Diffusion Morphs (DiM)

The Power of Iterative Generative Models for Attacking FR Systems

Zander W. Blasingame

Clarkson University
Potsdam, NY, USA

A talk presented to the Biometrics Security & Privacy Group at Idiap

23.07.2024

Outline

Introduction

Diffusion Morphs (DiM)

Fast-DiM

Greedy-DiM

AdjointDEIS

Conclusion

Introduction

Face Morphing

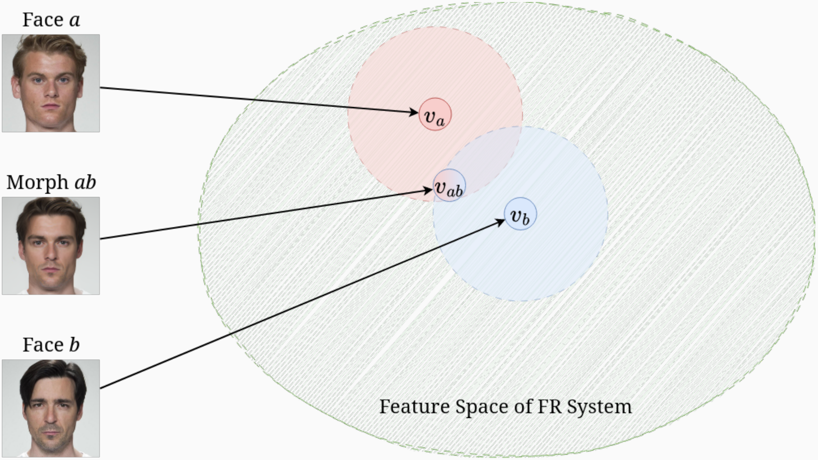


Figure 1: Images from FRLL[1] dataset. Morph generated by us.

[1] Lisa DeBruine and Benedict Jones. "Face Research Lab London Set". In: (May 2017). DOI: 10.6084/m9.figshare.5047666.v5. URL: https://figshare.com/articles/dataset/Face_Research_Lab_London_Set/5047666.

Generative AI Morph Creation Pipeline

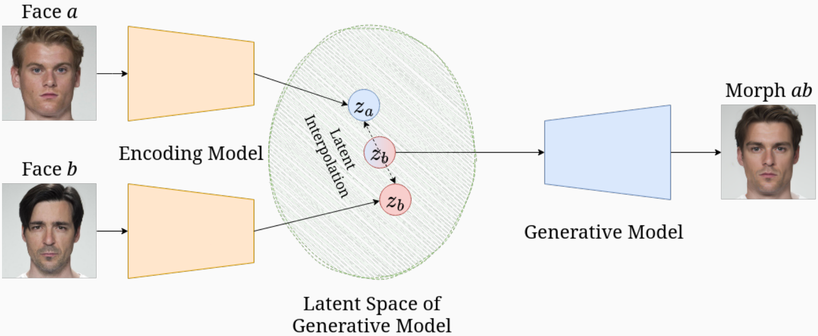
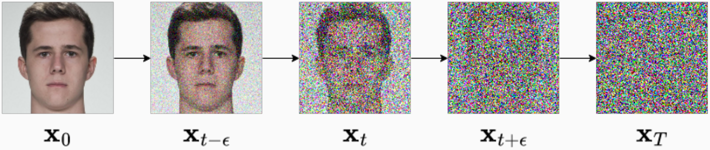


Figure 2: General morph creation pipeline using generative AI models.

Diffusion Models



- Forward diffusion process is governed by the Itô SDE

$$\mathrm{d}\mathbf{x}_t = f(t)\mathbf{x}_t \, \mathrm{d}t + g(t) \, \mathrm{d}\mathbf{w}_t \tag{1}$$

where $\{\mathbf{w}_t\}_{t\in[0,T]}$ is the standard Wiener process on $[0, T]$

Reverse Diffusion Process

- The diffusion equation can be reversed with

$$\mathrm{d}\mathbf{x}_t = \left[f(t)\mathbf{x}_t - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x}_t)\right] \mathrm{d}t + g(t)\, \mathrm{d}\check{\mathbf{w}}_t \tag{2}$$

  where $\check{\mathbf{w}}_t$ is the *backwards* Wiener process defined as $\check{\mathbf{w}}_t \coloneqq \mathbf{w}_t - \mathbf{w}_T$

- The marginal distributions $p_t(\mathbf{x})$ of eq. (2) follow an associated ODE known as the *probability flow* ODE[2]

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = f(t)\mathbf{x}_t - \frac{1}{2} g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x}_t) \tag{3}$$

[2] Yang Song et al. "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=PxTIG12RRHS.

Learning The Reverse Diffusion SDE

- Often the Variance Preserving (VP) framework is used where the drift and diffusion coefficients are

$$f(t) = \frac{\mathrm{d}\log \alpha_t}{\mathrm{d}t} \tag{4}$$

$$g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log \alpha_t}{\mathrm{d}t}\sigma_t^2 \tag{5}$$

  for some noise schedule α_t, σ_t

- Sampling the forward trajectory then simplifies to

$$\mathbf{x}_t = \alpha_t \mathbf{x}_0 + \sigma_t \boldsymbol{\epsilon} \tag{6}$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{7}$$

Learning The Reverse Diffusion SDE

- Learning the score $\nabla_{\mathbf{x}} \log p_t(\mathbf{x}_t)$ is similar to learning the noise $\boldsymbol{\epsilon}$

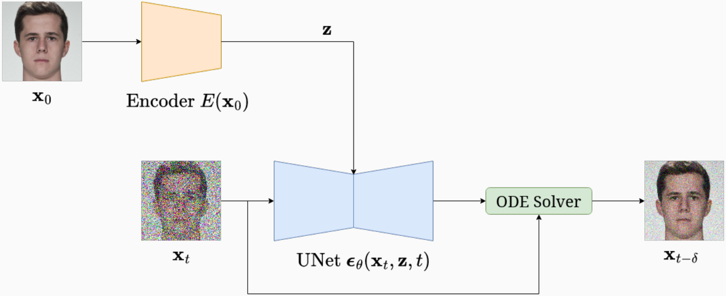
$$\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \approx -\sigma_t \nabla_{\mathbf{x}} \log p_t(\mathbf{x}_t) \tag{8}$$

  or some other closely related quantity like $\mathbf{x}_0$-prediction[3]
- Train a U-Net, $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$, to learn the added noise

$$\hat{\theta} = \arg\min_\theta \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0)}[||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)||_2^2] \tag{9}$$

[3] Tim Salimans and Jonathan Ho. "Progressive Distillation for Fast Sampling of Diffusion Models". In: *International Conference on Learning Representations*. 2022. URL: `https://openreview.net/forum?id=TIdIXIpzhoI`.

Conditional Generation with Diffusion Autoencoders



- Learn an encoder $\mathbf{z} = E(\mathbf{x}_0)$
- Condition the noise prediction model on $\mathbf{z}$
- To get a consistent $\mathbf{x}_T$ run ODE solver in reverse from $\mathbf{x}_0$
- We use Diffusion Autoencoders[4] to create face morphs

[4] Konpat Preechakul et al. "Diffusion Autoencoders: Toward a Meaningful and Decodable Representation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 10619–10629.

Diffusion Morphs (DiM)

Face Morphing with Diffusion

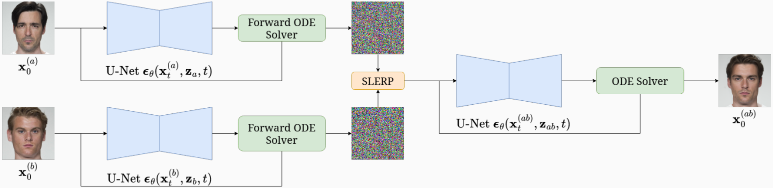


Figure 3: Face morphing pipeline[5]

- Encode bona fide images to get $\mathbf{z}_a, \mathbf{z}_b$
- Encode images by running ODE solver backwards
- Morph encoded images with spherical interpolation to get $\mathbf{x}_T^{(ab)}$
- Morph latent codes with linear interpolation to get $\mathbf{z}_{ab}$
- Run ODE solver to get morphed image

[5] Zander W. Blasingame and Chen Liu. "Leveraging Diffusion for Strong and High Quality Face Morphing Attacks". In: *IEEE Transactions on Biometrics, Behavior, and Identity Science* 6.1 (2024), pp. 118–131. DOI: 10.1109/TBIOM.2024.3349857.

MMPMR

- Mated Morph Presentation Match Rate (MMPMR)[6]
- Measure of vulnerability of an FR system to a morphing attack
- Defined as

$$M(\delta) = \frac{1}{M} \sum_{n=1}^{M} \left\{ \left[\min_{n \in \{1,\ldots,N_m} S_m^n \right] > \delta \right\} \tag{10}$$

where δ is the verification threshold, S_m^n is the similarity score of the n-th subject of morph m, N_m is the total number of contributing subjects to morph m, and M is the total number of morphed images.

[6] Ulrich Scherhag et al. "Biometric Systems under Morphing Attacks: Assessment of Morphing Techniques and Vulnerability Reporting". In: *2017 International Conference of the Biometrics Special Interest Group (BIOSIG)*. 2017, pp. 1–7. DOI: 10.23919/BIOSIG.2017.8053499.

Visual Comparison to Other Morphing Attacks

(a) Identity a **(b)** OpenCV **(c)** StyleGAN2 **(d)** DiM **(e)** MIPGAN-II **(f)** Face-Morpher **(g)** Identity b

(h) Identity c **(i)** OpenCV **(j)** StyleGAN2 **(k)** DiM **(l)** MIPGAN-II **(m)** Face-Morpher **(n)** Identity d

Figure 4: Comparison across different morphing algorithms of two identity pairs from the FRLL dataset.

Quantitative Comparison

Table 1: Vulnerability of different FR systems across different morphing attacks on the SYN-MAD 2022 dataset. FMR $= 0.1\%$.

	MMPMR($\uparrow$)		
Morphing Attack	**AdaFace**	**ArcFace**	**ElasticFace**
FaceMorpher [8]	89.78	87.73	89.57
OpenCV [8]	94.48	92.43	94.27
MIPGAN-I [19]	72.19	77.51	66.46
MIPGAN-II [19]	70.55	72.19	65.24
DiM [5]	92.23	90.18	93.05

Summary - DiM

- High visual fidelity
- Outperforms GAN-based morphs
- Flexible generation due to iterative nature
- Slow inference speed due to multiple iterations
- Greater computational requirements

Fast-DiM

DiM is *Relatively* Slow

- DiM takes 250 Network Function Evaluations (NFE) of the U-Net to preform the encoding step
- DiM take an additional 100 NFE to generate the morphed image
- Fewer NFE will degrade morphing performance in terms of MMPMR
- Can we reduce the NFE while maintaining MMPMR?
- Yes, use high-order ODE solvers[7]

[7] Cheng Lu et al. "DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps". In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo et al. Vol. 35. Curran Associates, Inc., 2022, pp. 5775–5787. URL: `https://proceedings.neurips.cc/paper_files/paper/2022/file/260a14acce2a89dad36adc8eefe7c59e-Paper-Conference.pdf`, Qinsheng Zhang and Yongxin Chen. "Fast Sampling of Diffusion Models with Exponential Integrator". In: *International Conference on Learning Representations*. 2023.

Transforming the Probability Flow ODE

- Performing $\mathbf{x}_0$-prediction over $\boldsymbol{\epsilon}$-prediction can improve performance of diffusion models[8]
- The $\boldsymbol{\epsilon}$-prediction U-Net used in Diffusion Autoencoders and DiM can be transformed into a $\mathbf{x}_0$-prediction network by

$$\mathbf{x}_\theta(\mathbf{x}_t, \mathbf{z}, t) = \frac{\mathbf{x}_t - \sigma_t \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{z}, t)}{\alpha_t} \tag{11}$$

- The empirical probability flow ODE in the VP scenario can be expressed as

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = \left(f(t) + \frac{g^2(t)}{2\sigma_t^2}\right)\mathbf{x}_t - \frac{\alpha_t g^2(t)}{2\sigma_t^2}\mathbf{x}_\theta(\mathbf{x}_t, \mathbf{z}, t) \tag{12}$$

[8] Tim Salimans and Jonathan Ho. "Progressive Distillation for Fast Sampling of Diffusion Models". In: *International Conference on Learning Representations*. 2022. URL: `https://openreview.net/forum?id=TIdIXIpzhoI`.

Higher-order ODE Solver

- Let $\lambda_t \coloneqq \log(\alpha_t/\sigma_t)$ denote one half the log-SNR
- Lu *et al.*[9] show that the empirical probability flow ODE can be rewritten in of λ_t such that

$$\mathbf{x}_t = \frac{\sigma_t}{\sigma_s}\mathbf{x}_s + \sigma_t \int_{\lambda_s}^{\lambda_t} e^{\lambda}\mathbf{x}_\theta(\mathbf{x}_\lambda, \mathbf{z}, \lambda)\,\mathrm{d}\lambda \tag{13}$$

- Take the $(k-1)$-th Taylor expansion at λ_s with step size $h \coloneqq \lambda_t - \lambda_s$

$$\begin{aligned}\mathbf{x}_t &= \frac{\sigma_t}{\sigma_s}\mathbf{x}_s \\ &+ \sigma_t \sum_{n=0}^{k-1} \underbrace{\frac{\mathrm{d}^n}{\mathrm{d}\lambda^n}\Big[\mathbf{x}_\theta(\mathbf{x}_\lambda, \mathbf{z}, \lambda)\Big]_{\lambda=\lambda_s}}_{\text{estimated}} \underbrace{\int_{\lambda_s}^{\lambda_t} e^{\lambda}\frac{(\lambda-\lambda_s)^n}{n!}\,\mathrm{d}\lambda}_{\text{analytically computed}} \\ &+ \underbrace{\mathcal{O}(h^{k+1})}_{\text{omitted}}\end{aligned} \tag{14}$$

[9] Cheng Lu et al. *DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models*. 2023. arXiv: 2211.01095 [cs.LG].

Multi-step Methods

- Multi-step methods like Adams-Bashforth[10] can be used to reduce comptuational overhead and estimated n-th order derivatives
- For the empirical probability flow ODE Lu *et al.*[11] propose a second-order multi-step method
- Assume previous solution $\mathbf{x}_r$ at time $t < r < s$ and let $\rho = \frac{\lambda_r - \lambda_s}{h}$, then the solution $\mathbf{x}_t$ predicted from $\mathbf{x}_s$ is given as

$$\mathbf{D} = \left(1 + \frac{1}{2\rho}\right)\mathbf{x}_\theta(\mathbf{x}_r, \mathbf{z}, r) - \frac{1}{2\rho}\mathbf{x}_\theta(\mathbf{x}_s, \mathbf{z}, s) \tag{15}$$

$$\mathbf{x}_t = \frac{\sigma_t}{\sigma_r}\mathbf{x}_r - \alpha_t(e^{-h} - 1)\mathbf{D} \tag{16}$$

[10] K. Atkinson, W. Han, and D.E. Stewart. *Numerical Solution of Ordinary Differential Equations*. Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts. Wiley, 2011. ISBN: 9781118164525. URL: https://books.google.com/books?id=QzjGgLlKCYQC.

[11] Cheng Lu et al. *DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models*. 2023. arXiv: 2211.01095 [cs.LG].

Impact of ODE Solver on Face Morphing

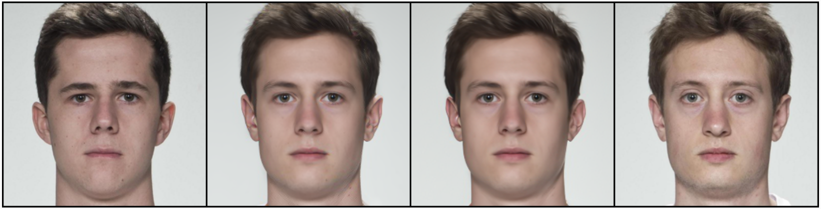


Figure 5: From left to right: identity a, morph generated with DDIM ($N = 100$), morph generated with DPM++ 2M ($N = 20$), identity b.

Table 2: Impact of ODE Solver on the DiM-A algorithm.

		MMPMR(↑)		
ODE Solver	**NFE(↓)**	**AdaFace**	**ArcFace**	**ElasticFace**
DDIM	100	92.23	90.18	93.05
DPM++ 2M	50	92.02	90.18	93.05
DPM++ 2M	20	91.62	89.98	93.25

How Important is the Initial Noise

- The Diffusion Autoencoder uses the conditional information $\mathbf{z}$ and initial noise $\mathbf{x}_T$ to generate an image
- How important is the initial noise $\mathbf{x}_T$ in creating a face morph?
- Specifically, do we need to encode the bona fide images into $\mathbf{x}_T^{(a)}$ and $\mathbf{x}_T^{(b)}$?

Encoded Noise vs White Noise

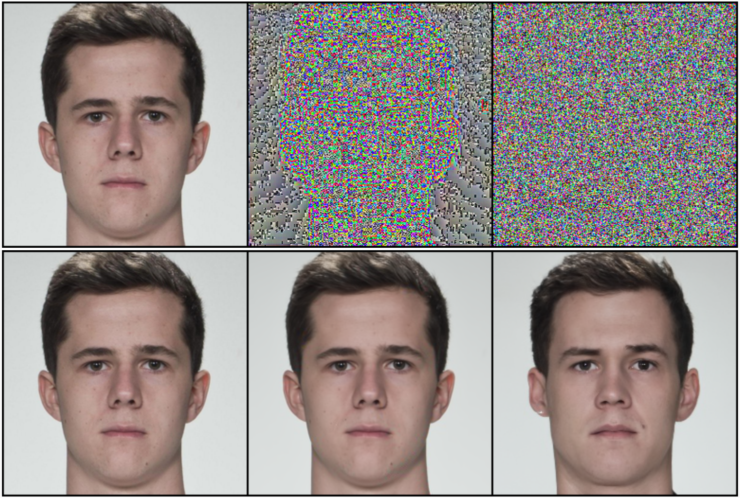


Figure 6: From top left to lower right: original image, output from the DiffAE forward solver, white noise, original image, DDIM sampled image from DiffAE approach, DDIM sampled image from pure white noise.

Sampling from a Noisy Representation

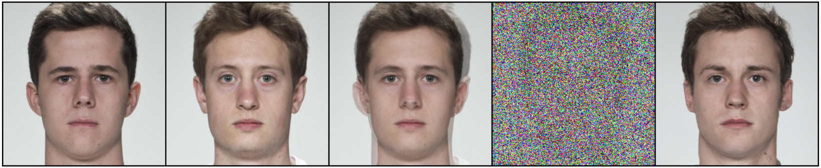


Figure 7: From left to right: identity a, identity b, pixel-wise averaged image, noisy image, final morphed image.

Table 3: Amount of added noise versus MMPMR (↑) using the DPM++ 2M solver with $N = 50$.

Noise Level	AdaFace	ArcFace	ElasticFace
1.0	4.5	3.48	2.04
0.6	9.41	6.75	4.91
0.5	15.13	12.27	9.41
0.4	27.61	21.68	21.27
0.3	45.81	40.49	37.01

Solving the Forward ODE

- Preechakul *et al.*[12] the authors simply rearrange the DDIM update equation to obtain

$$\mathbf{x}_{t_i} = \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}}\left(\mathbf{x}_{t_{i-1}} + \alpha_{t_{i-1}}(e^{h_i} - 1)\mathbf{x}_\theta(\mathbf{x}_{t_i}, \mathbf{z}, t_i)\right) \tag{17}$$

  for some timesteps $t_{i-1} < t_i$ where $\mathbf{x}_{t_{i-1}}$ is the current and known sample
- Can't be used as the U-Net depends on $\mathbf{x}_{t_i}$, Preechakul *et al.* instead simply evaluate the U-Net on $\mathbf{x}_{t_{i-1}}$ yielding

$$\mathbf{x}_{t_i} = \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}}\left(\mathbf{x}_{t_{i-1}} + \alpha_{t_{i-1}}(e^{h_i} - 1)\mathbf{x}_\theta(\mathbf{x}_{t_{i-1}}, \mathbf{z}, t_{i-1})\right) \tag{18}$$

- Is this a reasonably substitution?
- But how much does this error impact small N?

[12] Konpat Preechakul et al. "Diffusion Autoencoders: Toward a Meaningful and Decodable Representation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 10619–10629.

Impact of Forward ODE Solver on Autoencoding

Table 4: Study of the effects on autoencoding reconstruction quality across different forward ODE solvers on the FRLL dataset. Sampling is done with the DPM++ 2M solver with $N = 20$ steps.

	LPIPS(↓)			
Forward ODE Solver	$N_F = 20$	$N_F = 50$	$N_F = 100$	$N_F = 250$
DiffAE	**0.2370**	0.1404	0.1211	0.1113
DDIM	0.2953	0.1843	0.1173	0.0760
DPM++ 2M	0.3247	**0.1159**	**0.1120**	**0.0752**

Impact of Forward ODE Solver on Face Morphing

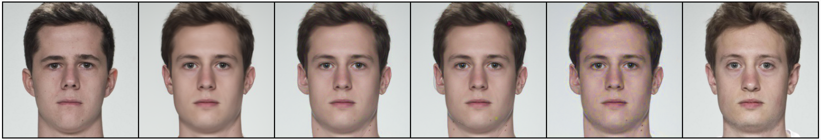


Figure 8: From left to right: identity a, DiffAE forward solver $N_F = 250$, DDIM forward ODE solver $N_F = 100$, DPM++ 2M forward ODE solver $N_F = 100$, DPM++ 2M forward ODE solver $N_F = 50$, and identity b.

Table 5: Impact of forward ODE Solver on MMPMR.

		MMPMR(↑)		
ODE Solver	**NFE(↓)**	**AdaFace**	**ArcFace**	**ElasticFace**
DiffAE	250	92.02	90.18	93.05
DDIM	100	91.82	88.75	91.21
DPM++ 2M	100	90.59	87.12	90.8
DDIM	50	89.78	86.3	89.37
DPM++ 2M	50	90.18	86.5	88.96

Summary - Fast-DiM

- The NFE can be reduced by using higher-order ODE solvers for sampling with little to no reduction in performance
- For a small decrease in performance the NFE can be greatly reduced by using higher-order ODE solvers for encoding
- The initial noises $\mathbf{x}_T^{(a)}, \mathbf{x}_T^{(b)}$ and morphed noise $\mathbf{x}_T^{(ab)}$ are *exceedingly* important to creating high quality morphs[13]

[13] Zander W. Blasingame and Chen Liu. "Fast-DiM: Towards Fast Diffusion Morphs". In: *IEEE Security & Privacy* (2024), pp. 2–13. DOI: 10.1109/MSEC.2024.3410112.

Greedy-DiM

Guided Optimization for Morphing

- MIPGAN[14] showed the power in using guided optimization
- MIPGAN far outperforms the unguided GAN architecture
- Can we do this for DiMs?
- It is difficult to find the optimal $\mathbf{x}_T^{(ab)}$ and $\mathbf{x}_{ab}$ in DiMs
- Morph-PIPE solves this via brute force search[15]
- Can we do better?

[14] Haoyu Zhang et al. "MIPGAN—Generating Strong and High Quality Morphing Attacks Using Identity Prior Driven GAN". In: *IEEE Transactions on Biometrics, Behavior, and Identity Science* 3.3 (2021), pp. 365–383. DOI: 10.1109/TBIOM.2021.3072349.

[15] Haoyu Zhang et al. "Morph-PIPE: Plugging in Identity Prior to Enhance Face Morphing Attack Based on Diffusion Model". In: *Norwegian Information Security Conference (NISK)*. 2023.

Yes, by being greedy

Greedy-DiM

Table 6: Comparison of existing DiM methods in the literature and our proposed algorithm.

	DiM [5]	**Fast-DiM [3]**	**Morph-PIPE [20]**	**Ours (Greedy-DiM)**
ODE solver	DDIM	DPM++ 2M	DDIM	DDIM
Forward ODE solver	DiffAE	DDIM	DiffAE	DiffAE
Number of sampling steps	100	50	2100	20
Heuristic function	✗	✗	$\mathcal{L}_{ID}^*$	$\mathcal{L}_{ID}^*$
Search strategy	✗	✗	Brute-force search	Greedy optimization
Search space	$\emptyset$	$\emptyset$	Set of 21 blend values	Image space
Optimal solution in search space	✗	✗	0	1

- Greedily search for the optimal ϵ at each time step which minimizes the identity loss defined as the sum of two sub-losses:

$$\mathcal{L}_{ID} = d(v_{ab}, v_a) + d(v_{ab}, v_b) \tag{19}$$

$$\mathcal{L}_{diff} = \Big| d(v_{ab}, v_a) - d(v_{ab}, v_b)) \Big| \tag{20}$$

$$\mathcal{L}_{ID}^* = \mathcal{L}_{ID} + \mathcal{L}_{diff} \tag{21}$$

where $v_a = F(\mathbf{x}_0^{(a)}), v_b = F(\mathbf{x}_0^{(b)}), v_{ab} = F(\mathbf{x}_0^{(ab)})$, and $F : \mathcal{X} \to V$ is an FR system which embeds images into a vector space V which is equipped with a measure of distance, d.

Greedy-DiM-S

Table 7: Comparison of search strategies with the identity loss as the heuristic function.

		MMPMR(↑)		
Search Strategy	**NFE(↓)**	**AdaFace**	**ArcFace**	**ElasticFace**
None	350	92.23	90.18	93.05
Brute-force	2350	**95.91**	92.84	**95.5**
Greedy	350	95.71	**93.87**	95.3

Table 8: Greedy search over $\{w_n\}_{n=1}^{21} \subseteq [0,1]$ vs greedy gradient descent over $[0,1]$.

	MMPMR(↑)		
Search Space	**AdaFace**	**ArcFace**	**ElasticFace**
$\{w_n\}_{n=1}^{21} \subseteq [0,1]$	95.71	93.87	95.3
$[0,1]$	95.5	94.07	95.09

Greedy-DiM*

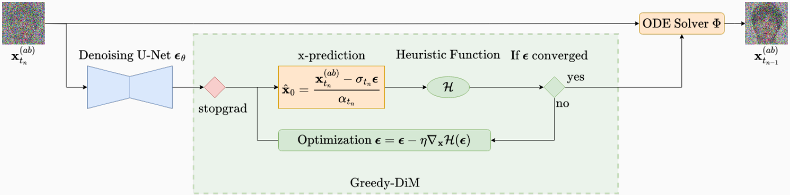


Figure 9: Overview of a single step of the Greedy-DiM* algorithm. Proposed changes highlighted in green.

- Perform gradient descent to find the optimal ϵ at each time step

Visual Results

(a) Identity a (b) DiM-A (c) Fast-DiM (d) Morph-PIPE (e) Greedy-DiM-S (f) Greedy-DiM* (g) Identity b

Figure 10: Comparison of DiM morphs on the FRLL dataset.

Results

Table 9: Vulnerability of different FR systems across different morphing attacks on the SYN-MAD 2022 dataset. FMR = 0.1%.

		MMPMR(↑)		
Morphing Attack	**NFE(↓)**	**AdaFace**	**ArcFace**	**ElasticFace**
FaceMorpher [8]	-	89.78	87.73	89.57
Webmorph [8]	-	97.96	96.93	98.36
OpenCV [8]	-	94.48	92.43	94.27
MIPGAN-I [19]	-	72.19	77.51	66.46
MIPGAN-II [19]	-	70.55	72.19	65.24
DiM-A [5]	350	92.23	90.18	93.05
DiM-C [5]	350	89.57	83.23	86.3
Fast-DiM [3]	300	92.02	90.18	93.05
Fast-DiM-ode [3]	150	91.82	88.75	91.21
Morph-PIPE [20]	2350	95.91	92.84	95.5
Greedy-DiM-S	350	95.71	93.87	95.3
Greedy-DiM*	270	**100**	**100**	**100**

Justification of the Unreasonably Good Performance

Theorem (Optimality of Greedy-DiM*)

Given a sequence of monotonically descending timesteps, $\{t_n\}_{n=1}^{N}$, from T to 0, the DDIM solver to the Probability Flow ODE, and a heuristic function $\mathcal{H}$, the locally optimal solution admitted by Greedy-DiM at time t_n is globally optimal.*

Theorem (Search Space of Greedy-DiM* is well-posed)

Let $\mathbb{P}$ be a probability distribution on a compact subset $\mathcal{X} \subseteq \mathbb{R}^n$ with full support on $\mathcal{X}$ which models the distribution of the optimal $\mathbf{x}_0^*$ and is absolutely continuous w.r.t. the n-dimensional Lebesgue measure λ^n on $\mathcal{X}$. Let $\mathcal{S}_P, \mathcal{S}_S, \mathcal{S}^*$ denote the search spaces of the Morph-PIPE, Greedy-DiM-S, and Greedy-DiM algorithms. Then the following statements are true.*

1. $\mathbb{P}(\mathcal{S}_P) = \mathbb{P}(\mathcal{S}_S) = 0.$
2. $\mathbb{P}(\mathcal{S}^*) = 1.$

Visual Justification

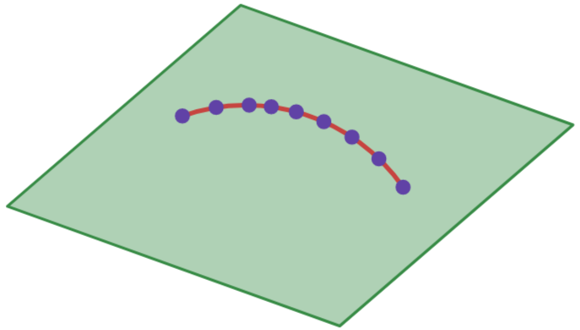


Figure 11: Illustration of the search space in $\mathbb{R}^2$ of different DiM algorithms at a single step. Purple denotes Morph-PIPE/Greedy-DiM-S, red denotes Greedy-DiM-S continuous, and green denotes Greedy-DiM*. Note the search spaces of the algorithms other than Greedy-DiM* lie in a low dimensional manifold.

Summary - Greedy-DiM

- SOTA performance on SYN-MAD 2022 dataset
- Adds only a little overhead to vanilla DiM
- Guiding heuristic $\mathcal{H}$ can be swapped for another differentiable function

AdjointDEIS

Backpropagation through Diffusion Models

- Suppose we have a differentiable loss function $\mathcal{L}$ operating on the output of a diffusion model
- How do we calculate $\partial\mathcal{L}/\partial\mathbf{x}_T$, $\partial\mathcal{L}/\partial\mathbf{z}$, and $\partial\mathcal{L}/\partial\theta$?
- Naïvely, performing backpropagation throughout the iterative calls of the U-Net is memory intensive
- Diffusion models can be thought of as a type of Neural ODE[16]
- Can we use the method of adjoint sensitivity[17]?

[16] Ricky T. Q. Chen et al. "Neural Ordinary Differential Equations". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf.

[17] Lev Semenovich Pontryagin et al. "The Mathematical Theory of Optimal Processes.". In: *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik* 43.10-11 (1963), pp. 514–515. DOI: https://doi.org/10.1002/zamm.19630431023. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/zamm.19630431023. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/zamm.19630431023.

Method of Adjoint Sensitivity

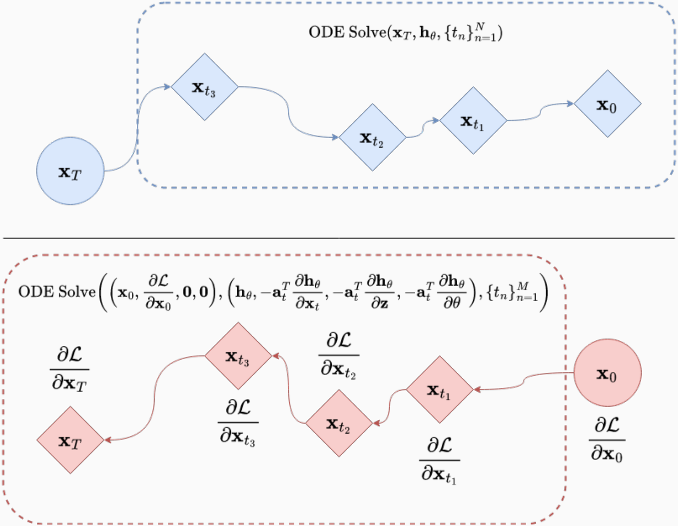


Figure 12: Overview of Adjoint Diffusion ODE

Method of Adjoint Sensitivity

- Let $\mathbf{h}_\theta$ denote the empirical probability flow ODE

$$\mathbf{h}_\theta(\mathbf{x}_t, \mathbf{z}, t) = f(t)\mathbf{x}_t + \frac{g^2(t)}{2\sigma_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{z}, t) \tag{22}$$

- Let $\mathbf{a}_t \coloneqq \partial\mathcal{L}/\partial\mathbf{x}_t$ denote the adjoint state, the dynamics of which evolve with another ODE

$$\frac{\mathrm{d}\mathbf{a}_t}{\mathrm{d}t} = -\mathbf{a}_t^\top \frac{\partial\mathbf{h}_\theta(\mathbf{x}_t, \mathbf{z}, t)}{\partial\mathbf{x}_t} \tag{23}$$

as time flows forwards from 0 to T

- The gradients of the loss w.r.t. $\mathbf{z}$ and θ are expressed as the solution to another set of ODEs

$$\frac{\partial\mathcal{L}}{\partial\mathbf{z}} = -\int_0^T \mathbf{a}_t^\top \frac{\partial\mathbf{h}_\theta(\mathbf{x}_t, \mathbf{z}, t)}{\partial\mathbf{z}} \tag{24}$$

$$\frac{\partial\mathcal{L}}{\partial\theta} = -\int_0^T \mathbf{a}_t^\top \frac{\partial\mathbf{h}_\theta(\mathbf{x}_t, \mathbf{z}, t)}{\partial\theta} \tag{25}$$

AdjointDEIS

- Adjoint Diffusion Exponential Integrator Sampler - AdjointDEIS
- Use exponential integrators to solve adjoint diffusion ODEs
- Given an initial value $\partial \mathcal{L}/\partial \mathbf{x}_0$ we can calculate *all* the gradients by solving the adjoint diffusion ODEs
- Let $\mathbf{y}_t = e^{-\int_0^t f(\tau)\,\mathrm{d}\tau}\mathbf{x}_t$ and let $\hat{\mathbf{a}}_t \coloneqq \partial \mathcal{L}/\partial \mathbf{y}_t$
- Given an initial value $\hat{\mathbf{a}}_t$ at time $t \in [0, T]$, the solution $\hat{\mathbf{a}}_s$ at time $s \in (t, T]$ is

$$\hat{\mathbf{a}}_s = \hat{\mathbf{a}}_t + \alpha_0 \int_{\lambda_t}^{\lambda_s} e^{-\lambda} \hat{\mathbf{a}}_\lambda^\top \frac{\partial \boldsymbol{\epsilon}_\theta(\frac{\alpha_\lambda}{\alpha_0}\mathbf{y}_\lambda, \mathbf{z}, \lambda)}{\partial \mathbf{y}_\lambda}\,\mathrm{d}\lambda \tag{26}$$

AdjointDEIS-1

- Taking the $(k-1)$-th Taylor expansion at λ_t yields

$$\hat{\mathbf{a}}_s = \hat{\mathbf{a}}_t + \alpha_0 \sum_{n=0}^{k-1} \underbrace{\frac{\mathrm{d}^n}{\mathrm{d}\lambda^n}\left[\hat{\mathbf{a}}_\lambda^\top \frac{\partial \boldsymbol{\epsilon}_\theta(\frac{\alpha_\lambda}{\alpha_0}\mathbf{y}_\lambda, \mathbf{z}, \lambda)}{\partial \mathbf{y}_\lambda}\right]_{\lambda=\lambda_t}}_{\text{estimated}} \underbrace{\int_{\lambda_t}^{\lambda_s} \frac{(\lambda-\lambda_t)^n}{n!} e^{-\lambda}\,\mathrm{d}\lambda}_{\text{analytically computed}} + \underbrace{\mathcal{O}(h^{k+1})}_{\text{omitted}} \tag{27}$$

  where $h \coloneqq \lambda_s - \lambda_t$
- With $k = 1$ we have

$$\hat{\mathbf{a}}_s = \hat{\mathbf{a}}_t + \alpha_0 \frac{\sigma_s}{\alpha_s}(e^h - 1)\hat{\mathbf{a}}_t^\top \frac{\partial \boldsymbol{\epsilon}_\theta(\frac{\alpha_t}{\alpha_0}\mathbf{y}_t, \mathbf{z}, t)}{\partial \mathbf{y}_t} \tag{28}$$

AdjointDEIS-1

- Switching back to $\mathbf{x}_t$ yields[18]

$$\mathbf{a}_s = \frac{\alpha_t}{\alpha_s}\mathbf{a}_t + \alpha_t^2\sigma_s(e^h - 1)\mathbf{a}_t^\top \frac{\partial \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{z}, t)}{\partial \mathbf{x}_t} \tag{29}$$

- Likewise for the integrals for the gradients w.r.t. $\mathbf{z}$ and θ

$$\Gamma_{t,s}^{\mathbf{z}}\left(\frac{\partial \mathcal{L}}{\partial \mathbf{z}}\right) = \frac{\partial \mathcal{L}}{\partial \mathbf{z}} + \alpha_t^2\sigma_s(e^h - 1)\mathbf{a}_t^\top \frac{\partial \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{z}, t)}{\partial \mathbf{z}} \tag{30}$$

$$\Gamma_{t,s}^{\theta}\left(\frac{\partial \mathcal{L}}{\partial \theta}\right) = \frac{\partial \mathcal{L}}{\partial \theta} + \alpha_t^2\sigma_s(e^h - 1)\mathbf{a}_t^\top \frac{\partial \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{z}, t)}{\partial \theta} \tag{31}$$

- A similar approach to Lu *et al.*[19] is used to construct AdjointDEIS-2M

[18] Zander W. Blasingame and Chen Liu. "AdjointDEIS: Efficient Gradients for Diffusion Models". In: *arXiv e-prints*, arXiv:2405.15020 (May 2024), arXiv:2405.15020. arXiv: 2405.15020 [cs.CV].

[19] Cheng Lu et al. *DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models*. 2023. arXiv: 2211.01095 [cs.LG].

What about SDEs?

- Diffusion SDEs can be more useful than ODEs for image editing[20]
- Can we use AdjointDEIS-1 to calculate gradients for diffusion SDEs?
- No!
- The adjoint method requires solving the dynamics backwards from the end state $\mathbf{x}_0$
- Reversing an Itô SDE in time is not trivial!
- Use the backwards Stratonovich SDE to calculate adjoint flow[21]

[20] Shen Nie et al. "The Blessing of Randomness: SDE Beats ODE in General Diffusion-based Image Editing". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=DesYwmUG00.

[21] Xuechen Li et al. "Scalable Gradients for Stochastic Differential Equations". In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, 26–28 Aug 2020, pp. 3870–3882. URL: https://proceedings.mlr.press/v108/li20i.html.

Stochastic Flow

- There exists a smooth mapping Φ such that $\Phi_{s,t}(\mathbf{x}_s)$ is the solution at time t for the process starting at $\mathbf{x}_s$ at time $s \geq t$ for

$$\mathrm{d}\mathbf{x}_t = \left[f(t)\mathbf{x}_t + \frac{g^2(t)}{\sigma_t} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{z}, t) \right] \mathrm{d}t + g(t) \circ \mathrm{d}\breve{\mathbf{w}}_t \tag{32}$$

- Φ is known as the *stochastic flow*
- These maps are diffeomorphisms which satisfy the flow property

$$\Phi_{s,t}(\mathbf{x}_s) = \Phi_{u,t}(\Phi_{s,u}(\mathbf{x}_s)) \quad s \leq u \leq t, \mathbf{x}_s \in \mathcal{X} \tag{33}$$

- There exists a backwards flow $\breve{\Psi}_{s,t} \coloneqq \Phi_{s,t}^{-1}$ which satisfies

$$\breve{\Psi}_{s,t}(\mathbf{x}_t) = \mathbf{x}_t - \int_s^t f(t)\breve{\Psi}_{u,t}(\mathbf{x}_t) + \frac{g^2(u)}{\sigma_u} \boldsymbol{\epsilon}_\theta(\breve{\Psi}_{u,t}(\mathbf{x}_t), \mathbf{z}, u) \,\mathrm{d}u - \int_s^t g(u) \circ \mathrm{d}\mathbf{w}_u \tag{34}$$

Adjoint Diffusion SDE

- Let $\mathbf{A}_{s,t}(\mathbf{x}_s) = \partial\mathcal{L}(\Phi_{s,t}(\mathbf{x}_s))/\partial\mathbf{x}_s$ denote the *adjoint flow*
- Let the backwards adjoint flow be denoted by $\breve{\mathbf{A}}_{s,t}(\mathbf{x}_t) \coloneqq \mathbf{A}_{s,t}(\breve{\Psi}_{s,t}(\mathbf{x}_t))$
- The backwards adjoint flow satisfies the Stratonovich SDE

$$\begin{aligned}\breve{\mathbf{A}}_{s,t}(\mathbf{x}_t) = \nabla_{\mathbf{x}_t}\mathcal{L}(\mathbf{x}_t) \\ + \int_s^t \breve{\mathbf{A}}_{u,t}(\mathbf{x}_t)\nabla_{\mathbf{x}_u}\left[f(t)\breve{\Psi}_{u,t}(\mathbf{x}_t) + \frac{g^2(u)}{\sigma_u}\boldsymbol{\epsilon}_\theta(\breve{\Psi}_{u,t}(\mathbf{x}_t), \mathbf{z}, u)\right]\mathrm{d}u \\ + \underbrace{\int_s^t \breve{\mathbf{A}}_{u,t}(\mathbf{x}_t)\nabla_{\mathbf{x}_u} g(u) \circ \mathrm{d}\mathbf{w}_u}_{\text{becomes zero!}}\end{aligned} \tag{35}$$

- Adjoint Diffusion SDE simplifies to an ODE!

SDE-AdjointDEIS-1

- Similar to the AdjointDEIS-1 the first-order solvers become

$$\mathbf{a}_s = \frac{\alpha_t}{\alpha_s}\mathbf{a}_t + 2\alpha_t^2\sigma_s(e^h - 1)\mathbf{a}_t^\top \frac{\partial \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{z}, t)}{\partial \mathbf{x}_t} \tag{36}$$

$$\Gamma_{t,s}^{\mathbf{z}}\left(\frac{\partial \mathcal{L}}{\partial \mathbf{z}}\right) = \frac{\partial \mathcal{L}}{\partial \mathbf{z}} + 2\alpha_t^2\sigma_s(e^h - 1)\mathbf{a}_t^\top \frac{\partial \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{z}, t)}{\partial \mathbf{z}} \tag{37}$$

$$\Gamma_{t,s}^{\theta}\left(\frac{\partial \mathcal{L}}{\partial \theta}\right) = \frac{\partial \mathcal{L}}{\partial \theta} + 2\alpha_t^2\sigma_s(e^h - 1)\mathbf{a}_t^\top \frac{\partial \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{z}, t)}{\partial \theta} \tag{38}$$

- N.B., the backwards flow is still an SDE

Solving the Backwards Diffusion SDE

- Lu *et al.*[22] propose a first-order solver for the diffusion SDE

$$\mathbf{x}_t = \frac{\alpha_t}{\alpha_s}\mathbf{x}_s - 2\sigma_t(e^h - 1)\boldsymbol{\epsilon}_\theta(\mathbf{x}_s, \mathbf{z}, s) + \sigma_t\sqrt{e^{2h} - 1}\boldsymbol{\epsilon}_s \tag{39}$$

  where $\boldsymbol{\epsilon}_s \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $h = \lambda_t - \lambda_s$

- Wu and De la Torre[23] propose to solve for $\boldsymbol{\epsilon}_s$ by rearranging the SDE solver

$$\boldsymbol{\epsilon}_s = \frac{\mathbf{x}_t - \frac{\alpha_t}{\alpha_s}\mathbf{x}_s + 2\sigma_t(e^h - 1)\boldsymbol{\epsilon}_\theta(\mathbf{x}_s, \mathbf{z}, s)}{\sigma_t\sqrt{e^{2h} - 1}} \tag{40}$$

- This technique is referred to as *Cycle-SDE*

[22] Cheng Lu et al. *DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models*. 2023. arXiv: `2211.01095 [cs.LG]`.

[23] Chen Henry Wu and Fernando De la Torre. "A Latent Space of Stochastic Diffusion Models for Zero-Shot Image Editing and Guidance". In: *ICCV*. 2023.

Application - Face Morphing

(a) Identity a (b) AdjointDEIS (c) SDE-AdjointDEIS (d) Identity b

Figure 13: Example of DiM + AdjointDEIS

- We can use AdjointDEIS by optimizing $\mathbf{x}_T$ and $\mathbf{z}$ w.r.t. the identity loss $\mathcal{L}^*_{ID}$
- Note the more realistic skin texture using the SDE over ODE solver

DiM + AdjointDEIS - Visual Results

Figure 14: Comparison of DiM morphs on the FRLL dataset. From left to right, identity a, DiM-A, Fast-DiM, Morph-PIPE, AdjointDEIS, SDE-AdjointDEIS, and identity b.

DiM + AdjointDEIS - Vulnerability

Table 10: Vulnerability of different FR systems across different morphing attacks on the SYN-MAD 2022 dataset. FMR = 0.1%.

		MMPMR(↑)		
Morphing Attack	**NFE(↓)**	**AdaFace**	**ArcFace**	**ElasticFace**
DiM-A [5]	350	92.23	90.18	93.05
Fast-DiM [3]	300	92.02	90.18	93.05
Morph-PIPE [20]	2350	95.91	92.84	95.5
DiM + AdjointDEIS	1250	**96.32**	93.25	**96.32**
DiM + SDE-AdjointDEIS	750	95.71	**94.68**	**96.32**

Summary - AdjointDEIS

- New technique to calculate gradients of diffusion models w.r.t. a differentiable function
- Adjoint ODE solver is decoupled from sampling ODE solver
- Gradients can be calculated for Adjoint SDE
- SOTA face morphing performance on SYN-MAD 2022 dataset
- AdjointDEIS can be applied to many different problems with a differentiable loss function

Conclusion

Summary

DiM[24] Face morphing with diffusion models
Fast-DiM[25] Higher-order ODE solvers for faster sampling
Greedy-DiM[26] Greedy optimization for more potent morphs
AdjointDEIS[27] Efficient gradients for diffusion ODEs/SDEs

[24] Zander W. Blasingame and Chen Liu. "Leveraging Diffusion for Strong and High Quality Face Morphing Attacks". In: *IEEE Transactions on Biometrics, Behavior, and Identity Science* 6.1 (2024), pp. 118–131. DOI: 10.1109/TBIOM.2024.3349857.

[25] Zander W. Blasingame and Chen Liu. "Fast-DiM: Towards Fast Diffusion Morphs". In: *IEEE Security & Privacy* (2024), pp. 2–13. DOI: 10.1109/MSEC.2024.3410112.

[26] Zander W. Blasingame and Chen Liu. "Greedy-DiM: Greedy Algorithms for Unreasonably Effective Face Morphs". In: *arXiv e-prints*, arXiv:2404.06025 (Apr. 2024), arXiv:2404.06025. DOI: 10.48550/arXiv.2404.06025. arXiv: 2404.06025 [cs.CV].

[27] Zander W. Blasingame and Chen Liu. "AdjointDEIS: Efficient Gradients for Diffusion Models". In: *arXiv e-prints*, arXiv:2405.15020 (May 2024), arXiv:2405.15020. arXiv: 2405.15020 [cs.CV].

?

References

[1] K. Atkinson, W. Han, and D.E. Stewart. *Numerical Solution of Ordinary Differential Equations*. Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts. Wiley, 2011. ISBN: 9781118164525. URL: https://books.google.com/books?id=QzjGgLlKCYQC.

[2] Zander W. Blasingame and Chen Liu. "AdjointDEIS: Efficient Gradients for Diffusion Models". In: *arXiv e-prints*, arXiv:2405.15020 (May 2024), arXiv:2405.15020. arXiv: 2405.15020 [cs.CV].

[3] Zander W. Blasingame and Chen Liu. "Fast-DiM: Towards Fast Diffusion Morphs". In: *IEEE Security & Privacy* (2024), pp. 2–13. DOI: 10.1109/MSEC.2024.3410112.

[4] Zander W. Blasingame and Chen Liu. "Greedy-DiM: Greedy Algorithms for Unreasonably Effective Face Morphs". In: *arXiv e-prints*, arXiv:2404.06025 (Apr. 2024), arXiv:2404.06025. DOI: 10.48550/arXiv.2404.06025. arXiv: 2404.06025 [cs.CV].

[5] Zander W. Blasingame and Chen Liu. "Leveraging Diffusion for Strong and High Quality Face Morphing Attacks". In: *IEEE Transactions on Biometrics, Behavior, and Identity Science* 6.1 (2024), pp. 118–131. DOI: 10.1109/TBIOM.2024.3349857.

[6] Ricky T. Q. Chen et al. "Neural Ordinary Differential Equations". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf.

[7] Lisa DeBruine and Benedict Jones. "Face Research Lab London Set". In: (May 2017). DOI: 10.6084/m9.figshare.5047666.v5. URL: https://figshare.com/articles/dataset/Face_Research_Lab_London_Set/5047666.

[8] Marco Huber et al. "SYN-MAD 2022: Competition on Face Morphing Attack Detection Based on Privacy-aware Synthetic Training Data". In: *2022 IEEE International Joint Conference on Biometrics (IJCB)*. 2022, pp. 1–10. DOI: 10.1109/IJCB54206.2022.10007950.

[9] Xuechen Li et al. "Scalable Gradients for Stochastic Differential Equations". In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, 26–28 Aug 2020, pp. 3870–3882. URL: https://proceedings.mlr.press/v108/li20i.html.

References iv

[10] Cheng Lu et al. "DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps". In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo et al. Vol. 35. Curran Associates, Inc., 2022, pp. 5775–5787. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/260a14acce2a89dad36adc8eefe7c59e-Paper-Conference.pdf.

[11] Cheng Lu et al. *DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models*. 2023. arXiv: 2211.01095 [cs.LG].

[12] Shen Nie et al. "The Blessing of Randomness: SDE Beats ODE in General Diffusion-based Image Editing". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=DesYwmUG00.

[13] Lev Semenovich Pontryagin et al. "The Mathematical Theory of Optimal Processes.". In: *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik* 43.10-11 (1963), pp. 514–515. DOI: https://doi.org/10.1002/zamm.19630431023. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/zamm.19630431023. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/zamm.19630431023.

[14] Konpat Preechakul et al. "Diffusion Autoencoders: Toward a Meaningful and Decodable Representation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 10619–10629.

[15] Tim Salimans and Jonathan Ho. "Progressive Distillation for Fast Sampling of Diffusion Models". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=TIdIXIpzhoI.

[16] Ulrich Scherhag et al. "Biometric Systems under Morphing Attacks: Assessment of Morphing Techniques and Vulnerability Reporting". In: *2017 International Conference of the Biometrics Special Interest Group (BIOSIG)*. 2017, pp. 1–7. DOI: 10.23919/BIOSIG.2017.8053499.

[17] Yang Song et al. "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=PxTIG12RRHS.

[18] Chen Henry Wu and Fernando De la Torre. "A Latent Space of Stochastic Diffusion Models for Zero-Shot Image Editing and Guidance". In: *ICCV*. 2023.

[19] Haoyu Zhang et al. "MIPGAN—Generating Strong and High Quality Morphing Attacks Using Identity Prior Driven GAN". In: *IEEE Transactions on Biometrics, Behavior, and Identity Science* 3.3 (2021), pp. 365–383. DOI: 10.1109/TBIOM.2021.3072349.

[20] Haoyu Zhang et al. "Morph-PIPE: Plugging in Identity Prior to Enhance Face Morphing Attack Based on Diffusion Model". In: *Norwegian Information Security Conference (NISK)*. 2023.

[21] Qinsheng Zhang and Yongxin Chen. "Fast Sampling of Diffusion Models with Exponential Integrator". In: *International Conference on Learning Representations*. 2023.

Itô vs Stratonovich

- Itô integral of a continuous semi-martingale $\{X_t\}_{t\in[0,T]}$ adapted to the forward filtration generated by the Wiener process $\{W_t\}_{t\in[0,T]}$ on the real interval $[s,t] \subseteq [0,T]$ is defined as

$$\int_s^t X_t \,\mathrm{d}W_t = \lim_{|\Pi|\to 0} \sum_{k=1}^{N} X_{t_{k-1}}(W_{t_k} - W_{t_{k-1}}) \tag{41}$$

  where Π is the N-part partition of the real interval $[s,t]$, and $|\Pi| = \max_k t_k - t_{k-1}$ denotes the norm of the partition.
- Stratonovich Integral

$$\int_s^t X_t \circ \mathrm{d}W_t = \lim_{|\Pi|\to 0} \sum_{k=1}^{N} \frac{X_{t_k} - X_{t_{k-1}}}{2}(W_{t_k} - W_{t_{k-1}}) \tag{42}$$