Supplementary Materials for DreamMover

Anonymous ECCV 2024 Submission

Paper ID #2366

To see the dynamic effect of our method and visual comparisons, please refer to our supplementary video. This document includes the following contents:

- Math explanation for fusion.
- Details about *InterpBench* dataset
- More details of baselines and metrics.
- Effects of inversion step T.
- More implementation details of our method.
- More qualitative results.
- Execution time.
- Details of user study.
- Limitation discussion and failure cases.

A Math explanation for our fusion

In the main paper, we point out that the fusion between two input images including Softmax Splatting [10] and time-weighted averaging loses details and high-frequency information. Here, we give an easy mathematical explanation.

In the theories of Denoising Diffusion Probabilistic Models [4], a clean image can be turned into a full Gaussian noise by constant noise addition. Since our fusion process is performed at the noisy timestamp, we may as well assume that the noisy latent codes are standard Gaussian noise.

Now we have the noisy latent code for each of the two images to be interpolated, $z_T^0 \sim \mathcal{N}(0,1)$ and $z_T^1 \sim \mathcal{N}(0,1)$, where T is the noisy timestamp. According to the nature of the Gaussian distribution, if the whole latent obeys the standard Gaussian distribution, each pixel should also conform to the standard Gaussian distribution and each pixel is independent of each other. Consider the Softmax Splatting operation and assume that the pixel points $r_1, r_2, \dots r_n \sim \mathcal{N}(0,1)$ from z_T^0 are mapped to the same location by optical flow $F^{0\to\delta}$, where $\delta \in (0,1)$ is the middle time to be interpolated. Then the value of this location is the weighted average of the pixel points, and the formula is:

$$\mu_1 r_1 + \mu_2 r_2 + \cdots + \mu_n r_n \ \sim \mathcal{N}(0, \mu_1^2 + \mu_2^2 + \cdots + \mu_n^2) \ , \tag{1}$$

where $\mu_i > 0$ is the weight of r_i and $\mu_1 + \mu_2 + \cdots + \mu_n = 1$. Due to the additivity of independent Gaussian distributions, the mean at this location remains unchanged but the variance becomes $\mu_1^2 + \mu_2^2 + \cdots + \mu_n^2$, which is far less than the original 1. Thus, the variance of all pixels of $z_T^{0\to\delta}$ obtained by Softmax Splatting of z_T^0 is much less than 1. In the same way, we can get $z_T^{1\to\delta}$ obtained by Softmax Splatting of z_T^1.

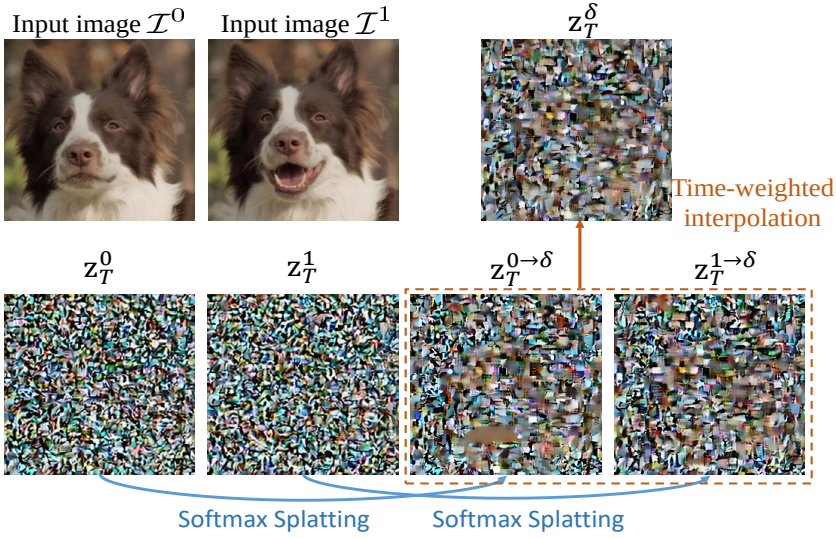

Fig. A.1: Visualization of the directly fusion process, if we do not divide it into two spaces, high-level and low-level. We can see the noise getting blurred and changes from high frequencies to low frequencies.

The reduction in the variance of noisy latent codes means that the fluctuation of the noise becomes smaller and tends to be more stable. However, the noisy latent code are obtained by constantly adding noise to the clean latent codes via DDIM inversion [14], and the process forms a uniquely determined Markov chain [9]. Thus the detailed information in the image is embedded in the noise. Noise from high to low frequencies results in the loss of high frequency information in the image space.

Next, we discuss the operation of time-weighted averaging:

$$z_T^\delta = (1-\delta)\cdot z_T^{0\to\delta} + \delta \cdot z_T^{1\to\delta} \sim \mathcal{N}(0, (1-\delta)^2\sigma_0^2 + \delta^2\sigma_1^2) \ , \tag{2}$$

where σ_0^2 and σ_1^2 are the variance of $z_T^{0\to\delta}$ and $z_T^{1\to\delta}$. Due to $(1-\delta)^2+\delta^2 < 1$, we can find that the variance of the final noisy latent code z_T^δ at time δ further decreases and the detailed information is further lost. The whole process is shown in Fig. A.1, and the qualitative results are shown in Fig. A.2(a).

A natural but incorrect idea is to normalize the variance back to 1 using Gaussian normalization. Thus we can rewrite Eq. 1 and Eq. 2 in the following form:

$$\frac{\mu_1 r_1 + \mu_2 r_2 + \cdots + \mu_n r_n}{\sqrt{\mu_1^2 + \mu_2^2 + \cdots + \mu_n^2}} \sim \mathcal{N}(0,1) \ , \tag{3}$$

$$z_T^\delta = \frac{(1-\delta)\cdot z_T^{0\to\delta} + \delta \cdot z_T^{1\to\delta}}{\sqrt{(1-\delta)^2\sigma_0^2 + \delta^2\sigma_1^2}} \sim \mathcal{N}(0,1) \ , \tag{4}$$

However, this simple normalization supplements some wrong high-frequency information and sharpens the final generated images as shown in Fig. A.2(b).

In addition, Diffmorpher [17] tries to adopt Adaptive Instance Normalization (AdaIN) [6] to ensure the coherence in color and brightness between generated images and input images. However, applying it to our task would also result in erroneous high-frequency information, just as shown in Fig. A.2(c).

Our key insight is to divide the original z_T^δ into the high-level space $z_{T\to 0}$ and the low-level space $\epsilon_\theta(z_T, T)$. We then fuse as usual but in the high-level space, which only contains little high-frequency information of images, mitigating the loss of details. As for $\epsilon_\theta(z_T, T)$, in order to retain the high-frequency information, we employ the "winner-take-all"(WTA) strategy and replace all weighted average operations in Softmax Splatting and temporal interpolation with it. However, it is worth noting that if we use WTA for both high-level and low-level spaces, it will result in an unsmooth spatial transition between images interpolations, as illustrated in Fig. A.2(d).

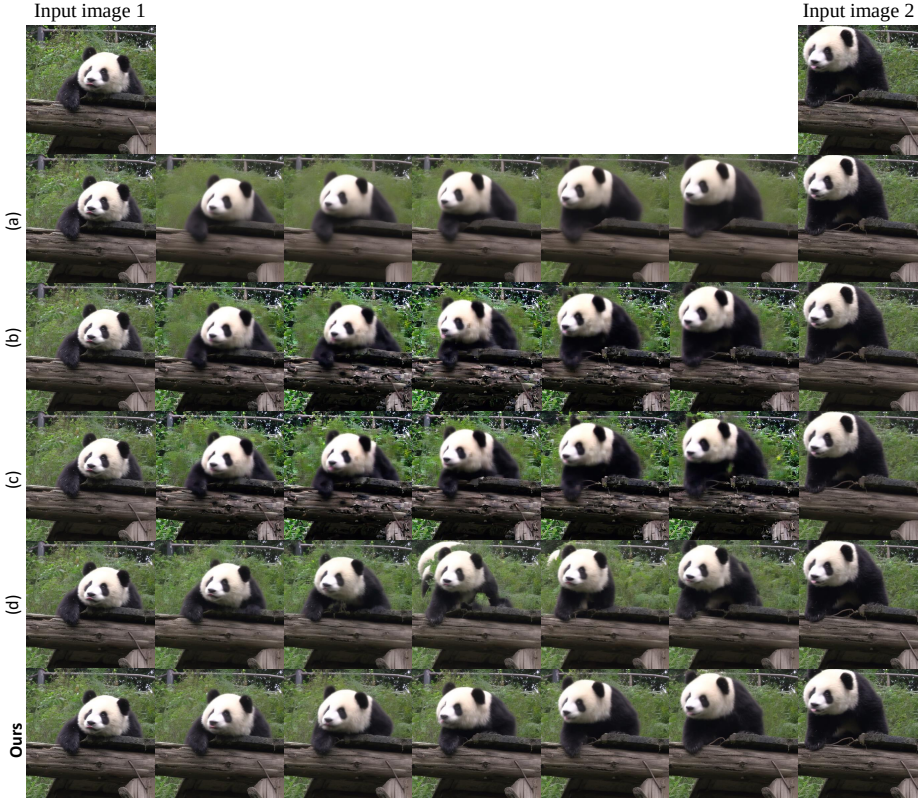

Fig. A.2: Visual comparison of different fusion strategies. (a)Directly fusion, (b) using Gaussian normalisation, (c) adopting Adaptive Instance Normalization (AdaIN), (d) employing "WTA" in both spaces, and **our strategy.**

B Details about *InterpBench* Dataset

We have collected 100 pairs of images in total, which include a wide variety of large motion of objects. We used off-the-shelf image editing tools such as MasaCtrl [1] and DragDiffusion [13] to obtain 10 pairs of edited images. All the other 90 pairs of images are real images downloaded from Pixabay (https://pixabay.com/) and Mixkit (https://mixkit.co/). We hope future research on this task can benefit from *InterpBench*. The dataset will be released soon.

C More Details of Baselines and metrics

In experiment part, we comprehensively compare our method with previous state-of-the-art methods, including frame interpolation for large motion and image morphing techniques. We offer more details of the baselines and metrics that we use here:

Diffinterp [15]: *Interpoation between Images with Diffusion Models* is a recent state-of-the-art image interpolation method based on diffusion models. They employ latent interpolation, text embedding interpolation and pose guidance based on ControlNet [18]. However, they focus on transitions between different objects, but cannot work well in the same objects with large motion. We utilize the official code (https://github.com/clintonjwang/ControlNet) and the pretrained Stable Diffusion v1.5 base model as our baseline.

Diffmorpher [17]: Similar to Diffinterp, they worked on image morphing through diffusion model with two images of topologically similar objects as input. The key idea is to capture the semantics of the two images by fitting two LoRAs [5]. Since their interpolation is performed directly by superimposing two images, if there is large motion of the same object, the results are often distorted. We adopt the official code (https://github.com/Kevin-thu/DiffMorpher) with default settings and the pretrained Stable Diffusion v1.5 base model as our baseline.

Film [12]: They try to apply frame interpolation between near-duplicate photos, and accommodate larger motion than the previous method of video frame interpolation. Nevertheless, artifacts appear when the photos are not near duplicates. We employed the code of the Pytorch version and the official pretrained model (https://github.com/dajes/frame-interpolation-pytorch).

In the main paper, we apply FID [3], LPIPS [19], WE [8] and WE_{mid} as our evaluation metrics. We compute the FID score between the distribution of the two input images and the distribution of the two middle interpolated images. As for LPIPS, we take each of the two middle images and apply the perceptual similarity with the two input images respectively and calculate the mean values. We employ WE to evaluate the temporal consistency of the generated videos. In addition, for WE_{mid}, we separately warp the intermediate two images to the input image pair and compute the MSE loss. For the sake of fairness, in all methods, we take the two middle-most generated images to calculate the metrics.

D Effects of inversion step T

We conducted a qualitative comparison to elucidate the impact of varying T (i.e., the total number of inversion steps) during the latent optimization stage of our method. We set T to be $T = 10, 20, 30, 40, 50$ steps and run our approach on *InterpBench* to obtain the interpolation results ($T = 50$ corresponds to the pure noisy latent). We can observe qualitative visualization in Fig. D.3. Considering generation effects and inference time, $T = 30$ steps outperforms other steps, we set this as our default setting.

Fig. D.3: Effects of different inversion steps T. We set DDIM inversion step to be $T = 10, 20, 30, 40, 50$ steps, and compare the interpolation results.

E More implementation details of our method.

E.1 Fine-tuning LORA

Low-Rank Adaption (LoRA) [5] is an efficient tuning method initially developed for fine-tuning large language models, and more recently applied to diffusion

models. Rather than adjusting the complete diffusion model directly, LoRA refines the model parameters θ through the training of a low-rank residual component $\Delta\theta$, which can be broken down into low-rank matrices. In addition to its inherent efficiency in fine-tuning, LoRA demonstrates a remarkable ability to capture the essence of provided images within the low-rank parameter space. By simply fitting a LoRA on the two input images, the fine-tuned model can generate images with consistent semantic identity.

Hence, we train a Lora $\Delta\theta$ on the diffusion UNet ϵ_θ for the input image pair $\mathcal{I}_0$ and $\mathcal{I}_1$. Formally, the learning objective for training $\Delta\theta$ is:

$$\mathcal{L}(\Delta\theta) = \mathbb{E}_{\epsilon,t}[||\epsilon - \epsilon_{\theta+\Delta\theta}(\sqrt{\bar{\alpha}_t}\mathbf{z}_{0i} + \sqrt{1-\bar{\alpha}}\epsilon, t, \mathbf{c}_i)||^2]$$

where $\mathbf{z}_{0i} = \epsilon(\mathcal{I}_i)$ is the VAE encoded latent embedding associated with the input images, $\epsilon \sim \mathcal{N}(0, I)$ is the random sampled Gaussian noise, c_i is the text embedding encoded from the text prompt $\mathcal{P}_i$ and $\epsilon_{\theta+\Delta\theta}$ represents the LoRA-intergrated UNet. The fine-tuning objective is optimized separately via gradient descent in $\Delta\theta$. After fine-tuning, we apply the UNet with LoRA $\Delta\theta$ as the noise prediction network in the denoising steps.

E.2 WTA

In the main paper, we introduce "Winner-Takes-All" (WTA) to replace all weighted-averaging operations in Softmax Splatting and time-weighted interpolation for low-level space. Here we go into more details about WTA. As the name implies, WTA stands for its literally meaning. As shown in Eq. 1 and 3, both Softmax Splatting and time-weighted interpolation involve weighted-averaging operations. In other words, the results are derived by adding together the relevant weights of pixels or latent codes, and the sum of the weights is 1. Nevertheless, as demonstrated in A, the weighted-averaging operations result in a reduction of variance. We therefore decide to take the value of whoever has the largest weight directly, thus circumventing the loss of high-frequency information caused by the variance getting smaller in the low-level space. For Eq. 1, if r_i has the highest weight μ_i, the final result is r_i instead of $\mu_1 r_1 + \mu_2 r_2 + \cdots + \mu_n r_n$. Similarly, for Eq. 3, if δ is the greater than $1-\delta$, the final result is $z_t^{0\to\delta}$ instead of $(1-\delta)\cdot z_T^{0\to\delta} + \delta \cdot z_T^{1\to\delta}$.

E.3 Text prompt

In our framework, $\mathcal{I}_0$ and $\mathcal{I}_1$ can be either real images or diffusion-generated images with text prompts $\mathcal{P}_0$ and $\mathcal{P}_1$. Text prompts can simply be described as "a photo of [something]". For the input image pair and the generated results, the text prompts are all the same.

E.4 Inter-frame processing module

Most recent works [2, 7, 11, 16, 20] attempt to extend the existing advanced diffusion models for text-to-image generation to a text-to-video editing model by

inflating spatial self-attention into spatio-temporal self-attention. Specifically, the features of the patches from different frames are combined in the extended spatio-temporal attention module. By capturing spatial and temporal context in this way, we introduce this strategy to improve the inter-frame consistency without training.

F More qualitative results.

We present a range of scenarios to illustrate the effectiveness of our method in both image quality and semantic coherence. Additionally, we encourage readers to refer to the accompanying video for a more comprehensive visual comparison.

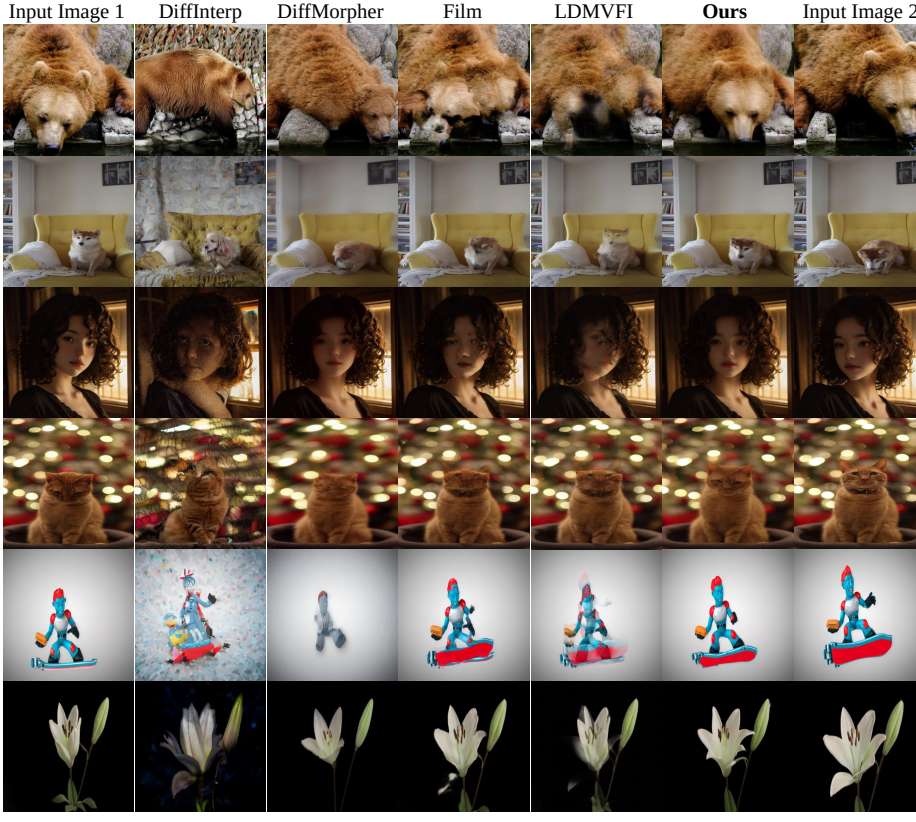

Fig. F.4: More qualitative results.

Fig. F.5: More qualitative results.

G Execution time

As shown in Table G.1, we counted the execution time of baselines and our method on a single NVIDIA RTX 3090 GPU. All methods generate 32 intermediate images of 512x512 resolution. Besides inference time, our method and Diffmorpher cost additional time in fine-tuning Loras [5]. They need to fine-tune two, while we only need one. Our total runtime includes 42.15s of fine-tuning Lora time and 138.39s of inference time. Our approach is based on a diffusion model but the inference time is essentially the same as Film, due to the fact that our method only needs to compute the optical flow once to generate all frames, but Film requires computing the flow every time a frame is generated. DiffInterp employs additional texture optimization and Diffmorpher applies a resampling strategy which takes a lot of time. LDMVFI trains a diffusion model for video interpolation with many more denoising steps, which significantly increases inference time.

Table G.1: Execution time.

	DiffInterp	Diffmorpher	Film	LDMVFI	**ours**
Time(s)	886.76	83.24+387.09	126.68	1560.47	42.15+138.39

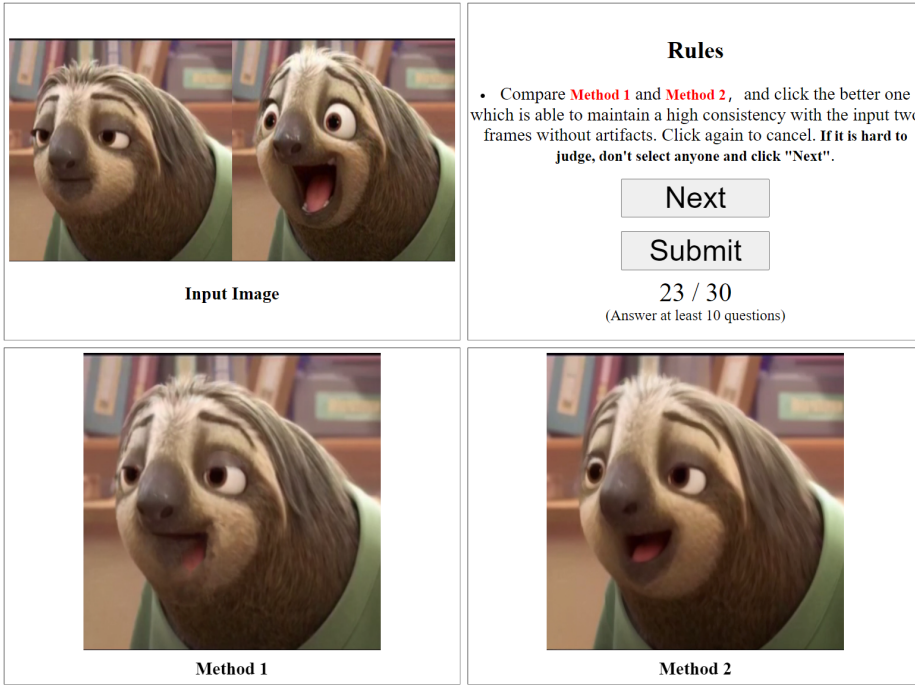

Fig. H.6: Interface of the user study website.

H User Study

We conduct a user study to assess the effectiveness of our method as perceived by human observers. The study comprises 30 pairs of photos from *InterpBench* dataset. We create an online website for the user study, and a screenshot of the website interface is shown in Fig. H.6. Method 1 and Method 2 exhibit the synthesized videos of two different methods. One of the methods is ours, and the other was randomly selected from DiffInterp [15], DiffMorpher [17] and Film [12]. Note that the positions of the two methods are not fixed in a specific order, but are randomly arranged for each example. We use these methods to generate videos by interpolating frames between two images. Participants are required to select the method that can generate videos with high fidelity and high consistency. If the judgment is difficult, they can choose to skip to the next example without selecting any method. The user study is completely anonymous and it does not involve the collection of any personally identifiable data.

I Limitations

Our approach builds upon the foundation of the pre-trained diffusion model; however, it also carries forward some of its constraints. By utilizing the diffusion model in a low-resolution latent space, we risk encountering issues such as texture sticking and challenges in capturing subtle movements, as shown in Fig I.7. Furthermore, our method may struggle with significant camera motions due to the wide range of viewing angles, making it challenging to obtain optical flow accurately, as shown in Fig. I.8.

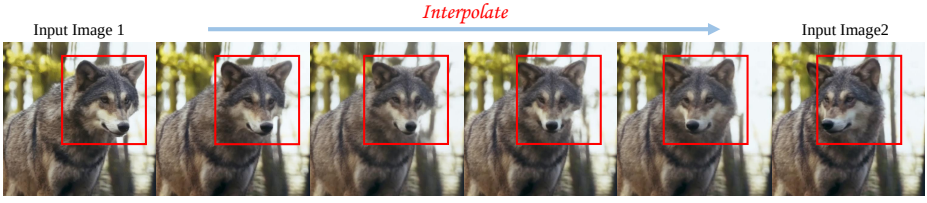

Fig. I.7: Limitations of our method. Our method may encounter texture sticking, where the background near the subject moves along with it.

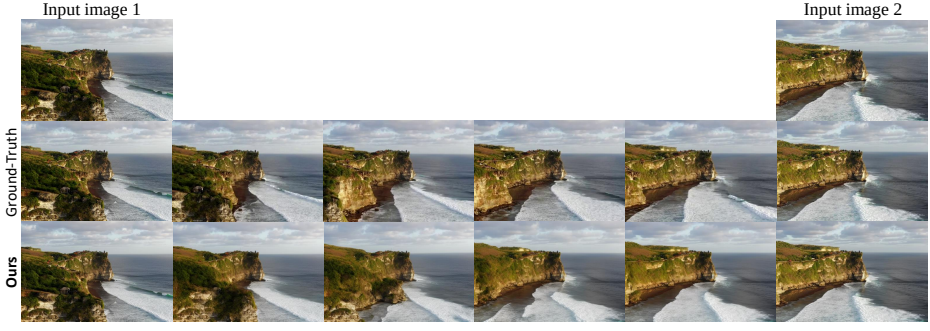

Fig. I.8: Limitations of our method. The camera is rotating to advance in the ground-truth video, but the animation we generated based on the two images is more like changing in a plane.

References

1. Cao, M., Wang, X., Qi, Z., Shan, Y., Qie, X., Zheng, Y.: Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing. arXiv preprint arXiv:2304.08465 (2023) 4
2. Ceylan, D., Huang, C.H.P., Mitra, N.J.: Pix2video: Video editing using image diffusion. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 23206–23217 (2023) 6
3. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017) 4
4. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020) 1
5. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685 (2021) 4, 5, 8
6. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: Proceedings of the IEEE international conference on computer vision. pp. 1501–1510 (2017) 3
7. Khachatryan, L., Movsisyan, A., Tadevosyan, V., Henschel, R., Wang, Z., Navasardyan, S., Shi, H.: Text2video-zero: Text-to-image diffusion models are zero-shot video generators. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15954–15964 (2023) 6
8. Lai, W.S., Huang, J.B., Wang, O., Shechtman, E., Yumer, E., Yang, M.H.: Learning blind video temporal consistency. In: Proceedings of the European conference on computer vision (ECCV). pp. 170–185 (2018) 4
9. Markov: An example of statistical investigation of the text eugene onegin concerning the connection of samples in chains. Science in Context **19**(4), 591–600 (2006) 2
10. Niklaus, S., Liu, F.: Softmax splatting for video frame interpolation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5437–5446 (2020) 1
11. Qi, C., Cun, X., Zhang, Y., Lei, C., Wang, X., Shan, Y., Chen, Q.: Fatezero: Fusing attentions for zero-shot text-based video editing. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15932–15942 (2023) 6
12. Reda, F., Kontkanen, J., Tabellion, E., Sun, D., Pantofaru, C., Curless, B.: Film: Frame interpolation for large motion. In: European Conference on Computer Vision. pp. 250–266. Springer (2022) 4, 9
13. Shi, Y., Xue, C., Pan, J., Zhang, W., Tan, V.Y., Bai, S.: Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. arXiv preprint arXiv:2306.14435 (2023) 4
14. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. arXiv preprint arXiv:2010.02502 (2020) 2
15. Wang, C.J., Golland, P.: Interpolating between images with diffusion models. arXiv preprint arXiv:2307.12560 (2023) 4, 9
16. Wu, J.Z., Ge, Y., Wang, X., Lei, S.W., Gu, Y., Shi, Y., Hsu, W., Shan, Y., Qie, X., Shou, M.Z.: Tune-a-video: One-shot tuning of image diffusion models for text-to-video generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7623–7633 (2023) 6

17. Zhang, K., Zhou, Y., Xu, X., Pan, X., Dai, B.: Diffmorpher: Unleashing the capability of diffusion models for image morphing. arXiv preprint arXiv:2312.07409 (2023) 3, 4, 9
18. Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3836–3847 (2023) 4
19. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: 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) 4
20. Zhang, Y., Wei, Y., Jiang, D., Zhang, X., Zuo, W., Tian, Q.: Controlvideo: Training-free controllable text-to-video generation. arXiv preprint arXiv:2305.13077 (2023) 6