DNPMORPH: DIRECT NOISE-PATH OPTIMIZATION FOR SMOOTH AND CONSISTENT DIFFUSION MORPHING

Anonymous authors
Paper under double-blind review

ABSTRACT

Text-to-image diffusion models possess vast knowledge of our visual world, however unlocking this prior for the task of image morphing remains a significant challenge. Here, we introduce *DNPMorph*, a diffusion-based morphing method that utilizes a pretrained diffusion model and, unlike existing methods, directly optimizes the noise latents of a morph path so as to minimize smoothness objectives on the corresponding diffusion-generated frames. To make path-level optimization practical, we use a type of zero-order optimization strategy which does not require differentiating through the diffusion sampling process, essentially treating it as a black box. The path is initialized by spherical linear interpolation between the initial noises corresponding to the user-provided endpoint images, and is then refined in a coarse-to-fine manner. We provide a theoretical conditional descent analysis for our method, and empirically compare it to existing diffusion-based morphing methods on four benchmarks. DNPMorph outperforms competing methods on all datasets in terms of path-smoothness metrics and on three of four in terms of FID, establishing a new state-of-the-art for diffusion-based image morphing.

1 INTRODUCTION

Image morphing aims to generate a sequence that changes smoothly and plausibly between two endpoints. Denoising Diffusion Models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021a;b) provide a strong image prior for this task, but morphing with them remains challenging: interpolating in latent noise space does not necessarily decode to a smooth or semantically coherent trajectory in image space. Existing diffusion-based morphing methods therefore improve smoothness only indirectly, *e.g.*, by interpolating adapted model parameters (Zhang et al., 2024), modifying attention and denoising dynamics (Cao et al., 2025), or constructing prior-aware latent paths (Yu et al., 2025). While effective, these approaches do not directly optimize decoded-image smoothness.

We instead formulate diffusion morphing as *direct optimization of the noise path*. We introduce *DNPMorph* (*Direct Noise-Path Morphing*), which keeps the diffusion model fixed and optimizes the interior latent codes of the morph path to improve a smoothness objective evaluated on the decoded frames. Unlike direct decoded-frame optimization, the frames are not free variables: they remain outputs of the fixed generative model, so the diffusion prior continues to constrain the resulting images. To make this practical, we use a surrogate update that applies a zero-order approximation to the diffusion chain, using stop-gradient through the denoising process while still backpropagating through the decoder and the decoded-path objective. In our experiments, we initialize the path from a noise spherical linear interpolation (SLERP) trajectory (Shoemake, 1985), which provides a prior-aligned reference path in Gaussian noise space.

We also provide a conditional analysis of this surrogate update, identifying sufficient alignment and regularity conditions under which omitting the denoising-chain Jacobian still yields a descent-producing direction for the target objective. To optimize longer morphs more robustly, we further use a two-stage coarse-to-fine schedule that first optimizes a sparse path and then densifies it, reducing mid-path drift and improving consistency with the endpoint-defined transition.

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.

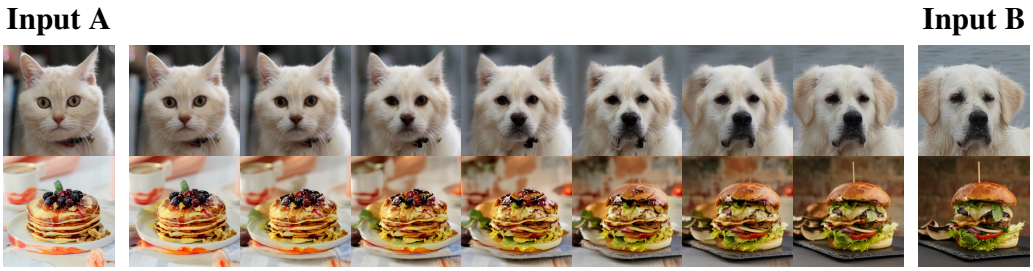


Figure 1: **Qualitative examples from DNPMorph.** We show two representative morph sequences produced by our method, each with seven intermediate frames between the endpoint images.

Empirically, DNPMorph substantially improves smoothness metrics across all datasets and improves FID (Heusel et al., 2017) on three of four datasets, suggesting that these gains usually preserve distributional compatibility with the endpoint-image domain. Moreover, the optimized latents remain close to the diffusion-prior geometry, supporting the view that the method refines a prior-aligned path rather than freely drifting in latent space.

To summarize, our contributions are as follows. (i) We propose DNPMorph, a diffusion-morphing method that directly optimizes the noise path rather than relying on model adaptation or modified sampling dynamics. (ii) We make path-level optimization practical using a zero-order approximation of the diffusion process while optimizing decoded morph-sequence losses. (iii) We identify conditions under which the resulting surrogate update decreases the target objective. (iv) Across four benchmarks, DNPMorph improves path smoothness on all datasets and achieves the best FID on three of four, setting a new state-of-the-art for diffusion-based image morphing.

2 Related work

Diffusion-based image morphing. Generating a seamless transition between two images with diffusion models is challenging, since naive interpolation in latent noise space often produces abrupt semantic changes or unrealistic intermediate frames. Existing diffusion-based morphing methods can be broadly grouped into three categories. The first category adapts or interpolates conditioning and model parameters to induce smoother transitions, for example by interpolating latent or feature representations (Wang & Golland, 2023), optimizing text embeddings and LoRA modules (Yang et al., 2023), or fitting image-specific LoRAs and interpolating them together with latent noises and attention statistics (Zhang et al., 2024). A second line of works avoids per-instance tuning and instead modifies the inference-time attention or denoising process itself, for example by injecting endpoint guidance into self-attention and shaping the denoising dynamics (Cao et al., 2025), or by correcting interpolation through modified noising and denoising steps (Zheng et al., 2024). A third, broader category constructs latent trajectories using explicit priors or diffusion-space geometry, including norm-aware interpolation paths (Samuel et al., 2023), probability-density geodesics (Yu et al., 2025), and diffusion-space geodesics derived from score-based or Riemannian metrics (Saito & Matsubara, 2025a;b). Despite their differences, these approaches improve morphing smoothness only indirectly, by manipulating conditioning, internal dynamics, or latent trajectories rather than directly optimizing a decoded-image smoothness objective. In contrast, our method keeps the diffusion model fixed and formulates morphing as direct smoothness optimization on the decoded path.

Jacobian-free generative optimization. Zero-order and Jacobian-free approximations are common in diffusion-based generative optimization. Score Distillation Sampling (SDS) (Poole et al., 2023) and its variants, including Delta Denoising Score (DDS) (Hertz et al., 2023), Posterior Distillation Sampling (PDS) (Koo et al., 2024), and inverse Rectified Flow Distillation Sampling (iRFDS) (Yang et al., 2025), avoid the diffusion-model Jacobian through timestep-wise score distillation rather than optimization through an entire denoising chain. D-Flow (Ben-Hamu et al., 2024) optimizes through the whole generative process by differentiating through it, but this requires backpropagation through the full chain and is therefore memory- and compute-intensive. FlowOpt (Ronai et al., 2025) optimizes latent variables through a complete denoising or flow process without backpropagating through the model, primarily for inversion and text-based image editing. In contrast to these works, our setting and objective are different: we study image morphing and jointly optimize a set of path latents under a shared sequence-level smoothness objective over adjacent decoded frames,

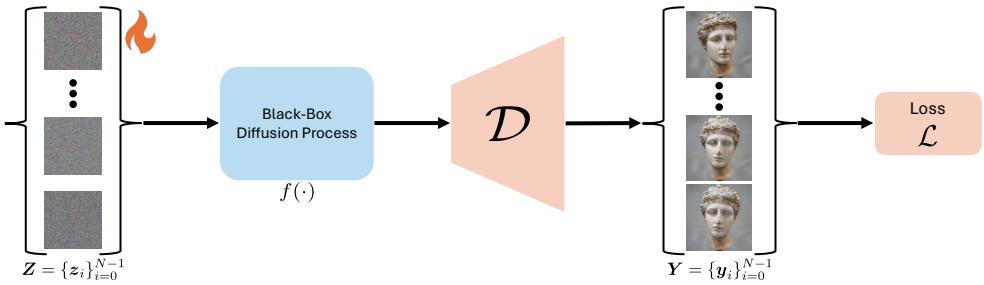


Figure 2: **DNPMorph framework.** A morph is represented by a noise-latent path, passed through a fixed diffusion-process map and a decoder to obtain a sequence of frames in pixel space. DNPMorph optimizes the interior latents under a sequence-level objective on the decoded frames. Gradients flow through the decoder and decoded-path objective, but not through the diffusion process. Orange and blue mark the components that are differentiated and not differentiated, respectively.

rather than optimizing a single output under a standalone image-level loss. The fixed endpoint latents and coarse-to-fine refinement schedule further make the optimization a structured trajectory problem rather than an independent single-image control problem. Finally, our method only partially avoids gradients: the denoising chain is not differentiated through, yet gradients still propagate through the decoder and the downstream image-space smoothness objective.

3 Problem Setup and Diffusion Mapping

Given two endpoint images, our goal is to construct an N-frame morph sequence indexed by $i = 0, \dots, N-1$, where frames 0 and $N-1$ are the endpoints and the intermediate frames form a smooth, perceptually natural, and endpoint-consistent transition.

We work in the setting of a latent diffusion model, where diffusion is performed in a latent space and a decoder maps denoised latent representations to image space. We use ODE-based samplers, which induce a deterministic mapping from the latent noise space to the latent image space. Specifically, let C_ℓ, H_ℓ, W_ℓ denote the latent dimensions, and let $H_{\text{im}}, W_{\text{im}}$ denote the spatial dimensions of the decoded image. We denote by $\boldsymbol{z}_i \in \mathbb{R}^{C_\ell \times H_\ell \times W_\ell}$ the initial-noise latent of frame i, and by $f_T(\cdot)$ the fixed full T-step diffusion-process map in latent space. Applying this map to $\boldsymbol{z}_i$ yields a clean latent

$$\boldsymbol{x}_i = f_T(\boldsymbol{z}_i), \qquad \boldsymbol{x}_i \in \mathbb{R}^{C_\ell \times H_\ell \times W_\ell}. \tag{1}$$

The decoder $\mathcal{D}$ then maps the clean latent to image space, *i.e.*, $\boldsymbol{y}_i = \mathcal{D}(\boldsymbol{x}_i) \in \mathbb{R}^{3 \times H_{\text{im}} \times W_{\text{im}}}$.

We collect the frame-level variables into path-level sequences,

$$\boldsymbol{Z} = (\boldsymbol{z}_0, \dots, \boldsymbol{z}_{N-1}), \qquad \boldsymbol{X} = (\boldsymbol{x}_0, \dots, \boldsymbol{x}_{N-1}), \qquad \boldsymbol{Y} = (\boldsymbol{y}_0, \dots, \boldsymbol{y}_{N-1}). \tag{2}$$

At the path level, the diffusion-process relation becomes

$$\boldsymbol{X} = F(\boldsymbol{Z}) := \big(f_T(\boldsymbol{z}_i)\big)_{i=0}^{N-1}, \tag{3}$$

and the decoded morph path is $\boldsymbol{Y} = \mathcal{D}(\boldsymbol{X})$, where $\mathcal{D}$ is applied frame-wise to the elements of $\boldsymbol{X}$.

The endpoint latents $\boldsymbol{z}_0$ and $\boldsymbol{z}_{N-1}$ are obtained by inverting the endpoint images and remain fixed during optimization. Consequently, the morphing problem reduces to optimizing the interior latents $\{\boldsymbol{z}_i\}_{i=1}^{N-2}$ such that the decoded path $\boldsymbol{Y}$ forms a smooth and consistent transition between them.

4 Method

In this section, we formulate DNPMorph as optimization of a decoded-sequence objective over the interior noise-path latents, introduce its surrogate update, which omits the diffusion-process Jacobian while retaining gradients through the decoder and decoded-path objective, and present the coarse-to-fine path refinement shown in Fig. 3.

4.1 Direct Optimization of Decoded Morph Paths

We formulate diffusion morphing as optimization over the noise path rather than over decoded frames. To our knowledge, this is the first formulation of diffusion morphing that optimizes the

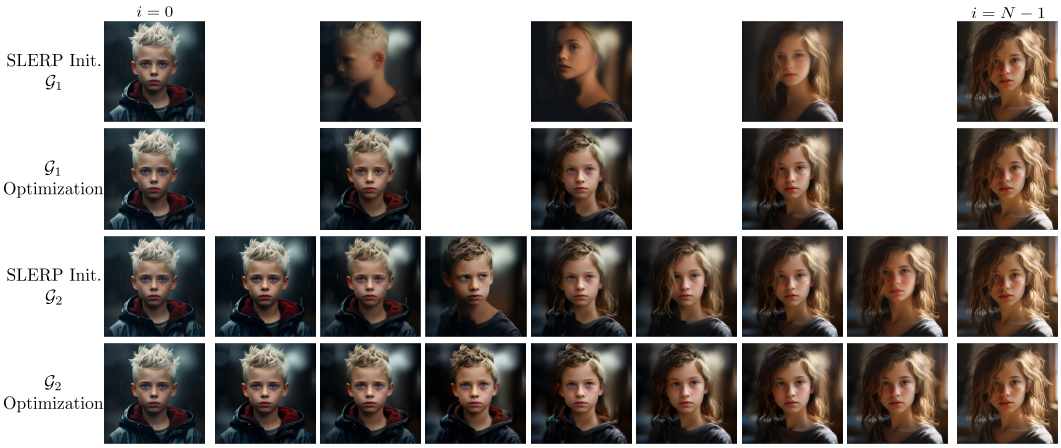


Figure 3: **DNPMorph coarse-to-fine refinement.** Starting from the noise SLERP path, DNPMorph first optimizes the sparse frames $\mathcal{G}_1$ while keeping the endpoints fixed. The path is then densified by local SLERP within the optimized coarse segments, followed by refinement of all non-endpoint frames $\mathcal{G}_2$. This coarse-to-fine schedule reduces mid-path drift and yields a smoother transition that remains more consistent with the endpoints.

interior noise latents of a fixed diffusion model under a decoded sequence-level objective. We introduce a path-quality functional $\mathcal{S}(\boldsymbol{Y})$ and formulate DNPMorph as

$$\min_{\boldsymbol{z}_1,\dots,\boldsymbol{z}_{N-2}} \Psi_\lambda(\boldsymbol{Z}), \qquad \Psi_\lambda(\boldsymbol{Z}) := \mathcal{S}\big(\mathcal{D}(F(\boldsymbol{Z}))\big) + \frac{\lambda}{2}\|\boldsymbol{Z} - \bar{\boldsymbol{Z}}\|_2^2. \tag{4}$$

Here, $\mathcal{S}$ is defined on the decoded sequence, and may combine local smoothness between adjacent frames, semantic consistency along the path, and regularity of the transition speed. The second term is a regularization term that attracts the optimized path toward an anchor path $\bar{\boldsymbol{Z}}$, such as an initialization to which we wish the optimized path to remain close.

Although the objective is defined on the decoded sequence $\boldsymbol{Y}$, the optimization variables are the noise-path latents $\boldsymbol{Z}$. Treating $\boldsymbol{Y}$ directly as the variable turns morphing into an unconstrained image-space fitting problem. In Sec. 5.3, we show that direct decoded-frame optimization can drastically reduce the optimized losses while leaving the visual sequence almost unchanged, suggesting that it adversarially exploits $\mathcal{S}(\boldsymbol{Y})$ without perceptually improving the transition. In contrast, optimizing $\boldsymbol{Z}$ through the diffusion mapping F keeps each decoded frame tied to the diffusion process prior, while still allowing direct optimization of a sequence-level smoothness objective.

The full gradient of Ψ with respect to the path variables is

$$\nabla_{\boldsymbol{Z}} \Psi_\lambda(\boldsymbol{Z}) = J_F(\boldsymbol{Z})^\top J_{\mathcal{D}}(F(\boldsymbol{Z}))^\top \nabla_{\boldsymbol{Y}} \mathcal{S}(\boldsymbol{Y}) + \lambda\left(\boldsymbol{Z} - \bar{\boldsymbol{Z}}\right), \tag{5}$$

where $J_{\mathcal{D}}$ and J_F are the Jacobians of the decoder $\mathcal{D}$ and the diffusion process F, respectively. Computing this gradient is expensive because it requires backpropagation through the whole diffusion process. We therefore use a surrogate update that approximates $J_F(\boldsymbol{Z})^\top$ by the identity map, while retaining gradients through the decoder and the decoded-path objective:

$$q_\lambda(\boldsymbol{Z}) := J_{\mathcal{D}}(F(\boldsymbol{Z}))^\top \nabla_{\boldsymbol{Y}} \mathcal{S}(\boldsymbol{Y}) + \lambda\left(\boldsymbol{Z} - \bar{\boldsymbol{Z}}\right). \tag{6}$$

We set to zero the elements of $q_\lambda(\cdot)$ corresponding to the endpoint images, as we do not wish to update them, and with slight abuse of notation regard this modified direction as $q_\lambda(\cdot)$. In the following, we omit λ from $\Psi_\lambda(\cdot)$ and $q_\lambda(\cdot)$ for notational convenience. We then use $q(\boldsymbol{Z})$ as a surrogate update direction for the noise-path variables $\boldsymbol{Z}$ and update the interior latents by

$$\boldsymbol{Z} \leftarrow \boldsymbol{Z} - \eta q(\boldsymbol{Z}), \tag{7}$$

where $\eta > 0$ is some step size.

This update rule is the zero-order type approximation used by DNPMorph: the denoising-chain map is evaluated in the forward pass but is not backpropagated through, while gradients still propagate

through the decoder and the decoded-path objective. Intuitively, this surrogate can work when the omitted diffusion-process Jacobian does not strongly rotate the update directions encountered during optimization. In that case, moving the noise latents along the surrogate direction changes the denoised latents in a compatible direction, so replacing $J_F(\boldsymbol{Z})^\top$ by the identity can still yield a direction aligned with the true gradient. The following theorem formalizes this intuition: under alignment and regularity assumptions, the surrogate direction $q(\boldsymbol{Z})$ leads to a sufficient decrease of the target objective $\Psi(\boldsymbol{Z})$ (see App. A for the proof).

Theorem 1 (Sufficient decrease of the target objective)**.** *Let $\boldsymbol{Z}$ be as defined in Eq. (2), and let the iterates be generated by $\boldsymbol{Z}^{k+1} = \boldsymbol{Z}^k - \eta q(\boldsymbol{Z}^k)$. Assume that $\Psi(\boldsymbol{Z}) \geq 0$, that Ψ is L_Ψ-smooth,* i.e.*,*

$$\|\nabla\Psi(\boldsymbol{Z}_1) - \nabla\Psi(\boldsymbol{Z}_2)\|_2 \leq L_\Psi \|\boldsymbol{Z}_1 - \boldsymbol{Z}_2\|_2, \qquad \forall \boldsymbol{Z}_1, \boldsymbol{Z}_2, \tag{8}$$

and that there exists a constant $\gamma > 0$ such that

$$\langle \nabla\Psi(\boldsymbol{Z}),\, q(\boldsymbol{Z})\rangle \geq \gamma \|q(\boldsymbol{Z})\|_2^2, \qquad \forall \boldsymbol{Z}. \tag{9}$$

If the step size satisfies $0 < \eta < \frac{2\gamma}{L_\Psi}$, then $c_\eta := \gamma\eta - \frac{L_\Psi}{2}\eta^2 > 0$, and for every $k \geq 0$,

$$\Psi(\boldsymbol{Z}^{k+1}) \leq \Psi(\boldsymbol{Z}^k) - c_\eta \|q(\boldsymbol{Z}^k)\|_2^2 \leq \Psi(\boldsymbol{Z}^k). \tag{10}$$

Theorem 1 identifies the key condition under which the zero-order surrogate is descent-producing: alignment between $q(\boldsymbol{Z})$ and the full gradient $\nabla\Psi(\boldsymbol{Z})$, as in Eq. (9). Under the theorem assumptions, the sequence of losses $\{\Psi(\boldsymbol{Z}^k)\}_{k\geq 0}$ is monotonically non-increasing. Since Ψ is bounded below, the surrogate updates vanish asymptotically, *i.e.*, $q(\boldsymbol{Z}^k) \to 0$. If, in addition, there exists $\delta > 0$ such that

$$\|\nabla\Psi(\boldsymbol{Z})\|_2 \leq \delta \|q(\boldsymbol{Z})\|_2, \tag{11}$$

then also $\nabla\Psi(\boldsymbol{Z}^k) \to 0$.

In App. B, we empirically probe the smoothness condition Eq. (8), the alignment condition Eq. (9), and the gradient-comparison condition Eq. (11) in the regime where exact full gradients are computationally feasible. Concretely, we sample many noise-path pairs $\boldsymbol{Z}_1, \boldsymbol{Z}_2$ to evaluate the ratio in Eq. (8), and many noise paths $\boldsymbol{Z}$ to estimate the ratios in Eq. (9) and Eq. (11). Empirically, the ratio in Eq. (8) remains finite and its highest value occurs when $\|\boldsymbol{Z}_1 - \boldsymbol{Z}_2\|_2$ is small, the ratio in Eq. (9) admits a positive lower bound, and the ratio in Eq. (11) admits a finite upper bound. We note that these validations were obtained for a particular choice of the loss term, specifically for the objective we optimize in Sec. 5.

The formulation also supports viewing DNPMorph as a controlled refinement of a reference path. The regularization term in Eq. (4) provides an explicit way to bias the solution toward a reference path when $\lambda > 0$. In App. A, we further present Prop. S1, which assumes a local error bound and characterizes when a limiting stationary point remains locally close to the chosen reference path. Thus, under the stated assumptions, the surrogate update decreases the target objective while the optimized path remains tied to the initial reference path.

4.2 Objective instantiation and coarse-to-fine refinement

The conceptual framework above does not depend on a particular initialization, path-quality functional $\mathcal{S}$, or optimization schedule. In our experiments, we instantiate $\mathcal{S}$ using commonly used morphing smoothness criteria on adjacent decoded frames: LPIPS-based (Zhang et al., 2018) perceptual path length (LPIPS-PPL), LPIPS-based path-distance variance (PDV), and a DINO-based (Caron et al., 2021) semantic path-length term (DINO-PPL). We initialize the path by SLERP in the model's native Gaussian noise space and update the noise latents using Eq. (7) inside the two-stage coarse-to-fine schedule described below. Thus, the practical instantiation refines a prior-aligned reference path using decoded-sequence losses. The formulas for the instantiated objective are provided in App. C.1.

Optimizing the path in a single-stage may induce mid-path drift (we refer to the first and last rows of Fig. S2). To mitigate this, we propose a two-stage coarse-to-fine optimization framework, as detailed in Alg. S1 (App. E). In our 17-frame experiments, frames 0 and 16 are fixed endpoints. Let $\mathcal{G}$ denote the set of frame indices optimized at a given stage. In the sparse stage, we optimize the coarse interior frames $\mathcal{G}_1 = \{4, 8, 12\}$ and evaluate the objective on consecutive frames of the sorted set $\mathcal{G}_1$ along with the endpoints. We then initialize the remaining frames by local SLERP within the

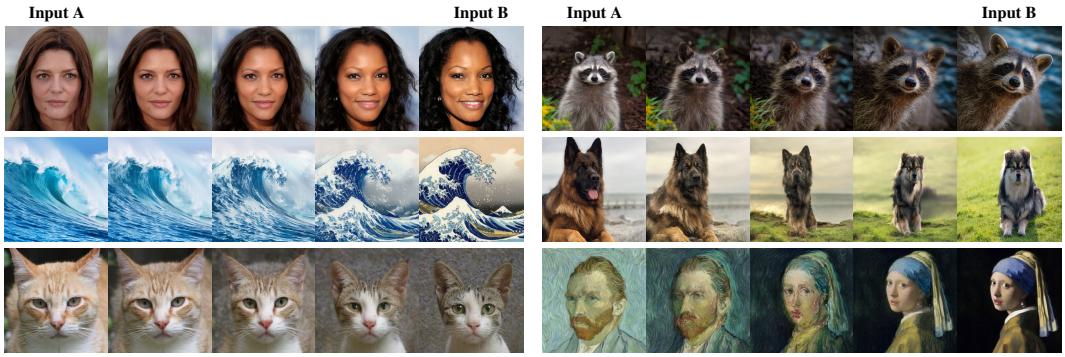


Figure 4: **Qualitative examples produced by DNPMorph.** We show six representative morphs; for each pair, we display the two endpoints and three intermediate frames.

optimized coarse segments. In the dense stage, we refine all non-endpoint frames $\mathcal{G}_2 = \{1, \ldots, 15\}$ and evaluate the objective on the full set of adjacent-pairs. As illustrated in Fig. 3, this schedule helps preserve endpoint consistency before producing the dense morph. A step-by-step visualization of the intermediate frames along the optimization trajectory is provided in App. C.9, Fig. S6.

5 Experiments

In this section, we evaluate DNPMorph empirically and show that it improves morph quality over prior diffusion-morphing methods.

5.1 Experimental setup

We evaluate DNPMorph on the MorphBench (MB) (Zhang et al., 2024) and Morph4Data (M4D) (Cao et al., 2025) datasets, which contain 90 and 76 pairs, respectively, as well as on 50-pair subsets of AFHQ (AF) (Choi et al., 2020) and CAHQ (CA) (Karras et al., 2018), which we constructed as described in App. C.3.

We compare against the following baselines: plain SLERP, DiffMorpher (Zhang et al., 2024), IMPUS (Yang et al., 2023), FreeMorph (Cao et al., 2025), and the diffusion-latent geodesic method of Yu et al. (2025), which we denote as *GeoMorph*. We rerun all compared methods under the same Stable Diffusion 2.1 (SD 2.1) backbone and with the same dataset-specific prompts. For DNPMorph, intermediate frame conditioning is obtained by linearly interpolating between the endpoint text embeddings.

We follow the standard 17-frame evaluation protocol used in prior diffusion-morphing works. All methods generate sequences of 17 frames (15 intermediate frames). All experiments are run on an NVIDIA A100 GPU. The sparse stage is optimized for 100 steps. We report two dense-stage optimization budgets, DNPMorph-50 and DNPMorph-150, which use 50 and 150 dense-stage optimization steps, respectively. Since DNPMorph is a finite-budget refinement initialized from the SLERP reference path, we use the unregularized objective, $\lambda = 0$ in Eq. (4); positive anchor-regularization weights are more conservative and thus require more iterations to reach comparable losses, as discussed in App. C.4. Unless explicitly stated otherwise, we use DNPMorph to refer to the main DNPMorph-150 setting. Additional implementation details are provided in App. C.2.

5.2 Qualitative results

Figures 1 and 4 show representative morph sequences produced by DNPMorph across multiple pairs, while Fig. 5 compares all methods on a representative image pair. Additional qualitative results are provided in App. D. Qualitatively, DNPMorph produces smoother transitions than competing methods, with fewer abrupt semantic changes, while usually preserving coherent intermediate frames that remain visually consistent with the endpoints.

5.3 Quantitative results

Metrics. We report five quantitative metrics. As a distributional-fidelity proxy, FID is computed at the dataset level between the distribution of endpoint images and the distribution of generated

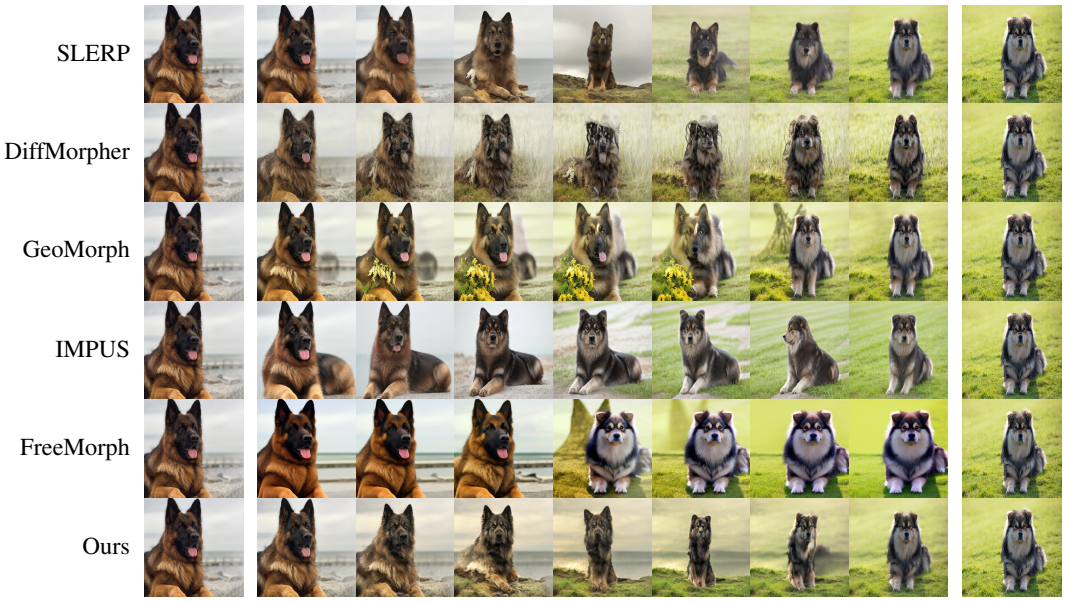


Figure 5: **Qualitative comparison.** Compared with prior methods, our approach yields smoother transitions while maintaining intermediate frames that remain more consistent with the endpoints.

intermediate images, following the benchmark protocol of Zhang et al. (2024): we randomly sample two intermediate frames per pair over 10 trials and report the mean FID. LPIPS-PPL is the sum of adjacent LPIPS distances over the full 17-frame sequence. DINO-PPL is defined analogously, using DINOv2 patch-token cosine distance. DreamSim-PPL is defined analogously using DreamSim (Fu et al., 2023). Following Yu et al. (2025), we report PDV as the standard deviation of adjacent LPIPS distances in the sequence. Lower is better for all metrics.

Quantitative comparison. Table 1 reports a quantitative comparison on all four datasets. Across all datasets, DNPMorph achieves the best LPIPS-PPL, DINO-PPL, and DreamSim-PPL, and obtains the best or second-best PDV on most datasets. On three datasets, the DNPMorph variants also achieve the best FID. On Morph4Data, DNPMorph does not obtain the best FID, but it substantially improves the smoothness metrics, suggesting a dataset-dependent tradeoff between direct path smoothing and distributional fidelity. Overall, the quantitative results are consistent with the qualitative examples: DNPMorph substantially improves path-smoothness metrics, with FID improvements on three of the four datasets. Statistical uncertainty for the per-pair smoothness metrics is analyzed in App. C.7, where we report paired bootstrap confidence intervals over endpoint pairs for both DNPMorph-50 and DNPMorph-150 against each competitor.

User study. To further validate our method, we conducted a pairwise user study in which each participant was shown two image-morphing results – ours and a competing method – in random order. We compared our method to GeoMorph and DiffMorpher, which achieve the most comparable results to DNPMorph. Participation in the user study was voluntary. Each questionnaire contained 20 comparisons, where from each dataset 5 comparisons were randomly selected. Users were asked a two-alternative forced-choice question to select their preferred morphing result, w.r.t. the following criteria: (i) smoothness of the transitions, (ii) consistency to the endpoints, and (iii) visual plausibility of the intermediate frames, *i.e.*, naturalness. We collected 70 user responses, covering a sample size of 700 questions for each method. The results are reported in Fig. 6, where the error bars correspond to 95% Wilson's confidence intervals (Wilson, 1927). These results support the quantitative results in Tab. 1. Additional details and the results are provided in App. C.8.

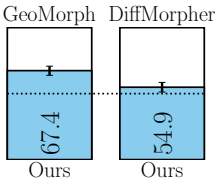


Figure 6: **Human perceptual study.** The bar plots report the percentage of users who preferred our method over competing methods overall w.r.t. (i) transition smoothness, (ii) endpoint consistency, and (iii) naturalness. Error bars show Wilson's 95% confidence intervals.

Table 1: **Quantitative comparison across the four evaluation datasets.** Lower is better for all shown metrics. MB: MorphBench, M4D: Morph4Data, AF: AFHQ subset, CA: CAHQ subset. DM: DiffMorpher, FM: FreeMorph, GM: GeoMorph, DNP-50: DNPMorph-50, DNP-150: DNPMorph-150. Bold indicates the best value and underline indicates the second-best value.

Dataset	Metric	SLERP	DM	IMPUS	FM	GM	DNP-50	DNP-150
MB	FID↓	64.1	58.2	60.1	64.6	56.2	**52.2**	52.3
	LPIPS-PPL↓	1.33	0.98	1.75	1.24	0.80	0.67	**0.53**
	PDV↓	0.056	0.015	0.037	0.082	0.023	0.018	**0.014**
	DINO-PPL↓	1.67	1.12	1.64	1.55	1.08	0.66	**0.50**
	DreamSim-PPL↓	0.59	0.33	0.48	0.57	0.30	0.29	**0.25**
M4D	FID↓	112.8	119.8	104.3	**93.1**	93.9	103.6	103.4
	LPIPS-PPL↓	2.37	1.80	3.22	1.71	1.45	1.33	**1.05**
	PDV↓	0.087	**0.025**	0.041	0.111	0.037	0.039	0.036
	DINO-PPL↓	3.24	2.62	3.40	2.21	2.12	1.48	**1.12**
	DreamSim-PPL↓	1.50	1.00	1.34	0.96	0.70	0.77	**0.65**
AF	FID↓	44.1	42.8	44.3	46.5	43.7	**40.4**	41.0
	LPIPS-PPL↓	1.45	1.10	1.80	1.48	1.00	0.62	**0.51**
	PDV↓	0.053	**0.016**	0.034	0.099	0.022	0.024	0.024
	DINO-PPL↓	1.39	0.95	1.37	1.38	0.95	0.48	**0.37**
	DreamSim-PPL↓	0.39	0.19	0.29	0.51	0.23	0.17	**0.15**
CA	FID↓	56.1	52.5	52.7	68.2	102.6	50.0	**49.9**
	LPIPS-PPL↓	0.96	0.74	1.17	1.24	1.10	0.50	**0.40**
	PDV↓	0.038	**0.013**	0.032	0.078	0.024	0.020	0.020
	DINO-PPL↓	1.03	0.80	1.09	1.35	1.55	0.42	**0.32**
	DreamSim-PPL↓	0.38	0.22	0.34	0.63	0.73	0.21	**0.18**

Metric transfer and direct decoded-frame optimization. DNPMorph directly optimizes LPIPS-PPL, DINO-PPL, and an LPIPS-based PDV term, and we further assess whether these gains transfer beyond the optimized feature losses in two ways. First, Tab. 1 shows that DNPMorph also improves DreamSim-PPL, which is not used in the objective; App. C.6 provides a more detailed DreamSim-based analysis, including the optimization trajectory and DreamSim-PDV. Second, we compare against direct decoded-frame optimization: optimizing $\boldsymbol{Y}$ directly with the same LPIPS/DINO/PDV objective drives the optimized losses even lower than DNPMorph, but changes the visual sequence only weakly, transfers poorly to DreamSim-PPL, and substantially worsens FID. This suggests adversarial-like exploitation of the optimized feature losses rather than a meaningful improvement in morph quality. Full results and a qualitative example are provided in App. C.5. Together with the FID results, qualitative comparisons, and user study, these analyses indicate that DNPMorph improves both the optimized objective and complementary measures of morph quality.

Ablations. In App. C.4, we report ablations on the coarse-to-fine schedule, the contribution of individual loss terms, and the effect of the anchor-regularization weight.

Runtime. DNPMorph directly optimizes a decoded-sequence objective at test time, whereas prior methods typically improve smoothness indirectly through conditioning/model adaptation, inference-time dynamics, or prior-guided latent trajectories. This direct optimization enables the improved smoothness reported in Tab. 1, but also increases runtime because each iteration invokes the full diffusion-process map. To make this quality-compute tradeoff explicit, we evaluate DNPMorph along the dense-stage optimization trajectory and compare it with competing methods at their measured runtimes (App. F, Fig. S14). The results show that prior methods are faster, while DNPMorph trades additional test-time computation for stronger final smoothness.

5.4 Preservation of the diffusion-prior geometry

As discussed in Sec. 4, DNPMorph can be viewed as a controlled refinement of a prior-aligned reference path. We therefore examine whether the optimized noise latents preserve the diffusion model's native prior geometry: they should remain close to the Gaussian noise distribution and to the geometry of the initial SLERP paths, rather than freely drifting in latent space.

We first validate prior preservation by testing Gaussianity of the optimized noise latents. On the AFHQ subset, we follow the random-projection protocol of Tasker et al. (2026). For each path type, we randomly select one interior noise latent from each of the 50 AFHQ pairs, yielding 50 high-dimensional latent samples. For each of 5,000 random one-dimensional directions, we project

Table 2: **Gaussianity via random 1D projections.** AD is the mean Anderson–Darling statistic; DP and SW are mean p-values. Percent columns are projection pass rates under AD<0.752 and DP/SW $p > 0.05$. SD 2.1 is reported in Tasker et al. (2026); the remaining rows are computed on AFHQ.

Path	AD↓	AD%↑	DP↑	DP%↑	SW↑	SW%↑
SD 2.1	0.381	95.80	0.496	94.76	0.497	94.96
SLERP	0.378	95.38	0.509	94.16	0.499	94.95
DNPMorph	0.379	95.31	0.507	94.22	0.497	94.90

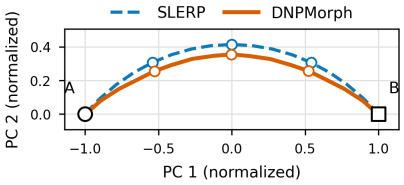


Figure 7: **Mean latent-path geometry.** Projected mean paths in the pair-specific SLERP PCA plane over 50 AFHQ pairs.

these samples to 50 scalar values and run Anderson–Darling (AD) (Anderson & Darling, 1954), D'Agostino–Pearson (DP) (D'agostino & Pearson, 1973), and Shapiro–Wilk (SW) (Shapiro & Wilk, 1965) normality tests on the resulting 1D sample. Table 2 reports the mean AD statistic, the mean DP/SW p-values across projections, and the projection pass rates using AD<0.752, the 5% AD critical threshold used by Tasker et al. (2026), and DP/SW with $p > 0.05$. Under the null hypothesis of Gaussian projected samples, calibrated DP/SW p-values are approximately uniform on $[0, 1]$, so their mean is expected to be near 0.5 and the pass rate for $p > 0.05$ near 95%. The first row reports the SD 2.1 reference values from Tasker et al. (2026); the remaining rows are computed on our AFHQ latent paths. The SLERP and optimized latents obtain nearly identical Gaussianity statistics, indicating that the optimization largely preserves the Gaussian noise geometry.

We also test whether the optimized paths preserve the geometry of the initial noise-SLERP arcs. For each AFHQ pair, using the same 17-frame paths as in our evaluation protocol, we fit a two-dimensional PCA basis to the original SLERP noise-latent path and project the optimized path onto the same two directions. Averaged over the 50 AFHQ pairs, these two SLERP directions explain $75.2\% \pm 2.0\%$ of the optimized-path variance, reported as mean $\pm$ standard deviation across pairs. Fig. 7 visualizes the corresponding mean projection: for visualization only, each pair is centered at the SLERP midpoint and rescaled by the projected endpoint distance before averaging. The optimized trajectory follows the same arc-shaped geometry as the original SLERP path, supporting the view that the optimization behaves as a refinement of the prior path rather than a free drift in latent space.

5.5 Limitations

As with existing diffusion-morphing methods, endpoint pairs with large semantic, structural, or viewpoint changes can remain challenging, since local smoothness alone may not resolve all correspondence ambiguities. In particular, the adjacent-frame smoothness objective does not explicitly enforce long-range correspondence or physically plausible motion. DNPMorph improves path quality by directly optimizing the decoded sequence, which naturally requires more computation than methods that construct or guide the path without iterative sequence-level optimization. This gives a controllable quality-compute tradeoff: the runtime analysis in App. F compares DNPMorph optimization trajectory with the competing methods at their measured runtimes, showing that additional iterations improve smoothness but increase runtime. Appendix G provides additional failure cases.

6 Conclusion

We presented DNPMorph, a diffusion-morphing method that directly optimizes noise-path latents for decoded-sequence smoothness. DNPMorph optimizes a sequence-level objective through a fixed diffusion model, keeping the morph tied to the generative prior rather than treating decoded frames as free variables. A surrogate update makes this practical by omitting backpropagation through the denoising-chain map while retaining gradients through the decoder and decoded-path objective. Starting from a noise SLERP path, DNPMorph refines the morph with a two-stage coarse-to-fine schedule. Empirically, it produces smoother and more consistent transitions than competing methods, while the optimized latents remain close to the diffusion-prior geometry. In terms of broader impact, DNPMorph offers a framework for advancing generative modeling, yet these advancements require continued vigilance toward the ethical risks inherent in media synthesis and manipulation.

AI USE STATEMENT

In this work, we used generative AI tools only for minor text polishing to improve readability, to modify software code, and to validate theoretical proofs. We have not used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods from scratch, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, and interpret results. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Our results can be fully reproduced by implementing the method as described in Sec. 4, and summarized in App. E, with the hyperparameters as provided in App. C.2. Code will be released upon acceptance.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Theodore W Anderson and Donald A Darling. A test of goodness of fit. *Journal of the American statistical association*, 49(268):765–769, 1954.

Heli Ben-Hamu, Omri Puny, Itai Gat, Brian Karrer, Uriel Singer, and Yaron Lipman. D-flow: Differentiating through flows for controlled generation. In *International Conference on Machine Learning*, pp. 3462–3483. PMLR, 2024.

Yukang Cao, Chenyang Si, Jinghao Wang, and Ziwei Liu. Freemorph: Tuning-free generalized image morphing with diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18111–18120, 2025. Morph4Data dataset is released under CC BY-NC 4.0.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8188–8197, 2020. AFHQ dataset is released under CC BY-NC 4.0.

RALPH D'agostino and Egon S Pearson. Tests for departure from normality. empirical results for the distributions of b_2 and $\sqrt{b_1}$. *Biometrika*, 60(3):613–622, 1973.

Thomas J DiCiccio and Bradley Efron. Bootstrap confidence intervals. *Statistical science*, 11(3): 189–228, 1996.

Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. Dreamsim: Learning new dimensions of human visual similarity using synthetic data. *arXiv preprint arXiv:2306.09344*, 2023.

Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit Haim Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. In *The Eleventh International Conference on Learning Representations*, 2023.

Yvette Graham, Nitika Mathur, and Timothy Baldwin. Randomized significance tests in machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pp. 266–274, 2014.

Amir Hertz, Kfir Aberman, and Daniel Cohen-Or. Delta denoising score. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2328–2337, 2023.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

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

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. CelebA dataset is released under a custom non-commercial research license.

Philipp Koehn. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pp. 388–395, 2004.

Juil Koo, Chanho Park, and Minhyuk Sung. Posterior distillation sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13352–13361, 2024.

Zehong Ma, Ruihan Xu, and Shiliang Zhang. Pixelgen: Pixel diffusion beats latent diffusion with perceptual loss. *arXiv preprint arXiv:2602.02493*, 2026.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *The Eleventh International Conference on Learning Representations*, 2023.

Or Ronai, Vladimir Kulikov, and Tomer Michaeli. Flowopt: Fast optimization through whole flow processes for training-free editing. *arXiv preprint arXiv:2510.22010*, 2025.

Shinnosuke Saito and Takashi Matsubara. Image interpolation with score-based riemannian metrics of diffusion models. *arXiv preprint arXiv:2504.20288*, 2025a.

Shinnosuke Saito and Takashi Matsubara. Be tangential to manifold: Discovering riemannian metric for diffusion models. *arXiv preprint arXiv:2510.05509*, 2025b.

Dvir Samuel, Rami Ben-Ari, Nir Darshan, Haggai Maron, and Gal Chechik. Norm-guided latent space exploration for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:57863–57875, 2023.

Samuel Sanford Shapiro and Martin B Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3-4):591–611, 1965.

Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pp. 245–254, 1985.

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.

Chen Tasker, Roy Betser, Eyal Gofer, Meir Yossef Levi, and Guy Gilboa. The universal normal embedding. *arXiv preprint arXiv:2603.21786*, 2026.

Clinton J Wang and Polina Golland. Interpolating between images with diffusion models. *arXiv preprint arXiv:2307.12560*, 2023.

Edwin B Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927.

Xiaofeng Yang, Chen Cheng, Xulei Yang, Fayao Liu, and Guosheng Lin. Text-to-image rectified flow as plug-and-play priors. In *The Thirteenth International Conference on Learning Representations*, 2025.

Zhaoyuan Yang, Zhengyang Yu, Zhiwei Xu, Jaskirat Singh, Jing Zhang, Dylan Campbell, Peter Tu, and Richard Hartley. Impus: Image morphing with perceptually-uniform sampling using diffusion models. *arXiv preprint arXiv:2311.06792*, 2023.

Qingtao Yu, Jaskirat Singh, Zhaoyuan Yang, Peter Henry Tu, Jing Zhang, Hongdong Li, Richard Hartley, and Dylan Campbell. Probability density geodesics in image diffusion latent space. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 27989–27998, 2025.

Kaiwen Zhang, Yifan Zhou, Xudong Xu, Bo Dai, and Xingang Pan. Diffmorpher: Unleashing the capability of diffusion models for image morphing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7912–7921, 2024. MorphBench dataset is released under a custom non-commercial release license.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

PengFei Zheng, Yonggang Zhang, Zhen Fang, Tongliang Liu, Defu Lian, and Bo Han. NoiseDiffusion: Correcting noise for image interpolation with diffusion models beyond spherical linear interpolation. In *The Twelfth International Conference on Learning Representations*, 2024.

Sharon Zhou, Mitchell Gordon, Ranjay Krishna, Austin Narcomey, Li F Fei-Fei, and Michael Bernstein. Hype: A benchmark for human eye perceptual evaluation of generative models. *Advances in neural information processing systems*, 32, 2019.

SUPPLEMENTARY MATERIAL

A THEORETICAL EXTENSIONS AND PROOFS

A.1 PATH DRIFT BOUND

Proposition S1 (Local bounded drift)**.** *Let $\boldsymbol{Z}_q^\star$ be a stationary point, and let U be a neighborhood of $\boldsymbol{Z}_q^\star$. Suppose there exist constants $\kappa, \delta > 0$ such that $\|\boldsymbol{Z} - \boldsymbol{Z}_q^\star\|_2 \le \kappa \|\nabla\Psi(\boldsymbol{Z})\|_2$ and $\|\nabla\Psi(\boldsymbol{Z})\|_2 \le \delta\|q(\boldsymbol{Z})\|_2$ for all $\boldsymbol{Z} \in U$. Then any initialization $\boldsymbol{Z}^0 \in U$ satisfies*

$$\|\boldsymbol{Z}^0 - \boldsymbol{Z}_q^\star\|_2 \le \kappa\delta\|q(\boldsymbol{Z}^0)\|_2.$$

A.2 PROOF OF THEOREM 1

Proof. By L_Ψ-smoothness of Ψ,

$$\Psi(\boldsymbol{Z}^{k+1}) \le \Psi(\boldsymbol{Z}^k) + \langle \nabla\Psi(\boldsymbol{Z}^k),\, \boldsymbol{Z}^{k+1} - \boldsymbol{Z}^k\rangle + \frac{L_\Psi}{2}\|\boldsymbol{Z}^{k+1} - \boldsymbol{Z}^k\|_2^2. \tag{S1}$$

Using $\boldsymbol{Z}^{k+1} - \boldsymbol{Z}^k = -\eta q(\boldsymbol{Z}^k)$ gives

$$\Psi(\boldsymbol{Z}^{k+1}) \le \Psi(\boldsymbol{Z}^k) - \eta\langle \nabla\Psi(\boldsymbol{Z}^k),\, q(\boldsymbol{Z}^k)\rangle + \frac{L_\Psi}{2}\eta^2\|q(\boldsymbol{Z}^k)\|_2^2. \tag{S2}$$

Using the alignment assumption Eq. (9), *i.e.*,

$$\langle \nabla\Psi(\boldsymbol{Z}),\, q(\boldsymbol{Z})\rangle \ge \gamma\|q(\boldsymbol{Z})\|_2^2,$$

we obtain

$$\Psi(\boldsymbol{Z}^{k+1}) \le \Psi(\boldsymbol{Z}^k) - \Big(\gamma\eta - \frac{L_\Psi}{2}\eta^2\Big)\|q(\boldsymbol{Z}^k)\|_2^2. \tag{S3}$$

Since $0 < \eta < 2\gamma/L_\Psi$, the coefficient in parentheses is strictly positive, which proves the claim in Eq. (10). Consequently, the sequence $\{\Psi(\boldsymbol{Z}^k)\}_{k\ge 0}$ is monotonically non-increasing. Since Ψ is bounded below by zero, summing Eq. (10) over k gives $\sum_{k=0}^{\infty}\|q(\boldsymbol{Z}^k)\|_2^2 < \infty$, and therefore $q(\boldsymbol{Z}^k) \to 0$. If the gradient-comparison condition Eq. (11) also holds, then $\|\nabla\Psi(\boldsymbol{Z}^k)\|_2 \le \delta\|q(\boldsymbol{Z}^k)\|_2 \to 0$, completing the proof.

□

A.3 PROOF OF PROPOSITION S1

Proof. Since $\boldsymbol{Z}^0 \in U$, the local error bound $\|\boldsymbol{Z} - \boldsymbol{Z}_q^\star\|_2 \le \kappa\|\nabla\Psi(\boldsymbol{Z})\|_2$ gives

$$\|\boldsymbol{Z}^0 - \boldsymbol{Z}_q^\star\|_2 \le \kappa\|\nabla\Psi(\boldsymbol{Z}^0)\|_2. \tag{S4}$$

Using the gradient-comparison assumption Eq. (11), *i.e.*, $\|\nabla\Psi(\boldsymbol{Z})\|_2 \le \delta\|q(\boldsymbol{Z})\|_2$ at $\boldsymbol{Z}^0$ yields

$$\|\nabla\Psi(\boldsymbol{Z}^0)\|_2 \le \delta\|q(\boldsymbol{Z}^0)\|_2. \tag{S5}$$

Combining the two inequalities proves the result. □

B EMPIRICAL DIAGNOSTICS FOR THE ASSUMPTIONS OF THEOREM 1 (WITH $\lambda = 0$)

Due to the significant memory overhead of backpropagating through $T = 50$ denoising timesteps, we probe the assumptions of Thm. 1 with $\lambda = 0$ in a reduced setting with $T = 10$ timesteps, where exact full-gradient computation is still feasible. Memory usage is further reduced using gradient checkpointing and slicing for both attention and the VAE. These diagnostics do not prove that the assumptions hold throughout the full $T = 50$ experimental setting. However, the $T = 10$ case still differentiates through a nontrivial multi-step denoising chain, and the $T = 50$ sampler uses the same denoising family with a finer discretization. We therefore view this experiment as a tractable diagnostic of whether the surrogate direction can remain aligned with the true gradient when full gradients can be computed explicitly.

We use randomly sampled Gaussian noise paths and random prompt pairs rather than optimized trajectories, since this probes a broader portion of the model's native prior space rather than only trajectories selected by the method.

We first probe the smoothness condition Eq. (8). We conduct a diagnostic experiment similar to Ronai et al. (2025), but with each $\boldsymbol{Z}$ containing $N = 5$ random latent samples. Specifically, we drew many pairs of samples $\boldsymbol{Z}_1, \boldsymbol{Z}_2$, as we detail next. We generated 2000 nonidentical text prompts using ChatGPT-4 (Achiam et al., 2023), drew 5 white Gaussian noise samples for both $\boldsymbol{Z}_1$ and $\boldsymbol{\varepsilon}$ for each of the 2 randomly chosen text prompts, and defined $\boldsymbol{Z}_2$ as

$$\boldsymbol{Z}_2 = \sqrt{\alpha}\boldsymbol{Z}_1 + \sqrt{1-\alpha}\boldsymbol{\varepsilon}, \tag{S6}$$

for various α values, so that both $\boldsymbol{Z}_1$ and $\boldsymbol{Z}_2$ are distributed as $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ (an isotropic Gaussian). The text embeddings of each $\boldsymbol{Z}$ follow Eq. (S17).

By substituting $\boldsymbol{Z}_2$ into the left-hand side of L_{Ψ_0}-smoothness assumption Eq. (8) we obtain, for each α value

$$\max_{\boldsymbol{Z}_1, \boldsymbol{\varepsilon}} \frac{\|\nabla\Psi_0(\boldsymbol{Z}_1) - \nabla\Psi_0(\sqrt{\alpha}\boldsymbol{Z}_1 + \sqrt{1-\alpha}\boldsymbol{\varepsilon})\|_2}{\|(1-\sqrt{\alpha})\boldsymbol{Z}_1 - \sqrt{1-\alpha}\boldsymbol{\varepsilon}\|_2}. \tag{S7}$$

As illustrated in Fig. S1, the obtained maximum in Eq. (S7) remains finite over the sampled points, providing empirical evidence consistent with the smoothness condition in this reduced setting. Note that both axes in Fig. S1 are plotted on logarithmic scales.

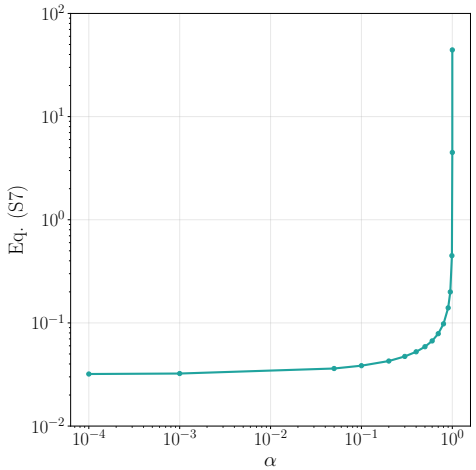


Figure S1: **Diagnostic for the smoothness condition Eq. (8) in Theorem 1.** The maximum obtained over 2000×5 noise realizations for each $\boldsymbol{Z}_1, \boldsymbol{\varepsilon}$ in Eq. (S7), achieved for various α values. Both x- and y-axes are plotted on a logarithmic scale.

We next probe the alignment condition Eq. (9) and the gradient-comparison condition Eq. (11). For this part, we use the same experimental setup as above, but evaluate the ratios on single sampled

latents $\boldsymbol{Z}$, *i.e.*, $\boldsymbol{Z} = \boldsymbol{Z}_1$. For the alignment condition Eq. (9), we estimate the empirical lower bound

$$\gamma = \min_{\boldsymbol{Z}} \frac{\langle \nabla \Psi_0(\boldsymbol{Z}), q_0(\boldsymbol{Z}) \rangle}{\|q_0(\boldsymbol{Z})\|_2^2} , \quad \forall \boldsymbol{Z}. \tag{S8}$$

The minimum obtained value is $\gamma = 0.1236$, giving a positive empirical lower bound in this diagnostic. Additionally, for the gradient-comparison condition Eq. (11), we calculated the upper bound, *i.e.*,

$$\delta = \max_{\boldsymbol{Z}} \frac{\|\nabla \Psi_0(\boldsymbol{Z})\|_2}{\|q_0(\boldsymbol{Z})\|_2} , \quad \forall \boldsymbol{Z}. \tag{S9}$$

The maximum obtained value is $\delta = 107.6272$, giving a finite empirical upper bound in this diagnostic.

Overall, these finite-sample diagnostics provide empirical evidence consistent with the smoothness condition Eq. (8), the alignment condition Eq. (9), and the gradient-comparison condition Eq. (11) of Thm. 1 in the reduced $T = 10$ setting.

C Additional experimental details

C.1 Instantiated path objective

In the experiments, we instantiate the generic path-quality functional $\mathcal{S}$ with three terms defined on decoded-frame pairs. We use $\mathcal{G}_1$ along with the endpoints in the sparse stage and all the frames in the dense stage, as defined in Sec. 4. For each pair (i, j) in the relevant pair set, we compute the LPIPS distance

$$d_{ij}^{\mathrm{L}} = \mathrm{LPIPS}(\boldsymbol{y}_i, \boldsymbol{y}_j) \tag{S10}$$

and a DINO distance d_{ij}^{D}. Following DINO-based perceptual similarity measures, our DINO path length measures semantic smoothness through distances between DINOv2 (Oquab et al., 2024) features of adjacent frames (Ma et al., 2026). Following PixelGen (Ma et al., 2026), the DINO distance uses patch-token features $\phi(\cdot) \in \mathbb{R}^{P\times C}$, excluding the CLS token, and is defined by

$$d_{ij}^{\mathrm{D}} = \frac{1}{P}\sum_{p=1}^{P}\Big(1 - \langle \bar{\phi}_p(\boldsymbol{y}_i), \bar{\phi}_p(\boldsymbol{y}_j)\rangle\Big), \tag{S11}$$

where $\bar{\phi}_p(\boldsymbol{y}_i) = \phi_p(\boldsymbol{y}_i)/\|\phi_p(\boldsymbol{y}_i)\|_2$.

We define an ordered sorted set of consecutive pairs as $\mathcal{P}$. For the first stage, $\mathcal{P}_1$ is defined as the consecutive ordered pairs of $\mathcal{G}_1 \cup \{0, N-1\}$. For the second stage, $\mathcal{P}_2$ is defined similarly, but for all frames.

For a pair set $\mathcal{P} \in \{\mathcal{P}_1, \mathcal{P}_2\}$, we define the LPIPS and DINO path-length terms as

$$\mathcal{L}_{\mathrm{LPIPS}}(\mathcal{P}) = \frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}} d_{ij}^{\mathrm{L}}, \tag{S12}$$

$$\mathcal{L}_{\mathrm{DINO}}(\mathcal{P}) = \frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}} d_{ij}^{\mathrm{D}}. \tag{S13}$$

To encourage uniform transition speed, we also penalize the variance of adjacent LPIPS distances:

$$\mathcal{L}_{\mathrm{PDV}}(\mathcal{P}) = \frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}}\big(d_{ij}^{\mathrm{L}} - \mu_{\mathcal{P}}\big)^2, \qquad \mu_{\mathcal{P}} = \frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}} d_{ij}^{\mathrm{L}}. \tag{S14}$$

This variance penalty is used during optimization; reported PDV values use the standard-deviation convention defined in Sec. 5.

The instantiated objective used in each stage is

$$\mathcal{L}(\mathcal{P}) = \lambda_{\mathrm{LPIPS}}\mathcal{L}_{\mathrm{LPIPS}}(\mathcal{P}) + \lambda_{\mathrm{DINO}}\mathcal{L}_{\mathrm{DINO}}(\mathcal{P}) + \lambda_{\mathrm{PDV}}\mathcal{L}_{\mathrm{PDV}}(\mathcal{P}). \tag{S15}$$

We use $\mathcal{P} = \mathcal{P}_1$ in the sparse stage and $\mathcal{P} = \mathcal{P}_2$ in the dense stage. In all experiments, $(\lambda_{\mathrm{LPIPS}}, \lambda_{\mathrm{DINO}}, \lambda_{\mathrm{PDV}}) = (0.5, 0.5, 3)$.

C.2 Implementation details

All experiments use SD 2.1-base with DDIM inversion and denoising using 50 steps. Images are processed at 512×512 resolution, and all morph sequences contain $N = 17$ frames. The initial path is constructed by SLERP between the inverted endpoint noises at full noise level. For two vectors $\boldsymbol{a}, \boldsymbol{b}$ and interpolation parameter $\tau \in [0, 1]$, we use

$$\mathrm{SLERP}(\boldsymbol{a}, \boldsymbol{b}; \tau) = \frac{\sin((1-\tau)\omega)}{\sin\omega}\,\boldsymbol{a} + \frac{\sin(\tau\omega)}{\sin\omega}\,\boldsymbol{b}, \qquad \omega = \arccos\left(\frac{\langle \boldsymbol{a}, \boldsymbol{b}\rangle}{\|\boldsymbol{a}\|_2\,\|\boldsymbol{b}\|_2}\right), \tag{S16}$$

where for the ith frame we define τ_i as the normalized position of the ith frame for a morph with N frames, *i.e.*, $\tau_i = \frac{i}{N-1}$. We apply Eq. (S16) to construct the initial interior latent path between the fixed endpoint noises. In the main configuration, we optimize the latent path using LPIPS-PPL, DINO-PPL, and PDV with weights $(0.5, 0.5, 3)$. As defined in Sec. 5, DNPMorph-150 is the main setting and DNPMorph-50 is the shorter compute-budget setting. LPIPS uses the AlexNet backbone, while DINO-PPL uses `facebook/dinov2-base` at input resolution 224.

For efficiency, we use batching at several stages of the optimization pipeline. In the reported experiments, we batch all 15 intermediate frames together when applying the diffusion model, so each denoising step is evaluated in parallel across the full set of optimized frames. We additionally use a batch size of 4 for the decoder, LPIPS, and DINO computations. Under the black-box stop-gradient approximation, we do not backpropagate through the diffusion model, so its evaluations are cheaper and can be batched more aggressively, whereas the decoder and the LPIPS/DINO networks remain in the backward pass and are therefore more memory-intensive. Overall, this batching strategy speeds up test-time optimization by evaluating multiple frames or adjacent-frame pairs together whenever memory allows, while leaving the optimization procedure itself unchanged.

For text conditioning, we perform textual inversion (Gal et al., 2023) independently for the two endpoint images. Specifically, for each endpoint prompt, we optimize the corresponding CLIP text embedding for 500 steps using AdamW with learning rate 0.005, yielding endpoint conditioning vectors c_0 and c_{N-1}. For the intermediate frames, we linearly interpolate these conditioning vectors:

$$c_i = (1 - \tau_i)c_0 + \tau_i c_{N-1}, \qquad i = 0, \dots, N-1. \tag{S17}$$

We use an initial learning rate of 100.0 for both optimization stages. For the reported 17-frame setting, we use the sparse and dense optimization sets defined in Sec. 4: $\mathcal{G}_1 = \{4, 8, 12\}$ and $\mathcal{G}_2 = \{1, \dots, 15\}$. The sparse stage optimizes $\{\boldsymbol{z}_i\}_{i \in \mathcal{G}_1}$ for 100 steps and evaluates the objective on $\mathcal{P}_1 = \{(0, 4), (4, 8), (8, 12), (12, 16)\}$. We then densify the path by segment-wise SLERP within the optimized coarse segments. In the dense stage, we jointly optimize $\{\boldsymbol{z}_i\}_{i \in \mathcal{G}_2}$ for 150 steps in the main DNPMorph-150 setting and for 50 steps in the shorter DNPMorph-50 setting, with the objective evaluated on $\mathcal{P}_2 = \{(i, i+1)\}_{i=0}^{15}$. For both stages, we use the same reduce-on-plateau rule: if the total weighted stage objective does not improve for 12 consecutive iterations, we restore the latent residuals to the best state found so far, and multiply the learning rate by 0.1. We allow at most one such decay, so the learning rate can decrease from 100.0 to 10.0.

C.3 Dataset and prompt construction

MorphBench and Morph4Data are used as provided datasets, while AFHQ and CAHQ are custom evaluation subsets that we construct for this work.

MorphBench and Morph4Data. For MorphBench and Morph4Data, prompts are generated automatically with LLaVA. Concretely, we use `llava-hf/llava-1.5-7b-hf` to produce one short caption per image.

AFHQ and CAHQ. In constructing the AFHQ and CAHQ subsets, we follow the same subset-construction procedure used in prior work on curated morphing evaluations (Saito & Matsubara, 2025b). The datasets are built by randomly sampling candidate image pairs and accepting only pairs whose LPIPS distance falls below a dataset-specific threshold, while enforcing that each source image appears in at most one accepted pair. We construct 50 accepted pairs for AFHQ and 50 accepted pairs for CAHQ, using LPIPS on images resized to 512×512, with thresholds 0.7 and 0.6, respectively. Prompts for these two datasets are not generated by LLaVA; instead, they are assigned deterministically from the filename prefixes, *e.g.*, `cat_*` $\mapsto$ "a photo of a cat" and `female_*` $\mapsto$ "a photo of a woman".

C.4 Ablation studies

We report three ablation groups on MorphBench. First, under the shorter DNPMorph-50 compute budget, we evaluate the contribution of the coarse-to-fine schedule by replacing it with a single-stage variant that optimizes all non-endpoint frames jointly from the start; we denote this variant as *Single-stage*. Second, we evaluate the contribution of each loss term by zeroing one of the three weights in Eq. (S15). Third, we examine the anchor regularizer in Eq. (4) by varying λ.

Coarse-to-fine ablation. Table S1 compares the full model to the single-stage variant. Removing the coarse-to-fine schedule degrades LPIPS-PPL, DINO-PPL, and especially FID, while leaving PDV unchanged. This indicates that the coarse-to-fine schedule mainly improves path quality and endpoint consistency, rather than merely enforcing uniform step sizes.

Table S1: **Coarse-to-fine ablation on MorphBench.** Lower is better for all metrics.

Variant	LPIPS-PPL ↓	DINO-PPL ↓	PDV ↓	FID ↓
Single-stage	0.73	0.73	0.018	61.2
Full model	0.67	0.66	0.018	52.2

Table S2: **Loss-term ablations on MorphBench.** Lower is better for all metrics.

Variant	LPIPS-PPL ↓	DINO-PPL ↓	PDV ↓	FID ↓
Without DINO-PPL	0.54	1.03	0.017	53.3
Without LPIPS-PPL	1.40	0.73	0.018	57.4
Without PDV	0.66	0.66	0.028	49.1
Full model	0.67	0.66	0.018	52.2

Loss-term ablations. Table S2 studies the effect of removing each loss term. Removing DINO-PPL yields the best LPIPS-PPL, but worsens DINO-PPL and slightly degrades FID, suggesting that LPIPS alone already encourages some semantic smoothness, while the explicit DINO term is needed for the strongest semantic consistency. Removing LPIPS-PPL substantially degrades LPIPS-PPL and also worsens DINO-PPL and FID, indicating that LPIPS is a central component of the objective. Removing PDV leaves LPIPS-PPL and DINO-PPL unchanged but increases PDV from 0.01 to 0.03, confirming that this term mainly enforces transition uniformity; at the same time, it slightly improves FID, likely because without the uniformity constraint the optimization can shorten the path by pulling intermediate frames closer to the endpoints.

Figure S2 provides a qualitative comparison of the ablation variants on a representative MorphBench pair.

Anchor-regularization weight. The objective in Eq. (4) includes an optional anchor regularizer that biases the noise path toward the SLERP initialization. We evaluate $\lambda \in \{10^{-2}, 10^{-3}, 10^{-4}\}$ on MorphBench using the low-budget setting of 100 sparse steps and 50 dense steps. To quantify the compute effect of this conservative regularization, we also run the unregularized objective ($\lambda = 0$) with the same 100 sparse steps and report the minimum number of dense steps required to match or improve each positive-λ setting on all reported metrics. As shown in Tab. S3, positive anchor regularization yields more conservative updates and can require additional optimization to reach comparable smoothness.

C.5 Direct decoded-frame optimization

The objective optimized by DNPMorph is defined on the decoded morph sequence, but the optimization variables are the initial-noise latents $\boldsymbol{Z}$. To isolate the role of this parametrization, we compare against direct decoded-frame optimization. Specifically, we initialize from the same decoded path, keep the endpoint frames fixed, and optimize the intermediate decoded frames $\boldsymbol{Y}$ directly using the same LPIPS-PPL, DINO-PPL, and LPIPS-based PDV terms used by DNPMorph.

Table S4 reports results on MorphBench after 150 optimization iterations. Direct $\boldsymbol{Y}$ optimization achieves substantially lower values on the optimized metrics, but performs much worse on the held-out DreamSim-PPL metric and on FID. This indicates that unconstrained decoded-frame optimization can exploit the optimized feature losses without producing a correspondingly better morph.

Figure S3 shows a representative qualitative example. Although direct $\boldsymbol{Y}$ optimization strongly reduces the optimized losses, the visual sequence changes only weakly and does not yield the same perceptual improvement as optimizing the latent noise path. This behavior is consistent with adversarial-like exploitation of LPIPS/DINO feature distances. In contrast, DNPMorph optimizes through the fixed diffusion generator and yields substantially better held-out/fidelity behavior.



Figure S2: **Qualitative ablation comparison on a representative MorphBench pair.** We show the effect of using a single-stage optimization and of removing each loss term, compared with the full model.

Table S3: **Effect of anchor-regularization weight on MorphBench.** Each positive-λ row uses 100 sparse steps and 50 dense steps. The last column reports the minimum number of dense steps required by the unregularized setting ($\lambda = 0$), after the same 100 sparse steps, to match or improve that row on all reported metrics. Lower is better for all metrics.

λ	LPIPS-PPL ↓	DINO-PPL ↓	DreamSim-PPL ↓	PDV ↓	$\lambda = 0$ dense steps
10^{-2}	1.25	1.47	0.55	0.049	0
10^{-3}	1.00	1.04	0.44	0.034	5
10^{-4}	0.70	0.71	0.32	0.021	45

C.6 Transfer beyond optimized metrics

The main objective used by DNPMorph directly optimizes LPIPS-PPL, DINO-PPL, and an LPIPS-based PDV term. Therefore, improvements on these metrics primarily show that the proposed latent-path optimization improves its intended decoded-sequence objective. App. C.5 shows that directly optimizing decoded frames can reduce these optimized losses even further, but transfers poorly to held-out DreamSim-PPL and FID. We therefore additionally evaluate whether the gains obtained by DNPMorph transfer to metrics that are not used in the optimization objective.

To test transfer beyond the optimized losses, we evaluate DreamSim-based metrics, which are not used in the optimization objective. First, Fig. S4 shows the evolution of LPIPS-PPL, DINO-PPL, and DreamSim-PPL along the dense stage on MorphBench. Each curve is normalized by its value at iteration 0 of the dense stage. LPIPS-PPL and DINO-PPL are part of the objective, while DreamSim-PPL is held out. The held-out DreamSim-PPL decreases together with the optimized path-length terms, suggesting that the optimization improves perceptual smoothness beyond the particular losses used directly.

We also evaluate transition-speed uniformity using DreamSim. Since DNPMorph optimizes an LPIPS-based transition-speed regularizer and reports PDV as the standard deviation of adjacent LPIPS distances, we compute the analogous standard-deviation metric with adjacent DreamSim distances. For the full evaluated path, we use the dense-stage adjacent-pair set $\mathcal{P}_2$ defined in Sec. 4. For $(i, j) \in \mathcal{P}_2$, define

$$d_{ij}^{\mathrm{DS}} = \mathrm{DreamSim}(\boldsymbol{y}_i, \boldsymbol{y}_j), \qquad \bar{d}^{\mathrm{DS}} = \frac{1}{|\mathcal{P}_2|} \sum_{(i,j)\in\mathcal{P}_2} d_{ij}^{\mathrm{DS}}.$$

Table S4: **Direct decoded-frame optimization on MorphBench.** We compare DNPMorph-150 to direct optimization of decoded frames $\boldsymbol{Y}$ for 150 iterations using the same LPIPS/DINO/PDV objective. Lower is better.

	Optimized metrics			Held-out / fidelity	
Variant	LPIPS-PPL ↓	DINO-PPL ↓	PDV ↓	DreamSim-PPL ↓	FID ↓
DNPMorph-150	0.53	0.50	0.014	**0.25**	**52.3**
Direct $\boldsymbol{Y}$ optimization	**0.27**	**0.26**	**0.006**	0.50	80.12

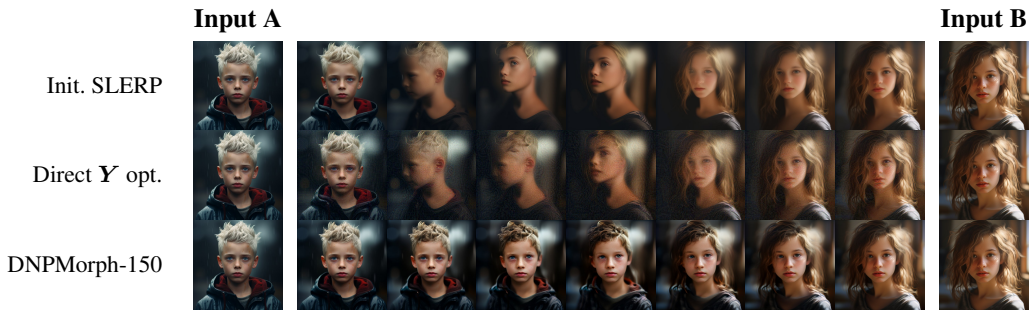


Figure S3: **Direct decoded-frame optimization can exploit optimized metrics without visibly improving the morph.** We compare the initialization (noise-space SLERP), direct $\boldsymbol{Y}$ optimization, and DNPMorph-150 on a representative MorphBench pair. Direct $\boldsymbol{Y}$ optimization strongly reduces the optimized LPIPS/DINO/PDV objective, but yields worse held-out DreamSim-PPL and FID, as reported in Tab. S4.

We define DreamSim-PDV as

$$\text{DreamSim-PDV} = \sqrt{\frac{1}{|\mathcal{P}_2|} \sum_{(i,j)\in\mathcal{P}_2} \left(d_{ij}^{\text{DS}} - \bar{d}^{\text{DS}}\right)^2}.$$

This metric follows the same standard-deviation convention as the reported LPIPS-based PDV metric, but replaces adjacent LPIPS distances with adjacent DreamSim distances.

Table S5 reports this additional held-out measure of transition-speed uniformity. DNPMorph obtains the best DreamSim-PDV on three of four datasets and is close to the best method on AFHQ. Together with the DreamSim-PPL results in Tab. 1 and the optimization trajectory in Fig. S4, these results suggest that the improvements are not confined to the LPIPS- and DINO-based quantities used in the optimization objective.

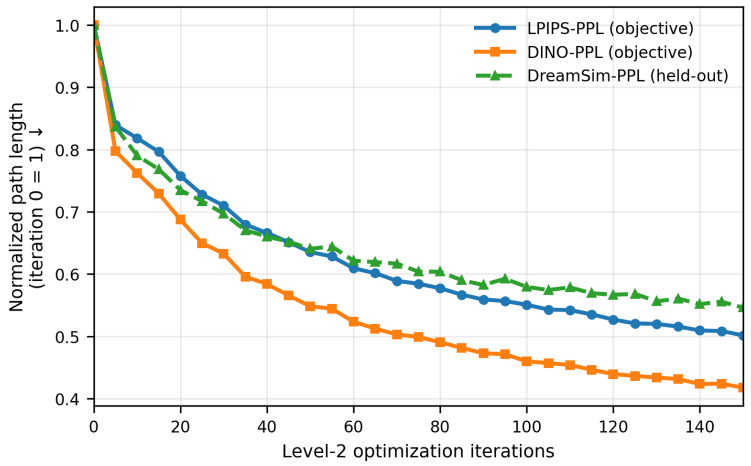


Figure S4: **Relative path-length reduction during the dense stage on MorphBench.** Each curve is normalized by its value at iteration 0 of the dense stage. LPIPS-PPL and DINO-PPL are part of the DNPMorph objective, while DreamSim-PPL is held out.

Table S5: **DreamSim-based path-distance variance.** DreamSim-PDV measures the standard deviation of adjacent DreamSim distances along the generated path. Lower is better. This metric is not used in the optimization objective. The DNPMorph column uses the main DNPMorph-150 setting.

Dataset	SLERP	DiffMorpher	IMPUS	FreeMorph	GeoMorph	DNPMorph
MB	0.04	0.013	0.019	0.052	0.019	**0.011**
M4D	0.08	0.038	0.044	0.089	0.041	**0.031**
AF	0.021	**0.005**	0.009	0.05	0.009	0.006
CA	0.026	0.01	0.019	0.052	0.035	**0.008**

C.7 Paired bootstrap confidence intervals

Prior diffusion-morphing benchmarks typically report aggregate metric values over fixed endpoint pairs, following the standard reporting convention for this task. We include an additional paired uncertainty analysis to quantify whether the observed improvements are stable with respect to the sampled endpoint pairs. A raw standard deviation over image pairs is not the most informative uncertainty estimate here, because it is dominated by endpoint-pair difficulty: some endpoint pairs naturally require long perceptual paths, while others require short paths. Such variability mainly reflects the diversity of the benchmark pairs, rather than uncertainty in the method comparison.

We therefore use a paired nonparametric bootstrap over endpoint pairs. This follows the standard idea of paired bootstrap resampling for system comparison, where two methods are evaluated on the same test examples and the examples are resampled with replacement (Koehn, 2004; Graham et al., 2014). More generally, bootstrap confidence intervals provide a nonparametric way to estimate uncertainty for statistics whose sampling distribution is not available in closed form (DiCiccio & Efron, 1996). Bootstrap confidence intervals are also used in generative-model evaluation, e.g., in HYPE (Zhou et al., 2019).

For each dataset, metric, competitor b, and endpoint pair i, let $M_{i,\text{ours}}$ and $M_{i,b}$ denote the corresponding per-pair metric values, where lower is better. We compute the paired log-ratio

$$r_i = \log \frac{M_{i,\text{ours}} + \epsilon}{M_{i,b} + \epsilon},$$

with $\epsilon = 10^{-8}$ for numerical stability. We report the geometric mean ratio

$$R = \exp\left(\frac{1}{N}\sum_{i=1}^{N} r_i\right).$$

Table S6: **Paired bootstrap confidence intervals for DNPMorph-50 against each competitor.** For each dataset and metric, each cell compares DNPMorph-50 against the indicated competitor using paired per-pair log-ratios on matched endpoint pairs. We report the geometric mean ratio with the 95% paired bootstrap confidence interval in brackets. Values below 1 indicate that DNPMorph-50 is better.

Dataset	Metric	SLERP	DiffMorpher	IMPUS	FreeMorph	GeoMorph
MB	LPIPS-PPL	0.459 [0.421, 0.498]	0.670 [0.623, 0.725]	0.368 [0.335, 0.404]	0.453 [0.406, 0.504]	0.761 [0.707, 0.822]
MB	PDV	0.293 [0.260, 0.328]	1.110 [0.966, 1.274]	0.407 [0.353, 0.473]	0.175 [0.153, 0.200]	0.744 [0.652, 0.846]
MB	DINO-PPL	0.350 [0.314, 0.389]	0.588 [0.535, 0.650]	0.375 [0.337, 0.419]	0.341 [0.302, 0.385]	0.547 [0.504, 0.596]
MB	DreamSim-PPL	0.463 [0.408, 0.521]	0.917 [0.847, 0.995]	0.561 [0.506, 0.623]	0.397 [0.348, 0.452]	0.878 [0.815, 0.951]
M4D	LPIPS-PPL	0.555 [0.524, 0.587]	0.716 [0.672, 0.763]	0.418 [0.388, 0.451]	0.736 [0.671, 0.805]	0.904 [0.845, 0.969]
M4D	PDV	0.411 [0.374, 0.450]	1.445 [1.268, 1.656]	0.839 [0.722, 0.977]	0.305 [0.270, 0.346]	1.027 [0.908, 1.163]
M4D	DINO-PPL	0.456 [0.423, 0.493]	0.564 [0.521, 0.608]	0.444 [0.405, 0.489]	0.623 [0.558, 0.695]	0.687 [0.630, 0.751]
M4D	DreamSim-PPL	0.539 [0.499, 0.581]	0.797 [0.738, 0.861]	0.625 [0.566, 0.691]	0.736 [0.652, 0.828]	1.083 [0.999, 1.181]
AF	LPIPS-PPL	0.425 [0.403, 0.449]	0.552 [0.524, 0.582]	0.336 [0.318, 0.355]	0.404 [0.382, 0.428]	0.598 [0.571, 0.627]
AF	PDV	0.470 [0.417, 0.528]	1.525 [1.319, 1.754]	0.693 [0.620, 0.778]	0.233 [0.216, 0.251]	1.081 [0.990, 1.174]
AF	DINO-PPL	0.340 [0.319, 0.364]	0.482 [0.454, 0.512]	0.330 [0.308, 0.355]	0.322 [0.296, 0.351]	0.473 [0.445, 0.505]
AF	DreamSim-PPL	0.485 [0.445, 0.525]	0.914 [0.864, 0.966]	0.598 [0.561, 0.639]	0.326 [0.305, 0.350]	0.731 [0.702, 0.762]
CA	LPIPS-PPL	0.520 [0.485, 0.568]	0.672 [0.627, 0.735]	0.421 [0.391, 0.463]	0.395 [0.363, 0.434]	0.456 [0.409, 0.508]
CA	PDV	0.591 [0.513, 0.685]	1.707 [1.480, 1.988]	0.658 [0.586, 0.749]	0.270 [0.249, 0.299]	0.983 [0.855, 1.125]
CA	DINO-PPL	0.407 [0.373, 0.454]	0.522 [0.477, 0.583]	0.377 [0.342, 0.422]	0.303 [0.272, 0.340]	0.275 [0.239, 0.316]
CA	DreamSim-PPL	0.588 [0.546, 0.645]	0.995 [0.932, 1.086]	0.627 [0.581, 0.692]	0.325 [0.295, 0.361]	0.305 [0.262, 0.356]

Values below 1 indicate that our method improves over the competitor on the same endpoint pairs. To estimate uncertainty, we draw $B = 10{,}000$ bootstrap samples by resampling the N endpoint pairs with replacement. For each bootstrap sample, we recompute R, yielding bootstrap ratios $R^{*(1)}, \ldots, R^{*(B)}$. We then report the percentile 95% confidence interval,

$$\left[Q_{0.025}\left(\{R^{*(j)}\}_{j=1}^{B}\right), Q_{0.975}\left(\{R^{*(j)}\}_{j=1}^{B}\right) \right],$$

where Q_α denotes the empirical α-quantile. Thus, the interval contains the central 95% of the bootstrap distribution, leaving 2.5% mass in each tail.

These intervals capture variability with respect to benchmark endpoint-pair composition, not training randomness or model initialization. Since FID is computed at the dataset level rather than per pair, we apply this paired analysis only to the per-pair smoothness metrics.

Tables S6 and S7 report the results for DNPMorph-50 and DNPMorph-150, respectively. For both settings, the LPIPS-PPL and DINO-PPL improvements are supported at the nominal 95% bootstrap-CI level across all datasets and competitors, with all confidence intervals below 1. The held-out DreamSim-PPL improvements are strongest for DNPMorph-150, where all confidence intervals are also below 1. PDV is more mixed: DNPMorph improves over several competitors but does not consistently outperform DiffMorpher on PDV, matching the aggregate results in Tab. 1. Overall, the paired analysis supports the main smoothness claim while clarifying that the strongest statistical evidence is for path-length smoothness rather than transition-uniformity against every baseline.

Table S7: **Paired bootstrap confidence intervals for DNPMorph-150 against each competitor.** For each dataset and metric, each cell compares DNPMorph-150 against the indicated competitor using paired per-pair log-ratios on matched endpoint pairs. We report the geometric mean ratio with the 95% paired bootstrap confidence interval in brackets. Values below 1 indicate that DNPMorph-150 is better.

Dataset	Metric	SLERP	DiffMorpher	IMPUS	FreeMorph	GeoMorph
MB	LPIPS-PPL	0.363 [0.330, 0.398]	0.530 [0.487, 0.580]	0.291 [0.263, 0.323]	0.358 [0.319, 0.401]	0.602 [0.554, 0.656]
MB	PDV	0.236 [0.209, 0.268]	0.896 [0.775, 1.037]	0.328 [0.284, 0.384]	0.141 [0.124, 0.161]	0.601 [0.521, 0.690]
MB	DINO-PPL	0.266 [0.236, 0.299]	0.446 [0.401, 0.499]	0.285 [0.252, 0.323]	0.259 [0.228, 0.294]	0.415 [0.377, 0.459]
MB	DreamSim-PPL	0.394 [0.345, 0.446]	0.779 [0.714, 0.851]	0.477 [0.428, 0.531]	0.338 [0.295, 0.385]	0.747 [0.691, 0.811]
M4D	LPIPS-PPL	0.426 [0.396, 0.458]	0.550 [0.510, 0.593]	0.321 [0.293, 0.351]	0.565 [0.513, 0.622]	0.694 [0.640, 0.754]
M4D	PDV	0.373 [0.334, 0.415]	1.312 [1.136, 1.519]	0.762 [0.651, 0.895]	0.277 [0.245, 0.314]	0.932 [0.818, 1.068]
M4D	DINO-PPL	0.328 [0.297, 0.364]	0.406 [0.370, 0.445]	0.319 [0.285, 0.360]	0.449 [0.400, 0.503]	0.494 [0.447, 0.550]
M4D	DreamSim-PPL	0.435 [0.401, 0.474]	0.644 [0.595, 0.698]	0.506 [0.457, 0.561]	0.595 [0.528, 0.669]	0.876 [0.805, 0.959]
AF	LPIPS-PPL	0.348 [0.326, 0.374]	0.452 [0.425, 0.482]	0.275 [0.259, 0.294]	0.330 [0.311, 0.353]	0.489 [0.463, 0.524]
AF	PDV	0.479 [0.422, 0.542]	1.556 [1.342, 1.797]	0.707 [0.630, 0.797]	0.238 [0.221, 0.256]	1.103 [1.016, 1.194]
AF	DINO-PPL	0.263 [0.242, 0.286]	0.371 [0.347, 0.403]	0.255 [0.236, 0.278]	0.248 [0.227, 0.272]	0.365 [0.340, 0.398]
AF	DreamSim-PPL	0.438 [0.395, 0.481]	0.824 [0.774, 0.880]	0.540 [0.504, 0.582]	0.294 [0.276, 0.315]	0.659 [0.628, 0.697]
CA	LPIPS-PPL	0.421 [0.399, 0.445]	0.544 [0.518, 0.572]	0.341 [0.323, 0.362]	0.320 [0.298, 0.343]	0.370 [0.334, 0.408]
CA	PDV	0.568 [0.501, 0.642]	1.643 [1.444, 1.859]	0.633 [0.575, 0.698]	0.260 [0.246, 0.275]	0.945 [0.824, 1.084]
CA	DINO-PPL	0.305 [0.285, 0.328]	0.391 [0.364, 0.422]	0.282 [0.260, 0.307]	0.227 [0.207, 0.249]	0.206 [0.181, 0.234]
CA	DreamSim-PPL	0.500 [0.473, 0.527]	0.847 [0.810, 0.884]	0.533 [0.506, 0.560]	0.276 [0.256, 0.297]	0.259 [0.223, 0.302]

C.8 User study

A screenshot from the user study is provided in Fig. S5.

The exact instructions we provided to each participant in the user study were as follows: "In each section, you will see two morphing results, each showing the same two endpoint images ('Input A' and 'Input B'), and the intermediate morph between them. Please answer the question 'Which morph do you think is better overall?', according to the following criteria: Smoothness - the sequence should change gradually from frame to frame, with no sudden jumps or abrupt semantic changes. Endpoint consistency - the intermediate frames should stay visually related to the two endpoint images and not drift to unrelated content. Naturalness - the intermediate frames should look visually plausible and free of obvious artifacts. Please select 'First morph' or 'Second morph' according to which one you think is better overall. If possible, zoom in on the images to get a better view."

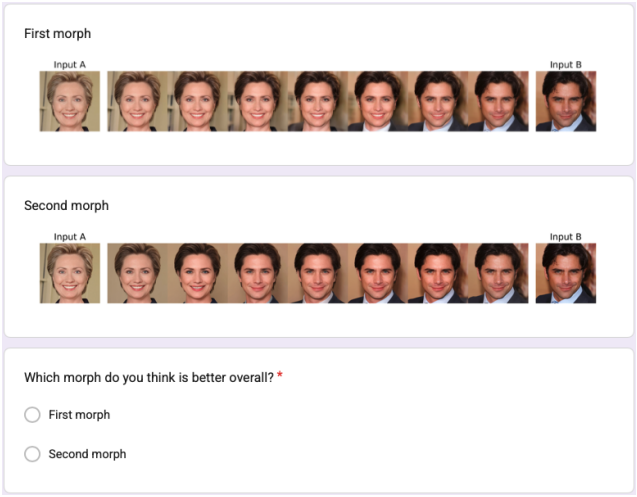


Figure S5: **A screenshot of a representative example from the human perceptual study.** The order of the pairwise comparison is random, and users were asked a two-alternative forced-choice question to select their preferred morphing results.

C.9 Optimization curves and diagnostics

We provide additional optimization diagnostics for the representative boy-to-girl pair. Figure S6 visualizes the evolution of the optimized frames during the sparse stage. Figure S7 reports the corresponding sparse-stage losses, and Fig. S8 reports the losses during the subsequent dense refinement stage. The curves are stage-local: in the sparse stage, losses are computed on the coarse segment pairs, while in the dense stage they are computed on all adjacent frame pairs.

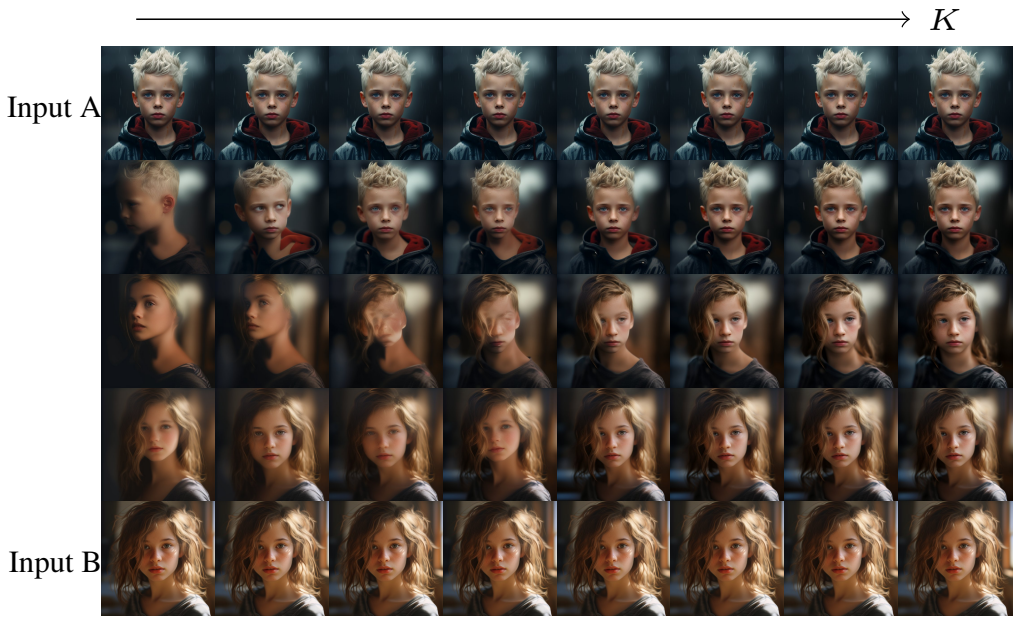
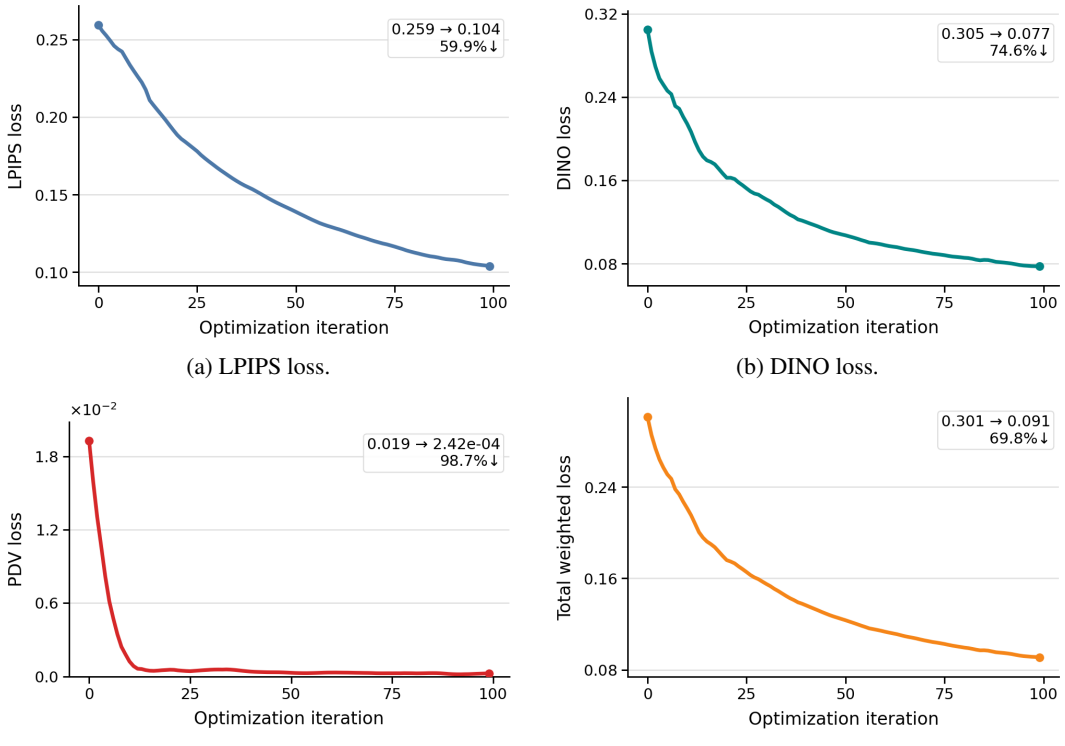


Figure S6: **Intermediate results along the sparse stage for the representative boy-to-girl pair.** Rows correspond to the two endpoints and the three intermediate frames optimized in the sparse stage. The inner frames are progressively refined toward a smoother and more endpoint-consistent transition.



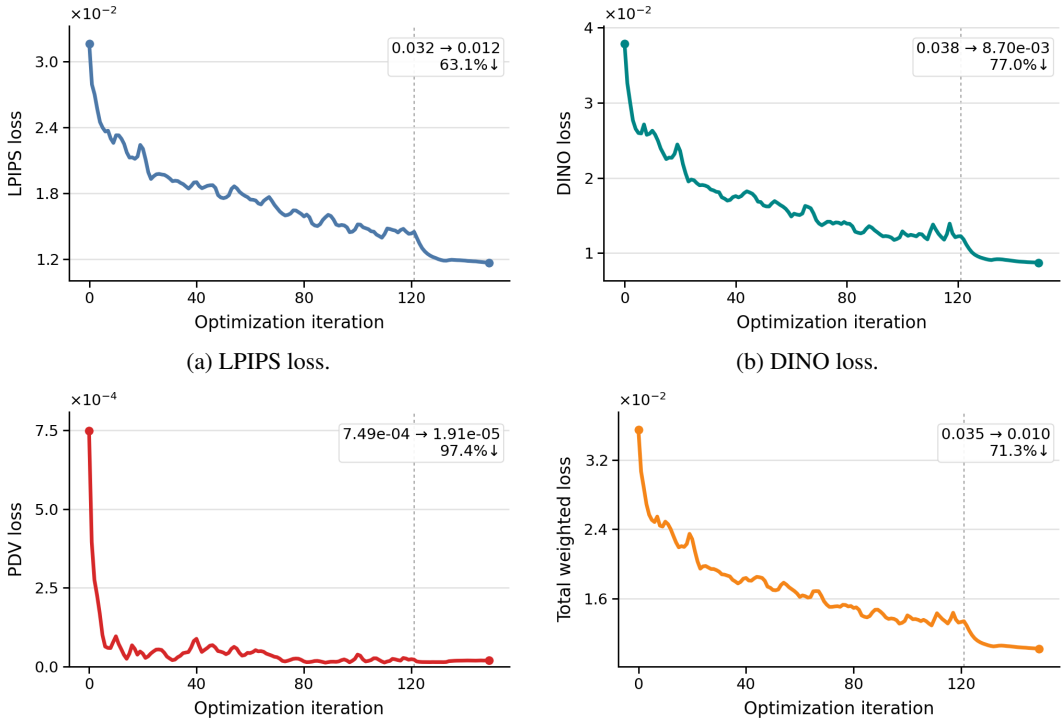
(a) LPIPS loss.

(b) DINO loss.

(c) PDV loss.

(d) Total weighted loss.

Figure S7: **Sparse-stage optimization curves for the representative boy-to-girl pair.** We show the LPIPS, DINO, PDV, and total weighted losses during the sparse stage. This stage optimizes only the three intermediate frames shown in Fig. S6, and evaluates the objective on the corresponding coarse segment pairs.



(a) LPIPS loss.

(b) DINO loss.

(c) PDV loss.

(d) Total weighted loss.

Figure S8: **Dense refinement-stage optimization curves for the representative boy-to-girl pair.** After the sparse stage, the path is densified and refined over all non-endpoint frames. We show the LPIPS, DINO, PDV, and total weighted losses during this refinement stage. The dashed vertical line marks the learning-rate reduction and rollback step used by the optimizer.

D ADDITIONAL RESULTS

In addition to the static examples shown below, we provide supplementary GIF visualizations comparing morph sequences produced by DNPMorph and competing methods.

Visualization note. To satisfy the submission PDF-size limit, the qualitative grids in this appendix are displayed using JPEG-compressed copies of the generated frames. All quantitative metrics are computed from the original generated images, not from the compressed visualization files.

Figure S9 presents another representative example of the coarse-to-fine refinement shown in Fig. 3.



Figure S9: **DNPMorph coarse-to-fine refinement.** Another representative example of the two-stage refinement process shown in Fig. 3. Starting from a sparse path, we first optimize frames $4, 8, 12$, densify the path by local SLERP, and then refine all non-endpoint frames.

Figures S10, S11, and S12 present the same comparison as Fig. 5, each for a different representative example.

Figure S13 shows the same six pairs as Fig. 4, but with seven intermediate frames instead of three, together with five additional examples.



Figure S10: **Additional qualitative comparison on another representative pair.** The same comparison as in Fig. 5, shown here for a different representative pair. Our approach yields smoother transitions while maintaining intermediate frames that remain more consistent with the endpoints. We show 7 intermediate frames.



Figure S11: **Additional qualitative comparison on another representative pair.** The same comparison as in Fig. 5, shown here for a different representative pair. Our approach yields smoother transitions while maintaining intermediate frames that remain more consistent with the endpoints. We show 7 intermediate frames.



Figure S12: **Additional qualitative comparison on another representative pair.** The same comparison as in Fig. 5, shown here for a different representative pair. Our approach yields smoother transitions while maintaining intermediate frames that remain more consistent with the endpoints. We show 7 intermediate frames.



Figure S13: **Extended qualitative examples produced by DNPMorph.** We show the same six pairs as in Fig. 4, but with seven intermediate frames instead of three, together with five additional representative morphs.

E ALGORITHM

Algorithm S1 Coarse-to-fine noise-path optimization.

1: **Input:** fixed endpoint latents $\boldsymbol{z}_0, \boldsymbol{z}_{N-1}$; stage sets $\mathcal{G}_1, \mathcal{G}_2$ and pair sets $\mathcal{P}_1, \mathcal{P}_2$; steps $K_{\text{sparse}}, K_{\text{dense}}$
2: **Output:** decoded morph sequence $\{\boldsymbol{y}_i\}_{i=0}^{N-1}$
3: Initialize the sparse frames $\{\boldsymbol{z}_i\}_{i\in\mathcal{G}_1}$ by SLERP between $\boldsymbol{z}_0$ and $\boldsymbol{z}_{N-1}$.
4: **Sparse stage:** optimize $\{\boldsymbol{z}_i\}_{i\in\mathcal{G}_1}$ for K_{sparse} steps, with the path objective evaluated on $\mathcal{P}_1$.
5: Initialize the remaining non-endpoint frames $\{\boldsymbol{z}_i\}_{i\in\mathcal{G}_2\setminus\mathcal{G}_1}$ by local SLERP within the optimized coarse segments.
6: **Dense stage:** optimize $\{\boldsymbol{z}_i\}_{i\in\mathcal{G}_2}$ for K_{dense} steps, with the path objective evaluated on $\mathcal{P}_2$.
7: Decode the optimized path and return $\{\boldsymbol{y}_i\}_{i=0}^{N-1}$.

F RUNTIME AND COMPUTE–QUALITY TRADEOFF

We measure wall-clock runtime for all methods under the same SD 2.1 backbone and 17-frame protocol used in the main experiments. All measurements are performed on a single NVIDIA A100 GPU, excluding one-time model loading and including all method-specific per-pair optimization, adaptation, path construction, and generation steps.

Table S8: **Runtime comparison.** All methods are evaluated for one 17-frame morph under the SD 2.1 backbone on a single NVIDIA A100 GPU. Runtime excludes one-time model loading and includes all method-specific per-pair optimization, adaptation, path construction, and generation steps.

Method	Time / pair (min)
FreeMorph	0.75
GeoMorph	2.5
DiffMorpher	4.5
IMPUS	17
DNPMorph-50	20.0
DNPMorph-150	45.0

The higher runtime of DNPMorph follows from its optimization target: rather than constructing a path once and then generating it, DNPMorph repeatedly evaluates the decoded sequence during test-time optimization. In our implementation, each sparse-stage iteration takes approximately 4.5 seconds, and each dense-stage iteration takes approximately 15 seconds when optimizing all frames jointly. With 100 sparse-stage iterations, DNPMorph-50 uses 50 dense-stage iterations and takes about 20 minutes per pair, while DNPMorph-150 uses 150 dense-stage iterations and takes about 45 minutes per pair.

Figure S14 makes this quality–compute tradeoff explicit on MorphBench. Starting from the sparse-stage output, dense-stage optimization rapidly improves the smoothness metrics. The competing-method markers show the runtime–quality operating points of DiffMorpher, IMPUS, FreeMorph, and GeoMorph under the same SD 2.1 backbone and 17-frame protocol. Among these baselines, GeoMorph is the closest competitor on most MorphBench metrics and ties DiffMorpher on PDV. Moving from DNPMorph-50 to DNPMorph-150 further improves LPIPS-PPL, PDV, DINO-PPL, and DreamSim-PPL, showing that runtime can be traded for stronger smoothness. In contrast, FID is best early in the dense stage and then mildly worsens as optimization increasingly prioritizes path smoothness. This suggests a tradeoff between adjacent-frame smoothness and distributional fidelity, rather than a monotonic improvement in all aspects of morph quality.

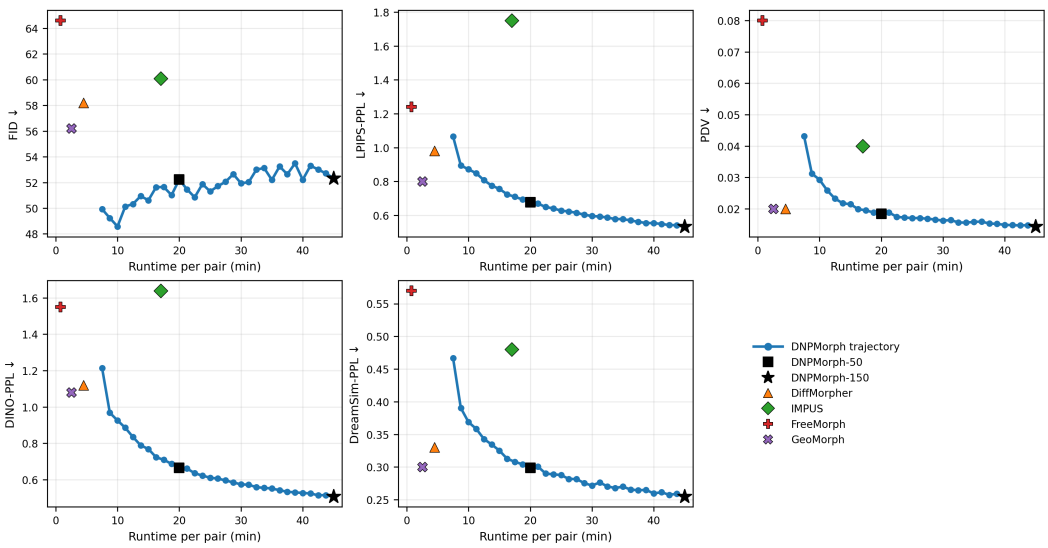


Figure S14: **Compute–quality tradeoff along the dense stage on MorphBench.** The sparse-stage output is kept fixed, and we evaluate the generated paths every 5 iterations during the dense stage. The horizontal axis shows total runtime per pair, including the fixed sparse-stage runtime. The blue curve shows the DNPMorph trajectory as additional dense-stage optimization iterations are used. The black square marks DNPMorph-50, and the black star marks DNPMorph-150. The remaining markers show the non-SLERP competing methods, each plotted at its measured runtime from Tab. S8 and its MorphBench metric value from Tab. 1. This visualizes the runtime–quality landscape: competing methods are faster, while DNPMorph trades additional test-time optimization for stronger smoothness.

G Limitations and Failure Cases

While DNPMorph substantially improves smoothness metrics and the DNPMorph variants improve FID on three of four datasets, several limitations remain. First, difficult endpoint pairs with large semantic, structural, or viewpoint gaps can still produce abrupt changes, hybridized intermediate states, or unstable correspondences. More generally, our objective enforces adjacent-frame perceptual smoothness, but does not explicitly enforce temporally natural motion or long-range object correspondence. Thus, low adjacent-frame distances indicate local perceptual smoothness, not necessarily physically plausible motion or globally correct correspondence.

Representative failure cases are shown in Fig. S15. In the dog-jump example, the frames are locally smooth, but the jump itself remains temporally unnatural: the dog gradually becomes taller, the main jump occurs abruptly, and only the later leg motion is smooth. In the bird example, the method does not recover a natural wing-opening motion; instead, the wings gradually appear and fade. In the chair example, the endpoint gap is likely too large, producing plausible intermediate frames that still undergo large transitions and are not always well aligned with either endpoint. In the flower-to-monkey example, the large semantic gap yields locally smooth transitions through intermediate objects that are plausible but not clearly identifiable as either endpoint.

Second, like other diffusion-based morphing methods, our approach inherits biases and weaknesses of the underlying diffusion backbone. In particular, transitions involving humans, articulated limbs, thin structures, or fine geometric details can be less stable and may contain local artifacts.

Third, as shown in Tab. S8, directly optimizing the latent path requires more test-time computation than single-pass interpolation or inference-control methods. Thus, DNPMorph is most suitable for offline high-quality morph generation.

Finally, DNPMorph is initialization-dependent. Our method starts from the noise SLERP path, which is a natural and principled initialization: it lies in the Gaussian prior geometry of the diffusion model, provides a simple reference trajectory, and in our analysis and experiments the optimization

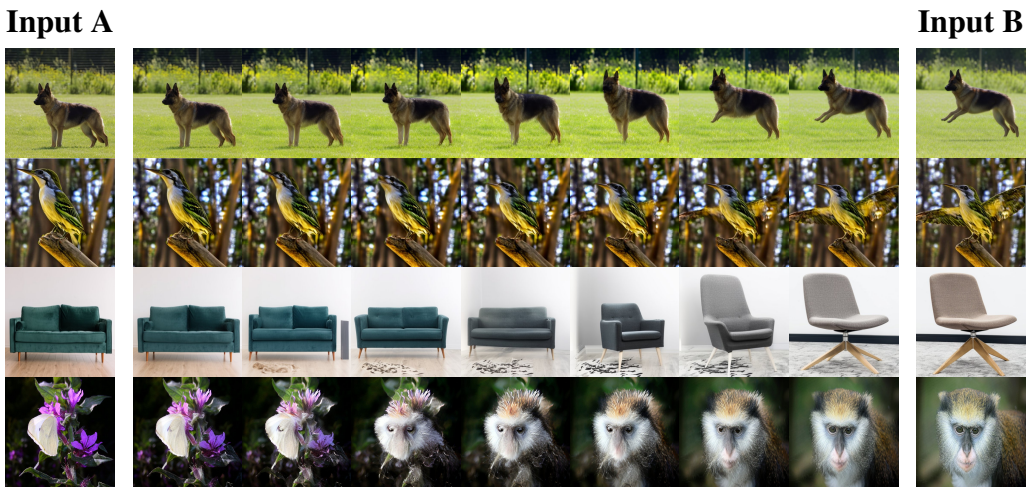


Figure S15: **Representative failure cases of DNPMorph.** Even with direct noise-path optimization, difficult endpoint pairs can still exhibit temporally unnatural motion, abrupt semantic changes, unstable correspondences, or intermediate frames that are locally plausible but not well aligned with either endpoint.

behaves largely as a controlled correction of this prior path rather than a free drift. Nevertheless, the subsequent optimization is still local. Different path initializations may therefore lead to different optimized trajectories, and our current analysis does not guarantee convergence to a global optimum. Thus, while SLERP is a strong default initialization, better initial paths could further improve the final optimized morph. Designing such initializations and developing stronger global guarantees remain interesting directions for future work.