velocity forecasting.

FT steps	Timesteps
18	0, 1, 2, 3, 4, 5, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 49
11	0, 1, 2, 4, 6, 9, 14, 23, 30, 37, 44
9	0, 1, 2, 4, 9, 17, 26, 35, 43

A Detailed Experimental Configurations

Full-Transformer schedules. Table 4 lists the full-Transformer (FT) schedules used by SGSE throughout the analyses and experiments. All configurations retain the original 50-step sampling process, with timesteps indexed from 0 to 49. At the timesteps listed in Table 4, we perform an FT evaluation to obtain the current velocity, while the remaining timesteps use velocity forecasting. The 18-FT schedule is adopted from the medium configuration of VDE (Tan et al., 2026), while the 11-FT and 9-FT schedules are adopted from the $N = 5$ and $N = 7$ configurations of L2P (Shen et al., 2026), respectively.

Efficiency measurement. All methods are evaluated using the same environment configuration and hardware. The reported latency and FLOPs speedup ratios are computed by dividing the corresponding latency or FLOPs of FT inference by those of each accelerated method. A larger speedup ratio therefore indicates higher inference efficiency.

B Calibration Details

SGSE introduces only a small number of timestep-specific coefficients while keeping the pretrained generative model frozen. These coefficients govern velocity forecasting, state forecasting, and the scaling of the state-guided direction. We calibrate them offline using 50 FT trajectories and keep them fixed during inference.

Forecasting coefficients. At each forecasting timestep i, we use the four most recent velocities and states as history. Velocity and state forecasting employ separate sets of timestep-specific coefficients, denoted by $\alpha_{i,j}$ and $\beta_{i,j}$, respectively. These coefficients are calibrated offline on FT trajectories by minimizing the reconstruction error between each forecast and its corresponding FT reference. Specifically, we optimize $\alpha_{i,j}$ and $\beta_{i,j}$ using the following reconstruction losses:

$$\mathcal{L}_\alpha = \left\| v_i^{\mathrm{FT}} - \sum_{j=i-4}^{i-1} \alpha_{i,j} v_j^{\mathrm{FT}} \right\|_2^2, \quad \mathcal{L}_\beta = \left\| z_{i+1}^{\mathrm{FT}} - \sum_{j=i-3}^{i} \beta_{i,j} z_j^{\mathrm{FT}} \right\|_2^2, \tag{21}$$

where v_i^{FT} denotes the FT velocity at timestep i along the FT trajectory, and z_{i+1}^{FT} denotes the corresponding state at the next timestep. Both reconstruction losses are averaged over the 50 calibration trajectories. For both $\alpha_{i,j}$ and $\beta_{i,j}$, we initialize the coefficient associated with the most recent history entry to 1 and all other coefficients to 0. Under this initialization, velocity and state forecasting are equivalent to reusing the most recent velocity and state, respectively. We optimize both sets of coefficients for 500 epochs using Adam with a learning rate of 0.01.

SGSE coefficient. After calibrating the velocity and state forecasting coefficients, we further calibrate the timestep-specific scaling coefficient γ_i, which adjusts the sign and magnitude of the state-guided direction in the final velocity forecast. At each forecasting timestep i, γ_i is optimized so that the scaled direction $\gamma_i q_i$ approximates the missing velocity component g_i that cannot be represented by the velocity-history span. We use the following reconstruction objective:

$$\mathcal{L}_\gamma = \frac{\|g_i - \gamma_i q_i\|_2^2}{\|g_i\|_2^2 + \epsilon} + \lambda_\gamma \left(\gamma_i - 1\right)^2. \tag{22}$$

where $\epsilon = 10^{-8}$ and $\lambda_\gamma = 0.001$. The regularization term mildly penalizes deviations of γ_i from 1, encouraging the calibrated direction $\gamma_i q_i$ to remain close to the original unscaled direction q_i. We initialize $\gamma_i = 1$ and optimize it for 500 epochs using Adam with a learning rate of 0.02.

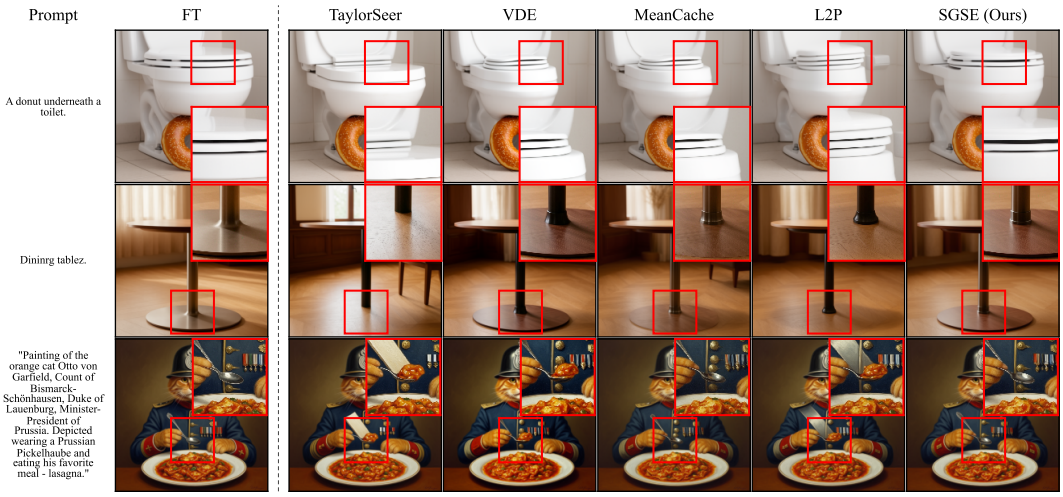


Figure 7: Qualitative comparisons on FLUX.2 Klein Base 4B with 18 FT evaluations.

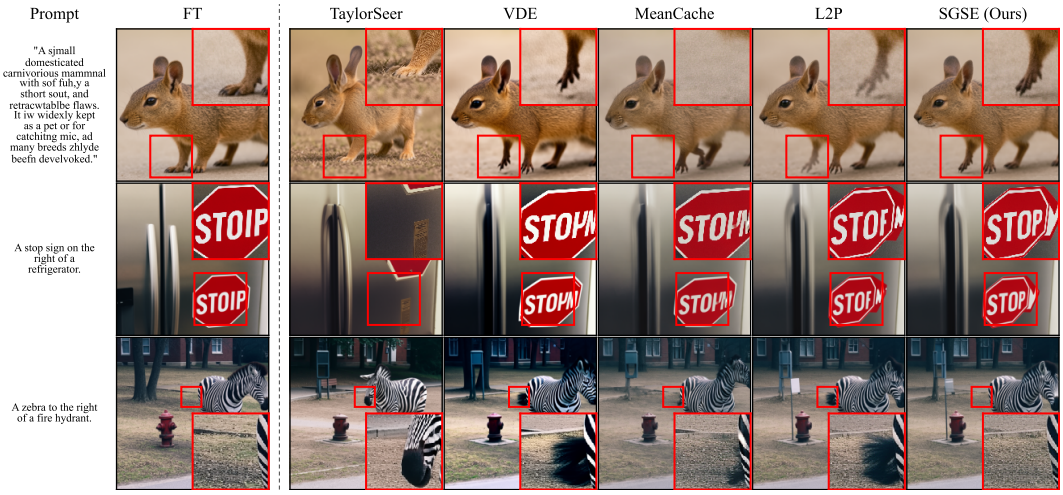


Figure 8: Qualitative comparisons on FLUX.2 Klein Base 4B with 9 FT evaluations.

C Additional Qualitative Results

We provide additional qualitative comparisons on FLUX.2 with 18 and 9 FT evaluations and on Wan2.1 with 17/18 FT evaluations. Figs. 7 and 8 show the FLUX.2 results, while Fig. 9 presents the Wan2.1 results. Across these settings, SGSE consistently produces outputs that are visually closer to the corresponding FT outputs, demonstrating better fidelity under different computational budgets.

D More Ablations

Mechanism ablation. Table 5 evaluates different choices of the forecasted target, guidance signal, and calibration method. The first three rows compare state and velocity forecasting. Directly forecasting the state leads to a substantial performance drop. Although Fig. 1(c) shows that the next state can be accurately forecasted from FT state history, errors accumulate rapidly when the forecasted states are recursively fed back into the history. As shown in the second row, adding velocity-based guidance improves state forecasting only modestly and does not recover the large performance loss. In contrast, forecasting the velocity in the third row provides a much more stable basis for consecutive forecasting and substantially better preserves fidelity to the FT outputs. Therefore, SGSE retains velocity as the forecasting target and uses the state only as an additional source



Figure 9: Qualitative comparisons on Wan2.1-T2V-1.3B with 17/18 FT evaluations.

Table 5: Mechanism ablation on FLUX.2 (Black Forest Labs, 2025) under the 11-FT schedule. FT denotes calibration using full-Transformer trajectories, while rollout denotes calibration on recursively generated accelerated trajectories.

Forecast	Guide	Calibration	SSIM ↑	PSNR ↑	LPIPS ↓	CLIPScore ↑
z	-	FT	0.4070	10.9344	0.8954	0.6897
z	v	FT	0.4632	11.5046	0.8219	0.7553
v	-	FT	0.8265	20.8545	0.1469	0.8218
v	-	rollout	0.8123	20.4501	0.1652	0.8214
v	noise	-	0.8243	20.8345	0.1468	0.8213
v	noise	FT	0.8294	20.8690	0.1461	0.8217
v	z	FT	**0.8358**	**21.0309**	**0.1432**	**0.8228**

of information. The third and fourth rows further compare calibration on FT and rollout trajectories. For rollout calibration, we recursively generate accelerated trajectories and use the resulting forecasted velocities to calibrate the forecasting coefficients. This setting slightly degrades performance compared with calibration on FT trajectories. This degradation arises because accumulated rollout errors introduce sample-dependent perturbations into the calibration data, making the calibrated forecasting coefficients less reliable. Finally, the last three rows examine the role of the guidance signal in span expansion. In the fifth row, we replace the state-guided term $\gamma_i q_i$ in Eq. (15) with uncalibrated Gaussian noise. This setting does not improve over velocity forecasting alone, showing that simply introducing an additional signal is ineffective. We then replace the state z_i in Eq. (10) with a standard Gaussian vector independently sampled for each sample and forecasting timestep, while retaining the remaining SGSE operations and FT calibration, as shown in the sixth row. This setting yields only modest gains, indicating that calibration can partially exploit the additional degree of freedom. In contrast, state guidance in the final row achieves the best performance across all metrics. This is consistent with our analysis that the state-guided direction exhibits directional similarity to the missing velocity component, supporting state information as an informative signal for effective span expansion.

Ablation for history-window size. Table 6 evaluates SGSE with different history-window sizes. Performance generally improves as the window size increases from 1 to 4, with a window size of 4 achieving the best results across all metrics. Further increasing the window size yields no additional improvement and can even reduce fidelity, indicating that more distant historical velocities provide limited useful information for forecasting the current dynamics. A window size of 4 therefore achieves the best overall performance, consistent with the forecasting analysis in Fig. 2(a).

E Discussion on Sampler

State-implied velocity. SGSE derives the state-implied velocity from the update rule of the employed sampler. For FLUX.2, which uses an Euler sampler, we divide the difference between the

Table 6: Ablation on the history-window size of SGSE on FLUX.2 (Black Forest Labs, 2025) under the 11-FT schedule.

Window size	SSIM ↑	PSNR ↑	LPIPS ↓	CLIPScore ↑
1	0.8120	19.9937	0.1752	0.8225
2	0.8226	20.7489	0.1631	0.8204
4	**0.8358**	**21.0309**	**0.1432**	**0.8228**
8	0.8283	20.9095	0.1486	0.8208
12	0.8134	20.5209	0.1647	0.8174
All	0.8317	20.9487	0.1457	0.8216

forecasted next state $\hat{z}_{i+1}$ and the current state z_i by the sampling-time interval Δt_i. Specifically, the Euler update is

$$z_{i+1} = z_i + \Delta t_i v_i, \tag{23}$$

and the state-implied velocity is therefore

$$\tilde{v}_i = \frac{\hat{z}_{i+1} - z_i}{\Delta t_i}. \tag{24}$$

For Wan2.1, which uses Flow-UniPC, we treat one scheduler step as an affine mapping from the current model output to the next state, with the current state, timestep, and solver history fixed. Let $\Phi_i(u)$ denote this update, where u is the current model output. Under the fixed solver history,

$$\Phi_i(u) = a_i u + b_i, \tag{25}$$

where a_i and b_i denote the element-wise gain and bias, respectively. We obtain them using unit-output and zero-output probes:

$$a_i = \Phi_i(\mathbf{1}) - \Phi_i(0), \qquad b_i = \Phi_i(0), \tag{26}$$

The scheduler history is restored after each probe so that the actual sampling trajectory remains unchanged. The state-implied velocity is then recovered by

$$\tilde{v}_i = \frac{\hat{z}_{i+1} - b_i}{a_i}. \tag{27}$$

Thus, SGSE implements the inverse mapping $\mathcal{I}_i$ in Eq. (10) in a sampler-aware manner, mapping the next-state forecast into the same output space as the reference velocity.

Euler specialization. For the Euler sampler, the general formulation admits a closed-form single-state representation. For any historical state z_j in the state-forecasting window,

$$z_j = z_i - \sum_{\ell=j}^{i-1} \Delta t_\ell v_\ell. \tag{28}$$

Substituting the state forecast in Eq. (3) with $m = 4$ into Eq. (24) gives

$$\tilde{v}_i = \frac{1}{\Delta t_i} \left(\sum_{j=i-3}^{i} \beta_{i,j} - 1 \right) z_i + h_i, \tag{29}$$

where

$$h_i = -\frac{1}{\Delta t_i} \sum_{j=i-3}^{i} \beta_{i,j} \sum_{\ell=j}^{i-1} \Delta t_\ell v_\ell. \tag{30}$$

Since h_i is a linear combination of historical velocities, $h_i \in \mathcal{H}_i$ and therefore

$$(I - P_i)h_i = 0. \tag{31}$$

The out-of-span component of the state-implied velocity thus reduces to

$$q_i = (I - P_i)\tilde{v}_i = \frac{1}{\Delta t_i} \left(\sum_{j=i-3}^{i} \beta_{i,j} - 1 \right) (I - P_i) z_i. \tag{32}$$

Table 7: Comparison between the general SGSE formulation and direct calibration of the Euler single-state coefficient η_i on FLUX.2 under the 11-FT schedule.

Methods	SSIM ↑	PSNR ↑	LPIPS ↓	CLIPScore ↑
Direct η_i	0.8347	21.0262	0.1442	0.8215
SGSE	**0.8358**	**21.0309**	**0.1432**	**0.8228**

Accordingly, SGSE can be equivalently represented as

$$\bar{v}_i = \hat{v}_i + \eta_i(I - P_i)z_i. \tag{33}$$

Therefore, under Euler updates, SGSE admits an exact single-state representation while preserving its geometric role of expanding the velocity-history span. This simplification is specific to the Euler update. For multi-step solvers such as Flow-UniPC, the state-to-model-output mapping additionally depends on the solver history and the above derivation does not directly apply.

Direct calibration of the single-state form. We further investigate whether the Euler specialization can be applied by directly calibrating η_i in Eq. (33). As shown in Table 7, directly calibrating η_i performs slightly worse across all evaluation metrics. These results indicate that collapsing the formulation into a single coefficient provides no empirical advantage, whereas the general state-forecasting formulation retains sampler compatibility and achieves better performance.