Supplementary Material for MotionBridge: Dynamic Video Inbetweening with Flexible Controls

Maham Tanveer[1,2*] Yang Zhou[2] Simon Niklaus[2]
Ali Mahdavi Amiri[1] Hao Zhang[1] Krishna Kumar Singh[2] Nanxuan Zhao[2]
[1]Simon Fraser University [2]Adobe Research

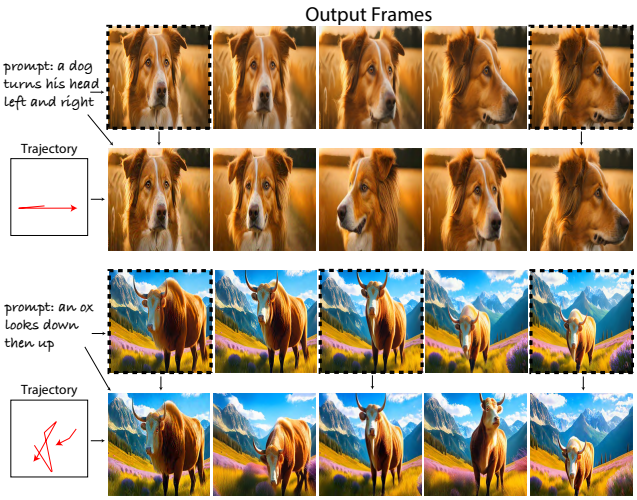

Figure 1. Results demonstrate that capturing detailed motion through prompts alone can be challenging. However, providing trajectories simplifies the process. The top row in both examples illustrates a T2V result generated using only a prompt, while the bottom row highlights how adding a trajectory to extracted keyframes achieves the desired motion effectively.

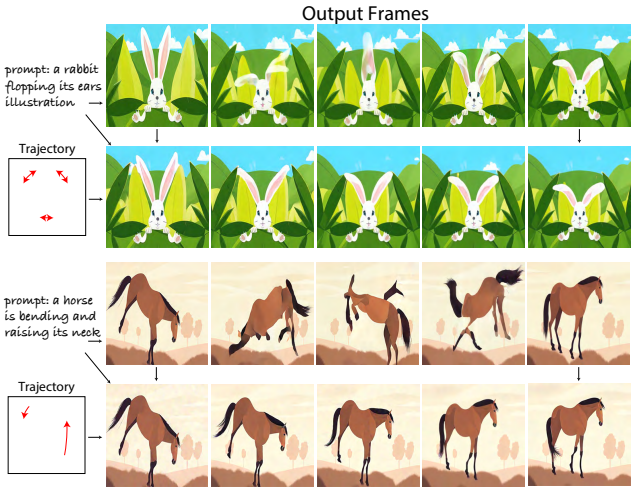

Figure 2. In some cases, the T2V model may generate flickers, artifacts, or undesired deformations. Incorporating trajectories can help refine and improve the output.

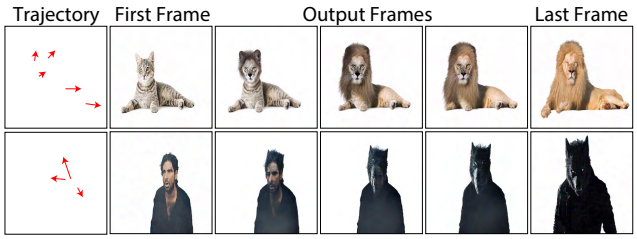

Figure 3. Results showcasing image deformation.

1. Applications

Refining Text-to-Video Outputs With a basic Text-to-Video (T2V) model, text serves as the sole mode of control. However, in cases requiring detailed motion, this often proves insufficient. Additionally, the model may produce erratic or erroneous outputs, particularly when dealing with larger motions. Here, we present examples demonstrating how to refine the outputs of a T2V model, improving both control in Fig. 1 and quality in Fig. 2.

Seamless Image Deformation An application of guiding interpolation is deformation between two images of different characters/subjects. We show some results in Fig. 3 on how our method can be used for this purpose.

Camera Motion Generation Given two input frames of the same object from different camera locations, we use automatic motion extraction via optical flow to generate the inbetween frames in Fig. 4.

Video Temporal Inpainting Given a set of video frames with missing frames in between, our model generates the intermediate frames using automatically extracted sparse motion guidance as shown in Fig. 5.

*Work done during an internship at Adobe.

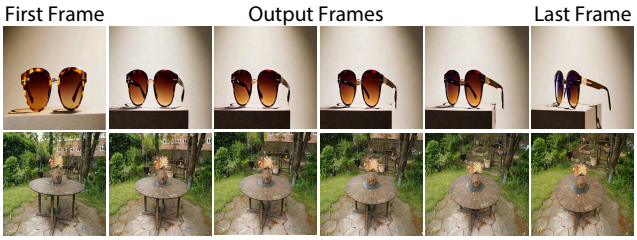

Figure 4. Camera motion results showing a smooth panning to interpolate the two frames.

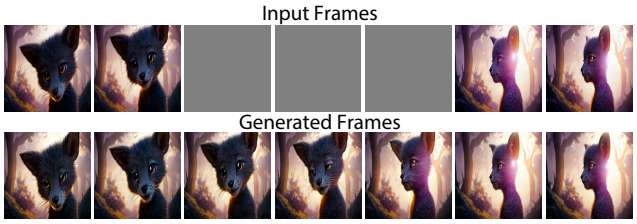

Figure 5. Temporal inpainting results: Given an input sequence with missing frames, the generated frames fill in the gaps.

2. Ablation

Details on Sparse Motion Generator For our Sparse Motion Generator, we aimed to generate motion that seamlessly integrates with the image branch. As in DiT [1], similar to vision transformers, the input image is divided into patches, which complicates maintaining the optical flow's spatial connection to the image. To address this, we sought to replicate the process of image inputs for motion inputs to ensure compatibility. To achieve this, we decided to use the same Variational Autoencoder (VAE) for both image and optical flow/motion inputs. For this analysis, we focus on optical flow, as it provides a clearer means of assessing the effects and calculating output quality.

Since optical flow has two channels, while images have three, the first step was to make it compatible with the VAE requirements. The most straightforward approach was to append a zero channel to the optical flow, effectively creating a three-channel input. Some results from this method are shown in Fig. 6.

Figure 6. Reconstruction results from zero-layer appending method compared to our method. While the appending method follows the motion, it also exhibits more distortions and blurring.

Method	PSNR ↑	LPIPS ↓	SSIM ↑
Appended Layer	17.77	0.34	0.45
Ours	**19.24**	**0.26**	**0.55**

Table 1. Converting to RGB shows a clear improvement in quality compared to appending a zero-layer.

While this approach demonstrated positive results, occasional glitches, delays, and content distortions were observed, likely because the VAE was originally trained for image inputs. To address these issues, we converted the optical flow into an RGB format and repeated the training process. We then tested the recreation of 50 randomly selected videos using full dense optical flow as input. Table 1 illustrates how the RGB conversion improved the quality of the recreated outputs.

#Frames	PSNR ↑	LPIPS ↓	SSIM ↑
1	11.0	0.67	0.15
2	13.8	0.42	0.27
3	15.4	0.35	0.35
4	**15.7**	**0.33**	**0.36**

Table 2. Increasing number of keyframes increases the reconstruction quality.

Multi-Keyframe Input As the model is capable of accepting more than two input images, we present a quantitative analysis in Table 2 showing how an increasing number of images improves reconstruction quality, based on a random set of 25 videos.

References

[1] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. 2