Modeling GAN Latent Dynamics using Neural ODEs

Weihao Xia
University College London

Yujiu Yang
Tsinghua University

Jing-Hao Xue
University College London

Abstract

In this paper, we propose DynODE, a method to model the video dynamics by learning the trajectory of independently inverted latent codes from GANs. The entire sequence is seen as discrete-time observations of a continuous trajectory of the initial latent code. The latent codes representing different frames are therefore reformulated as state transitions of the initial frame, which can be modeled by neural ordinary differential equations. Our DynODE learns the holistic geometry of the video dynamic space from given sparse observations and specifies continuous latent states, allowing us to engage in various video applications such as frame interpolation and video editing. Extensive experiments demonstrate that our method achieves state-of-the-art performance but with much less computation. Code is available at https://github.com/weihaox/dynode_released.

1 Introduction

GAN inversion [1, 38, 29] allows us to invert images into the latent space of pretrained GAN models, facilitating further attribute editing of these images [22, 11]. Recent GAN-inversion based video editing methods [34, 24] have demonstrated that even using a non-temporal StyleGAN [9, 10, 8], the temporal consistencies can be preserved through identical operations across all frames. However, applying the same operations to each frame is redundant; more importantly, the temporal relationships of the independently-inverted latent codes in a GAN's latent space are not exploited.

In this paper, we present DynODE, a method to model the latent Dynamics with neural ordinary differential equations (ODEs) [2]. Our work borrows intuition from dynamical systems and treats the video dynamics as the solution to a first-order non-autonomous ODE. To be specific, if considering the latent space as a dynamical system, the changes in latent codes along a certain direction can be likened to the trajectory of a moving particle. The non-temporal latent codes (in the latent space) becomes discrete-time observations of a continuous trajectory of the initial latent state (in the dynamic space). The subsequent latent codes are reformulated as state transitions from the initial one. The entire video is therefore determined by the first frame and its temporal trajectory.

Given sparse observations, our method is encouraged to learn and recover the holistic geometry of the video dynamic space. Our DynODE specifies continuous latent states. This design allows time-oriented and motion-coherent frame interpolation at unseen timesteps and liberates video editing from the laborious and repetitive frame-by-frame processing, demonstrating the benefits of modeling the latent dynamics. Our contributions are summarized as follows: (1) we model the video dynamics by learning the temporal trajectory of the non-temporal latent codes with neural ODEs; (2) we present a dynamical view of the GAN latent space, in which video frames are seen as discrete-time observations of a continuous trajectory of the initial latent code; (3) the learned video dynamics facilitate various video applications such as frame interpolation and video editing.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

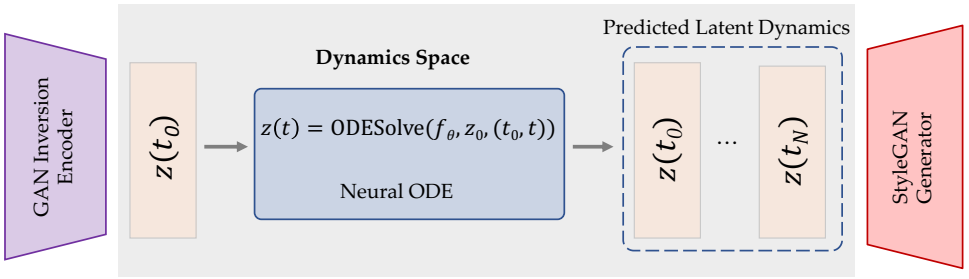


Figure 1: Our goal with DynODE is to model latent dynamics of GANs (*e.g.* StyleGANs [10]) using a neural ODE. The learned continuous trajectory is then used for frame interpolation and video editing.

2 Background

GAN Inversion [1, 38] aims to invert a given image back into the latent space of a pretrained GAN model so that the image can be faithfully reconstructed from the inverted code by the generator. The generator of an unconditional GAN learns the mapping $\mathcal{G} : \mathcal{Z} \to \mathcal{X}$. When $\mathbf{z}_1, \mathbf{z}_2 \in \mathcal{Z}$ are close in the $\mathcal{Z}$ space, the corresponding images $x_1, x_2 \in \mathcal{X}$ are visually similar. GAN inversion, denoted as $\mathcal{E}$, maps data x back to latent representation $\mathbf{z}^*$ or, equivalently, finds an image x^* that can be entirely synthesized by the well-trained generator $\mathcal{G}$ and can remain close to the real image x. Formally, denoting the signal to be inverted as $x \in \mathbb{R}^n$, the well-trained generative model as $\mathcal{G} : \mathbb{R}^{n_0} \to \mathbb{R}^n$, and the latent vector as $\mathbf{z} \in \mathbb{R}^{n_0}$, GAN inversion studies the following problem:

$$\mathbf{z}^* = \arg\min_{\mathbf{z}} \ell(\mathcal{G}(\mathbf{z}), x), \tag{1}$$

where $\ell(\cdot)$ is a distance metric in the image or feature space, and $\mathcal{G}$ is assumed to be a feed-forward neural network. With the inverted $\mathbf{z}^*$, we can obtain the original and manipulated images.

Neural ODEs [2] are a family of continuous-time models which define a hidden state $h(t)$ to be the solution to an ODE initial-value problem:

$$\dot{h}(t) = \frac{dh(t)}{dt} = f_\theta(h(t), t; \theta) \quad s.t. \quad h(t_0) = h_0. \tag{2}$$

The function f_θ specifies the dynamics of the hidden state, using a neural network with parameters θ. $t \in [0, T]$ is time and $h(t) \in \mathbb{R}^d$. The hidden state $h(t)$ is defined at all times, and can be evaluated at any desired timestep by using a numerical ODE solver denoted as `ODESolve`:

$$h_0, \ldots, h_N = \texttt{ODESolve}(f_\theta, h_0, (t_0, \ldots, t_N)), \tag{3}$$

where $(t_0, \ldots, t_N)$ are timesteps where $h(t)$ is evaluated. The gaps between consecutive timesteps t_i are not necessarily equal. Neural ODEs specify $h(t)$ as a continuous function over time, even though it is evaluated at discrete timesteps $(t_0, \ldots, t_N)$.

3 Method

Given the inverted codes $\mathbf{z}_0, \cdots, \mathbf{z}_n$ corresponding to each frame in the sequence $c_0, \cdots, c_n$, our objective is to learn the video dynamics in the GAN latent space. This work specifically aims to model the latent dynamics of StyleGAN [10], as illustrated in Fig. 1. We consider the dynamics of a state $\mathbf{z}(t)$ in the phase space Ω $(= \mathbb{R}^{2n})$ of a dynamical system. These non-temporal latent codes $\{\mathbf{z}_0, \cdots, \mathbf{z}_n\}$ become observations $\{\mathbf{z}(t_0), \cdots, \mathbf{z}(t_n)\}$ of a motion trajectory of $\mathbf{z}_0$ at specified times $t_0, \cdots, t_n$. The initial state, denoted as $\mathbf{z}(t_0)$, is equal to $\mathbf{z}_0$. This trajectory can be treated as the solution to a non-autonomous dynamical system determined by

$$\frac{d\mathbf{z}}{dt} = f(\mathbf{z}(t), t) \quad \text{for } t \in \mathbb{R},\ \mathbf{z} \in \Omega, \tag{4}$$

where $f : \Omega \times \mathbb{R} \mapsto T\Omega$ is assumed to be continuous, and $T\Omega$ is the tangent space. By approximating the differential with an estimator $f_\theta \simeq f$, where f_θ is a θ-parameterized neural network, the neural ODEs allow to model the evolution across time of such a dynamical system and learn the dynamics

(or trajectories) from relevant data. For an arbitrary time t_i, the `ODESolve` computes a numerical approximation of the integral of the dynamics from the initial time value t_0 to t_i by Eq. (4) that is equal to

$$\begin{aligned} \mathbf{z}_i = \tilde{\mathbf{z}}(t_i) &= \texttt{ODESolve}\left(f_\theta, \mathbf{z}_0, (t_0, t_i)\right) \\ &\simeq \mathbf{z}_0 + \int_{t_0}^{t_i} f_\theta\left(\mathbf{z}(t), t\right) dt = \mathbf{z}(t_i), \\ \text{where} \quad \mathbf{z}_0 &= \mathcal{E}(c_0), \quad \hat{c}_i = \mathcal{G}(\mathbf{z}_i). \end{aligned} \tag{5}$$

$\tilde{\mathbf{z}}(t_i)$ is a prediction of $\mathbf{z}(t_i)$ using a neural ODE network; $\hat{c}_i$ is a reconstructed video frame.

For training the neural ODE network, given a source video consisting of N frames $\{x_i\}_{i=1}^N$, we first obtain the preprocessed frames $\{c_i\}_{i=1}^N$ after a standard cropping-and-alignment step [34, 24]. We then use an off-the-shelf inversion method [1] to produce the latent inversion $\{\mathbf{z}_i\}_{i=1}^N = \{\mathcal{E}\left(c_i\right)\}_{i=1}^N$ for all frames. These latent codes are used to train a neural ODE network (`ODEfunc`, f_θ), which aims to predict the subsequent frames. Specifically, we randomly sample a small batch with the same size, $\mathbf{z}_0, \cdots, \mathbf{z}_n$ ($n < N$), for each iteration. Given the initial state $\mathbf{z}_0$, the neural ODE network is trained to predict the known observations, $\mathbf{z}_1, \cdots, \mathbf{z}_n$, at the corresponding times by minimizing the losses in the latent, feature, and image spaces. The loss in the latent space is defined as the optimization between original latent codes and those predicted by f_θ at the same timesteps:

$$\mathcal{L}_{latent} = \sum_{i=1}^{n} \| \underbrace{\texttt{ODESolve}\left\{f_\theta, \tilde{\mathbf{z}}\left(t_i\right), \left(t_i, t_{i+1}\right)\right\}}_{\tilde{\mathbf{z}}(t_{i+1})} - \mathbf{z}\left(t_{i+1}\right) \|_2^2. \tag{6}$$

The losses in the image and feature spaces are similarly defined using the pixel-wise mean squared error (MSE) and the learned perceptual image patch similarity (LPIPS) [35], respectively:

$$\begin{aligned} \mathcal{L}_{image} &= \sum_{i=1}^{n} \| \mathcal{G}\left(\tilde{\mathbf{z}}\left(t_{i+1}\right)\right) - \mathcal{G}\left(\mathbf{z}\left(t_{i+1}\right)\right) \|_2^2, \\ \mathcal{L}_{lpips} &= \sum_{i=1}^{n} \| \mathcal{F}\left(\mathcal{G}\left(\tilde{\mathbf{z}}\left(t_{i+1}\right)\right)\right) - \mathcal{F}\left(\mathcal{G}\left(\mathbf{z}\left(t_{i+1}\right)\right)\right) \|_2^2, \end{aligned} \tag{7}$$

where $\mathcal{G}$ is the generator [10] and $\mathcal{F}$ is a pretrained model for feature extraction [5]. The final loss is a weighted combination of these terms. Objective functions introduced for training may differ depending on the dataset. For instance, a dedicated face recognition loss [3] that measures the cosine similarity between the predicted image and its source could be incorporated to preserve facial identity.

Once trained, dividing the time interval by k, the neural ODE network produces the states at specified timesteps in the form of $\tilde{\mathbf{z}}_1, \ldots, \tilde{\mathbf{z}}_k = \texttt{ODESolve}(f_\theta, \mathbf{z}_0, (t_1, \ldots, t_k))$. The predicted latent codes $\tilde{\mathbf{z}}_i$ are then fed into the generator $\mathcal{G}$ in order to produce frames $\mathcal{G}(\tilde{\mathbf{z}}_i)$. Our method is agnostic to particular GAN inversion and latent editing techniques, and can thus be seamlessly integrated as a versatile plug-and-play module for video dynamics modeling.

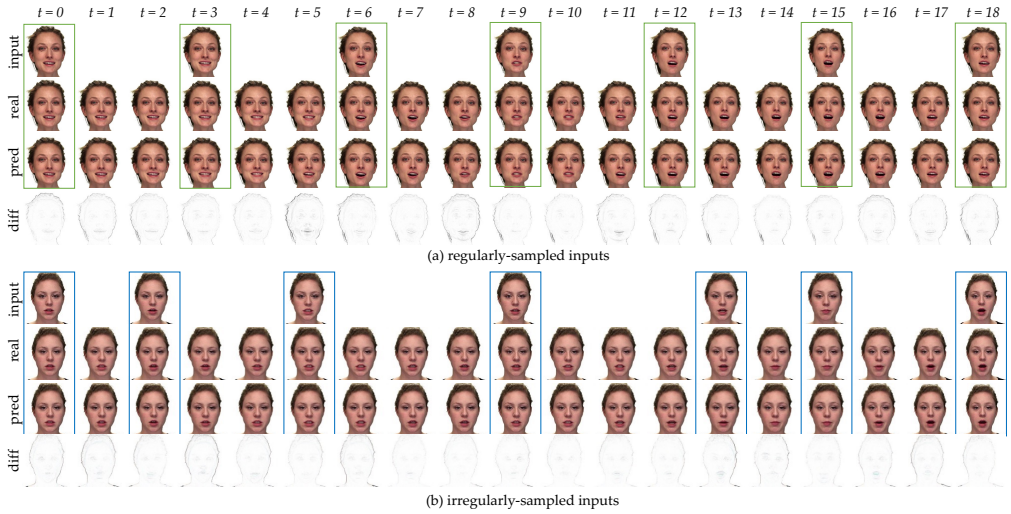
4 Experiments

This section describes experiments on dynamics modeling. We present the video reconstructions from sparse observations, which is an important example of inverse problems. Two downstream video applications of GAN latent dynamics modeling can be found in Sec. C.

Data. Our experiments use four video categories: face [9], outdoor scene [37], bird [26], and a synthetic dataset Isaac3D [17]. The real videos in the first three categories are sourced from publicly available datasets [27, 16] or obtained from YouTube. The synthetic Isaac3D videos are created by generating consecutive frames depicting the movement of the robot or camera. For additional details regarding the pretrained StyleGAN2 [10] and data preparation, please refer to Sec. B.

Evaluation. Neural ODEs are expected to learn the temporal properties of a trajectory, allowing it to approximate the actual states at the observed timesteps and even those between observations. Therefore, dynamics modeling performance can be assessed based on the reconstruction quality of predicted images at certain timesteps, which is typically evaluated by using MSE and SSIM [28].

Results. We sample frames at regular and irregular time intervals and compare the predicted frames with the actual ones at both observed and unobserved timesteps. Fig. 2 displays both the sampled frames from the original videos (*real*) and their predicted counterparts (*pred*), where the subtle differences (*diff*) indicate accurate reconstructions. Tab. 1 shows the performance of dynamics modeling characterized by the reconstruction quality. The quantitative evaluation on four categories demonstrates that our method not only models the video dynamics for the known observations but also generalizes to the unobserved time. This capability indicates that neural ODEs are able to learn the entire geometry of video dynamic rather than simply remembering given observations, enabling them to accept irregularly sampled frames or create video frames at unknown timesteps.



(a) regularly-sampled inputs

(b) irregularly-sampled inputs

Figure 2: Qualitative results of dynamics modeling on face videos. We sample frames at both regular and irregular time intervals and compare the predicted frames with the actual observations.

Dataset	Face		Scene		Bird		Isaac3D	
Metric	MSE ↓	SSIM ↑	MSE ↓	SSIM ↑	MSE ↓	SSIM ↑	MSE ↓	SSIM ↑
Observed	6.752	98.9	22.956	97.6	25.407	98.1	15.655	99.2
Unobserved	35.397	93.2	48.323	92.4	55.368	90.3	27.651	96.5

Table 1: Quantitative evaluation of dynamics modeling on four datasets. The performance of dynamics modeling is assessed based on the reconstruction quality of predicted images at certain timesteps, reported as MSE ↓ (lower is better, scaled by $\times$e-3) and SSIM ↑ (higher is better).

5 Conclusion

In this paper, we present `DynODE`, a method to model video dynamics by learning the trajectory of independently-inverted latent codes using neural ODEs. Our method estimates time-oriented and motion-coherent frames at unseen timesteps by accounting for the holistic geometry of the video dynamic space. Such a design enables continuous frame interpolation and consistent video editing, freeing video editing from tedious and redundant frame-by-frame processing. Experiments on a wide range of datasets show that our method improves upon prior state-of-the-art methods.

Limitations and Future Directions. Our framework has several limitations. For example, we use the simplest implement of the neural differential equations to show their potential applications. From the vast variants, we can provide the dynamic adjustment to the trajectory based on future observations using the neural controlled differential equations (CDEs) [12], or introduce the stochasticity using the neural stochastic differential equations (SDEs) [14]. Furthermore, since the dynamics is modeled from a single video, the learned trajectory is deterministic, which contradicts the stochastic nature of the time-varying videos. This can be addressed by training an encoder on large-scale video datasets.

Acknowledgements

This work was supported by the Engineering and Physical Sciences Research Council [grant number EP/W523835/1].

References

[1] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. In *ICCV*, pages 6711–6720, 2021.

[2] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *NeurIPS*, volume 31, page 6572–6583, 2018.

[3] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. In *CVPR*, 2019.

[4] John R Dormand and Peter J Prince. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[6] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super SloMo: High quality estimation of multiple intermediate frames for video interpolation. In *CVPR*, pages 9000–9008, 2018.

[7] David Kanaa, Vikram Voleti, Samira Ebrahimi Kahou, and Christopher Pal. Simple video generation using neural ODEs. In *NeurIPS Workshop*, 2019.

[8] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *NeurIPS*, 2020.

[9] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, pages 4401–4410, 2019.

[10] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *CVPR*, pages 8110–8119, 2020.

[11] Valentin Khrulkov, Leyla Mirvakhabova, Ivan Oseledets, and Artem Babenko. Latent transformations via NeuralODEs for GAN-based image editing. In *ICCV*, pages 14428–14437, 2021.

[12] Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural Controlled Differential Equations for Irregular Time Series. In *NeurIPS*, volume 33, pages 6696–6707, 2020.

[13] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[14] Xuechen Li, Ting-Kam Leonard Wong, Ricky TQ Chen, and David Duvenaud. Scalable gradients for stochastic differential equations. In *AISTATS*, pages 3870–3882, 2020.

[15] Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *CVPR*, pages 4463–4471, 2017.

[16] Steven R Livingstone and Frank A Russo. The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english. *PloS one*, 13(5):e0196391, 2018.

[17] Weili Nie, Tero Karras, Animesh Garg, Shoubhik Debnath, Anjul Patney, Ankit Patel, and Animashree Anandkumar. Semi-supervised StyleGAN for disentanglement learning. In *ICLR*, 2020.

[18] Sunghyun Park, Kangyeol Kim, Junsoo Lee, Jaegul Choo, Joonseok Lee, Sookyung Kim, and Edward Choi. Vid-ode: Continuous-time video generation with neural ordinary differential equation. In *AAAI*, pages 2412–2422, 2021.

[19] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. StyleCLIP: Text-driven manipulation of StyleGAN imagery. In *ICCV*, pages 2085–2094, 2021.

[20] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. In *SIGGRAPH*, pages 313–318, 2003.

[21] Yulia Rubanova, Ricky TQ Chen, and David Duvenaud. Latent ODEs for irregularly-sampled time series. In *NeurIPS*, page 5320–5330, 2019.

[22] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of GANs for semantic face editing. In *CVPR*, pages 9243–9252, 2020.

[23] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. MoCoGAN: Decomposing motion and content for video generation. In *CVPR*, pages 1526–1535, 2018.

[24] Rotem Tzaban, Ron Mokady, Rinon Gal, Amit Bermano, and Daniel Cohen-Or. Stitch it in time: Gan-based facial editing of real videos. In *SIGGRAPH Asia*, pages 1–9, 2022.

[25] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

[26] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[27] Ting-Chun Wang, Arun Mallya, and Ming-Yu Liu. One-shot free-view neural talking-head synthesis for video conferencing. In *CVPR*, pages 10039–10049, 2021.

[28] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: From error visibility to structural similarity. *TIP*, 13, 2004.

[29] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. GAN inversion: A survey. *TPAMI*, 2022.

[30] Xiangyu Xu, Li Siyao, Wenxiu Sun, Qian Yin, and Ming-Hsuan Yang. Quadratic video interpolation. In *NeurIPS*, pages 1647–1656, 2019.

[31] Yangyang Xu, Yong Du, Wenpeng Xiao, Xuemiao Xu, and Shengfeng He. From continuity to editability: Inverting gans with consecutive images. In *ICCV*, pages 13910–13918, 2021.

[32] Yangyang Xu, Shengfeng He, Kwan-Yee K Wong, and Ping Luo. Rigid: Recurrent gan inversion and editing of real face videos. In *ICCV*, 2023.

[33] Yiran Xu, Badour AlBahar, and Jia-Bin Huang. Temporally consistent semantic video editing. In *ECCV*, pages 357–374. Springer, 2022.

[34] Xu Yao, Alasdair Newson, Yann Gousseau, and Pierre Hellier. A latent transformer for disentangled face editing in images and videos. In *ICCV*, pages 13789–13798, 2021.

[35] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 2018.

[36] Bolei Zhou, Àgata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *NeurIPS*, pages 487–495, 2014.

[37] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *CVPR*, pages 633–641, 2017.

[38] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *ECCV*, 2016.

Appendices

A Related Work

Neural ODE [2] and its variants have recently been explored for video generation. Kanaa *et al.* [7] combine a typical encoder-decoder architecture with Neural ODEs. Park *et al.* [18] propose an ODE convolutional GRU as the encoder for continuous-time video generation. Unlike [7, 18], we use a neural ODE network to model the temporal correlation among latent codes, which are derived separately by using existing GAN inversion algorithms.

There are some ways to analyze the spatio-temporal dynamics of videos in addition to ODE-based models. Recent studies [18, 12] have demonstrated that they are not as effective as ODE-based models. The RNN-based models, for example, assuming fixed time intervals, are limited to learn the representations only at the observed times. These methods can barely handle datasets collected from wild environments with irregular time intervals and missing states.

B Experimental Settings

B.1 Pretrained Models and Datasets

Given their widespread popularity, most of the GAN inversion and latent editing methods focus on StyleGANs [9, 10]. Therefore, we use StyleGAN2 [10] as the pretrained model in all experiments. Our method is not dependent on StyleGANs and can be applied to any GAN model. The latent codes are obtained by inverting each frame into $\mathcal{W}^+$ space of StyleGAN2 [10]. Other latent spaces [29], *e.g.* $\mathcal{Z}$ or $\mathcal{W}$ space, are also supported. Experiments are conducted on several categories of publicly available datasets to demonstrate the effectiveness of our proposed method.

Face. The StyleGAN2 model is trained on FFHQ [9] at a resolution of 1024×1024. The real face videos are collected from public talking-head datasets [27, 16] or downloaded from YOUTUBE.

Scene. The StyleGAN2 is trained using Place365 [37] at the resolution of 256×256. We use [11] to generate temporally-consistent frames by editing the time-varying attributes, *e.g.*, NIGHT, DAWNDUSK, and SUNRISESUNSET. We also download real videos of outdoor natural scenes from YOUTUBE and obtain latent codes of sampled frames by direct optimization.

Bird. The StyleGAN2 model is trained using CUB-200-2011 dataset [26] at the resolution of 256×256. We collect bird videos from YOUTUBE.

Isaac3D. The StyleGAN2 model is trained using synthetic images with a resolution of 128×128 from Isaac3D [17]. The dataset contains 9 factors of variation, such as background color, object shape, robot movement, and camera height. We generate consecutive frames by moving the robot or camera.

B.2 Data Preparation

We collect some real videos of outdoor natural scenes, talking heads, and birds from YOUTUBE and obtain latent codes of sampled frames by direct optimization. These in-the-wild images may not be well suited to current GAN models and inversion methods due to their limitations. To overcome the limitations of both datasets and off-the-shelf methods and maintain our focus on validation of our proposed method, in some cases, we synthesize data by using [11], which is capable of editing attributes of images generated by StyleGAN through latent editing. This process of data synthesis is mainly performed on two categories: outdoor scenes [36] and Isaac3D [17].

Scene. The images of outdoor scenes are extracted from [36] and contain annotations for 40 binary attributes, such as dusk, autumn, flowers, dull, colorful, midday, fog, snow, windy, and rain.

Isaac3D. Isaac3D [17] contains nine factors of variation, such as background color, lighting intensity, object shape, robot movement, or camera height. We notice that some attributes are time-related, such as NIGHT, DAWNDUSK, and SUNRISESUNSET for outdoor scenes [36], or ROBOT MOVEMENT for Isaac3D [17]. To be specific, we use [11] to generate temporally-consistent frames by editing the time-varying attributes, *e.g.*, NIGHT, DAWNDUSK, and SUNRISESUNSET. As Isaac3D [17] is a synthetic dataset, we generate consecutive frames by simulating the camera height adjustment or the

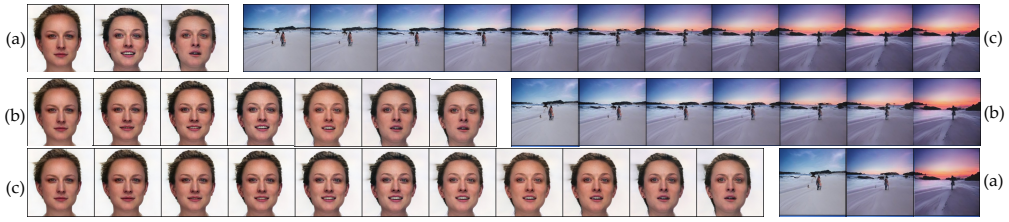


Figure 1: Results of continuous frame interpolation. Based on given frames if talking faces or outdoor natural scenes in (a), our method can generate in-between video frames in diverse time intervals.

robot movement along the x-/y-axis. The rest of the attributes are used for consistent video editing. This data preparation process can alleviate the limitation of datasets and help us focus on validating that the edited videos present identical video dynamics and maintain temporal coherence.

B.3 Implementation Details

We implement the proposed method in PyTorch on an Nvidia GeForce RTX 2080. The neural ODE network is parameterized by a Multi-Layer Perceptron (MLP). The second and third layers together form a repeating module. The parameter DEPTH that determines the number of repetitions is adjusted depending on the complexity of the dataset. The parameters in the neural ODE function are optimized using the Adam optimizer [13]. We train the neural ODE network for 5,000 gradient descent steps using a learning rate of 0.01, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1e^{-8}$. We use a adaptive-step DOPRI5 (Runge-Kutta [4] of order 5 of Dormand-Prince-Shampine) as the default ODE Solver. Details on StyleGANs [9, 10] and the inversion method [1] can be found in the respective papers.

B.4 Performance Metrics

For dynamics modeling, we use pixel-wise Mean Square Error (MSE) and Structural Similarity Index (SSIM) [28] as the metrics for quantitative comparisons. For video editing, we focus on the temporal coherence of edited videos, which is measured by the identity similarity between frame pairs as in [24]. Specifically, we use two metrics, temporally-local (TL-ID) and temporally-global (TG-ID) identity preservation introduced in [24], and a variant of the Average Content Distance (ACD) [23]. We use Fréchet Video Distance (FVD) [25] to evaluate the motion quality in the edited videos.

C Example of Downstream Tasks

Our `DynODE` learns the holistic geometry of the video dynamic space from given sparse observations and specifies continuous latent states, allowing us to engage in various video applications. This section demonstrates two example of applying the acquired latent dynamics to two downstream video processing tasks: frame interpolation (Sec. C.1) and video editing (Sec. C.2).

C.1 Continuous Frame Interpolation

The learned neural ODE specifies $z(t)$ as a continuous function over time, which facilitates infinite frame interpolation. It enables our framework to interpolate non-existent frames within the time interval from $t = 0$ to $t = N$ at arbitrary timesteps. Once trained, dividing the time interval $[0, N]$ by k, the neural ODE network produces the in-between states at the corresponding timesteps in the form of $\mathbf{z}_0, \ldots, \mathbf{z}_k = \texttt{Solve}(f_\theta, \mathbf{z}_0, (t_0, \ldots, t_k))$. The intrinsic properties of neural ODEs allow to achieve such infinite frame interpolation at a constant memory cost even when the frames are irregularly sampled or partially observed, which is the case that their RNN-based counterparts are often struggling to deal with [12, 21]. This is particularly helpful when a temporally smooth video is required. Fig. 1 shows the results of frame interpolation for talking heads and outdoor natural scenes at arbitrary timesteps. While the generators are trained on image datasets and not specifically tuned for the video, the predicted frames still exhibit high consistency across time.

It should be noted that intermediate video frames can also be generated by simply blending two adjacent latent codes. This procedure, often called image morphing in literature, is to fuse two images

by interpolating their latent codes, which also presents a continuous process. Given z_s and z_t, a series of semantically meaningful images can be generated following $z^* = z_s + \alpha(z_t - z_s)$, where α is a scale between 0 and 1. The term $(z_t - z_s)$ can also be considered as a direction, similar to those discovered by latent editing methods [34, 11, 22]. As opposed to the trajectory learned by neural ODEs, which could be viewed as the *temporal* directions, these are *spatial* or *manipulatable* directions for altering the semantics. Fig. 2 (adopted from [18]) illustrates a video dynamic space from t_s to t_t. The 2D instead of 1D structure of such space indicates the stochasticity of the trajectory between the two observed timesteps. The difference between direct morphing and ours is obvious in the video dynamic space. The direct morphing creates the intermediate states (green) at t_a, t_b, and t_c by accumulating the differences between $\mathbf{z}_s$ and $\mathbf{z}_t$, without considering the geometry structure of the video dynamic space. The obtained states may fall outside of the video dynamic space and thus produce frames that contain spatial changes instead of temporal motions. In contrast, our method estimates in-between states (red) at unknown times by accounting for the holistic geometry of the video dynamics. These states produce time-oriented and motion-coherent frames. The trajectory obtained from our method, illustrated as the blue arrow curve, fits closely with the video dynamic space and indicates the intrinsic motion of the video.

Due to limitations of StyleGAN [10], our method is suited more to videos with a specific category and may not perform well on existing frame interpolation benchmarks that include different objects or scenes. Existing frame interpolation methods [30, 6, 15] cannot be trained for specific categories due to the lack of such high-quality video datasets. As a result, we do not conduct comparisons of video interpolation. The comparison will be applicable if either constraint is lifted in the future.

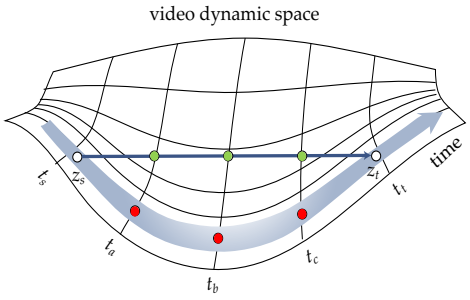


Figure 2: In the video dynamic space, the direct morphing creates the intermediate states (green) by accumulating the differences between the two states at t_s and t_t. In contrast, neural ODEs estimate in-between states (red) at unseen times by accounting for the holistic geometry of the video dynamics. These states produce time-oriented and motion-coherent frames.

C.2 Consistent Video Manipulation

In this section, we present experiments that apply the latent dynamics to video editing.

Baselines. We compare our method with five state-of-the-art face video editing baselines that rely on GAN inversion: IGCI [31], Latent Transformer [34], STIT [24], TCSVE [33], and RIGID [32]. These methods follow a pipeline that contains three key steps: pre-processing (inverting each cropped and aligned frame into the latent space); attribute manipulation (editing images by employing off-the-shelf latent based semantic editing techniques), and seamless cloning (blending the modified faces with the original input frames using [20]). These three methods achieve attribute editing in the video by applying the same redundant operations to every frame. In contrast, our method alters the initial frame and extends these modifications to the entire sequence by leveraging the learned trajectory.

Qualitative evaluation. Comparison with face video editing methods is shown in Fig. 3. The last two rows are results from implementing our method as a plug-and-play module for the video dynamics modeling in Latent Transformer [34] and TCSVE [33], respectively. The results demonstrate highly temporal coherence across all frames. However, it is worth noting that our results are obtained by exclusively applying core operations to the first frame only, thereby eliminating the need for redundant operations on all frames. Video editing results on talking heads and outdoor natural scenes are shown in Fig. 4, Fig. 5, Fig. 6, and Fig. 7. The target attribute is edited by employing off-the-shelf latent semantic editing methods [11, 34, 22, 19] on the first frame. The edited video frames show identical video dynamics and maintain temporal coherence.

Figure 3: Qualitative comparison with the state-of-the-art face video editing baselines: IGCI [31], Latent Transformer (Latent-T) [34], STIT [24], TCSVE [33], and RIGID [32]. The last two rows are obtained through plug-and-play integration of our method into Latent-T [34] and TCSVE [33].

Quantitative evaluation. Tab. 1 shows the quantitative evaluation comparisons. We show the result of our method embedded as a plug-and-play dynamics modeling module into Latent Transformer [34] and TCSVE [33], denoted as "`DynODE` (*w.* (with) [34])" and "`DynODE` (*w.* [33])" in Tab. 1, respectively. The best and second-best results are marked in **bold** and underline. Our method achieves state-of-the-art performance. It significantly improves performance and mitigates the drifting identity issue by holistically modeling the entire dynamics space. The quantitative evaluation of RIGID [32] is not reported as the code is not available. In contrast to other video editing studies, our method avoids repeating operations on every frame, without sacrificing editing quality or temporal consistency.

	IGCI [31]	STIT [24]	Latent-T [34]	TCSVE [33]	`DynODE` (*w.* [34])	`DynODE` (*w.* [33])
TL-ID ↑	0.969	0.989	0.957	0.992	0.965	**0.995**
TG-ID ↑	0.851	0.912	0.839	0.939	0.843	**0.957**
ACD ↓	1.485	0.846	1.352	0.798	1.331	**0.778**
FVD ↓	632.7	352.9	582.4	323.1	522.3	**304.2**

Table 1: Quantitative comparison on video editing. We reported results of three identity preservation metrics (TL-ID, TG-ID and ACD) and a motion quality metric FVD. The best and second-best results are marked in **bold** and underline. ↑ means higher is better, while ↓ means the opposite.

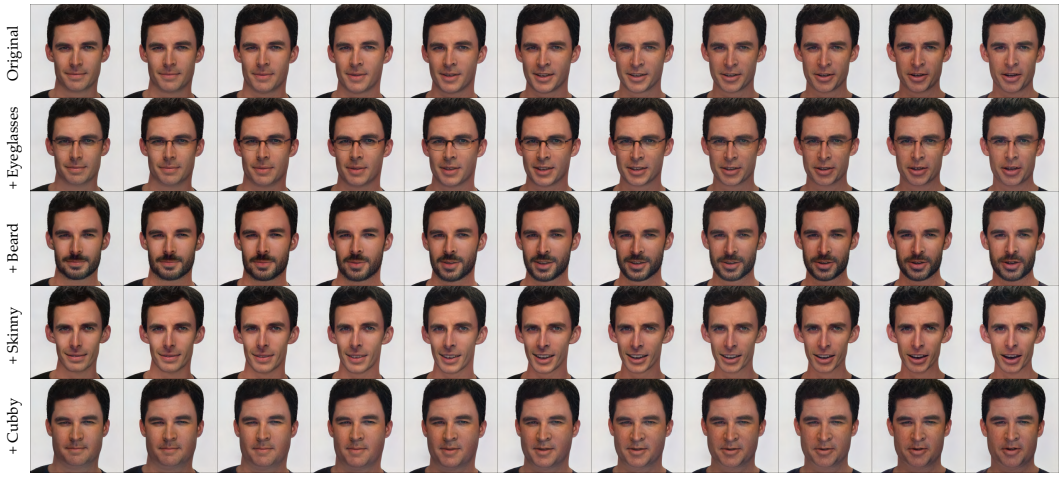


Figure 4: Video editing results on talking heads. We use [34] to alter facial attributes. Our method changes the desired attributes of the entire video by only editing the initial frame and extending such modifications to the entire sequence, without the need to apply redundant operations to every frame. The edited video frames show identical video dynamics and maintain temporal coherence.

Figure 5: Video editing results on outdoor scenes. Edited attributes are obtained by [11] except MOUNTAIN, which is edited by StyleCLIP [19]. The manipulated frames of the entire video show identical video dynamics and maintain temporal coherence.

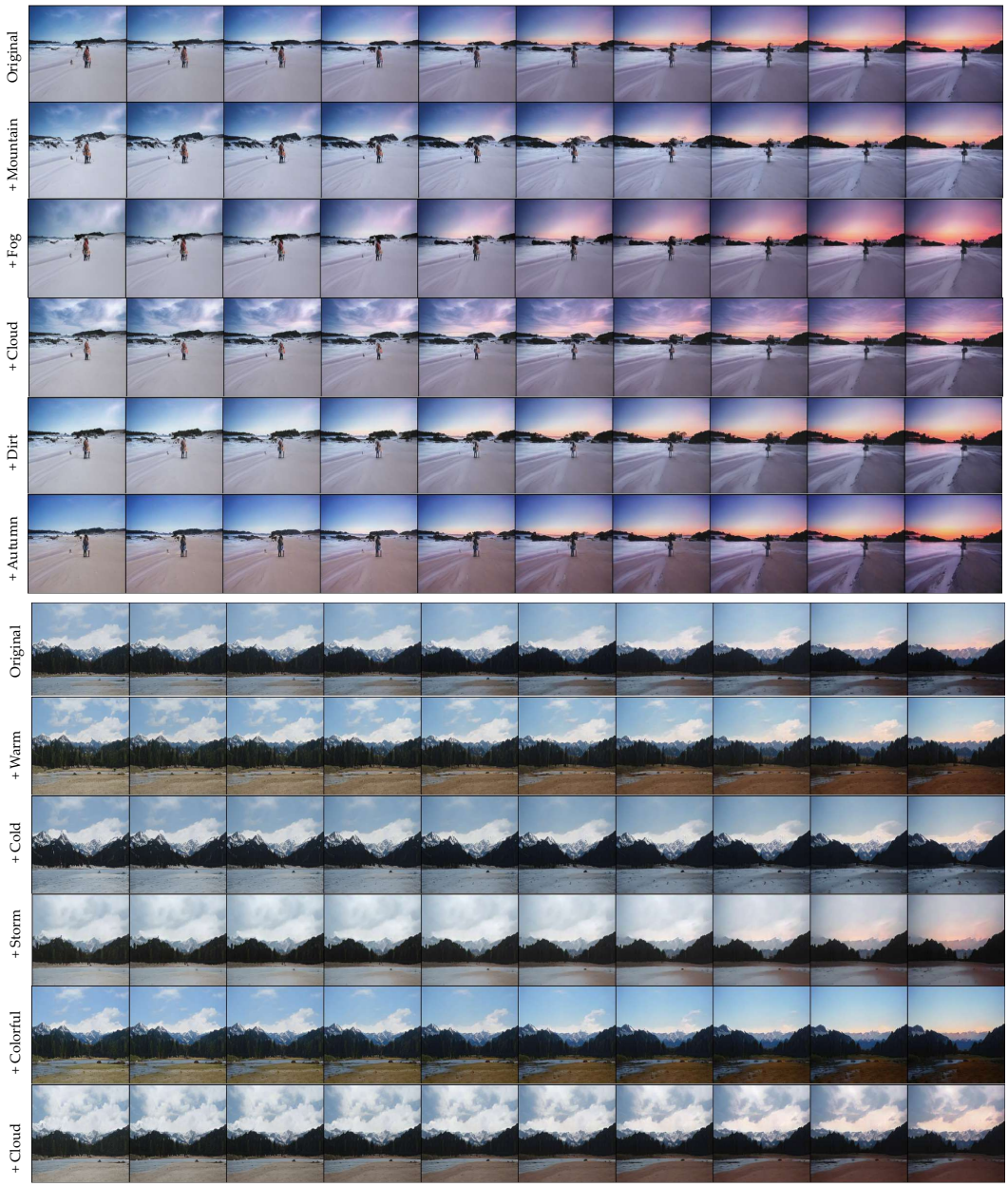


Figure 6: Video editing results on outdoor scenes. All edited attributes are obtained by using [11] except MOUNTAIN, which is edited by using StyleCLIP [19]. The manipulated frames of the entire video show identical video dynamics and maintain temporal coherence.

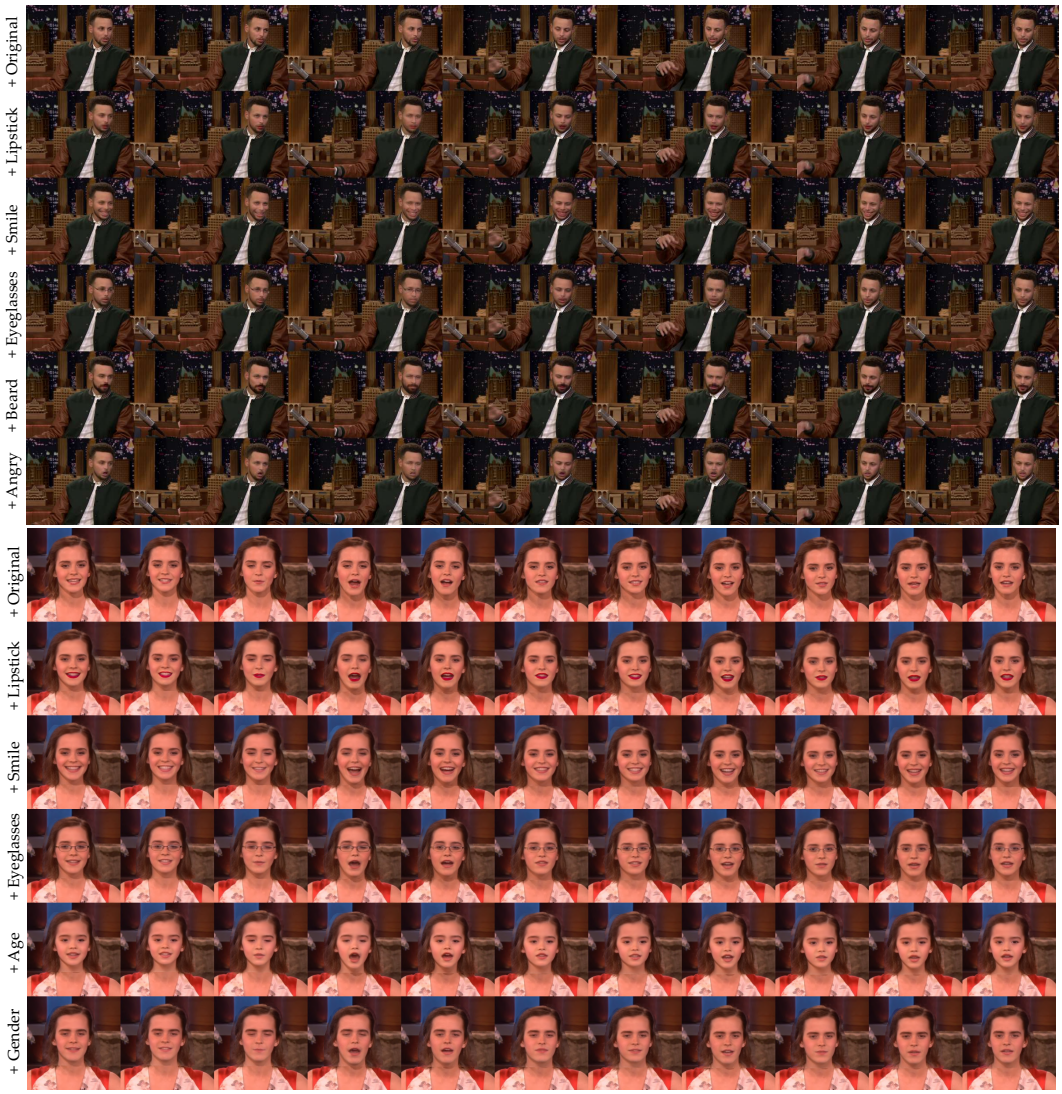


Figure 7: Video editing results on talking heads with complex backgrounds. The edited attributes of GENDER, AGE, and EYEGLASSES are obtained by InterFaceGAN [22]. The edited attributes of SMILE and LIPSTICK are obtained by StyleCLIP [19]. The manipulated frames of the entire video show identical video dynamics and maintain temporal coherence.