STIR-FM: SEGMENT-WISE TRAJECTORY INTERPOLATION REFINEMENT FOR FLOW MATCHING

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow-matching models require multiple sequential model evaluations for text-to-image generation. We propose **STIR-FM**, a training-free post-sampling framework that refines a cached few-step trajectory by inserting model evaluations into intervals selected using velocity variation and propagating local velocity corrections. STIR-FM scores each interval by the velocity change across it, inserts an evaluation at the midpoint of the highest-scoring interval, and corrects the cached velocities downstream from a coordinate-wise spatial slope estimated from three evaluated reference pairs. An initial trajectory costing a model evaluations and b inserted evaluations has a total cost of $a+b$ NFEs, with only b new evaluations needed when the initial trajectory is already available. Fixed-budget experiments on FLUX.1-dev and Stable Diffusion 3/3.5 show improved mean ImageReward over the initial trajectories; at matched total NFE, the refined means approach or exceed direct sampling once the initial trajectory is moderately resolved. A separate STORK experiment refines only samples falling below their per-sample STORK-12 CLIP reference and selects the highest-CLIP output in each history, reducing counted generator NFEs by 9.27% while mean held-out ImageReward remains numerically close to that of fixed STORK-12.

1 INTRODUCTION

Diffusion models (Ho et al., 2020; Kingma et al., 2021; Sohl-Dickstein et al., 2015; Song et al., 2021b) have become a dominant paradigm for high-quality image generation by gradually transforming noise into data through a sequence of denoising steps. More recently, flow matching models (Geng et al., 2026; Lipman et al., 2023; Liu et al., 2023) have provided an alternative formulation by learning a continuous velocity field (Chen et al., 2018) that transports samples from a simple prior distribution to the data distribution. This formulation naturally connects generative modeling with ordinary differential equation (ODE) (Chen et al., 2018) sampling, and has been adopted by recent large-scale text-to-image (Cao et al., 2025; Esser et al., 2024; Labs et al., 2025; Seedream et al., 2025; Wu et al., 2025b) or text-to-video models (Seedance et al., 2026; Wan et al., 2025; Wu et al., 2025a). Despite their strong generation quality, practical flow-matching samplers still require multiple sequential model evaluations, and reducing the number of sampling steps remains an important problem for efficient inference.

The effect of increasing the sampling budget varies across prompts and random seeds. We examine this variation by measuring the root mean square error (RMSE) between lower-step images and their corresponding 20-step images over 5,000 prompts (Figure 1). This quantifies how far lower-step outputs sit from a 20-step reference in normalized RGB space. The variation is sample-dependent, which means a fixed schedule spends the same number of evaluations on easy and hard samples alike. This motivates selecting trajectories whose generated images do not yet meet a chosen quality criterion and refining their highest-error intervals, so that additional evaluations target the least reliable parts of those trajectories.

We propose **STIR-FM** (*Segment-wise Trajectory Interpolation Refinement for Flow Matching*), a training-free post-sampling refinement framework. **Our central perspective is that an already generated trajectory is a reusable object for further optimization.** Cached states and velocities describe the path taken by each sample and inform where to spend additional evaluations. STIR-FM selects intervals by velocity variation, evaluates inserted points, and propagates local corrections to

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.

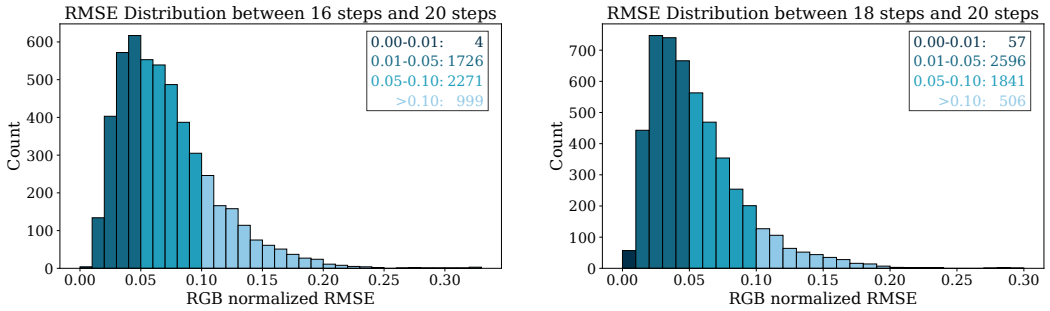


Figure 1: RMSE distributions between normalized lower-step generated images and the corresponding 20-step generated images on FLUX.1-dev over 5,000 prompts. For each generated image, pixel values are first normalized by dividing by 255; we then compute the root mean square error between the normalized lower-step image and its corresponding normalized 20-step image. Some 16-step images fall into the RMSE range $[0, 0.01]$, indicating numerical proximity to their 20-step reference images in normalized RGB space. Nevertheless, both settings still contain high-error samples, with some RMSE values exceeding 0.30 for 16 steps and 0.25 for 18 steps. The 20-step image serves as a matched finite-budget reference for this measurement.

downstream cached velocities. This enables incremental improvement of a retained trajectory without rerunning a complete longer sampling path, and can be adapted to different numerical solvers: interval selection and local velocity correction act on the trajectory itself, with solver-specific handling of history and boundaries.

The interval score determines *where* to refine; a user-specified budget or external stopping rule determines *how much* to refine. Given an a-NFE cached trajectory, b inserted evaluations cost b additional NFEs and $a + b$ in total, so every refined output is compared against direct sampling at the same total budget. This yields three contributions. First, we propose **STIR-FM**, a training-free post-sampling framework that treats an already generated trajectory as a reusable object: it inserts evaluations into the highest-error intervals and propagates local velocity corrections downstream, improving a retained trajectory without rerunning a longer sampling path. Second, we introduce a velocity-variation interval score that ranks intervals by the velocity change across them: a fixed step schedule spends the same steps at the same trajectory times for every sample, and is therefore insensitive to how strongly the velocity actually varies, whereas our score directs each new evaluation to the interval where it varies most. Third, our experiments show that at matched total NFE, STIR-FM approaches or exceeds direct sampling in mean ImageReward once the initial trajectory is moderately resolved, after 13 steps on FLUX.1-dev, 21 on Stable Diffusion 3, and 15 on Stable Diffusion 3.5, with a qualitative example in Figure 2; combined with STORK under per-sample reference CLIP stopping, it maintains numerically close mean ImageReward while cutting counted generator NFEs by 9.27%.

2 Related Work

Timestep schedules and sampling grids. The sampling grid is a key factor in the quality–efficiency trade-off of diffusion and flow-based generative models. Early samplers typically select inference steps from predefined training schedules or handcrafted noise schedules (Ho et al., 2020; Karras et al., 2022; Nichol & Dhariwal, 2021; Rombach et al., 2022). Recent work shows that this choice can significantly affect few-step generation: some studies analyze flaws in common noise schedules and sampling endpoints (Lin et al., 2024), others directly optimize the timestep grid for a given model, solver, or trajectory structure (Chen et al., 2024a;b; Sabour et al., 2024; Xue et al., 2024), and distribution-based approaches such as Beta Sampling allocate steps according to a Beta-distribution-like schedule motivated by spectral analysis (Lee et al., 2025). These methods establish the importance of step placement, but they fix the sampling grid before generation starts, so they cannot respond to the velocity changes that a particular sample develops along its trajectory.

Training-based acceleration. Another line of work reduces sampling cost by training or distilling a new few-step generator. Progressive distillation compresses a multi-step sampler into a student model



11 steps | 11+1 steps | 12 steps | 11+2 steps | 13 steps

Figure 2: Qualitative comparison on FLUX.1-dev at 512×512 resolution. All images are generated from the same prompt: "An ancient seawall, overgrown with sea grass and coral, stands as a testament to a long-forgotten era, its stone blocks worn smooth by the relentless tides." Here, "11+1 steps" denotes refining an 11-step trajectory by inserting one interpolation step, and "11+2 steps" denotes refining an 11-step trajectory by inserting two interpolation steps. The comparison shows that STIR-FM can improve the visual quality of an 11-step initial trajectory and make it approach that of standard sequential sampling with more steps.

with fewer evaluations (Salimans & Ho, 2022), while consistency models and latent consistency models learn trajectory-consistent mappings for one-step or few-step generation (Chen et al., 2025; Luo et al., 2023; Song et al., 2023). Related rectified-flow and trajectory-consistency distillation methods further improve few-step or one-step generation through additional training (Dao et al., 2025; Liu et al., 2024; Zheng et al., 2024). Although effective, these methods modify the model parameters and require additional training or distillation compute, whereas STIR-FM leaves the pretrained model untouched.

Training-free solvers and sampling algorithms. Training-free acceleration methods improve inference-time sampling without modifying model parameters. DDIM (Song et al., 2021a; Zhang et al., 2023) uses a non-Markovian reverse process, PNDM (Liu et al., 2022) treats diffusion sampling as numerical solving on manifolds, and DPM-Solver variants (Lu et al., 2022; 2025; Zheng et al., 2023) derive fast high-order ODE solvers. Other methods further reduce discretization error through exponential-integrator or predictor-corrector formulations (Zhang & Chen, 2023; Zhao et al., 2023), while EDM (Karras et al., 2022) studies the broader design space of noise parameterization, preconditioning, schedules, and solvers. These methods design a better global sampling rule and apply it to the whole trajectory from the start. STIR-FM instead leaves the sampler unchanged and refines the trajectory it has already produced, which makes the two directions complementary.

3 METHOD

Assume that a standard few-step sampler has already produced a latent trajectory

$$\{x_0, x_1, \ldots, x_N\},$$

with normalized timestamps

$$0 = t_0 < t_1 < \cdots < t_N = 1,$$

and corresponding velocity predictions

$$\{v_0, v_1, \ldots, v_{N-1}\}, \qquad v_k = f_\theta(x_k, t_k).$$

The equality $v_k = f_\theta(x_k, t_k)$ holds for the initially evaluated trajectory. During refinement, v_k may instead be a cached approximation at the updated state; it need not be a new model evaluation. In this work, we consider the simplest case where the original sampler uses the Euler (Hairer et al., 1993) update,

$$x_{k+1} = x_k + (t_{k+1} - t_k)\, v_k. \tag{1}$$

Our method then refines this already-generated trajectory through four components: error estimation, local velocity refinement, adaptive interpolation, and segment-wise propagation. Interval selection and local velocity correction act on the trajectory, so STIR-FM can be adapted to existing samplers with solver-specific history and boundary handling; derivations for other solvers, such as UniPC (Zhao et al., 2023) and STORK (Tan et al., 2026), are given in Appendix F.

3.1 Error Estimation

We first identify which intervals of the existing few-step trajectory are most unreliable. The basic intuition is that if the predicted velocity changes sharply between two neighboring steps, then the local linear update in Eq. (1) is more likely to incur a large approximation error.

For each interval $[t_k, t_{k+1}]$, we define a first-order error score

$$e_k = \|v_{k+1} - v_k\|_2 \, (t_{k+1} - t_k), \qquad k = 0, 1, \ldots, N-2. \tag{2}$$

A larger e_k indicates that the velocity field varies more strongly over that interval, and hence that interval is a better candidate for refinement.

For the last interval, we use a boundary extrapolation score:

$$e_{N-1} = \|v_{N-1} - v_{N-2}\|_2 \cdot \frac{(t_N - t_{N-1})^2}{t_{N-1} - t_{N-2}}. \tag{3}$$

After computing all scores, we rank the intervals and select the top M intervals with the largest errors for refinement, where M denotes the number of interpolation steps in one refinement pass. In this way, the extra computation is concentrated on the most problematic parts of the trajectory, rather than being distributed uniformly over the whole sampling path. The detailed procedure for error computation is given in Algorithm 2 in Appendix A.

A formal justification of this score as a proxy for the leading-order local truncation error of Euler integration is provided in Lemma B.1 in Appendix B.

3.2 Local Velocity Refinement

An inserted evaluation changes downstream states, so the cached velocity at an old state generally no longer equals the model velocity at the updated state. Re-evaluating the model at every affected node would remove the intended incremental-NFE benefit. Instead, we estimate a coordinate-wise spatial slope from three evaluated reference pairs, eliminating the locally linear time contribution, and multiply this slope by the displacement from the reference state. This is a local approximation, not a reconstruction of the full latent-space Jacobian. For example, inserting a midpoint between t_1 and t_2 in a five-step trajectory gives the following downstream sequence (subscripts here retain the original timestamps):

$$\underbrace{x_{\mathrm{mid}}, f_\theta(x_{\mathrm{mid}}, t_{\mathrm{mid}})}_{\text{one new model evaluation}} \longrightarrow \underbrace{x'_2, \widetilde{v}_2}_{\text{cached correction}} \longrightarrow \underbrace{x'_3, \widetilde{v}_3}_{\text{cached correction}} .$$

The remaining tail uses the boundary convention stated below; propagating or correcting an existing node adds no generator NFE.

Consider three consecutive timestamps (t_k, t_{k+1}, t_{k+2}). Let (x_k, x_{k+1}, x_{k+2}) denote the current refined latent states. Let $(x_k^{\mathrm{ori}}, x_{k+1}^{\mathrm{ori}}, x_{k+2}^{\mathrm{ori}})$ denote the corresponding reference states at which the original velocity predictions $(v_k^{\mathrm{ori}}, v_{k+1}^{\mathrm{ori}}, v_{k+2}^{\mathrm{ori}})$ are evaluated. These reference states are not necessarily taken from the initial trajectory; they may also include points inserted by interpolation in an earlier stage. However, once a reference state x_k^{ori} is associated with a velocity prediction v_k^{ori}, it is kept fixed in all subsequent refinement rounds. Accordingly, $(v_k^{\mathrm{ori}}, v_{k+1}^{\mathrm{ori}}, v_{k+2}^{\mathrm{ori}})$ are always interpreted as the velocity predictions evaluated at $(x_k^{\mathrm{ori}}, x_{k+1}^{\mathrm{ori}}, x_{k+2}^{\mathrm{ori}})$, rather than at the current refined states (x_k, x_{k+1}, x_{k+2}). We define a local correction for the middle velocity by

$$\Delta v_{k+1} = \frac{\left[(v_{k+1}^{\mathrm{ori}} - v_k^{\mathrm{ori}})(t_{k+2} - t_k) - (v_{k+2}^{\mathrm{ori}} - v_k^{\mathrm{ori}})(t_{k+1} - t_k)\right] \odot (x_{k+1} - x_{k+1}^{\mathrm{ori}})}{\left[(x_{k+1}^{\mathrm{ori}} - x_k^{\mathrm{ori}})(t_{k+2} - t_k) - (x_{k+2}^{\mathrm{ori}} - x_k^{\mathrm{ori}})(t_{k+1} - t_k)\right]}, \tag{4}$$

where $\odot$ denotes element-wise multiplication and the division is also performed element-wise. The refined velocity is then

$$v_{k+1} = v_{k+1}^{\mathrm{ori}} + \Delta v_{k+1}. \tag{5}$$

In the reported implementation, coordinates whose denominator magnitude falls at or below the ρ-quantile are suppressed, with $\rho = 0.20$ by default; Appendix C.1 states this step, and Appendix K.2

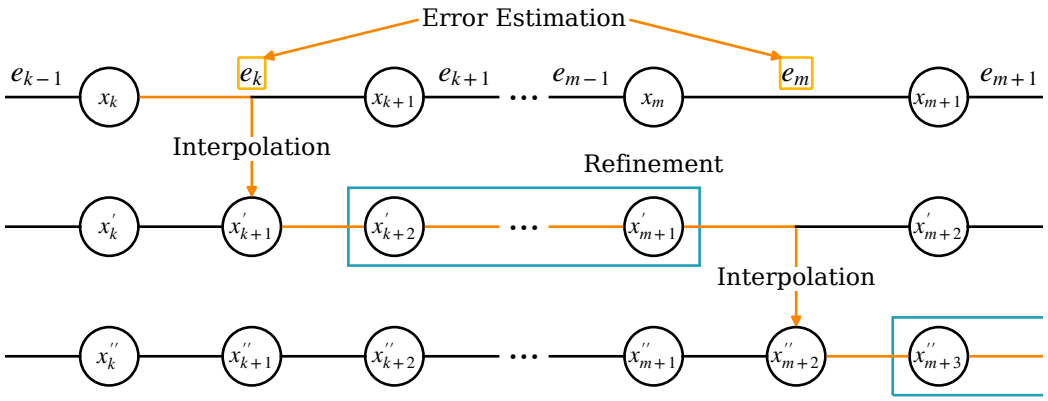


Figure 3: Overall segment-wise interpolation and refinement process. In the figure, x_k, x'_k, and x''_k correspond to the same timestamp, i.e., $t_k = t'_k = t''_k$, and x_{k+1}, x'_{k+2}, and x''_{k+2} also correspond to the same timestamp, i.e., $t_{k+1} = t'_{k+2} = t''_{k+2}$. The inserted midpoint states x'_{k+1} and x''_{k+1} satisfy $t'_{k+1} = t''_{k+1} = (t_k + t_{k+1})/2$. Similarly, x_m, x'_{m+1}, and x''_{m+1} correspond to the same timestamp, i.e., $t_m = t'_{m+1} = t''_{m+1}$, while x_{m+1}, x'_{m+2}, and x''_{m+3} correspond to the same timestamp, i.e., $t_{m+1} = t'_{m+2} = t''_{m+3}$. The inserted midpoint state x''_{m+2} satisfies $t''_{m+2} = (t_m + t_{m+1})/2$.

reports the sensitivity to $\rho \in \{0.10, 0.20, 0.30\}$. The local velocity refinement operator is summarized in Algorithm 4 in Appendix A, where the estimation of the spatial slope and its propagation to the affected nodes are spelled out step by step, and its formal derivation is given in Lemma C.1 in Appendix C.

3.3 Adaptive Interpolation

Given a selected high-error interval $[t_k, t_{k+1}]$, we insert one midpoint into this interval to locally densify the trajectory. The new timestamp is

$$t_{\text{mid}} = \frac{t_k + t_{k+1}}{2}. \tag{6}$$

Instead of directly interpolating between x_k and x_{k+1} in latent space, we generate the midpoint state by propagating from the left endpoint using the current refined velocity:

$$x_{\text{mid}} = x_k + (t_{\text{mid}} - t_k)\, v_k, \tag{7}$$

where v_k is the velocity currently used for trajectory propagation. We then evaluate the model at the midpoint to obtain

$$v_{\text{mid}} = f_\theta(x_{\text{mid}}, t_{\text{mid}}). \tag{8}$$

The newly constructed midpoint $(x_{\text{mid}}, v_{\text{mid}}, x^{\text{ori}}_{\text{mid}}, v^{\text{ori}}_{\text{mid}}, t_{\text{mid}})$ is inserted into the refined trajectory sequence $\{x_k\}$, refined velocity sequence $\{v_k\}$, reference-state sequence $\{x^{\text{ori}}_k\}$, original velocity sequence $\{v^{\text{ori}}_k\}$, and timestamp sequence $\{t_k\}$, respectively. At the moment of insertion, the reference quantities are initialized by the current refined quantities, namely, $x^{\text{ori}}_{\text{mid}} = x_{\text{mid}}$ and $v^{\text{ori}}_{\text{mid}} = v_{\text{mid}}$. This locally increases the trajectory resolution only in unreliable intervals, allowing us to improve poor few-step samples with only a small number of additional model evaluations. The detailed interpolation routine is summarized in Algorithm 3 in Appendix A.

3.4 Overall Refinement Propagation

One refinement round processes selected intervals in temporal order, propagating each insertion's effect up to the next selected interval before inserting another point (Figure 3). At initialization, the working states and velocities coincide with their evaluated reference pairs; subsequent rounds update the working sequences while preserving existing reference values.

Algorithm 1 Segment-Wise Trajectory Refinement

Require: Model f_θ, original reference points $\{x_k^{\text{ori}}\}_{k=0}^{N}$, original model predictions $\{v_k^{\text{ori}}\}_{k=0}^{N-1}$, previous refined trajectory $\{x_k\}_{k=0}^{N}$ (initialized as $\{x_k^{\text{ori}}\}_{k=0}^{N}$ in the first round), previous refined velocities $\{v_k\}_{k=0}^{N-1}$ (initialized as $\{v_k^{\text{ori}}\}_{k=0}^{N-1}$ in the first round), timestamps $\{t_k\}_{k=0}^{N}$, number of interpolation steps M

$\{e_k\}_{k=0}^{N-1} \leftarrow$ Algorithm 2$(\{v_k\}_{k=0}^{N-1}, \{t_k\}_{k=0}^{N})$ ▷ Error Estimation

$K \leftarrow \min(M, N)$

$\{p_m\}_{m=1}^{K} \leftarrow \text{sort}\big(\arg\text{TopK}_{0\le k\le N-1}\{e_k\}\big), \quad p_1 < p_2 < \cdots < p_K$

for $m = 1$ to K **do**

  $\hat{p}_m \leftarrow p_m + (m-1)$

  $(\{x_k\}_{k=0}^{N+m}, \{v_k\}_{k=0}^{N+m-1}, \{x_k^{\text{ori}}\}_{k=0}^{N+m}, \{v_k^{\text{ori}}\}_{k=0}^{N+m-1}, \{t_k\}_{k=0}^{N+m})$

    $\leftarrow$ Algorithm 3$\big(f_\theta, \hat{p}_m, \{x_k\}, \{v_k\}, \{x_k^{\text{ori}}\}, \{v_k^{\text{ori}}\}, \{t_k\}\big)$ ▷ Adaptive Interpolation

  $L \leftarrow \hat{p}_m + 2, \quad R \leftarrow \begin{cases} p_{m+1} + m, & m < K, \\ N + m, & m = K \end{cases}$

  for $j = L$ to R **do**

    $x_j \leftarrow x_{j-1} + (t_j - t_{j-1}) v_{j-1}$

    if $1 \le j \le N + m - 2$ **then**

      $v_j \leftarrow$ Algorithm 4$\big(\{x_k\}, \{x_k^{\text{ori}}\}, \{v_k^{\text{ori}}\}, \{t_k\}, j\big)$ ▷ Local Velocity Refinement

    end if

  end for

end for

return $\{x_k\}_{k=0}^{N+K}, \{v_k\}_{k=0}^{N+K-1}, \{x_k^{\text{ori}}\}_{k=0}^{N+K}, \{v_k^{\text{ori}}\}_{k=0}^{N+K-1}, \{t_k\}_{k=0}^{N+K}$

Given N current intervals and an insertion budget M, compute scores from the working velocities and select $K = \min(M, N)$ intervals. Their round-start indices $p_1 < \cdots < p_K$ remain fixed during the round. After $m-1$ insertions, the mth selected interval's left endpoint has current index $\hat{p}_m = p_m + m - 1$. Insert its midpoint at index $\hat{p}_m + 1$, evaluate the model once, and register the resulting state–velocity pair as a new immutable reference.

Starting at $j = \hat{p}_m + 2$, propagate downstream states using the Euler update and correct their cached velocities where three evaluated references are available. For $m < K$, stop at $j = p_{m+1} + m$, the updated left endpoint of the next selected interval; its refined state and velocity then initialize the next midpoint insertion. For $m = K$, continue through the terminal state at $j = N + K$. All working and reference sequences share the same reindexing, while previously stored reference values remain unchanged. Appendix A.1 gives a symbol table and a worked two-insertion example.

At insertion m, local velocity correction applies only for $1 \le j \le N + m - 2$. The last stored velocity ($j = N + m - 1$) is retained because no right reference velocity is available; the terminal state ($j = N + m$) is propagated without evaluating a velocity. This is the Euler boundary convention used in Algorithm 1; Appendix F describes the corresponding choice for the other solvers.

The completed round adds K model evaluations and produces $N + K$ intervals. The next round recomputes scores on the updated working trajectory and reuses the enlarged reference set. Thus, new model calls occur only at inserted midpoints; downstream propagation and cached corrections incur no additional generator NFE. Appendix D provides the end-to-end error analysis.

4 Experiments

The fixed-budget Euler comparisons evaluate FLUX.1-dev and Stable Diffusion 3/3.5 on 5,000 samples per configuration; the conditional STORK study uses a separate set of **1,000 matched samples**. We report generator NFE and mean ImageReward, with CLIP additionally controlling reference-based stopping in the STORK study. Appendix G gives the complete experimental settings and cost conventions.

Table 1: Results on FLUX.1-dev for initial trajectory steps 9–18. All entries are average ImageReward (ImRe) scores evaluated over the same set of 5,000 prompts. Complete ImageReward and CLIP results for all evaluated step configurations are provided in Appendix H.

Inserted NFEs b by round	Initial NFE a									
	9	10	11	12	13	14	15	16	17	18
0	0.784	0.802	0.818	0.831	0.847	0.853	0.858	0.860	0.864	0.863
1	0.789	0.807	0.824	0.838	0.853	0.862	0.865	0.867	0.872	0.869
1+1	0.801	0.818	0.832	0.844	0.861	0.867	0.870	0.871	0.876	0.873
1+1+1	0.807	0.825	0.841	0.855	0.871	0.876	0.878	0.878	0.882	0.879
1+1+1+1	0.810	0.827	0.841	0.855	0.869	0.875	0.878	0.878	0.881	0.878
1+1+1+1+1	0.825	0.837	0.848	0.859	0.872	0.877	0.879	0.879	0.883	0.880
1+1+1+1+1+1	0.830	0.843	0.853	0.863	0.877	0.881	0.883	0.882	0.886	0.882
2	0.796	0.813	0.829	0.841	0.857	0.864	0.868	0.870	0.874	0.872
2+2	0.810	0.826	0.842	0.855	0.870	0.876	0.878	0.878	0.882	0.879
2+2+2	0.830	0.843	0.854	0.864	0.877	0.881	0.883	0.882	0.885	0.882
3	0.804	0.820	0.835	0.846	0.862	0.869	0.873	0.872	0.877	0.874
3+3	0.828	0.841	0.853	0.864	0.877	0.882	0.884	0.883	0.886	0.883
4	0.807	0.822	0.837	0.848	0.864	0.870	0.874	0.874	0.877	0.875
4+2	0.830	0.842	0.854	0.863	0.876	0.879	0.883	0.881	0.885	0.882

4.1 Experimental Results

In the fixed-budget tables, columns specify initial NFE a and rows specify inserted NFEs by round, whose sum is b; each cell therefore has total generator NFE $a + b$. Row 0 is the unrefined baseline, and direct equal-budget sampling is read from row 0 at column $a + b$ when that column is available.

In the main paper, we report ImageReward results for representative ranges: 9–18 steps for FLUX.1-dev (Tables 1), 20–24 steps for Stable Diffusion 3 (Tables 2), and 14–18 steps for Stable Diffusion 3.5 (Tables 2). Complete ImageReward and CLIP results for the fixed-budget Euler configurations are provided in Appendix H. For clearer visualization, we also plot representative ImageReward curves for selected interpolation strategies in Figure 4. We also provide additional qualitative comparisons in Appendix J. For these fixed-budget Euler comparisons, the main text focuses on ImageReward because it more clearly distinguishes refinement configurations; the STORK study below reports CLIP separately as its stopping metric and ImageReward as a held-out evaluation metric.

4.2 Analysis of Experimental Results

Effect of the number of refinement rounds. Mean ImageReward generally improves with additional refinement rounds across the three models, although gains are not monotonic in every configuration and do not establish a significant improvement at every round.

Effect of the initial trajectory length. An overly coarse initial trajectory retains displacement error that interpolation may not recover at equal total NFE. The observed crossover regions around 13, 15, and 21 initial steps for FLUX.1-dev, SD3.5, and SD3 depend on the tested schedule, resolution, and prompt set.

For a specified initial budget a, insertion strategy with total b, and matched evaluation set, the relevant comparison is

$$\Delta_{a,b} = \overline{\mathrm{ImRe}}_{\mathrm{STIR}(a,b)} - \overline{\mathrm{ImRe}}_{\mathrm{direct}(a+b)}. \tag{9}$$

A nonnegative reported $\Delta_{a,b}$ describes that setting's mean comparison, not a certified threshold for individual samples or other settings. For example, FLUX.1-dev at $13 + 1$ and direct 14 both report 0.853 after rounding, whereas $12 + 1$ reports 0.838 versus 0.847 for direct 13. The apparent crossover depends on b, the insertion strategy, and score precision; no confidence interval for its location is available. These observations therefore cannot be used as an automatic stopping rule.

Effect of interpolation strategies. At equal initial and insertion budgets, several strategies yield numerically close mean ImageReward. Inserting multiple points per round is sometimes slightly

Table 2: Results on Stable Diffusion 3 and Stable Diffusion 3.5. All entries are average ImageReward (ImRe) scores evaluated over the same set of 5,000 prompts. Complete ImageReward and CLIP results for all evaluated step configurations are provided in Appendix H.

Inserted NFEs b by round	Initial NFE a									
	Stable Diffusion 3					Stable Diffusion 3.5				
	20	21	22	23	24	14	15	16	17	18
0	0.615	0.626	0.633	0.642	0.644	0.574	0.593	0.596	0.602	0.617
1	0.626	0.636	0.643	0.650	0.652	0.590	0.608	0.610	0.613	0.628
1+1	0.632	0.643	0.649	0.656	0.658	0.597	0.614	0.617	0.620	0.635
1+1+1	0.636	0.646	0.651	0.658	0.659	0.608	0.623	0.626	0.627	0.640
1+1+1+1	0.639	0.648	0.654	0.660	0.662	0.608	0.624	0.626	0.628	0.641
1+1+1+1+1	0.641	0.650	0.656	0.662	0.663	0.614	0.629	0.631	0.632	0.645
1+1+1+1+1+1	0.642	0.651	0.656	0.663	0.664	0.617	0.632	0.634	0.635	0.645
2	0.634	0.643	0.649	0.656	0.658	0.605	0.620	0.623	0.625	0.637
2+2	0.639	0.649	0.654	0.661	0.662	0.611	0.625	0.628	0.629	0.641
2+2+2	0.643	0.652	0.656	0.664	0.664	0.620	0.633	0.636	0.636	0.647
3	0.638	0.647	0.652	0.659	0.661	0.611	0.626	0.628	0.631	0.641
3+3	0.642	0.651	0.656	0.663	0.664	0.620	0.634	0.635	0.637	0.647
4	0.640	0.649	0.654	0.661	0.662	0.615	0.631	0.632	0.634	0.645
4+2	0.643	0.652	0.656	0.662	0.665	0.621	0.635	0.636	0.637	0.649

better than distributing single insertions across more rounds, but the means alone do not establish perceptual equivalence or a significant preference.

Effect of model choice. The three backbones show improved equal-budget competitiveness as the initial trajectory becomes better resolved. The crossover region and preferred insertion strategy remain model- and budget-dependent.

Conditional refinement with STORK. We evaluate STIR+STORK on FLUX.1-dev using **1,000 matched samples**, separately from the 5,000-sample Euler study, using STORK for interpolation and propagation and STIR-FM for interval selection and cached velocity correction. Starting from STORK-10, samples receive up to four single-evaluation refinement rounds until their history-best CLIP score reaches a matched STORK-12 reference; ImageReward is held out from stopping and selection. As shown in Table 3, refinement uses $10{,}888$ generator NFEs (10.888 per sample), a 9.27% reduction relative to fixed STORK-12, with numerically close mean ImageReward (0.826233 versus 0.826220). At the cap, $974/1{,}000$ samples meet the reference, and all 26 remaining samples are included in the final means. This demonstrates conditional budget allocation under reference-based stopping; the protocol assumes a precomputed reference for each sample. Appendix K.1 provides the full protocol, NFE accounting, and detailed interpretation.

Interval selection and stabilization ablations. Starting from common 15-step FLUX.1-dev trajectories, we compare velocity-based interval selection with random-interval midpoint and uniform-time insertion at equal NFE, using the same correction and propagation operations (Table 4). After three insertions, mean ImageReward reaches 0.878, 0.869, and 0.865, respectively, favoring velocity-based placement in the reported means; paired uncertainty estimates and repeated random-selection trials are still needed. Appendix K.2 provides the detailed selection protocols, gain comparisons, and scope of the sensitivity result.

4.3 Limitations

Coarse-path errors limit refinement at very low initial NFE, consistent with Theorem E.1: an overly coarse initial trajectory retains a displacement floor that additional insertions cannot eliminate. The coordinate-wise correction estimates the spatial slope from three trajectory samples and therefore captures a first-order term along each coordinate, omitting cross-coordinate interactions.

Table 3: CLIP-controlled STIR+STORK on FLUX.1-dev with **1,000 matched samples**, separate from the 5,000-sample Euler study. From STORK-10, only samples below their matched STORK-12 CLIP reference receive another evaluation, up to four rounds. CLIP selects the history-best output; ImageReward (ImRe) is held out from stopping and selection. "Below" counts samples still below the reference; final means include all samples, including failures at the cap. A dash means unreported. NFE counts generator evaluations, excluding reference generation, decoding, and scoring.

Configuration / maximum refinement	Held-out ImRe	CLIP	Total NFE	Mean NFE	Below
Fixed STORK-12	0.826220	26.098	12,000	12.000	0
STORK-10	–	26.038	10,000	10.000	530
$10+1$	–	26.060	10,530	10.530	226
$10+1+1$	–	26.081	10,756	10.756	89
$10+1+1+1$	–	26.093	10,845	10.845	43
$10+1+1+1+1$	0.826233	26.099	10,888	10.888	26

Table 4: Interval-selection ablation on FLUX.1-dev: mean ImageReward from common 15-step initial trajectories. Each round adds one evaluation; strategies have equal NFE in each row. Velocity variation selects the highest-scoring interval's midpoint; random-interval selection samples an interval uniformly and inserts its midpoint; uniform-time insertion samples a time uniformly over $[0, 1]$. All strategies share correction and propagation, isolating the insertion-location rule. Reported differences describe means without uncertainty estimates.

Configuration	NFE	Velocity variation	Random interval midpoint	Uniform-time insertion
15	15	0.858	0.858	0.858
$15+1$	16	0.865	0.862	0.860
$15+1+1$	17	0.870	0.863	0.861
$15+1+1+1$	18	0.878	0.869	0.865

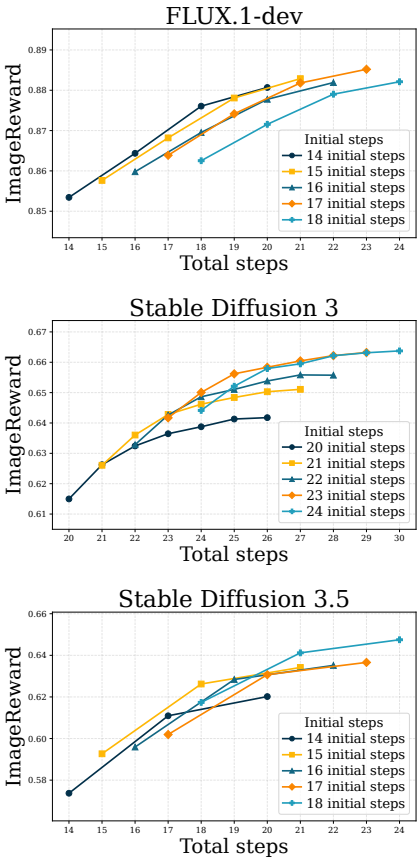


Figure 4: ImageReward curves, from top to bottom: FLUX.1-dev ($2+2+2$), SD3 ($1+1+1+1+1+1+1$), and SD3.5 ($3+3$). Curves correspond to initial trajectory lengths; the horizontal axis shows total NFE after refinement.

5 Conclusion

We proposed **STIR-FM**, a training-free framework for refining an existing flow-matching trajectory through interval selection, new midpoint evaluations, and cached local velocity corrections. The 5,000-sample Euler experiments assess quality at total NFE $a + b$ and report mean ImageReward improvements over the initial trajectories, with numerically competitive equal-budget results in several settings. The separate 1,000-sample STORK experiment reports a 9.27% reduction in counted generator NFEs under reference-based CLIP stopping and a mean ImageReward difference of 0.000013. The reference-based protocol requires a precomputed reference for each sample.

These results point to a way of reading the sampling budget: STIR-FM turns refinement error into an interval-level quantity, so the value of one further evaluation can be judged by the interval it addresses rather than by the step index it occupies. Since the interval score is computed from cached states and velocities, it costs no additional generator NFE and stays available throughout refinement, enabling sample-specific interval allocation within the fixed insertion budget. We hope this interval-selective interpolation strategy can inspire future work on adaptive post-sampling optimization and more efficient generative model inference.

AI USE STATEMENT

We used large language models to assist with refining mathematical proofs and polishing the writing, including improving the clarity and readability of the manuscript. The authors retain responsibility for verifying the mathematical arguments and for the accuracy, originality, and integrity of the final content, including material prepared with AI assistance.

REFERENCES

Daniel Bolya, Po-Yao Huang, Peize Sun, Jang Hyun Cho, Andrea Madotto, Chen Wei, Tengyu Ma, Jiale Zhi, Jathushan Rajasegaran, Hanoona Abdul Rasheed, Junke Wang, Marco Monteiro, Hu Xu, Shiyu Dong, Nikhila Ravi, Shang-Wen Li, Piotr Dollar, and Christoph Feichtenhofer. Perception encoder: The best visual embeddings are not at the output of the network. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026.

Siyu Cao, Hangting Chen, Peng Chen, Yiji Cheng, Yutao Cui, Xinchi Deng, Ying Dong, Kipper Gong, Tianpeng Gu, Xiusen Gu, et al. Hunyuanimage 3.0 technical report. *arXiv preprint arXiv:2509.23951*, 2025.

Defang Chen, Zhenyu Zhou, Can Wang, Chunhua Shen, and Siwei Lyu. On the trajectory regularity of ode-based diffusion sampling. In *International Conference on Machine Learning*, pp. 7905–7934. PMLR, 2024a.

Junsong Chen, Shuchen Xue, Yuyang Zhao, Jincheng Yu, Sayak Paul, Junyu Chen, Han Cai, Song Han, and Enze Xie. Sana-sprint: One-step diffusion with continuous-time consistency distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16185–16195, 2025.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Yuzhu Chen, Fengxiang He, Shi Fu, Xinmei Tian, and Dacheng Tao. Adaptive time-stepping schedules for diffusion models. In *The 40th Conference on Uncertainty in Artificial Intelligence*, 2024b.

Jang Hyun Cho, Andrea Madotto, Effrosyni Mavroudi, Triantafyllos Afouras, Tushar Nagarajan, Muhammad Maaz, Yale Song, Tengyu Ma, Shuming Hu, Suyog Jain, Miguel Martin, Huiyu Wang, Hanoona Abdul Rasheed, Peize Sun, Po-Yao Huang, Daniel Bolya, Nikhila Ravi, Shashank Jain, Tammy Stark, Seungwhan Moon, Babak Damavandi, Vivian Lee, Andrew Westbury, Salman Khan, Philipp Kraehenbuehl, Piotr Dollar, Lorenzo Torresani, Kristen Grauman, and Christoph Feichtenhofer. Perceptionlm: Open-access data and models for detailed visual understanding. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026.

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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 2654–2662, 2025.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning*, pp. 12606–12633. PMLR, 2024.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026.

Ernst Hairer, Gerhard Wanner, and Syvert P Nørsett. *Solving ordinary differential equations I: Nonstiff problems*. Springer, 1993.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 7514–7528, 2021.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, et al. Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025.

Haeil Lee, Hansang Lee, Seoyeon Gye, and Junmo Kim. Beta sampling is all you need: Efficient image generation strategy for diffusion models using stepwise spectral analysis. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 4215–4224. IEEE, 2025.

Shanchuan Lin, Bingchen Liu, Jiashi Li, and Xiao Yang. Common diffusion noise schedules and sample steps are flawed. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 5404–5411, 2024.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *International Conference on Learning Representations*, 2022.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023.

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 *The Twelfth International Conference on Learning Representations*, 2024.

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. *Advances in neural information processing systems*, 35:5775–5787, 2022.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 22 (4):730–751, 2025.

Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pp. 8162–8171. PMLR, 2021.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. Align your steps: Optimizing sampling schedules in diffusion models. In *International Conference on Machine Learning*, pp. 42947–42975. PMLR, 2024.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

Team Seedance, De Chen, Liyang Chen, Xin Chen, Ying Chen, Zhuo Chen, Zhuowei Chen, Feng Cheng, Tianheng Cheng, Yufeng Cheng, et al. Seedance 2.0: Advancing video generation for world complexity. *arXiv preprint arXiv:2604.14148*, 2026.

Team Seedream, Yunpeng Chen, Yu Gao, Lixue Gong, Meng Guo, Qiushan Guo, Zhiyao Guo, Xiaoxia Hou, Weilin Huang, Yixuan Huang, et al. Seedream 4.0: Toward next-generation multimodal image generation. *arXiv preprint arXiv:2509.20427*, 2025.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, pp. 32211–32252. PMLR, 2023.

Zheng Tan, Weizhen Wang, Andrea Bertozzi, and Ernest Ryu. Stork: Faster diffusion and flow matching sampling by resolving both stiffness and structure-dependence. In *International Conference on Learning Representations*, volume 2026, pp. 75478–75506, 2026.

Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. `https://github.com/huggingface/diffusers`, 2022.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Bing Wu, Chang Zou, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Jack Peng, Jianbing Wu, Jiangfeng Xiong, Jie Jiang, et al. Hunyuanvideo 1.5 technical report. *arXiv preprint arXiv:2511.18870*, 2025a.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025b.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:15903–15935, 2023.

Shuchen Xue, Zhaoqiang Liu, Fei Chen, Shifeng Zhang, Tianyang Hu, Enze Xie, and Zhenguo Li. Accelerating diffusion sampling with optimized time steps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8292–8301, 2024.

Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. In *The Eleventh International Conference on Learning Representations*, 2023.

Qinsheng Zhang, Molei Tao, and Yongxin Chen. gDDIM: Generalized denoising diffusion implicit models. In *The Eleventh International Conference on Learning Representations*, 2023.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *Advances in Neural Information Processing Systems*, 36:49842–49869, 2023.

Jianbin Zheng, Minghui Hu, Zhongyi Fan, Chaoyue Wang, Changxing Ding, Dacheng Tao, and Tat-Jen Cham. Trajectory consistency distillation: Improved latent consistency distillation by semi-linear consistency function with trajectory mapping. *arXiv preprint arXiv:2402.19159*, 2024.

Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. Dpm-solver-v3: Improved diffusion ode solver with empirical model statistics. *Advances in Neural Information Processing Systems*, 36: 55502–55542, 2023.

Algorithm 2 Error Estimation

Require: Refined velocities $\{v_k\}_{k=0}^{N-1}$, timestamps $\{t_k\}_{k=0}^{N}$

for $k = 0$ to $N-2$ **do**
  $\Delta v \leftarrow v_{k+1} - v_k$
  $\Delta t \leftarrow |t_{k+1} - t_k|$
  $e_k \leftarrow \|\Delta v\|_2 \cdot \Delta t$
end for
$\Delta v_{\text{last}} \leftarrow v_{N-1} - v_{N-2}$
$\Delta t_{\text{last}} \leftarrow |t_N - t_{N-1}|$
$\Delta t_{\text{prev}} \leftarrow |t_{N-1} - t_{N-2}|$
$e_{N-1} \leftarrow \|\Delta v_{\text{last}}\|_2 \cdot \Delta t_{\text{last}}^2 / \Delta t_{\text{prev}}$
return $\{e_k\}_{k=0}^{N-1}$

Algorithm 3 Adaptive Interpolation

Require: Model f_θ, insertion index k, refined trajectory $\{x_k\}_{k=0}^{N}$, refined velocities $\{v_k\}_{k=0}^{N-1}$, original reference points $\{x_k^{\text{ori}}\}_{k=0}^{N}$, original model predictions $\{v_k^{\text{ori}}\}_{k=0}^{N-1}$, timestamps $\{t_k\}_{k=0}^{N}$

$t_{\text{mid}} \leftarrow \dfrac{t_k + t_{k+1}}{2}$
$x_{\text{mid}} \leftarrow x_k + (t_{\text{mid}} - t_k) v_k$
$v_{\text{mid}} \leftarrow f_\theta(x_{\text{mid}}, t_{\text{mid}})$
Insert x_{mid} into $\{x_k\}$ at index $k+1$
Insert v_{mid} into $\{v_k\}$ at index $k+1$
Insert x_{mid} into $\{x_k^{\text{ori}}\}$ at index $k+1$
Insert v_{mid} into $\{v_k^{\text{ori}}\}$ at index $k+1$
Insert t_{mid} into $\{t_k\}$ at index $k+1$
return $\{x_k\}_{k=0}^{N+1}, \{v_k\}_{k=0}^{N}, \{x_k^{\text{ori}}\}_{k=0}^{N+1}, \{v_k^{\text{ori}}\}_{k=0}^{N}, \{t_k\}_{k=0}^{N+1}$

A Detailed Pseudocode for Core Subroutines

This appendix provides the detailed pseudocode for the three core subroutines used in STIR-FM. Algorithm 2 computes interval-wise error scores from the refined velocity sequence and is used to select high-error intervals for interpolation. Algorithm 3 inserts a midpoint into a selected interval, evaluates the model at the new state, and updates both the refined sequences and the fixed reference sequences. Algorithm 4 refines the velocity at an affected point using neighboring reference states and reference velocity predictions. Together with the overall segment-wise propagation procedure in Algorithm 1, these subroutines define one complete refinement round of STIR-FM.

A.1 Notation and a Worked Reindexing Example

Symbol	Meaning
N	Number of intervals at the beginning of the current round.
(x_j, v_j)	Working state and propagation velocity; both may change.
$(x_j^{\text{ori}}, v_j^{\text{ori}})$	Evaluated reference pair; values are immutable after creation, although its index shifts after insertion. The terminal state has no evaluated velocity.
M, K	Requested insertions and actual insertions $K = \min(M, N)$.
$p_m, \hat{p}_m$	Selected interval's left index on the round-start grid and its current index $\hat{p}_m = p_m + m - 1$.
L, R	Inclusive downstream propagation bounds in Algorithm 1.

Consider $N = 5$, $K = 2$, and selected round-start indices $p_1 = 1$, $p_2 = 3$. Selection is frozen within the round; scores are recomputed on the updated working trajectory in the next round. Use

Algorithm 4 Local Velocity Refinement

Require: Refined trajectory $\{x_k\}$, original reference points $\{x_k^{\mathrm{ori}}\}$, original model predictions $\{v_k^{\mathrm{ori}}\}$, timestamps $\{t_k\}$, target index j; suppression fraction $\rho = 0.20$

$k \leftarrow j - 1$

$\mathbf{n} \leftarrow \big((v_{k+1}^{\mathrm{ori}} - v_k^{\mathrm{ori}})(t_{k+2} - t_k) - (v_{k+2}^{\mathrm{ori}} - v_k^{\mathrm{ori}})(t_{k+1} - t_k)\big) \odot (x_{k+1} - x_{k+1}^{\mathrm{ori}})$

$\mathbf{d} \leftarrow (x_{k+1}^{\mathrm{ori}} - x_k^{\mathrm{ori}})(t_{k+2} - t_k) - (x_{k+2}^{\mathrm{ori}} - x_k^{\mathrm{ori}})(t_{k+1} - t_k)$

$\tau_\rho \leftarrow \mathrm{Quantile}_\rho(\{|d_i|\}_i)$ over this sample's latent coordinates

$\mathbf{r} \leftarrow \mathbf{0}$

for each coordinate i with $|d_i| > \tau_\rho$ **do**

    $r_i \leftarrow n_i / d_i$ ▷ Divide only on unmasked coordinates

end for

$v_j \leftarrow v_{k+1}^{\mathrm{ori}} + \mathbf{r}$

return v_j

$u = (t_1 + t_2)/2$ and $w = (t_3 + t_4)/2$ to denote the new timestamps:

Stage	Timestamps in current index order, starting at 0
Round start	$t_0, t_1, t_2, t_3, t_4, t_5$
First insertion	$t_0, t_1, u, t_2, t_3, t_4, t_5$
Second insertion	$t_0, t_1, u, t_2, t_3, w, t_4, t_5$

The first insertion has $\hat{p}_1 = 1$ and creates a reference pair at index 2 immediately after evaluating the model at u. Propagation runs from $L = 3$ through $R = p_2 + 1 = 4$, updating the states at the original t_2, t_3 and their cached velocities. It stops at the left endpoint of the next selected interval, which is now at index $\hat{p}_2 = 4$. The second insertion evaluates the model at w, creates its reference pair at index 5, and propagates indices $6, 7$ to the endpoint. Index 6 retains its stored velocity under the Euler tail rule, and index 7 is the terminal state with no stored velocity. Existing reference values never move with the working states: only their array indices change. The round makes exactly two new model evaluations, at u and w; downstream propagation and cached corrections make none. The new reference pairs persist into later rounds even if their working states subsequently change.

B Justification of the Error Indicator

In this section, we justify the first-order error score used in the main method:

$$e_k = \|v_{k+1} - v_k\|_2 \, (t_{k+1} - t_k). \tag{10}$$

Here, $\{v_k\}$ denotes the current refined velocity sequence used for trajectory propagation.

Lemma B.1. *Consider the latent dynamics*

$$\frac{dx(t)}{dt} = v(t), \tag{11}$$

on an interval $[t_k, t_{k+1}]$, where $v(t)$ is twice continuously differentiable. Let

$$\Delta t_k = t_{k+1} - t_k.$$

Then the local truncation error of the Euler update

$$x_{k+1}^{\mathrm{Euler}} = x(t_k) + \Delta t_k \, v(t_k) \tag{12}$$

satisfies

$$x(t_{k+1}) - x_{k+1}^{\mathrm{Euler}} = \frac{\Delta t_k^2}{2} \dot{v}(t_k) + O(\Delta t_k^3). \tag{13}$$

Moreover, if $\dot{v}(t_k)$ is approximated by the finite difference

$$\dot{v}(t_k) \approx \frac{v(t_{k+1}) - v(t_k)}{\Delta t_k}, \tag{14}$$

then

$$\left\|x(t_{k+1}) - x_{k+1}^{\text{Euler}}\right\|_2 \approx \frac{1}{2}\|v(t_{k+1}) - v(t_k)\|_2 \, \Delta t_k. \tag{15}$$

Therefore, the score

$$e_k = \|v_{k+1} - v_k\|_2 \, (t_{k+1} - t_k) \tag{16}$$

is proportional to the leading-order local truncation error of Euler integration on the interval $[t_k, t_{k+1}]$.

Proof. By the integral form of the dynamics, the exact solution at the right endpoint can be written as

$$x(t_{k+1}) = x(t_k) + \int_{t_k}^{t_{k+1}} v(t)\, dt. \tag{17}$$

We perform a second-order Taylor expansion of $v(t)$ around t_k:

$$v(t) = v(t_k) + \dot{v}(t_k)(t - t_k) + \frac{1}{2}\ddot{v}(\xi_t)(t - t_k)^2, \qquad \xi_t \in [t_k, t]. \tag{18}$$

Substituting Eq. (18) into Eq. (17) yields

$$\begin{aligned} x(t_{k+1}) &= x(t_k) + \int_{t_k}^{t_{k+1}} \left[v(t_k) + \dot{v}(t_k)(t - t_k) + \frac{1}{2}\ddot{v}(\xi_t)(t - t_k)^2 \right] dt \\ &= x(t_k) + \Delta t_k \, v(t_k) + \frac{\Delta t_k^2}{2}\dot{v}(t_k) + r_k, \end{aligned} \tag{19}$$

where the remainder term satisfies $r_k = O(\Delta t_k^3)$ because $v(t)$ is twice continuously differentiable on $[t_k, t_{k+1}]$.

From Eq. (12), the Euler approximation is

$$x_{k+1}^{\text{Euler}} = x(t_k) + \Delta t_k \, v(t_k).$$

Subtracting it from Eq. (19) gives

$$x(t_{k+1}) - x_{k+1}^{\text{Euler}} = \frac{\Delta t_k^2}{2}\dot{v}(t_k) + O(\Delta t_k^3), \tag{20}$$

which proves Eq. (13).

Next, we approximate the first derivative of the velocity by the endpoint finite difference:

$$\dot{v}(t_k) \approx \frac{v(t_{k+1}) - v(t_k)}{\Delta t_k}. \tag{21}$$

Substituting Eq. (21) into the leading term of Eq. (20) yields

$$x(t_{k+1}) - x_{k+1}^{\text{Euler}} \approx \frac{\Delta t_k}{2}\big(v(t_{k+1}) - v(t_k)\big). \tag{22}$$

Taking the ℓ_2 norm on both sides, we obtain

$$\left\|x(t_{k+1}) - x_{k+1}^{\text{Euler}}\right\|_2 \approx \frac{1}{2}\|v(t_{k+1}) - v(t_k)\|_2 \, \Delta t_k. \tag{23}$$

Finally, identifying

$$v_k = v(t_k), \qquad v_{k+1} = v(t_{k+1}),$$

we conclude that

$$\left\|x(t_{k+1}) - x_{k+1}^{\text{Euler}}\right\|_2 \approx \frac{1}{2}\|v_{k+1} - v_k\|_2 \, (t_{k+1} - t_k) = \frac{e_k}{2}. \tag{24}$$

Hence, e_k is proportional to the leading-order local truncation error, and is therefore a valid score for ranking unreliable intervals. □

C JUSTIFICATION OF THE LOCAL VELOCITY REFINEMENT

In this section, we justify the local velocity refinement rule used in the main method. The key idea is to locally linearize the velocity field around the reference middle point and use its neighboring reference samples to estimate the spatial derivative of the velocity. The refined velocity at the updated middle point is then obtained by a first-order correction along the spatial displacement.

Lemma C.1. *Let $v(x,t)$ be a twice continuously differentiable scalar velocity component in a neighborhood of $(x^{\mathrm{ori}}_{k+1}, t_{k+1})$. Assume that the neighboring samples*

$$(x^{\mathrm{ori}}_{k}, t_k, v^{\mathrm{ori}}_{k}), \qquad (x^{\mathrm{ori}}_{k+1}, t_{k+1}, v^{\mathrm{ori}}_{k+1}), \qquad (x^{\mathrm{ori}}_{k+2}, t_{k+2}, v^{\mathrm{ori}}_{k+2})$$

lie in this neighborhood. Then, after first-order Taylor expansion around $(x^{\mathrm{ori}}_{k+1}, t_{k+1})$ and neglecting higher-order remainders, the spatial derivative v_x can be approximated by

$$v_x \approx \frac{(v^{\mathrm{ori}}_{k} - v^{\mathrm{ori}}_{k+1})(t_{k+2} - t_{k+1}) - (v^{\mathrm{ori}}_{k+2} - v^{\mathrm{ori}}_{k+1})(t_k - t_{k+1})}{(x^{\mathrm{ori}}_{k} - x^{\mathrm{ori}}_{k+1})(t_{k+2} - t_{k+1}) - (x^{\mathrm{ori}}_{k+2} - x^{\mathrm{ori}}_{k+1})(t_k - t_{k+1})}. \tag{25}$$

Consequently, for an updated middle state x_{k+1} at the same timestamp t_{k+1}, the velocity at the updated point admits the first-order approximation

$$v_{k+1} \approx v^{\mathrm{ori}}_{k+1} + (x_{k+1} - x^{\mathrm{ori}}_{k+1})\, v_x. \tag{26}$$

Equivalently, Eq. (25) can be rewritten as

$$v_x \approx \frac{(v^{\mathrm{ori}}_{k+1} - v^{\mathrm{ori}}_{k})(t_{k+2} - t_k) - (v^{\mathrm{ori}}_{k+2} - v^{\mathrm{ori}}_{k})(t_{k+1} - t_k)}{(x^{\mathrm{ori}}_{k+1} - x^{\mathrm{ori}}_{k})(t_{k+2} - t_k) - (x^{\mathrm{ori}}_{k+2} - x^{\mathrm{ori}}_{k})(t_{k+1} - t_k)}, \tag{27}$$

which matches the implementation form used in the algorithm.

Proof. We expand the scalar velocity component $v(x,t)$ around the original middle point $(x^{\mathrm{ori}}_{k+1}, t_{k+1})$. For the left neighbor, a first-order Taylor expansion gives

$$v^{\mathrm{ori}}_{k} = v^{\mathrm{ori}}_{k+1} + (x^{\mathrm{ori}}_{k} - x^{\mathrm{ori}}_{k+1})\, v_x + (t_k - t_{k+1})\, v_t + R_k, \tag{28}$$

where $v_x := \partial v/\partial x$ and $v_t := \partial v/\partial t$ denote the partial derivatives of v with respect to x and t, respectively, evaluated at $(x^{\mathrm{ori}}_{k+1}, t_{k+1})$, and R_k is the higher-order remainder. Similarly, for the right neighbor,

$$v^{\mathrm{ori}}_{k+2} = v^{\mathrm{ori}}_{k+1} + (x^{\mathrm{ori}}_{k+2} - x^{\mathrm{ori}}_{k+1})\, v_x + (t_{k+2} - t_{k+1})\, v_t + R_{k+2}. \tag{29}$$

Neglecting the higher-order remainders R_k and R_{k+2}, we obtain the linear system

$$v^{\mathrm{ori}}_{k} - v^{\mathrm{ori}}_{k+1} = (x^{\mathrm{ori}}_{k} - x^{\mathrm{ori}}_{k+1})\, v_x + (t_k - t_{k+1})\, v_t, \tag{30}$$

$$v^{\mathrm{ori}}_{k+2} - v^{\mathrm{ori}}_{k+1} = (x^{\mathrm{ori}}_{k+2} - x^{\mathrm{ori}}_{k+1})\, v_x + (t_{k+2} - t_{k+1})\, v_t. \tag{31}$$

These are two equations with two unknowns, namely v_x and v_t. Eliminating v_t from Eq. (30) and Eq. (31) yields

$$v_x \approx \frac{(v^{\mathrm{ori}}_{k} - v^{\mathrm{ori}}_{k+1})(t_{k+2} - t_{k+1}) - (v^{\mathrm{ori}}_{k+2} - v^{\mathrm{ori}}_{k+1})(t_k - t_{k+1})}{(x^{\mathrm{ori}}_{k} - x^{\mathrm{ori}}_{k+1})(t_{k+2} - t_{k+1}) - (x^{\mathrm{ori}}_{k+2} - x^{\mathrm{ori}}_{k+1})(t_k - t_{k+1})}, \tag{32}$$

which proves Eq. (25).

Now consider the updated middle state x_{k+1} at the same time t_{k+1}. Applying a first-order expansion of the velocity field at $(x^{\mathrm{ori}}_{k+1}, t_{k+1})$ in the spatial direction gives

$$v(x_{k+1}, t_{k+1}) \approx v(x^{\mathrm{ori}}_{k+1}, t_{k+1}) + (x_{k+1} - x^{\mathrm{ori}}_{k+1})\, v_x. \tag{33}$$

Since

$$v(x^{\mathrm{ori}}_{k+1}, t_{k+1}) = v^{\mathrm{ori}}_{k+1},$$

we obtain

$$v_{k+1} \approx v^{\mathrm{ori}}_{k+1} + (x_{k+1} - x^{\mathrm{ori}}_{k+1})\, v_x, \tag{34}$$

which proves Eq. (26).

Finally, Eq. (27) follows from Eq. (25) by simple algebraic rearrangement of the time and state differences, so the appendix derivation is equivalent to the implementation formula used in the code. □

C.1 Numerical Stabilization of the Local Correction

The local correction in Eq. (4) divides by a reference-displacement bracket, which can be small for individual latent coordinates. We therefore apply the following mask in the reported implementation. Let A be the velocity-difference bracket in the numerator of Eq. (4), D its denominator bracket, and $\delta x = x_{k+1} - x^{\mathrm{ori}}_{k+1}$. For each sample and target node, the quantile is computed over all scalar coordinates of the latent tensor, not across samples or time:

$$\begin{cases} \tau_\rho = \mathrm{Quantile}_\rho\big(\{|D_i|\}_i\big), & \rho = 0.20, \\ \Delta v^{\mathrm{stab}}_{k+1,i} = \begin{cases} 0, & |D_i| \le \tau_\rho, \\ A_i\,\delta x_i / D_i, & |D_i| > \tau_\rho. \end{cases} \end{cases} \tag{35}$$

Using Eq. (35) in place of Δv_{k+1} in Eq. (5) gives the velocity update used in all reported experiments. The mask is applied before division, so zero denominators and entries equal to the threshold receive zero correction; ties can suppress more than 20% of coordinates. A small denominator makes the local slope poorly determined, so we suppress those coordinates instead. The sensitivity study in Appendix K.2 covers $\rho \in \{0.10, 0.20, 0.30\}$.

D End-to-End Error Analysis

In this section, we analyze the end-to-end error induced by the proposed refinement method. The analysis is decomposed into three parts: (i) the error of the locally estimated spatial derivative, (ii) the truncation error introduced by the single-point velocity refinement, and (iii) the accumulation of the Euler propagation error along the refined trajectory.

For notational simplicity, we first present the derivation for one scalar component of the velocity field. The vector-valued case follows component-wise, and the same norm bounds apply up to dimension-dependent constants.

D.1 Assumptions

Assumption D.1 (Smoothness)**.** *The velocity field $v(x,t)$ is twice continuously differentiable on a neighborhood of the exact trajectory. Moreover, there exists a constant $M_2 > 0$ such that all second-order derivatives of v are bounded in norm by M_2.*

Assumption D.2 (Lipschitz continuity)**.** *There exist constants $L_x, L_t > 0$ such that for all $x, y \in \mathbb{R}^d$ and $s, t \in [0,1]$,*

$$\|v(x,t) - v(y,t)\| \le L_x \|x - y\|, \qquad \|v(x,t) - v(x,s)\| \le L_t |t - s|. \tag{36}$$

Assumption D.3 (Local non-degeneracy)**.** *For each locally refined point, define*

$$D_k = (x^{\mathrm{ori}}_k - x^{\mathrm{ori}}_{k+1})(t_{k+2} - t_{k+1}) - (x^{\mathrm{ori}}_{k+2} - x^{\mathrm{ori}}_{k+1})(t_k - t_{k+1}). \tag{37}$$

There exists a constant $\kappa > 0$ such that

$$|D_k| \ge \kappa h_k^2, \tag{38}$$

where $h_k = \max\{|t_{k+1} - t_k|,\ |t_{k+2} - t_{k+1}|\}$.

D.2 Error of the Estimated Spatial Derivative

Lemma D.1. *Under Assumptions D.1 and D.3, the spatial derivative estimated by the local linear system satisfies*

$$|\hat{v}_x - v_x| \le C_D h_k, \tag{39}$$

where v_x is the exact spatial derivative evaluated at $(x^{\mathrm{ori}}_{k+1}, t_{k+1})$, and C_D is a constant independent of h_k.

Proof. Expand the left and right neighboring velocities around $(x^{\mathrm{ori}}_{k+1}, t_{k+1})$:

$$v^{\mathrm{ori}}_k = v^{\mathrm{ori}}_{k+1} + (x^{\mathrm{ori}}_k - x^{\mathrm{ori}}_{k+1}) v_x + (t_k - t_{k+1}) v_t + R_k, \tag{40}$$

$$v_{k+2}^{\text{ori}} = v_{k+1}^{\text{ori}} + (x_{k+2}^{\text{ori}} - x_{k+1}^{\text{ori}})v_x + (t_{k+2} - t_{k+1})v_t + R_{k+2}. \tag{41}$$

By Assumption D.1, the Taylor remainders satisfy

$$|R_k| + |R_{k+2}| \le C_R h_k^2 \tag{42}$$

for some constant $C_R > 0$.

Ignoring the remainder terms yields the linear estimator

$$\hat{v}_x = \frac{(v_k^{\text{ori}} - v_{k+1}^{\text{ori}})(t_{k+2} - t_{k+1}) - (v_{k+2}^{\text{ori}} - v_{k+1}^{\text{ori}})(t_k - t_{k+1})}{D_k}. \tag{43}$$

Substituting Eq. (40) and Eq. (41) into Eq. (43), we obtain

$$\hat{v}_x - v_x = \frac{R_k(t_{k+2} - t_{k+1}) - R_{k+2}(t_k - t_{k+1})}{D_k}. \tag{44}$$

Using Eq. (42), the fact that

$$|t_k - t_{k+1}| \le h_k, \qquad |t_{k+2} - t_{k+1}| \le h_k,$$

and Assumption D.3, we conclude

$$|\hat{v}_x - v_x| \le \frac{2C_R h_k^3}{\kappa h_k^2} = \frac{2C_R}{\kappa} h_k. \tag{45}$$

This proves the claim. □

D.3 Single-Point Truncation Error of the Velocity Refinement

Let

$$\delta_k = \|x_{k+1} - x_{k+1}^{\text{ori}}\|$$

denote the displacement between the updated point and the original point at the same timestamp.

Lemma D.2. *Under Assumption D.1, the locally refined velocity*

$$v_{k+1} = v_{k+1}^{\text{ori}} + (x_{k+1} - x_{k+1}^{\text{ori}})\hat{v}_x \tag{46}$$

satisfies

$$\|v(x_{k+1}, t_{k+1}) - v_{k+1}\| \le C_1 h_k \delta_k + C_2 \delta_k^2, \tag{47}$$

for some constants $C_1, C_2 > 0$. In particular, if $\delta_k = O(h_k)$, then

$$\|v(x_{k+1}, t_{k+1}) - v_{k+1}\| = O(h_k^2). \tag{48}$$

Proof. By Taylor expansion of $v(x, t_{k+1})$ around x_{k+1}^{ori},

$$v(x_{k+1}, t_{k+1}) = v(x_{k+1}^{\text{ori}}, t_{k+1}) + (x_{k+1} - x_{k+1}^{\text{ori}})v_x + R_{k+1}^{\text{sp}}, \tag{49}$$

where the remainder satisfies

$$\|R_{k+1}^{\text{sp}}\| \le C_2 \delta_k^2 \tag{50}$$

for some constant $C_2 > 0$. Since

$$v(x_{k+1}^{\text{ori}}, t_{k+1}) = v_{k+1}^{\text{ori}},$$

subtracting Eq. (46) from Eq. (49) yields

$$v(x_{k+1}, t_{k+1}) - v_{k+1} = (x_{k+1} - x_{k+1}^{\text{ori}})(v_x - \hat{v}_x) + R_{k+1}^{\text{sp}}. \tag{51}$$

Taking norms and applying Lemma D.1, we obtain

$$\|v(x_{k+1}, t_{k+1}) - v_{k+1}\| \le \|v_x - \hat{v}_x\|\, \delta_k + C_2 \delta_k^2 \le C_1 h_k \delta_k + C_2 \delta_k^2, \tag{52}$$

where C_1 absorbs the constant from Lemma D.1. The final statement follows immediately when $\delta_k = O(h_k)$. □

D.4 Euler Propagation Error

Let the refined time grid be

$$0 = \tau_0 < \tau_1 < \cdots < \tau_S = 1,$$

with step sizes

$$h_n = \tau_{n+1} - \tau_n.$$

The refined trajectory is propagated by

$$\hat{x}_{n+1} = \hat{x}_n + h_n v_n. \tag{53}$$

Let the exact solution of the ODE be $x(\tau_n)$, and define the global error

$$e_n = x(\tau_n) - \hat{x}_n.$$

Lemma D.3. *Under Assumptions D.1 and D.2, the error sequence satisfies*

$$\|e_{n+1}\| \le (1 + L_x h_n)\|e_n\| + C_E h_n^2 + h_n \eta_n, \tag{54}$$

where

$$\eta_n = \|v(\hat{x}_n, \tau_n) - v_n\| \tag{55}$$

is the local velocity approximation error, and $C_E > 0$ is a constant independent of h_n.

Proof. By the integral form of the exact solution,

$$x(\tau_{n+1}) = x(\tau_n) + \int_{\tau_n}^{\tau_{n+1}} v(x(s), s)\, ds. \tag{56}$$

A first-order Taylor expansion yields

$$x(\tau_{n+1}) = x(\tau_n) + h_n v(x(\tau_n), \tau_n) + R_n, \tag{57}$$

with

$$\|R_n\| \le C_E h_n^2. \tag{58}$$

Subtracting Eq. (53) from Eq. (57) gives

$$e_{n+1} = e_n + h_n\big(v(x(\tau_n), \tau_n) - v_n\big) + R_n. \tag{59}$$

Now split the velocity difference as

$$v(x(\tau_n), \tau_n) - v_n = \big(v(x(\tau_n), \tau_n) - v(\hat{x}_n, \tau_n)\big) + \big(v(\hat{x}_n, \tau_n) - v_n\big). \tag{60}$$

By Assumption D.2,

$$\|v(x(\tau_n), \tau_n) - v(\hat{x}_n, \tau_n)\| \le L_x \|e_n\|. \tag{61}$$

Substituting Eq. (60), Eq. (58), and Eq. (61) into Eq. (59) proves the recursion. □

D.5 End-to-End Error Bound

Let $\mathcal{C} \subset \{0, 1, \ldots, S-1\}$ denote the set of refined nodes where the local velocity correction is applied.

Theorem D.1. *Under Assumptions D.1–D.3, the terminal error of the refined method satisfies*

$$\|x(1) - \hat{x}_S\| \le \exp(L_x) \left(\|e_0\| + C_E \sum_{n=0}^{S-1} h_n^2 + \sum_{n=0}^{S-1} h_n \eta_n \right), \tag{62}$$

where η_n is defined in Eq. (55). Moreover, if the local refinement is applied only on nodes in $\mathcal{C}$ and Lemma D.2 holds at each such node, then

$$\|x(1) - \hat{x}_S\| \le \exp(L_x) \left(\|e_0\| + C_E \sum_{n=0}^{S-1} h_n^2 + \sum_{n \in \mathcal{C}} h_n \big(C_1 h_n \delta_n + C_2 \delta_n^2\big) \right). \tag{63}$$

If in addition

$$\delta_n \le C_\delta h_n, \qquad n \in \mathcal{C}, \tag{64}$$

then the method is globally first-order accurate:

$$\|x(1) - \hat{x}_S\| \le C_G \exp(L_x) \sum_{n=0}^{S-1} h_n^2 \le C_G \exp(L_x)\, h_{\max}, \tag{65}$$

where $h_{\max} = \max_n h_n$ and $C_G > 0$ is a constant independent of $h_{\max}$.

Proof. Starting from Lemma D.3,

$$\|e_{n+1}\| \le (1 + L_x h_n)\|e_n\| + C_E h_n^2 + h_n \eta_n.$$

Applying the discrete Grönwall inequality over the refined grid yields

$$\|e_S\| \le \exp\left(L_x \sum_{n=0}^{S-1} h_n\right) \left(\|e_0\| + C_E \sum_{n=0}^{S-1} h_n^2 + \sum_{n=0}^{S-1} h_n \eta_n\right). \tag{66}$$

Since $\sum_{n=0}^{S-1} h_n = 1$, Eq. (62) follows.

For unrefined nodes, one may take $\eta_n = 0$ in the idealized setting where the exact velocity is used. For refined nodes $n \in \mathcal{C}$, Lemma D.2 gives

$$\eta_n \le C_1 h_n \delta_n + C_2 \delta_n^2.$$

Substituting this into Eq. (62) yields Eq. (63).

Since the refined method starts from the same initial latent as the exact trajectory, we have $e_0 = 0$. Hence Eq. (63) reduces to

$$\|x(1) - \hat{x}_S\| \le \exp(L_x) \left(C_E \sum_{n=0}^{S-1} h_n^2 + \sum_{n \in \mathcal{C}} h_n \big(C_1 h_n \delta_n + C_2 \delta_n^2\big)\right). \tag{67}$$

Finally, if $\delta_n \le C_\delta h_n$ for all $n \in \mathcal{C}$, then

$$h_n \big(C_1 h_n \delta_n + C_2 \delta_n^2\big) \le (C_1 C_\delta + C_2 C_\delta^2) h_n^2.$$

Hence the whole right-hand side is bounded by a constant multiple of $\sum_n h_n^2$. Using the elementary inequality

$$\sum_{n=0}^{S-1} h_n^2 \le h_{\max} \sum_{n=0}^{S-1} h_n = h_{\max},$$

we obtain Eq. (65).

□

E Influence of the Initial Coarse Step Size on Multi-Round Refinement

In this section, we analyze how the displacement

$$\delta_n = \|\hat{x}_n - x^{(0)}(\tau_n)\|$$

behaves under multi-round interpolation, where $\hat{x}_n$ denotes the refined trajectory on the refined grid and $x^{(0)}(t)$ denotes the path induced by the initial coarse trajectory. We show that δ_n is controlled not only by the current refined step size, but also by the maximum step size of the *initial* coarse trajectory. As a consequence, an overly coarse initial trajectory may lead to a large displacement floor, and inserting too many interpolation points cannot eliminate this floor.

E.1 Setup

Let the initial coarse grid be

$$0 = s_0 < s_1 < \cdots < s_K = 1, \qquad H_0 := \max_{0 \le i \le K-1} (s_{i+1} - s_i).$$

Let $x_i^{(0)}$ be the trajectory generated on this coarse grid, and let $x^{(0)}(t)$ denote its piecewise Euler extension:

$$x^{(0)}(t) = x_i^{(0)} + (t - s_i)\, v(x_i^{(0)}, s_i), \qquad t \in [s_i, s_{i+1}]. \tag{68}$$

After multi-round interpolation, let the refined grid be

$$0 = \tau_0 < \tau_1 < \cdots < \tau_S = 1, \qquad h := \max_{0 \le n \le S-1} (\tau_{n+1} - \tau_n),$$

and let $\hat{x}_n$ be the refined numerical trajectory on this grid.

We assume the same smoothness and Lipschitz conditions as in Appendix D. In addition, assume that the velocity is bounded:

$$\|v(x,t)\| \le M_v \tag{69}$$

on the relevant neighborhood of the trajectory.

E.2 Upper Bound: δ_n Inherits the Initial Coarse Error

Lemma E.1. *Under the smoothness and Lipschitz assumptions, the piecewise coarse path satisfies*

$$\sup_{t\in[0,1]} \|x(t) - x^{(0)}(t)\| \le C_0 H_0, \tag{70}$$

where $x(t)$ is the exact solution and $C_0 > 0$ is a constant independent of H_0.

Proof. This is the standard global first-order error estimate for the explicit Euler method together with its piecewise linear extension. At coarse grid nodes, the Euler global error is $O(H_0)$. Inside each interval $[s_i, s_{i+1}]$, the exact solution and the piecewise extension differ by at most an additional $O(H_0)$ term due to local Taylor expansion. Combining the nodewise global error with the intra-interval interpolation error yields

$$\sup_{t\in[0,1]} \|x(t) - x^{(0)}(t)\| \le C_0 H_0.$$

□

Lemma E.2. *Let*

$$\varepsilon_{\mathrm{ref}}(h) := \max_{0\le n\le S} \|x(\tau_n) - \hat{x}_n\|.$$

Then

$$\delta_n \le C_0 H_0 + \varepsilon_{\mathrm{ref}}(h), \qquad 0 \le n \le S. \tag{71}$$

In particular,

$$\max_n \delta_n \le C_0 H_0 + \varepsilon_{\mathrm{ref}}(h). \tag{72}$$

Proof. By definition,

$$\delta_n = \|\hat{x}_n - x^{(0)}(\tau_n)\|.$$

Using the triangle inequality,

$$\delta_n \le \|\hat{x}_n - x(\tau_n)\| + \|x(\tau_n) - x^{(0)}(\tau_n)\|. \tag{73}$$

The first term is bounded by $\varepsilon_{\mathrm{ref}}(h)$ by definition, and the second term is bounded by $C_0 H_0$ by Lemma E.1. Thus Eq. (71) follows. □

Lemma E.2 shows that even if the refined step size h becomes small, the displacement δ_n still inherits an $O(H_0)$ contribution from the initial coarse path. Therefore, if the initial trajectory is too coarse, i.e., H_0 is too large, then δ_n can remain large throughout refinement.

E.3 Lower Bound: Why Over-Refinement Cannot Eliminate the Displacement Floor

The previous upper bound alone does not explain saturation. To show this, we additionally assume that the initial coarse trajectory is genuinely under-resolved.

Assumption E.1 (Nontrivial coarse-path mismatch)**.** *There exist a constant $c_0 > 0$ and a time $t^\star \in [0,1]$ such that*

$$\|x(t^\star) - x^{(0)}(t^\star)\| \ge c_0 H_0. \tag{74}$$

Lemma E.3. *Under Assumption E.1, there exists a refined node $\tau_{n^\star}$ such that*

$$\delta_{n^\star} \ge c_0 H_0 - C_L h - \varepsilon_{\mathrm{ref}}(h), \tag{75}$$

where $C_L > 0$ is a constant independent of h. Consequently,

$$\max_n \delta_n \ge c_0 H_0 - C_L h - \varepsilon_{\mathrm{ref}}(h). \tag{76}$$

Proof. By Assumption E.1, there exists $t^\star \in [0,1]$ such that

$$\|x(t^\star) - x^{(0)}(t^\star)\| \geq c_0 H_0.$$

Choose the refined node $\tau_{n^\star}$ closest to $t^\star$. Then

$$|\tau_{n^\star} - t^\star| \leq h. \tag{77}$$

Now write

$$\begin{aligned}\delta_{n^\star} &= \|\hat{x}_{n^\star} - x^{(0)}(\tau_{n^\star})\| \\ &\geq \|x(t^\star) - x^{(0)}(t^\star)\| - \|x(t^\star) - x(\tau_{n^\star})\| - \|x^{(0)}(t^\star) - x^{(0)}(\tau_{n^\star})\| \\ &\quad - \|x(\tau_{n^\star}) - \hat{x}_{n^\star}\|.\end{aligned} \tag{78}$$

Since $\dot{x}(t) = v(x(t), t)$ and $\|v(x,t)\| \leq M_v$, the exact trajectory is Lipschitz in time:

$$\|x(t^\star) - x(\tau_{n^\star})\| \leq M_v |t^\star - \tau_{n^\star}| \leq M_v h.$$

Similarly, the piecewise coarse extension $x^{(0)}(t)$ is also Lipschitz in time, with some constant $M_0 > 0$, hence

$$\|x^{(0)}(t^\star) - x^{(0)}(\tau_{n^\star})\| \leq M_0 h.$$

Finally, by the definition of $\varepsilon_{\mathrm{ref}}(h)$,

$$\|x(\tau_{n^\star}) - \hat{x}_{n^\star}\| \leq \varepsilon_{\mathrm{ref}}(h).$$

Substituting these bounds into Eq. (78) yields

$$\delta_{n^\star} \geq c_0 H_0 - (M_v + M_0)h - \varepsilon_{\mathrm{ref}}(h).$$

Setting $C_L := M_v + M_0$ proves the result. □

Combining Lemma E.2 and Lemma E.3, we obtain

$$c_0 H_0 - C_L h - \varepsilon_{\mathrm{ref}}(h) \leq \max_n \delta_n \leq C_0 H_0 + \varepsilon_{\mathrm{ref}}(h). \tag{79}$$

Therefore, when the refined step size h becomes small and the refined numerical error $\varepsilon_{\mathrm{ref}}(h)$ is also small, the maximum displacement remains of order H_0:

$$\max_n \delta_n = \Theta(H_0). \tag{80}$$

This shows that further reducing the refined step size cannot eliminate the displacement inherited from the initial coarse trajectory. Hence, if the initial path is too coarse, then over-refinement only samples the same coarse-reference mismatch more densely, rather than removing it.

E.4 Implication for the Local Velocity Correction

Recall the single-point refinement bound derived earlier:

$$\eta_n = \|v(\hat{x}_n, \tau_n) - v_n\| \leq C_1 h_n \delta_n + C_2 \delta_n^2. \tag{81}$$

If δ_n decreases together with h_n, then both terms become small. However, once δ_n saturates at the scale $O(H_0)$, reducing h_n alone can no longer remove the second term:

$$\eta_n \lesssim C_1 h_n H_0 + C_2 H_0^2. \tag{82}$$

As $h_n \to 0$, the first term vanishes, but the second term remains at the coarse-path scale $O(H_0^2)$. Therefore, excessive interpolation cannot indefinitely improve the local correction quality.

Theorem E.1. *Under the above assumptions, the following design principle holds.*

1. *The initial trajectory should not be overly coarse. In particular, to keep the displacement below a prescribed threshold $\bar{\delta}$, it is necessary that*

$$H_0 \lesssim \bar{\delta}. \tag{83}$$

2. *The number of inserted points should remain moderate. Once h is small enough such that*

$$C_L h + \varepsilon_{\mathrm{ref}}(h) \ll H_0, \tag{84}$$

the quantity $\max_n \delta_n$ is already dominated by the $O(H_0)$ coarse-path floor, and further interpolation yields diminishing returns.

Proof. The first claim follows directly from Eq. (72): keeping $\max_n \delta_n$ below a threshold requires controlling the coarse-path contribution $C_0 H_0$.

The second claim follows from Eq. (79). When Eq. (84) holds, the upper and lower bounds are both dominated by the terms proportional to H_0. Hence $\max_n \delta_n$ no longer decreases significantly as h decreases. The corresponding velocity-refinement error floor then follows from Eq. (82). □

F STIR-FM WITH UNIPC AND STORK

This section derives two concrete extensions of STIR-FM beyond the Euler instantiation in Section 3; the experimental protocols and results follow afterwards. The two constructions retain midpoint selection and persistent reference predictions, but replace trajectory propagation and rebuild the history required by the base solver. They specify numerical procedures; the derivations alone do not establish a quality improvement, an inherited convergence order, or equivalence to a previously evaluated implementation. We use the increasing-time convention of the main text, $\dot{x} = f_\theta(x, t)$, with noise at $t = 0$ and data at $t = 1$. If a model instead supplies $u_\theta(x, \sigma) = dx/d\sigma$ with decreasing noise level $\sigma = 1 - t$, then

$$f_\theta(x, t) = -u_\theta(x, 1 - t). \tag{85}$$

The actual integration coordinate must be used throughout; inserting a midpoint between model-conditioning indices is not equivalent when those indices are a nonlinear function of integration time.

For both constructions, let $(z_j^{\mathrm{ori}}, v_j^{\mathrm{ori}}, t_j)$ be an immutable record of an actual network evaluation, $v_j^{\mathrm{ori}} = f_\theta(z_j^{\mathrm{ori}}, t_j)$. The notation z_j^{ori} emphasizes that the evaluation state need not equal the accepted solver state, which is important for UniPC. To evaluate the cached velocity at another state z at the same timestamp, define

$$\mathcal{V}_j(z) = v_j^{\mathrm{ori}} + \widehat{J}_j \odot (z - z_j^{\mathrm{ori}}). \tag{86}$$

Here $\widehat{J}_j$ is the element-wise spatial slope from Eq. (4). More generally, for two other reference nodes ℓ, r, its numerator and denominator are

$$\begin{cases} A_j = (v_\ell^{\mathrm{ori}} - v_j^{\mathrm{ori}})(t_r - t_j) - (v_r^{\mathrm{ori}} - v_j^{\mathrm{ori}})(t_\ell - t_j), \\ D_j = (z_\ell^{\mathrm{ori}} - z_j^{\mathrm{ori}})(t_r - t_j) - (z_r^{\mathrm{ori}} - z_j^{\mathrm{ori}})(t_\ell - t_j), \\ (\widehat{J}_j)_d = \begin{cases} (A_j)_d/(D_j)_d, & |(D_j)_d| > \tau_j, \\ 0, & \text{otherwise}, \end{cases} \qquad \tau_j = \max\{\epsilon_D, \mathrm{Quantile}_\rho(\{|(D_j)_d|\}_d)\}. \end{cases} \tag{87}$$

The coordinate index is d, $\epsilon_D > 0$ is a numerical floor, and ρ is the suppression fraction; we use $\rho = 0.20$, matching the default suppression fraction in Appendix C.1. Use the immediate reference neighbors when available, or the nearest two other evaluated nodes for a one-sided boundary stencil. If fewer than three evaluated nodes exist, use $\widehat{J}_j = 0$. The unevaluated terminal state is never used as a reference velocity. After insertion, rebuild affected stencils for subsequent queries from the enlarged reference set while leaving every stored evaluation pair unchanged. History entries already consumed in the current sweep are retained; they are not changed retroactively when another reference is added.

Equation (87) follows by eliminating the time derivative from two local scalar Taylor equations. For a vector field it defines a diagonal surrogate, not an identification of its full Jacobian from three trajectory samples. Writing $J_j = \nabla_x f_\theta(z_j^{\mathrm{ori}}, t_j)$, smoothness gives

$$\mathcal{V}_j(z) - f_\theta(z, t_j) = \big(\mathrm{diag}(\widehat{J}_j) - J_j\big)(z - z_j^{\mathrm{ori}}) + \mathcal{O}(\|z - z_j^{\mathrm{ori}}\|_2^2). \tag{88}$$

Thus, off-diagonal coupling, poorly conditioned reference differences, and displacement from the reference can limit either extension.

For a current accepted trajectory $\{x_j\}$, set $w_j = \mathcal{V}_j(x_j)$ at nonterminal nodes and rank intervals using

$$e_j = \|w_{j+1} - w_j\|_2 \Delta t_j \quad (j < N-1), \qquad e_{N-1} = \|w_{N-1} - w_{N-2}\|_2 \frac{\Delta t_{N-1}^2}{\Delta t_{N-2}}, \tag{89}$$

where $\Delta t_j = t_{j+1} - t_j$. For a single interval, select that interval without the boundary formula. This deliberately preserves STIR's velocity-variation heuristic; it is not a derived high-order local-error estimator for UniPC or STORK. Within each round, select intervals on the pre-insertion grid, store their endpoint identities, and process them in increasing time. The two subsections below specify how to propagate from the first selected interval to the terminal state without reusing stale solver histories.

F.1 STIR-FM WITH UNIPC

Flow-to-data conversion. We use the data-prediction formulation of UniPC (Zhao et al., 2023). For the linear flow path, define

$$\alpha_t = t, \qquad \sigma_t = 1 - t, \qquad m_\theta(x,t) = x + \sigma_t f_\theta(x,t), \qquad \lambda_t = \log \frac{\alpha_t}{\sigma_t}. \tag{90}$$

Here m_θ denotes a predicted clean endpoint, not another network. Indeed, substituting $f_\theta = (m_\theta - x)/\sigma_t$ gives

$$\frac{d}{dt}\left(\frac{x(t)}{\sigma_t}\right) = \frac{m_\theta(x(t),t)}{\sigma_t^2}, \qquad \frac{d\lambda_t}{dt} = \frac{1}{\alpha_t \sigma_t}. \tag{91}$$

For $0 < t_k < t_+ < 1$, integrating this identity yields

$$\begin{cases} x(t_+) = \dfrac{\sigma_+}{\sigma_k} x(t_k) + \alpha_+ h \displaystyle\int_0^1 e^{h(u-1)} \widehat{m}(\lambda_k + hu)\, du, \\ \widehat{m}(\lambda) = m_\theta(x(t(\lambda)), t(\lambda)), \qquad h = \lambda_+ - \lambda_k. \end{cases} \tag{92}$$

Subscripts $+$ denote the next destination, which may be an inserted midpoint. This identity derives the exponential weights from the flow ODE and fixes their signs under the increasing-time convention. It assumes the linear α, σ path above; a different path requires its own conversion and integrating factor.

Predictor and corrector on the updated grid. Let y_j be the state at which the prediction used by UniPC is formed, and x_j the accepted, corrected state. Store the history values

$$v_j^P = \mathcal{V}_j(y_j), \qquad m_j = y_j + \sigma_j v_j^P, \qquad H_k = \{(t_j, \lambda_j, m_j) : j \le k\}. \tag{93}$$

An actual evaluation replaces $\mathcal{V}_j(y_j)$ at a newly inserted node. In particular, m_j is formed from y_j, not by combining x_j with a velocity evaluated at y_j. The propagation anchor is nevertheless the accepted state x_k, as in a predictor–corrector method.

For order parameter p, use $p-1$ preceding finite-λ history nodes, with

$$r_i = \frac{\lambda_{k-i} - \lambda_k}{h}, \qquad D_i = \frac{m_{k-i} - m_k}{r_i}, \quad i = 1, \dots, p-1, \qquad r_p = 1. \tag{94}$$

Define the exponential moments

$$\begin{cases} M_q(h) = h \displaystyle\int_0^1 e^{h(u-1)} u^q\, du, \\ M_0(h) = 1 - e^{-h}, \qquad M_q(h) = 1 - \dfrac{q}{h} M_{q-1}(h) \quad (q \ge 1). \end{cases} \tag{95}$$

For small h, evaluate these moments by their stable series $M_q(h) = \sum_{n=0}^{\infty} (-1)^n q! h^{n+1}/(q+n+1)!$ rather than by cancellation-prone subtraction. Choose weights satisfying

$$\sum_{i=1}^{p-1} \omega_i^P r_i^{q-1} = M_q(h) \quad (q = 1, \dots, p-1), \qquad \sum_{i=1}^{p} \omega_i^C r_i^{q-1} = M_q(h) \quad (q = 1, \dots, p). \tag{96}$$

These are small Vandermonde systems. To see their origin, expand the data prediction as $m(\lambda_k + hu) = m_k + \sum_{q\geq 1} a_q u^q$. Then $D_i = \sum_{q\geq 1} a_q r_i^{q-1}$, and matching its weighted sum to the integral in Eq. (92) gives Eq. (96). This selects an exact moment-matching member of the UniPC family; implementation variants that simplify low-order coefficients need not give identical finite-step outputs.

With

$$b_{k,+} = \frac{\sigma_+}{\sigma_k} x_k + \alpha_+ M_0(h) m_k, \tag{97}$$

the predictor and corrector become

$$\begin{cases} y_+ = b_{k,+} + \alpha_+ \sum_{i=1}^{p-1} \omega_i^P D_i, \\ m_+^P = y_+ + \sigma_+ v_+^P, \qquad D_p = m_+^P - m_k, \\ x_+ = b_{k,+} + \alpha_+ \sum_{i=1}^{p} \omega_i^C D_i. \end{cases} \tag{98}$$

At an existing node, $v_+^P = \mathcal{V}_+(y_+)$ uses no network call. At a new midpoint, $v_+^P = f_\theta(y_+, t_+)$ costs one NFE; register $(z_+^{\mathrm{ori}}, v_+^{\mathrm{ori}}) = (y_+, v_+^P)$ before applying the corrector. Do not overwrite z_+^{ori} by x_+ afterward. Append (t_+, λ_+, m_+^P) to the history and use x_+ as the next propagation anchor.

For the concrete choice $p = 2$, let $r = (\lambda_{k-1} - \lambda_k)/h$ and $D = (m_{k-1} - m_k)/r$. The full update is

$$\begin{cases} M_1 = 1 - \dfrac{1 - e^{-h}}{h}, \qquad M_2 = 1 - \dfrac{2}{h} + \dfrac{2(1 - e^{-h})}{h^2}, \\ y_+ = b_{k,+} + \alpha_+ M_1 D, \\ x_+ = b_{k,+} + \alpha_+ \left[\dfrac{M_2 - M_1}{r - 1} D + \dfrac{r M_1 - M_2}{r - 1} (m_+^P - m_k) \right]. \end{cases} \tag{99}$$

With only one finite-λ history value, use $p = 1$: $y_+ = b_{k,+}$ and $x_+ = b_{k,+} + \alpha_+ M_1 (m_+^P - m_k)$. After each insertion, recompute h, r_i, the weights, and every affected history value using the new timestamps and predictor states. Keeping coefficients or downstream model-history entries from the old grid would define a different, inconsistent procedure.

Endpoints and initialization. Since $\lambda_0 = -\infty$ and $\lambda_1 = +\infty$, the formulas above apply only to interior times. For a departure from $t = 0$, use $y_+ = x_0 + t_+ v_0$, evaluate or approximate the destination velocity, and accept $x_+ = y_+$ without an exponential corrector. The $t = 0$ record remains available for STIR stencils but is excluded from the finite-λ UniPC history. For the terminal destination $t_+ = 1$, use the first-order denoising limit

$$x_N = m_k, \tag{100}$$

and neither evaluate a terminal velocity nor apply a terminal corrector. This explicitly lowers the endpoint order rather than substituting an infinite h into Eq. (99). An initial UniPC run must retain the actual evaluation states y_j^{ori}, their velocities, accepted states, and timestamps. Saving only accepted states and then claiming that their cached velocities were evaluated there is insufficient.

Complete refinement procedure and cost. Algorithm 5 gives one round; additional rounds reuse the enlarged immutable reference set. The predictor–corrector discrepancy $\|x_+ - y_+\|$ can be logged as a diagnostic, but is not substituted for the selection rule in this construction. Every inserted nonterminal midpoint requires exactly one new velocity evaluation, while both the corrector and the refresh of old nodes use cached information. For an initial run costing a actual NFEs and b insertions, the total is $a + b$, provided the required reference cache was retained. Reconstructing a missing cache, refreshing old nodes with the network, or evaluating a terminal correction would add to this count.

Finally, for a fixed predictor state y, a velocity error η causes a data-prediction error $\sigma_t \eta$ by Eq. (90). The exponential weights propagate these errors in addition to the base integration error. Consequently, the standard UniPC order statements cannot simply be transferred to a refinement sweep that uses approximate cached velocities at old nodes.

Algorithm 5 One STIR-FM refinement round with UniPC

Require: Accepted states x_j, predictor states y_j, history H, immutable evaluation records, insertion budget M, maximum order $p_{\max}$

Select M intervals by Eq. (89); freeze their endpoint identities
Add their midpoint timestamps; mark new nodes and find the first selected left endpoint k_0
Keep the prefix through k_0; reconstruct its finite-λ history
for each destination t_+ after t_{k_0} in increasing time **do**
    if $t_+ = 1$ **then**
        Set $x_N = m_k$ and finish the sweep
    else
        If $t_k > 0$, choose $p \le p_{\max}$ from the finite history and rebuild Eq. (96)
        Predict y_+ by Eq. (98), or use the $t_k = 0$ startup rule
        if t_+ is new **then**
            Evaluate $v_+^P = f_\theta(y_+, t_+)$; register (y_+, v_+^P, t_+); rebuild affected stencils
        else
            Obtain $v_+^P = \mathcal{V}_+(y_+)$ from the persistent references
        end if
        Form m_+^P; correct x_+ by Eq. (98), except at startup
        Replace the destination's working history by (t_+, λ_+, m_+^P); advance the anchor
    end if
end for
return Updated trajectory, predictor states, history, and reference records

F.2 STIR-FM WITH STORK

Velocity history and virtual evaluations. STORK (Tan et al., 2026) replaces internal model evaluations of a stabilized Runge–Kutta construction by Taylor approximations from velocity history. For an accepted node, use $v_k = \mathcal{V}_k(x_k)$, and let $\delta_k = t_k - t_{k-1} > 0$. The first-order, nonuniform-grid construction is

$$\widehat{\dot{v}}_k = \frac{v_k - v_{k-1}}{\delta_k}, \qquad V_k(c;h) = v_k + ch\widehat{\dot{v}}_k, \qquad h = t_+ - t_k. \tag{101}$$

The dot denotes the derivative along the sampled trajectory, $df_\theta(x(t),t)/dt = \partial_t f_\theta + \nabla_x f_\theta f_\theta$, not the partial time derivative at fixed state. The quantity $V_k(c;h)$ is a virtual prediction at stage time $t_k + ch$ and incurs no NFE. Only the first macro-step, where no previous velocity exists, uses Euler propagation.

For completeness, a quadratic history interpolant on unequal intervals gives

$$\begin{cases} a = t_k - t_{k-1}, \qquad b = t_{k-1} - t_{k-2}, \\ \widehat{\dot{v}}_k = \dfrac{2a+b}{a(a+b)} v_k - \dfrac{a+b}{ab} v_{k-1} + \dfrac{a}{b(a+b)} v_{k-2}, \\ \widehat{\ddot{v}}_k = 2\left[\dfrac{v_k}{a(a+b)} - \dfrac{v_{k-1}}{ab} + \dfrac{v_{k-2}}{b(a+b)}\right], \\ V_k(c;h) = v_k + ch\widehat{\dot{v}}_k + \frac{1}{2}(ch)^2\widehat{\ddot{v}}_k. \end{cases} \tag{102}$$

These expressions follow by differentiating the Lagrange interpolant at t_k. More generally, for a degree-q history interpolant $P_k(t) = \sum_{i=0}^{q} v_{k-i} \prod_{\ell \neq i} (t - t_{k-\ell})/(t_{k-i} - t_{k-\ell})$, use $\widehat{v_k^{(r)}} = P_k^{(r)}(t_k)$ and its Taylor polynomial for V_k. Reduce q to the available history size during startup. In the complete procedure below we fix $q = 1$; Eq. (102) specifies an optional higher-history variant rather than a second default. After midpoint insertion, neither equal-step finite differences nor the pre-insertion δ_k are valid in general.

Stabilized stages and the finishing update. We now make the propagation operator $\mathcal{S}_k(h)$ explicit using the ROCK4 recurrence and four-stage finish underlying STORK-4. Let m denote the degree of its recurrence block; the number of virtual stage locations is $m + 4$. This separates the recurrence degree from the total stage count. Use the coefficient row for the chosen degree from the published

Table 5: A concrete ROCK4 recurrence block for the STIR–STORK construction. The first stage uses only μ_1.

j	μ_j	κ_j
1	0.018622507415261	–
2	0.022508615765388	0.006766202022281
3	0.025868058829245	0.031700423460328
4	0.028779530080862	0.063767140238631
5	0.031318677624508	0.098017835241231
6	0.033558640562019	0.132221859441700
7	0.035606882312253	0.165872490553963
8	0.037773716735773	0.201617703951743
9	0.041196987811917	0.253957284203575

STORK coefficient tables. With recurrence coefficients μ_1, $(\mu_j, \kappa_j)_{j=2}^{m}$, define

$$\begin{cases} Y_0 = x_k, \quad c_0 = 0, \qquad Y_1 = x_k + h\mu_1 v_k, \quad c_1 = \mu_1, \\ Y_j = (1+\kappa_j)Y_{j-1} - \kappa_j Y_{j-2} + h\mu_j V_k(c_{j-1}; h), \\ c_j = (1+\kappa_j)c_{j-1} - \kappa_j c_{j-2} + \mu_j, \qquad j = 2, \dots, m. \end{cases} \tag{103}$$

Let $a_{21}, a_{31}, a_{32}, a_{41}, a_{42}, a_{43}$ be the finishing coefficients and $b_1, \dots, b_4$ its output weights. The four virtual velocities and their associated stage states are

$$\begin{cases} Z_1 = Y_m, \qquad Z_2 = Y_m + ha_{21}F_1, \\ Z_3 = Y_m + h(a_{31}F_1 + a_{32}F_2), \\ Z_4 = Y_m + h(a_{41}F_1 + a_{42}F_2 + a_{43}F_3), \\ d_1 = c_m, \qquad d_2 = c_m + a_{21}, \\ d_3 = c_m + a_{31} + a_{32}, \qquad d_4 = c_m + a_{41} + a_{42} + a_{43}, \\ F_i = V_k(d_i; h), \quad i = 1, \dots, 4, \qquad \mathcal{S}_k(h) = Y_m + h\sum_{i=1}^{4} b_i F_i. \end{cases} \tag{104}$$

The Z_i specify the corresponding Runge–Kutta stages; evaluating $f_\theta(Z_i, t_k + d_i h)$ would be a different, more expensive method. Here every F_i is evaluated algebraically from the same Taylor history in Eq. (101).

A concrete coefficient choice is $m = 9$ (13 stage locations in this notation). Table 5 gives the recurrence coefficients, and the finishing coefficients are

$$\begin{cases} (a_{21}, a_{31}) = (-0.148972804305139,\ 0.475296959544537), \\ (a_{32}, a_{41}) = (-0.245899260153289,\ 0.012720301019110), \\ (a_{42}, a_{43}) = (-0.035750883975914,\ 0.481825981739298), \\ (b_1, b_2) = (0.678662206121957,\ -0.263538563990534), \\ (b_3, b_4) = (-0.347971227180309,\ 0.621040683222628). \end{cases} \tag{105}$$

These are the degree-9 row of the released `recf`, `fpa`, and `fpb` tables; full-precision table values should be used in an implementation. This coefficient choice specifies a reproducible instance of the numerical construction; its correspondence to the STORK configuration evaluated in Appendix K.1 has not been established.

An explicit simplification and its interpretation. For the first-order virtual field, Eq. (103) has the form $Y_j = x_k + hc_j v_k + h^2 q_j \widehat{v}_k$, where

$$q_0 = q_1 = 0, \qquad q_j = (1+\kappa_j)q_{j-1} - \kappa_j q_{j-2} + \mu_j c_{j-1}. \tag{106}$$

Substitution into Eq. (104) gives

$$\mathcal{S}_k(h) = x_k + h\left(c_m + \sum_i b_i\right) v_k + h^2 \left(q_m + \sum_i b_i d_i\right) \widehat{v}_k. \tag{107}$$

Algorithm 6 One STIR-FM refinement round with first-order Taylor STORK

Require: Working trajectory (x_j, v_j, t_j), immutable evaluation records, insertion budget M
Select intervals by Eq. (89); freeze their endpoint identities
Add midpoint timestamps; mark new nodes and find the first selected left endpoint k_0
Keep the prefix through k_0; initialize history from its working velocities
for each destination t_+ after t_{k_0} in increasing time **do**
    Set $h = t_+ - t_k$; recompute δ_k when $k \geq 1$ (otherwise use Euler)
    Propagate $x_+ = \mathcal{S}_k(h)$ using Eqs. (103)–(104) or Eq. (108)
    if $t_+ = 1$ **then**
        Store the terminal state and finish the sweep
    else if t_+ is new **then**
        Evaluate $v_+ = f_\theta(x_+, t_+)$ and register the immutable pair (x_+, v_+)
        Rebuild affected reference stencils
    else
        Refresh $v_+ = \mathcal{V}_+(x_+)$ using Eq. (87)
    end if
    Replace the old working history entry by (t_+, v_+) and advance the anchor
end for
return Updated trajectory, velocity history, and reference records

The constant and linear quadrature moments of a consistent ROCK4 tableau satisfy $c_m + \sum_i b_i = 1$ and $q_m + \sum_i b_i d_i = 1/2$. Consequently, in exact arithmetic the particular time-only first-order construction above reduces to

$$\boxed{\mathcal{S}_k(h) = x_k + hv_k + \frac{h^2}{2\delta_k}(v_k - v_{k-1})} \qquad (k \geq 1). \tag{108}$$

For $k = 0$, set $\mathcal{S}_0(h) = x_0 + hv_0$. Equation (108) also follows by integrating the linear extrapolant of velocity over $[t_k, t_k + h]$. For $h = \delta_k$ it becomes the two-step Adams–Bashforth formula. This simplification applies specifically to virtual velocities depending only on stage time; it does not identify a full state-dependent ROCK4 solver with Adams–Bashforth. It also shows why the name STORK-4 alone cannot establish fourth-order accuracy or the original ROCK4 stability region for a cached-velocity refinement. Different stage-dependent approximations require a separate analysis.

Midpoint insertion and propagation. For a selected interval, put $H = t_{k+1} - t_k$ and $t_m = t_k + H/2$. Using Eq. (108) yields the concrete midpoint update

$$x_m = x_k + \frac{H}{2}v_k + \frac{H^2}{8\delta_k}(v_k - v_{k-1}), \qquad v_m = f_\theta(x_m, t_m), \tag{109}$$

with the last term omitted at startup. Register (x_m, v_m, t_m) as a new immutable reference. Since the latest history interval is now $H/2$, propagation to the old right endpoint is

$$x_{k+1}^{\text{new}} = x_m + \frac{H}{2}v_m + \frac{H}{4}(v_m - v_k), \qquad v_{k+1}^{\text{new}} = \mathcal{V}_{k+1}(x_{k+1}^{\text{new}}). \tag{110}$$

If this is the terminal endpoint, only the state update is performed. Every later nonterminal node is reached with the updated velocity history, followed by Eq. (86). On reaching another selected interval, repeat midpoint insertion from its already updated left endpoint. The terminal state is propagated once without introducing a terminal network evaluation.

Cost and approximation limits. Algorithm 6 makes one actual model call per inserted midpoint and no model calls at virtual stages or existing nodes. Therefore its total generator cost is $a + b$ NFEs for an a-NFE cached base trajectory and b insertions, under the same cache convention as the UniPC construction. This count is independent of the number of algebraic stages and does not equate their wall-clock cost. For Eq. (108), velocity perturbations η_k, η_{k-1} induce the exact one-step perturbation, at a fixed anchor,

$$\Delta\mathcal{S}_k = \left(h + \frac{h^2}{2\delta_k}\right)\eta_k - \frac{h^2}{2\delta_k}\eta_{k-1}. \tag{111}$$

Very short preceding intervals can therefore amplify cache errors when followed by a long step. Restricting step ratios or reducing the order, for example through Euler fallback when h/δ_k exceeds a

fixed threshold, may mitigate this amplification; these modifications are not evaluated here. The stated construction does not include an order-reduction fallback. Together with Eq. (88), this explains why additional insertions, more virtual stages, or a higher nominal base-solver order do not by themselves remove the error inherited from the initial trajectory.

G Experimental Setup and Evaluation Protocols

All experiments are conducted on two NVIDIA A6000 GPUs with 48GB memory each. We evaluate our method on three representative text-to-image diffusion models: FLUX.1-dev, Stable Diffusion 3, and Stable Diffusion 3.5. For the main fixed-budget comparisons on all three models, we use the default `FlowMatchEulerDiscreteScheduler` in the Diffusers library (von Platen et al., 2022). For these fixed-budget comparisons, the guidance scale (Ho & Salimans, 2021) is set to 3.5. For each configuration in these fixed-budget comparisons, we use the same set of 5,000 prompts[1] to generate 5,000 samples at a resolution of 512×512, ensuring that all comparisons are conducted under an identical prompt set. We evaluate the generated images using two automatic metrics, ImageReward (Xu et al., 2023) and CLIP (Bolya et al., 2026; Cho et al., 2026; Hessel et al., 2021). For the initial few-step trajectory, we consider different step ranges for different models. Specifically, we evaluate FLUX.1-dev with 4–18 initial steps, Stable Diffusion 3 with 5–24 initial steps, and Stable Diffusion 3.5 with 4–18 initial steps. For each initial trajectory in the main fixed-budget comparisons, we insert a total of six interpolation steps. To study the effect of distributing the same interpolation budget across refinement rounds, we consider four interpolation strategies:

- six rounds with one interpolation step per round $(1 + 1 + 1 + 1 + 1 + 1)$;
- three rounds with two interpolation steps per round $(2 + 2 + 2)$;
- two rounds with three interpolation steps per round $(3 + 3)$;
- two rounds with four interpolation steps in the first round and two interpolation steps in the second round $(4 + 2)$.

We separately examine conditional refinement with STORK on **1,000 matched samples** in Section 4.2, rather than the 5,000 samples used in the main Euler comparisons. All STORK means, total NFEs, and acceptance counts are computed over this separate 1,000-sample evaluation; their absolute metric values should not be directly compared with the 5,000-sample Euler tables. We also isolate the interval-selection rule in Section 4.2, with the stabilization sensitivity reported in Appendix K.2. These additional studies use the budgets specified in their respective protocols rather than the uniform six-insertion budget above. Throughout, NFE counts generator velocity evaluations under the configured guidance implementation, not scoring-network calls or elapsed time. An a-NFE initial trajectory followed by b new midpoint evaluations costs $a + b$ generator NFEs in total, or b incremental NFEs if the initial trajectory is already cached. For S samples with insertion counts b_i, the aggregate is $Sa + \sum_{i=1}^{S} b_i$; the fixed-budget study sets $b_i = 6$ for every sample. Decoding, scoring, cache operations, and reference generation are outside these generator counts. The reported means are descriptive: paired per-sample uncertainty estimates and repeated random seeds are needed to assess small differences. Numerically close means alone establish neither statistical equivalence nor non-inferiority, which would require a predeclared tolerance and an appropriate interval estimate.

H Complete Experimental Results

This appendix provides the full experimental results for all evaluated models. A subset of these results is reported in the main experimental section, while the tables here present the complete results under different interpolation strategies and initial trajectory lengths. All tables report ImageReward (ImRe) and CLIP scores.

The first column of each table denotes the interpolation strategy. The strategy 0 represents the original trajectory without additional interpolation or refinement. The strategies $1, 1{+}1, \ldots, 1{+}1{+}1{+}1{+}1{+}1$ represent one to six refinement rounds, where one interpolation step is inserted in each round. Similarly, $2, 2 + 2$, and $2 + 2 + 2$ represent one, two, and three refinement rounds, respectively, with

[1] `https://huggingface.co/datasets/VincentGOURBIN/FluxPrompting`

Table 6: Complete experimental results on `FLUX.1-dev` for initial trajectory steps from 4 to 8. Rows denote interpolation strategies, and columns report the average ImageReward (ImRe) and CLIP scores over the same set of 5,000 prompts.

Inserted NFEs b by round	Initial NFE a									
	4		5		6		7		8	
	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP
0	0.071	25.1	0.482	25.9	0.621	26.1	0.688	26.2	0.739	26.3
1	0.195	25.2	0.530	25.9	0.642	26.1	0.700	26.2	0.746	26.3
1+1	0.249	25.2	0.569	25.9	0.670	26.1	0.722	26.2	0.763	26.3
1+1+1	0.274	25.3	0.571	25.9	0.674	26.1	0.726	26.2	0.768	26.3
1+1+1+1	0.312	25.4	0.597	26.0	0.684	26.2	0.733	26.3	0.773	26.3
1+1+1+1+1	0.356	25.4	0.621	26.0	0.716	26.2	0.759	26.3	0.792	26.4
1+1+1+1+1+1	0.381	25.4	0.638	26.0	0.720	26.2	0.767	26.3	0.799	26.4
2	0.244	25.2	0.557	25.9	0.659	26.1	0.711	26.2	0.755	26.3
2+2	0.327	25.4	0.588	26.0	0.683	26.2	0.732	26.3	0.773	26.3
2+2+2	0.409	25.5	0.649	26.1	0.728	26.2	0.768	26.3	0.800	26.4
3	0.302	25.4	0.567	25.9	0.670	26.1	0.723	26.2	0.766	26.3
3+3	0.419	25.6	0.645	26.1	0.724	26.3	0.764	26.3	0.797	26.4
4	0.495	25.9	0.605	26.0	0.682	26.1	0.730	26.2	0.771	26.3
4+2	0.588	26.0	0.651	26.1	0.729	26.3	0.771	26.3	0.799	26.4

Table 7: Complete experimental results on `FLUX.1-dev` for initial trajectory steps from 9 to 13. Rows denote interpolation strategies, and columns report the average ImageReward (ImRe) and CLIP scores over the same set of 5,000 prompts.

Inserted NFEs b by round	Initial NFE a									
	9		10		11		12		13	
	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP
0	0.784	26.3	0.802	26.3	0.818	26.3	0.831	26.4	0.847	26.4
1	0.789	26.4	0.807	26.4	0.824	26.4	0.838	26.4	0.853	26.4
1+1	0.801	26.4	0.818	26.4	0.832	26.4	0.844	26.4	0.861	26.4
1+1+1	0.807	26.4	0.825	26.4	0.841	26.4	0.855	26.4	0.871	26.4
1+1+1+1	0.810	26.4	0.827	26.4	0.841	26.4	0.855	26.4	0.869	26.4
1+1+1+1+1	0.825	26.4	0.837	26.4	0.848	26.4	0.859	26.4	0.872	26.4
1+1+1+1+1+1	0.830	26.4	0.843	26.4	0.853	26.4	0.863	26.4	0.877	26.4
2	0.796	26.4	0.813	26.4	0.829	26.4	0.841	26.4	0.857	26.4
2+2	0.810	26.4	0.826	26.4	0.842	26.4	0.855	26.4	0.870	26.4
2+2+2	0.830	26.4	0.843	26.4	0.854	26.4	0.864	26.4	0.877	26.4
3	0.804	26.4	0.820	26.4	0.835	26.4	0.846	26.4	0.862	26.4
3+3	0.828	26.4	0.841	26.4	0.853	26.4	0.864	26.4	0.877	26.4
4	0.807	26.4	0.822	26.4	0.837	26.4	0.848	26.4	0.864	26.4
4+2	0.830	26.4	0.842	26.4	0.854	26.4	0.863	26.4	0.876	26.4

two interpolation steps inserted in each round. The strategies 3 and $3 + 3$ represent one and two refinement rounds with three interpolation steps per round. The strategy 4 represents one refinement round with four inserted interpolation steps, while $4 + 2$ represents two refinement rounds, with four interpolation steps inserted in the first round and two in the second round. Thus, expressions joined by plus signs specify how the total interpolation budget is distributed across successive refinement rounds.

Table 8: Complete experimental results on `FLUX.1-dev` for initial trajectory steps from 14 to 18. Rows denote interpolation strategies, and columns report the average ImageReward (ImRe) and CLIP scores over the same set of 5,000 prompts.

Inserted NFEs b by round	Initial NFE a									
	14		15		16		17		18	
	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP
0	0.853	26.4	0.858	26.4	0.860	26.4	0.864	26.4	0.863	26.4
1	0.862	26.4	0.865	26.4	0.867	26.4	0.872	26.4	0.869	26.4
1+1	0.867	26.4	0.870	26.4	0.871	26.4	0.876	26.4	0.873	26.4
1+1+1	0.876	26.4	0.878	26.4	0.878	26.4	0.882	26.4	0.879	26.4
1+1+1+1	0.875	26.4	0.878	26.4	0.878	26.4	0.881	26.4	0.878	26.4
1+1+1+1+1	0.877	26.4	0.879	26.4	0.879	26.4	0.883	26.4	0.880	26.4
1+1+1+1+1+1	0.881	26.4	0.883	26.4	0.882	26.4	0.886	26.4	0.882	26.4
2	0.864	26.4	0.868	26.4	0.870	26.4	0.874	26.4	0.872	26.4
2+2	0.876	26.4	0.878	26.4	0.878	26.4	0.882	26.4	0.879	26.4
2+2+2	0.881	26.4	0.883	26.4	0.882	26.4	0.885	26.4	0.882	26.4
3	0.869	26.4	0.873	26.4	0.872	26.4	0.877	26.4	0.874	26.4
3+3	0.882	26.4	0.884	26.4	0.883	26.4	0.886	26.4	0.883	26.4
4	0.870	26.4	0.874	26.4	0.874	26.4	0.877	26.4	0.875	26.4
4+2	0.879	26.4	0.883	26.4	0.881	26.4	0.885	26.4	0.882	26.4

Table 9: Complete experimental results on `Stable Diffusion 3` for initial trajectory steps from 5 to 9. Rows denote interpolation strategies, and columns report the average ImageReward (ImRe) and CLIP scores over the same set of 5,000 prompts.

Inserted NFEs b by round	Initial NFE a									
	5		6		7		8		9	
	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP
0	-0.509	24.8	-0.139	25.5	0.081	25.8	0.219	26.0	0.307	26.1
1	-0.326	25.1	-0.013	25.7	0.170	25.9	0.285	26.0	0.361	26.2
1+1	-0.267	25.1	0.029	25.6	0.202	25.9	0.310	26.0	0.380	26.2
1+1+1	-0.303	24.9	0.045	25.5	0.230	25.8	0.336	26.0	0.405	26.1
1+1+1+1	-0.202	24.9	0.086	25.5	0.244	25.8	0.349	26.0	0.414	26.1
1+1+1+1+1	-0.151	25.1	0.122	25.6	0.269	25.8	0.362	26.0	0.423	26.1
1+1+1+1+1+1	-0.092	25.1	0.145	25.6	0.283	25.8	0.371	26.0	0.433	26.1
2	-0.398	24.6	0.022	25.5	0.219	25.8	0.329	26.0	0.399	26.1
2+2	-0.158	25.0	0.107	25.5	0.250	25.8	0.355	26.0	0.419	26.1
2+2+2	0.006	25.4	0.197	25.7	0.294	25.8	0.377	26.0	0.437	26.1
3	-0.251	24.7	0.003	25.3	0.228	25.8	0.340	26.0	0.413	26.1
3+3	0.030	25.2	0.204	25.6	0.307	25.8	0.373	26.0	0.436	26.1
4	0.103	25.4	0.159	25.6	0.241	25.8	0.352	26.0	0.422	26.1
4+2	0.120	25.5	0.204	25.6	0.313	25.8	0.391	26.0	0.434	26.1

H.1 FLUX.1-DEV

Tables 6, 7, and 8 report the complete results for `FLUX.1-dev` under different interpolation strategies and initial trajectory lengths.

H.2 STABLE DIFFUSION 3

Tables 9, 10, 11, and 12 report the complete results for `Stable Diffusion 3` under different interpolation strategies and initial trajectory lengths.

Table 10: Complete experimental results on `Stable Diffusion 3` for initial trajectory steps from 10 to 14. Rows denote interpolation strategies, and columns report the average ImageReward (ImRe) and CLIP scores over the same set of 5,000 prompts.

Inserted NFEs b by round	Initial NFE a									
	10		11		12		13		14	
	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP
0	0.381	26.2	0.438	26.3	0.475	26.3	0.502	26.4	0.532	26.4
1	0.423	26.2	0.472	26.3	0.504	26.3	0.526	26.4	0.553	26.4
1+1	0.438	26.2	0.484	26.3	0.514	26.3	0.535	26.3	0.561	26.4
1+1+1	0.457	26.2	0.499	26.3	0.528	26.3	0.548	26.3	0.571	26.4
1+1+1+1	0.464	26.2	0.503	26.3	0.531	26.3	0.549	26.3	0.574	26.4
1+1+1+1+1	0.472	26.2	0.513	26.3	0.539	26.3	0.557	26.3	0.580	26.4
1+1+1+1+1+1	0.476	26.2	0.515	26.3	0.541	26.3	0.559	26.3	0.581	26.4
2	0.451	26.2	0.494	26.3	0.523	26.3	0.542	26.4	0.566	26.4
2+2	0.467	26.2	0.506	26.3	0.534	26.3	0.551	26.3	0.575	26.4
2+2+2	0.480	26.2	0.518	26.3	0.545	26.3	0.561	26.3	0.583	26.4
3	0.465	26.2	0.507	26.3	0.532	26.3	0.551	26.3	0.574	26.4
3+3	0.481	26.2	0.519	26.3	0.544	26.3	0.560	26.3	0.582	26.4
4	0.470	26.2	0.512	26.3	0.539	26.3	0.556	26.3	0.578	26.4
4+2	0.482	26.2	0.520	26.3	0.545	26.3	0.561	26.3	0.584	26.4

Table 11: Complete experimental results on `Stable Diffusion 3` for initial trajectory steps from 15 to 19. Rows denote interpolation strategies, and columns report the average ImageReward (ImRe) and CLIP scores over the same set of 5,000 prompts.

Inserted NFEs b by round	Initial NFE a									
	15		16		17		18		19	
	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP
0	0.549	26.4	0.566	26.4	0.585	26.4	0.597	26.5	0.608	26.5
1	0.568	26.4	0.582	26.4	0.600	26.4	0.611	26.5	0.620	26.5
1+1	0.576	26.4	0.590	26.4	0.607	26.4	0.619	26.5	0.626	26.5
1+1+1	0.585	26.4	0.598	26.4	0.614	26.4	0.624	26.5	0.631	26.5
1+1+1+1	0.586	26.4	0.600	26.4	0.616	26.4	0.627	26.5	0.633	26.5
1+1+1+1+1	0.592	26.4	0.605	26.4	0.621	26.4	0.631	26.4	0.637	26.5
1+1+1+1+1+1	0.592	26.4	0.605	26.4	0.622	26.4	0.631	26.4	0.637	26.5
2	0.581	26.4	0.593	26.4	0.610	26.4	0.621	26.5	0.628	26.5
2+2	0.588	26.4	0.601	26.4	0.617	26.4	0.628	26.5	0.634	26.5
2+2+2	0.594	26.4	0.607	26.4	0.622	26.4	0.633	26.4	0.638	26.5
3	0.587	26.4	0.599	26.4	0.616	26.4	0.626	26.5	0.633	26.5
3+3	0.594	26.4	0.606	26.4	0.621	26.4	0.632	26.5	0.638	26.5
4	0.590	26.4	0.603	26.4	0.618	26.4	0.629	26.5	0.635	26.5
4+2	0.594	26.4	0.607	26.4	0.622	26.4	0.633	26.4	0.638	26.5

H.3 Stable Diffusion 3.5

Tables 13, 14, and 15 report the complete results for `Stable Diffusion 3.5` under different interpolation strategies and initial trajectory lengths.

I Computational Resources

All experiments were conducted on two NVIDIA RTX A6000 GPUs with 48GB memory each. The reported runtime excludes model-loading time. Since two GPUs were used in parallel, the wall-clock

Table 12: Complete experimental results on `Stable Diffusion 3` for initial trajectory steps from 20 to 24. Rows denote interpolation strategies, and columns report the average ImageReward (ImRe) and CLIP scores over the same set of 5,000 prompts.

Inserted NFEs b by round	Initial NFE a									
	20		21		22		23		24	
	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP
0	0.615	26.5	0.626	26.5	0.633	26.5	0.642	26.5	0.644	26.5
1	0.626	26.5	0.636	26.5	0.643	26.5	0.650	26.5	0.652	26.5
1+1	0.632	26.5	0.643	26.5	0.649	26.5	0.656	26.5	0.658	26.5
1+1+1	0.636	26.5	0.646	26.5	0.651	26.5	0.658	26.5	0.659	26.5
1+1+1+1	0.639	26.5	0.648	26.5	0.654	26.5	0.660	26.5	0.662	26.5
1+1+1+1+1	0.641	26.5	0.650	26.5	0.656	26.5	0.662	26.5	0.663	26.5
1+1+1+1+1+1	0.642	26.5	0.651	26.5	0.656	26.5	0.663	26.5	0.664	26.5
2	0.634	26.5	0.643	26.5	0.649	26.5	0.656	26.5	0.658	26.5
2+2	0.639	26.5	0.649	26.5	0.654	26.5	0.661	26.5	0.662	26.5
2+2+2	0.643	26.5	0.652	26.5	0.656	26.5	0.664	26.5	0.664	26.5
3	0.638	26.5	0.647	26.5	0.652	26.5	0.659	26.5	0.661	26.5
3+3	0.642	26.5	0.651	26.5	0.656	26.5	0.663	26.5	0.664	26.5
4	0.640	26.5	0.649	26.5	0.654	26.5	0.661	26.5	0.662	26.5
4+2	0.643	26.5	0.652	26.5	0.656	26.5	0.662	26.5	0.665	26.5

Table 13: Complete experimental results on `Stable Diffusion 3.5` for initial trajectory steps from 4 to 8. Rows denote interpolation strategies, and columns report the average ImageReward (ImRe) and CLIP scores over the same set of 5,000 prompts.

Inserted NFEs b by round	Initial NFE a									
	4		5		6		7		8	
	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP
0	-1.054	22.4	-0.413	24.6	-0.036	25.7	0.213	26.3	0.350	26.6
1	-0.774	23.3	-0.286	24.9	0.044	25.8	0.277	26.4	0.394	26.6
1+1	-0.744	23.4	-0.251	24.9	0.073	25.8	0.299	26.4	0.410	26.6
1+1+1	-0.712	23.3	-0.245	24.8	0.094	25.7	0.323	26.3	0.432	26.5
1+1+1+1	-0.634	23.4	-0.199	24.9	0.114	25.8	0.330	26.3	0.435	26.5
1+1+1+1+1	-0.554	23.7	-0.146	25.0	0.132	25.8	0.344	26.3	0.448	26.6
1+1+1+1+1+1	-0.535	23.7	-0.119	25.1	0.149	25.8	0.354	26.3	0.456	26.5
2	-0.679	23.2	-0.278	24.6	0.086	25.7	0.325	26.3	0.429	26.5
2+2	-0.500	23.9	-0.140	25.0	0.141	25.8	0.346	26.3	0.445	26.6
2+2+2	-0.482	23.9	-0.084	25.1	0.181	25.9	0.367	26.4	0.462	26.5
3	-0.457	24.1	-0.185	24.7	0.086	25.6	0.334	26.3	0.448	26.5
3+3	-0.244	24.6	-0.030	25.2	0.194	25.9	0.382	26.3	0.470	26.5
4	-0.455	24.1	0.139	25.8	0.151	25.8	0.320	26.2	0.442	26.5
4+2	-0.308	24.5	0.136	25.8	0.191	25.8	0.373	26.3	0.472	26.5

runtime is estimated as

$$T_{\text{wall}} = \frac{5000 \times T_{\text{prompt}}}{2}.$$

The total wall-clock runtime is approximately 315.03 hours, or 13.13 days.

J Additional Qualitative Results

This appendix provides additional qualitative comparisons for the proposed STIR-FM refinement strategy. We include examples from FLUX.1-dev, Stable Diffusion 3, and Stable Diffusion 3.5 under

Table 14: Complete experimental results on `Stable Diffusion 3.5` for initial trajectory steps from 9 to 13. Rows denote interpolation strategies, and columns report the average ImageReward (ImRe) and CLIP scores over the same set of 5,000 prompts.

Inserted NFEs b by round	Initial NFE a									
	9		10		11		12		13	
	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP
0	0.419	26.7	0.476	26.8	0.504	26.8	0.540	26.9	0.552	26.9
1	0.458	26.7	0.509	26.8	0.531	26.8	0.561	26.8	0.572	26.9
1+1	0.472	26.7	0.521	26.8	0.542	26.8	0.569	26.8	0.581	26.8
1+1+1	0.494	26.7	0.538	26.8	0.559	26.8	0.583	26.8	0.593	26.8
1+1+1+1	0.497	26.7	0.540	26.8	0.560	26.8	0.583	26.8	0.593	26.8
1+1+1+1+1	0.509	26.7	0.549	26.8	0.566	26.8	0.591	26.8	0.600	26.8
1+1+1+1+1+1	0.516	26.7	0.555	26.8	0.574	26.8	0.595	26.8	0.604	26.8
2	0.492	26.7	0.534	26.8	0.555	26.8	0.580	26.8	0.589	26.9
2+2	0.506	26.7	0.545	26.8	0.564	26.8	0.587	26.8	0.597	26.8
2+2+2	0.524	26.7	0.559	26.8	0.577	26.8	0.598	26.8	0.607	26.8
3	0.507	26.7	0.547	26.8	0.566	26.8	0.588	26.8	0.599	26.8
3+3	0.528	26.7	0.562	26.8	0.581	26.8	0.600	26.8	0.608	26.8
4	0.510	26.7	0.552	26.8	0.573	26.8	0.593	26.8	0.604	26.8
4+2	0.527	26.7	0.561	26.8	0.581	26.8	0.601	26.8	0.609	26.8

Table 15: Complete experimental results on `Stable Diffusion 3.5` for initial trajectory steps from 14 to 18. Rows denote interpolation strategies, and columns report the average ImageReward (ImRe) and CLIP scores over the same set of 5,000 prompts.

Inserted NFEs b by round	Initial NFE a									
	14		15		16		17		18	
	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP	ImRe	CLIP
0	0.574	26.9	0.593	26.9	0.596	26.9	0.602	26.9	0.617	26.9
1	0.590	26.9	0.608	26.9	0.610	26.9	0.613	26.9	0.628	26.9
1+1	0.597	26.9	0.614	26.9	0.617	26.9	0.620	26.9	0.635	26.9
1+1+1	0.608	26.9	0.623	26.9	0.626	26.9	0.627	26.9	0.640	26.9
1+1+1+1	0.608	26.8	0.624	26.9	0.626	26.9	0.628	26.9	0.641	26.9
1+1+1+1+1	0.614	26.8	0.629	26.9	0.631	26.9	0.632	26.9	0.645	26.9
1+1+1+1+1+1	0.617	26.8	0.632	26.9	0.634	26.9	0.635	26.9	0.645	26.9
2	0.605	26.9	0.620	26.9	0.623	26.9	0.625	26.9	0.637	26.9
2+2	0.611	26.9	0.625	26.9	0.628	26.9	0.629	26.9	0.641	26.9
2+2+2	0.620	26.9	0.633	26.9	0.636	26.9	0.636	26.9	0.647	26.9
3	0.611	26.9	0.626	26.9	0.628	26.9	0.631	26.9	0.641	26.9
3+3	0.620	26.9	0.634	26.9	0.635	26.9	0.637	26.9	0.647	26.9
4	0.615	26.9	0.631	26.9	0.632	26.9	0.634	26.9	0.645	26.9
4+2	0.621	26.9	0.635	26.9	0.636	26.9	0.637	26.9	0.649	26.9

different prompts and initial trajectory lengths. For each example, we compare standard sequential sampling with STIR-FM refinement starting from the same lower-step trajectory. Images labeled as $a + k$ are generated by first sampling an a-step trajectory and then inserting k additional interpolation steps using STIR-FM, while images labeled with a single number denote standard sequential sampling with that number of steps.

Figures 5, 6, 7, and 8 show representative qualitative results. These examples illustrate that STIR-FM can improve the visual quality of few-step trajectories and often approaches the appearance of longer sequential sampling. At the same time, they also show that interpolation-based refinement does not



Figure 5: Qualitative comparison on FLUX.1-dev at 512×512 resolution using the prompt "A serene lake at sunset." Images labeled as $6 + k$ are obtained by starting from a 6-step trajectory and inserting k additional interpolation steps, while 7, 8, and 9 steps denote standard sequential sampling with the corresponding number of steps. Although the refined results can approach the visual quality of standard sequential sampling with the same total number of steps, they do not necessarily reproduce identical image details. For example, fine structures such as the cloud patterns in the sky may still differ, since large errors introduced by an overly low-step initial trajectory cannot always be completely removed by interpolation.

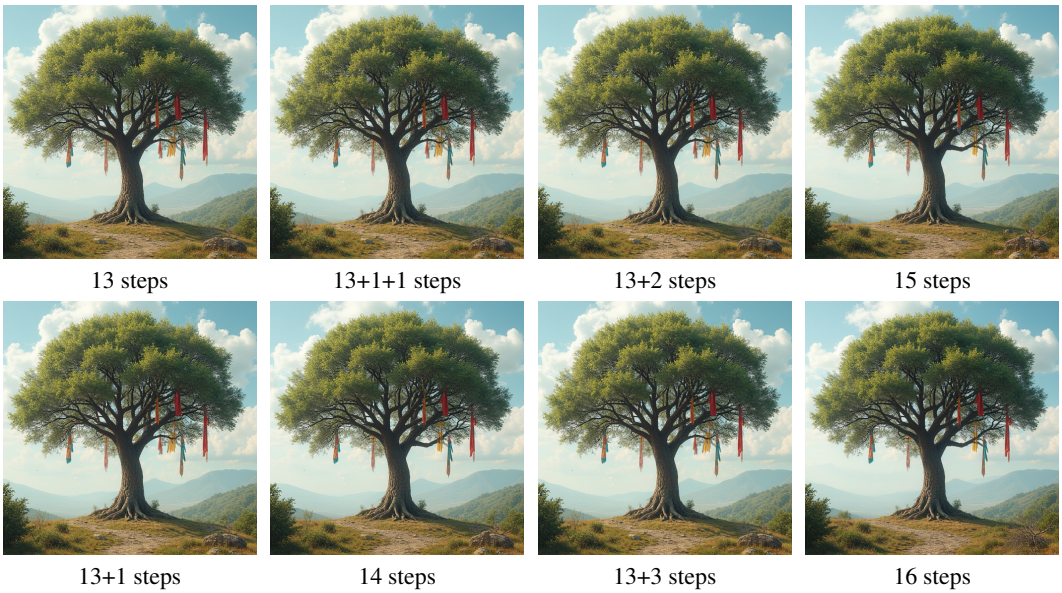


Figure 6: Qualitative comparison on FLUX.1-dev at 512×512 resolution using the prompt "A serene landscape with a large tree at its center, its branches adorned with prayer flags from various religions, including Hinduism, Buddhism, and Islam." Images labeled as $13 + k$ are obtained by starting from a 13-step trajectory and inserting k additional interpolation steps, while 14, 15, and 16 steps denote standard sequential sampling with the corresponding number of steps.

necessarily reproduce exactly the same image details as direct sequential sampling, especially when the initial trajectory is overly coarse.

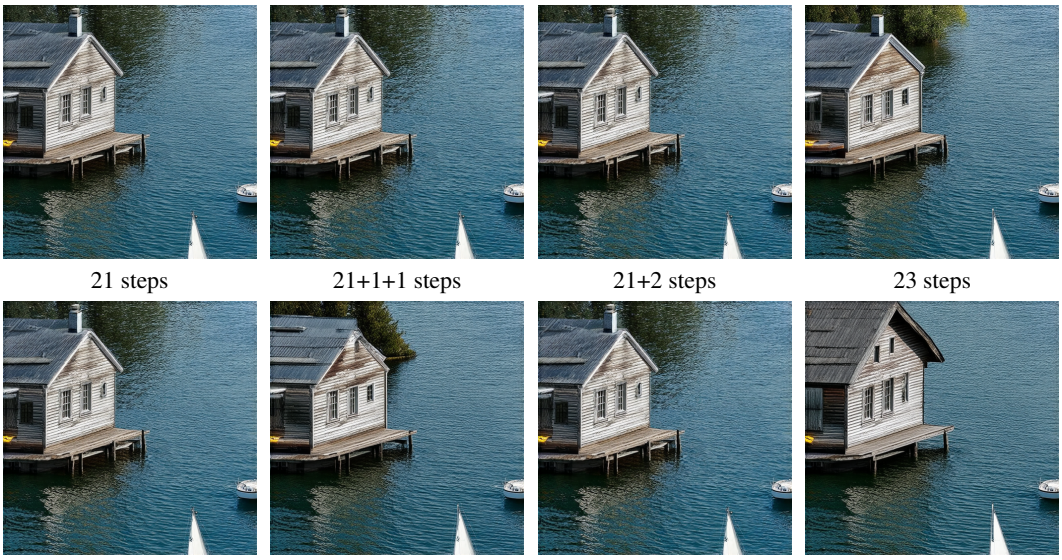


Figure 7: Qualitative comparison on Stable Diffusion 3 using the prompt "A serene Swedish lakeside cabin, its wooden exterior weathered to a soft silver gray, with a tranquil atmosphere and a single sailboat drifting on the calm water." Images labeled as $21 + k$ are obtained by starting from a 21-step trajectory and inserting k additional interpolation steps, while 22, 23, and 24 steps denote standard sequential sampling with the corresponding number of steps.



Figure 8: Qualitative comparison on Stable Diffusion 3.5 using the prompt "A romantic sunset sky with warm golden hues, set against a backdrop of rolling hills and fields of wildflowers swaying gently in the breeze." Images labeled as $15 + k$ are obtained by starting from a 15-step trajectory and inserting k additional interpolation steps, while 16, 17, and 18 steps denote standard sequential sampling with the corresponding number of steps.

K Additional Experimental Details

The following details accompany the parallel analyses in Section 4.2. The conditional STORK experiment and the interval-selection ablation have separate protocols; the ablation is not an additional evaluation of the solver constructions above.

Table 16: Computational resources used in the experiments. Runtime excludes model-loading time.

Model	Peak memory (MiB)	Avg. time / prompt (s)	Wall-clock time (h)
FLUX.1-dev	33620	190.45	132.26
Stable Diffusion 3	18458	97.52	67.72
Stable Diffusion 3.5	28362	165.67	115.05
Total	–	–	315.03

K.1 Conditional STORK Refinement: Protocol and Detailed Analysis

This subsection expands the 1,000-sample experiment summarized in Section 4.2 and Table 3.

Protocol. We examine conditional post-sampling refinement with STORK (Tan et al., 2026) on FLUX.1-dev. In this experiment, STORK supplies the interpolation and propagation operations, while STIR-FM retains velocity-variation interval selection, persistent reference predictions, and the local correction in Eq. (4). Each inserted midpoint requires one new generator evaluation; existing points reuse locally corrected predictions. The comparison uses **1,000 matched samples**, with a fixed STORK-12 output providing a separate reference for each sample. This evaluation is distinct from the 5,000-sample Euler experiments in Tables 1 and 2; all comparisons within Table 3 use the same 1,000 samples. All samples start from STORK-10 and receive at most four additional single-point refinement rounds.

Only CLIP controls acceptance and output selection: after each round, we retain the candidate with the highest CLIP score in that sample's refinement history. A sample stops when this score reaches its matched STORK-12 CLIP score; otherwise it receives another refinement, up to the four-round cap. At the cap, the history-best candidate is returned even if it remains below the reference. ImageReward is excluded from both stopping and history selection and is evaluated on the final selected outputs. This separates the metric used to request additional computation from the metric used to assess the final quality.

Results and interpretation. Table 3 shows that the final selected outputs obtain an average held-out ImageReward of 0.826233, compared with 0.826220 for fixed STORK-12. The difference of 0.000013 indicates numerical proximity, not evidence of a meaningful quality improvement or statistical equivalence. The procedure uses $10{,}888$ generator evaluations rather than $12{,}000$, saving $1{,}112$ evaluations, or 9.27%, with an average cost of 10.888 NFEs per sample. The total follows directly from the active sample counts:

$$\mathrm{NFE}_{\mathrm{total}} = 10 \times 1{,}000 + 530 + 226 + 89 + 43 = 10{,}888. \tag{112}$$

At the final cap, 974 of $1{,}000$ samples reach their matched CLIP reference; the remaining 26 are included in the reported aggregate metrics. Thus, this experiment exercises sample-dependent computation in addition to interval-dependent refinement, unlike a fixed insertion budget applied to every sample.

The stopping rule still requires a precomputed reference for each sample. Accordingly, the reported saving describes a controlled reference-based allocation experiment, not a reference-free stopping mechanism or an end-to-end speedup. Generating the reference outputs, decoding candidates, and evaluating CLIP incur costs outside the generator-NFE totals in Table 3. Holding out ImageReward prevents its direct use in output selection, but does not establish an unbiased estimate of every aspect of perceptual quality. Appendix F develops explicit solver-extension formulas; the experiment here evaluates STORK-based refinement and does not constitute an empirical evaluation of STIR+UniPC.

K.2 Refinement Ablations: Protocols and Detailed Analysis

Interval selection at a fixed refinement budget. To separate the effect of choosing insertion locations from simply spending more evaluations, we compare three strategies starting from the same 15-step FLUX.1-dev trajectories. The STIR-FM strategy selects the interval with the largest velocity-variation score and inserts its midpoint. The random-interval baseline samples an existing interval with equal probability and inserts its midpoint. The uniform-time baseline instead samples

an insertion time uniformly over $[0, 1]$ and propagates to that location within the containing interval. Uniform-time sampling is therefore distinct from deterministic equally spaced insertion. All three strategies use one new evaluation per round and the same subsequent local-correction and propagation operations.

As shown in Table 4, velocity-based selection attains the highest mean after each refinement round. After three insertions, its gain over the initial trajectory is 0.020, compared with 0.011 for random-interval midpoint insertion and 0.007 for uniform-time insertion. At the same total cost of 18 NFEs, the respective gaps are 0.009 and 0.013. These results support using observed velocity variation to place additional evaluations rather than selecting locations without this signal. These comparisons describe mean scores; confidence intervals and repeated random-selection trials are needed to quantify their uncertainty. This ablation isolates the insertion-location rule, while the multi-round comparisons in Section 4.2 examine how a fixed total insertion budget is distributed across rounds.

Sensitivity to numerical stabilization. The division in Eq. (4) can be unstable for small reference-displacement denominators. We apply the per-sample latent-coordinate mask in Eq. (35), suppressing corrections at or below the ρ-quantile of denominator magnitudes and retaining the corresponding reference velocities. In the sensitivity study, changing ρ among 0.10, 0.20, and 0.30 changes the average ImageReward by no more than 0.001; the reported refinement experiments use $\rho = 0.20$. This provides evidence of limited sensitivity over the tested range, rather than a general stability guarantee. It does not replace a separate ablation against removing the velocity correction or recomputing model velocities at displaced states.