LayerPack: Cross-Layer Consistency in Animation Inbetweening

Atsushi Maruyama
The University of Tokyo,
Tokyo, Japan
atsumaru1377@gmail.com
http://attic.ma-roo.com/

Sosui Koga
Nara Institute of
Science and Technology
Nara, Japan
koga.sosui.ko9@naist.ac.jp

Akinobu Maejima
OLM Digital, Inc.,
IMAGICA GROUP Inc.
Tokyo, Japan
akinobu.maejima@olm.co.jp

Yuki Koyama
The University of Tokyo,
Tokyo, Japan
koyama@pe.t.u-tokyo.ac.jp

I-Chao Shen
The University of Tokyo,
Tokyo, Japan
jdilyshen@gmail.com

Takeo Igarashi
The University of Tokyo,
Tokyo, Japan
takeo@acm.org

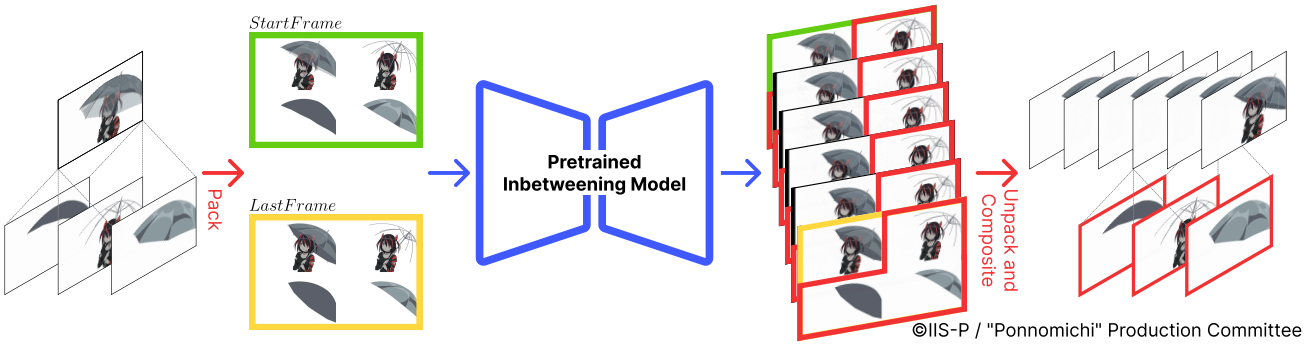

Figure 1: Overview of LayerPack. In this approach, multiple layers are packed into a single image, referred to as a LayerPack Image. The diffusion-based inbetweening model takes two LayerPack Images corresponding to the start and end frames, each containing all layers along with the composited image. The inbetweening model then performs the inbetweening task to generate multiple intermediate frames. Finally, the output LayerPack Images are unpacked into individual layers, resulting in a layer-separated video.

Abstract

In 2D hand-drawn animation, inbetweening, the process of generating intermediate frames between keyframes, requires tremendous manual effort. Recently, diffusion models have been explored for this task and have shown promising results. To apply these models to a professional animation production pipeline, it is necessary to consider support for layered data optimized for alpha-blending-based compositing. The inbetweening results for each layer must remain synchronized when composited, but applying diffusion models to each layer independently leads to a lack of cross-layer consistency. To address these issues, we propose LayerPack, a method for achieving cross-layer consistency in anime inbetweening using a pre-trained model. LayerPack achieves this through a simple yet effective strategy: packing multiple layers into a single input image. Both qualitative and quantitative evaluations demonstrate that LayerPack produces higher-quality and better-synchronized results than per-layer generation approaches.

Keywords: Animation inbetweening, Diffusion model, Layer generation

1. Introduction

Recently, diffusion models targeting image-conditioned video generation have been actively studied, showing impressive results [23, 19, 13, 6]. Inbetweening, one of the tasks in generating videos from images, is to interpolate the motion between two keyframes smoothly. Inbetweening is strongly associated with 2D hand-drawn animation production. Inbe-

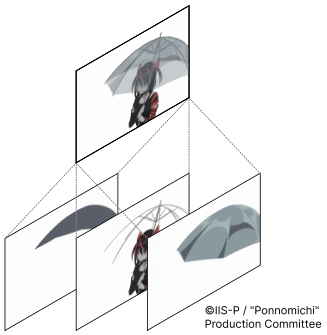

Figure 2: Separated layers which are commonly produced in 2D hand-drawn animation.

tweening work requires a significant amount of manual labor. Therefore, to consider an application in production, research that focuses on animation datasets for model training [10, 15] and research about animation-targeted inbetweening models [4, 16] is being conducted.

One major challenge when considering the application to production is the existence of layers. Production frequently utilizes layered data, as shown in Figure 2. By compositing these layers with alpha blending, the richness of expression in animation can be significantly expanded. Furthermore, it facilitates the explicit representation of foreground-background order and object structures, making it highly compatible with editing workflows. Consequently, layer data is treated as independent images, and inbetweening is performed independently for each layer. The inbetweening results for each layer must not only smoothly connect the start and end frames but also remain spatially synchronized when composited with other layers.

However, performing the inbetweening task with diffusion models while preserving the layer separation is technically challenging for the following reasons.

- Handling the transparency dimension (alpha channel) is inherently difficult. The appropriate method for representing layers (e.g, alpha channel, mask image) is still under investigation [2, 9, 3].
- Animation datasets containing multi-layer structures are unavailable. One exception is the dataset proposed in [18], but unfortunately, it is different from the data used in professional animation workflows.
- The number of layers varies depending on the scene. Designing and training diffusion-based models that handle varying numbers of layers is highly challenging.
- Modeling the cross-relationships between layers is difficult. One of the most straightforward approaches is to apply inbetweening diffusion models to each layer independently; however, the generated motions across layers become inconsistent both spatially and temporally. (See the baseline case in Section 5.2.4 and Section 5.2.5)

To address these issues, we propose LayerPack, a method that enables existing pretrained diffusion models to perform multi-layer inbetweening that ensures synchronized motion across layers (Figure 1). To achieve this, LayerPack simply packs multiple-layer content into a grid and sends it to a standard diffusion inbetweening model. This image packing approach allows layers to be represented without using transparency channels because each layer is spatially independent within the image. Furthermore, as other packing systems [21] demonstrate, information propagates between each grid cell, achieving cross-layer consistency. This approach does not require any additional training or fine-tuning of the diffusion model, facilitating the adoption of existing models.

We evaluated our method using a real-world anime dataset provided by a professional animation studio. Through both qualitative and quantitative evaluations, we demonstrate that LayerPack produces higher-quality results than the baseline approach of generating each layer's video independently in real-world settings. We also demonstrated that LayerPack's method works not only with specific generation models but also with a variety of generative models.

Our contributions are as follows.

- We propose LayerPack, a packing method that enables existing video inbetweening models to handle layer-structured data without additional training.
- We demonstrate that the proposed method achieves better cross-layer consistency in both temporal and spatial domains than existing approaches.

2. Related Work

2.1. Image-to-Video Generation

Video generation models have been extensively studied. In Image-to-Video generation, existing methods are broadly classified into two categories: extrapolation from a single-image condition and inbetweening between two given keyframes. These approaches have advanced through frameworks such as generative adversarial networks [1], autoencoders [5], and more recently, diffusion models utilizing U-Net or Transformer architectures [17, 12, 7].

In particular, the rise of diffusion transformers has significantly improved temporal coherence and visual quality. For example, CogVideoX adopts a 3D variational autoencoder and expert transformer for long-duration, high-resolution video generation [19]. Open-Sora [23], HunyuanVideo [6], and Wan [13] further expand the accessibility and scalability of video generation, enabling efficient text-to-video

and image-to-video synthesis. Additionally, FramePack proposes a context-packing strategy for efficient long-sequence generation [21].

While most of these models are trained on real-world video data, their direct application to 2D hand-drawn animation is not always appropriate. Several studies have developed specialized models and datasets for animation, such as ATD-12K [10] for animation interpolation and AVC [15] for super-resolution. Methods like ToonCrafter [16] leverage diffusion priors to animate keyframe-based cartoon images, while AniSora [4] presents an integrated framework for controllable animation generation and evaluation.

2.2. Layered dataset

Dataset containing layer information is extremely scarce. DAVIS datasets [8] was created as a benchmark for video segmentation and contains information closely resembling a layered structure, but 2D animation data is not included. MULAN [11] is a dataset that decomposes images into layers and performs instance completion to reconstruct occluded areas, but each data is a single image, not a video sequence.

LayerAnimate [18] attempted to create a layered dataset of 2D animation, but unfortunately, it is different from the data used in professional animation workflows. Holding different objects on the same layer and assigning only a one-hot layer ID to each pixel prevents the full benefits of the layer structure from being exploited. The issue of animation layered datasets in research remains unresolved.

2.3. Layer-Wise and Transparency-Aware Generation

Layered representations play an important role in both design and animation, allowing flexible composition and fine-grained control. However, the explicit modeling of transparency and multiple layers remains a challenging topic. LayerDiffuse [20] and Alfie [9] address RGBA image generation by enabling transparent-layer synthesis directly from pretrained diffusion models. At the video level, TransPixeler [14] extends diffusion transformers for consistent RGBA video generation through LoRA-based fine-tuning. LayerDiff [2] introduces a text-guided image diffusion framework for composable, multi-layered image synthesis using inter- and intra-layer attention. LayerFlow [3] extends this concept to video by generating per-layer sub-clips with layer embeddings and staged training, but they mentioned that generating a variable number of layers is their limitation. In the animation domain, LayerAnimate [18] is designed to control each layer with different magnitudes of motion or different user-input guidance, but it fails to maintain cross-layer consistency.

Large diffusion-transformer models have dramatically advanced video generation, and recent works have begun exploring RGBA and layer-aware synthesis. However, existing approaches rarely address multi-layer animation inbetweening with explicit cross-layer consistency, which is the focus of this study.

3. Background: Layers in Professional Animation Production

In the production of 2D hand-drawn animation, the layered data has been a common practice. This section discusses the purposes of layer separation in animation production and outlines the major challenges in research that focuses on layer-based representations.

3.1. Purpose of Layer Separation

A layer refers to a structural representation in which a single frame is decomposed into multiple objects or parts (Figure 2). Each layer is individually drawn and subsequently composited in a predefined order to construct the final scene. The use of layers in animation production serves multiple purposes.

One primary purpose of layer separation is to enhance the visual richness of the final composited frame. Each layer is typically represented in the RGBA format, enabling the depiction of transparent objects. For instance, as illustrated in the Figure 2, the transparent portion of a vinyl umbrella can be drawn on a separate layer; by adjusting its opacity and overlaying it on other layers, complex visual effects can be easily achieved.

Another advantage of layer separation lies in its ability to represent the structure of occluded objects explicitly. When layers are organized according to semantically meaningful units, such as individual characters, the connectivity and internal structure of objects within each character become clearer. Moreover, this approach allows for the explicit definition of spatial relationships, including foreground–background order.

From a production perspective, the use of layers also contributes to the reduction of labor. Static elements, for example, need not be redrawn in every frame. By separating static and dynamic components across layers, redundant drawing tasks can be minimized, thereby improving overall production efficiency. Even using diffusion models for generation, layers provide significant flexibility when checking and editing generated results. So it is highly valuable to generate each layer individually.

3.2. Challenges in Layer-Oriented Research

Despite its practical importance, research focusing on layer-based representations in 2D animation faces several challenges.

First, datasets containing multi-layered data are extremely limited. While some datasets, such as MULAN [11], decompose single photorealistic images into multiple layers, and others provide ground-truth foreground–background masks for video matting tasks, there currently exist no publicly

available datasets of multi-layered 2D hand-drawn animation. LayerAnimate [18] has proposed a Layer Curation Pipeline for constructing such a multi-layered 2D hand-drawn animation dataset; however, several issues remain. (1) The segmentation model employed was trained primarily on real-world images, which limits the accuracy of layer separation when applied to anime-style imagery. (2) The proposed approach focuses mainly on motion vectors, thus failing to capture the semantic and structural advantages of meaningful layer decomposition discussed above.

Second, the treatment of transparency poses a significant challenge. Most image and video diffusion models are trained on RGB data without an alpha channel, making it difficult to generate RGBA images or videos directly. Existing research on RGBA video generation [20, 9, 14] primarily targets text-to-video synthesis or video matting tasks, rather than hand-drawn animation.

Finally, the number of layers should ideally be variable, depending on the number and type of objects present within a scene. Furthermore, layers are not necessarily independent; cross-layer continuity and interactions across multiple layers must be modeled to represent complex motion and spatial relationships accurately.

4. Method: LayerPack

LayerPack is a method for generating multi-layer inbetweening with cross-layer consistency utilizing pretrained inbetweening models without additional training or finetuning. Section 4.1 describes the problem definition, and Section 4.2 describes our novel layer-wise frame representation.

4.1. Problem Definition

We target the *layer-wise inbetweening* task. The input consists of a start frame and an end frame, both separated into layers. We aim to generate the intermediate frames for each layer and composite them to create the inbetweening frames. Let the total number of animation frames be $T+1$ and the number of layers be L, where the frame index is $t \in \{0, 1, 2, \ldots, T\}$. The layer index l is ordered such that the smaller values correspond to the background and the larger values to the foreground.

The image of layer l in frame t is denoted as

$$X_t^{(l)} \in \mathbb{R}^{3\times H\times W}. \tag{1}$$

The input consists of all layer images for the keyframes $t = 0$ and $t = T$, i.e.,

$$\{X_t^{(l)} \mid t \in \{0, T\},\ l \in \{0, 1, \ldots, L-1\}\}. \tag{2}$$

The output is the set of all layer images for all frames:

$$\{X_t^{(l)} \mid t \in \{0, 1, \ldots, T\},\ l \in \{0, 1, \ldots, L-1\}\}. \tag{3}$$

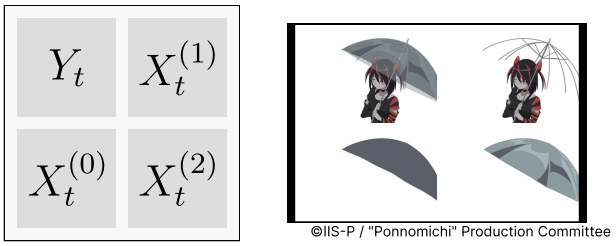

Figure 3: The composited image and individual layer images are packed into a single image. For example, for a sequence with three layers, the four images (three layers and one composited image) are arranged in a 2×2 grid.

For compositing, alpha-blending is applied in the ascending order of the layer indices. That is,

$$Y_t = C(X_t^{(0)}, \ldots, X_t^{(L-1)}), \tag{4}$$

where C represents the alpha compositing operation.

4.2. Layer-Wise Frame Representation

In the proposed method, LayerPack, we pack the input and output images into a grid. We pack multiple layers into a single image for the following reasons:

- It enables the representation of layer data as standard RGB images without requiring alpha channels.
- Because the packed image can be processed as a single image, existing inbetweening models can be directly applied without architectural modification.

The packed image is referred to as a *LayerPack Image* (Figure 3) and is defined as follows.

The LayerPack Image P_t for frame t is constructed by spatially concatenating all layer images for frame t along with the composited image obtained by their alpha blending. That is, P_t contains the L layer images

$$\{X_t^{(l)} \mid l \in \{0, 1, \ldots, L-1\}\} \tag{5}$$

and the composited image Y_t, resulting in a total of $L+1$ images. These images are arranged on a grid of K rows and K' columns ($K' = K$ or $K+1$). Each cell in the grid contains an image of size (H, W), so the entire LayerPack Image P_t forms a single image of size $(K \times H,\ K' \times W)$.

4.3. Inbetweening with LayerPack

By using the LayerPack Images at the keyframes, P_0 and P_T, as inputs to an inbetweening model, we can generate a set of LayerPack Images at the intermediate frames, $\{P_t\}_{t=0}^{T}$ (Figure 1). For this, since our method is model-agnostic, we can use existing pre-trained inbetweening models, such as FramePack [21], FCVG [24], ToonCrafter [16], and AniSora [4], as the backbone without additional training.

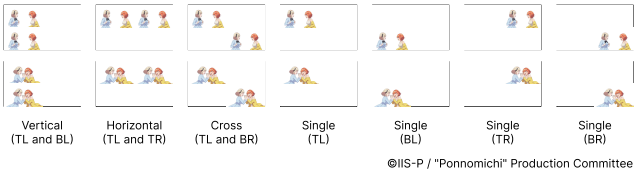

Figure 4: Comparison between packing patterns. Various packing patterns for the preliminary experiment to verify the effectiveness of the packing strategy.

Table 1: Quadrant synchronization evaluation results (mean $\pm$ std over $n = 30$ samples). Higher PSNR and lower LPIPS indicate stronger synchronization (more similar outputs).

Pattern	Pair	PSNR ↑	LPIPS ↓
vertical	tl–bl	37.91 ± 2.05	0.0226 ± 0.0070
horizontal	tl–tr	$\mathbf{37.95} \pm 1.70$	$\mathbf{0.0223} \pm 0.0086$
cross	tl–br	35.73 ± 1.95	0.0311 ± 0.0094
single	tl–tr	28.94 ± 3.16	0.0530 ± 0.0213
	tl–bl	29.83 ± 3.47	0.0468 ± 0.0223
	tl–br	29.20 ± 2.80	0.0545 ± 0.0225
	tr–bl	29.07 ± 3.05	0.0548 ± 0.0233
	tr–br	29.09 ± 3.22	0.0517 ± 0.0227
	bl–br	29.71 ± 3.21	0.0517 ± 0.0235

4.4. Post-processing

After obtaining the generated results, we conduct post-processing. Post-processing has three stages. First, we split the generated video based on the grid used during packing and treat each segment as a generated video for each layer.

Second, most diffusion-based inbetweening models are trained with a fixed resolution. Since our proposed method packs multiple images into a single image, resizing it to fit the input resolution of the inbetweening model reduces the resolution of each individual layer, resulting in lower-resolution outputs than the original keyframes. So we apply super-resolution, restoring each layer to its original resolution.

Finally, we perform alpha blending to obtain the composite video. Since the output image is RGB, we perform luminance-key-based matting to mask the white areas that are to be treated as the background.

5. Experiments and Results

5.1. Evaluation on Packing Effectiveness

Packing patterns. As shown in Figure 4, we evaluate seven packing patterns using a four-quadrant layout: Top-Left (TL), Top-Right (TR), Bottom-Left (BL), and Bottom-Right (BR). The patterns differ in how the same input image is placed across quadrants. In the **Vertical** pattern, the image is placed in the left two quadrants (TL and BL). In **Horizontal**, it is placed in the top two quadrants (TL and TR), while in **Cross**, it is placed in diagonal quadrants (TL and BR). In the **Single** pattern, the image is placed in only one quad-

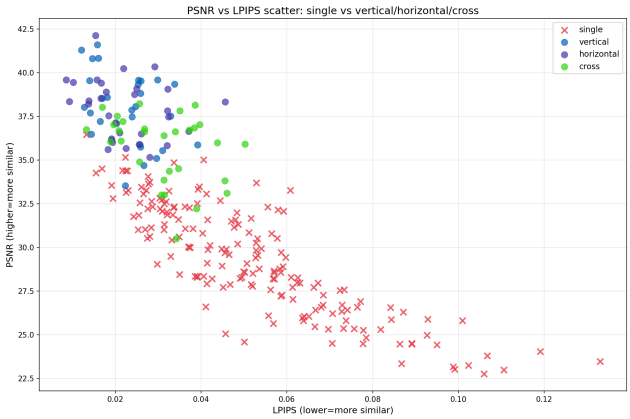

Figure 5: Comparison between packed and single-layer generation methods. PSNR and LPIPS distributions across all test cases are shown.

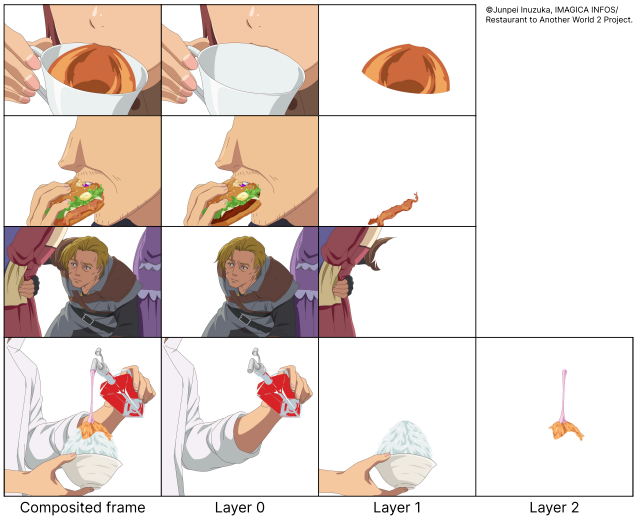

Figure 6: Examples within the evaluation dataset. Consists of animation data with two or three layers.

Table 2: Quantitative comparison. We conducted a quantitative evaluation on ten different scenes. Our method significantly outperforms baseline methods.

method	PSNR (↑)	LPIPS (↓)
LayerPack (Ours)	**20.63**	**0.174**
Baseline (Layer-by-Layer)	12.40	0.245

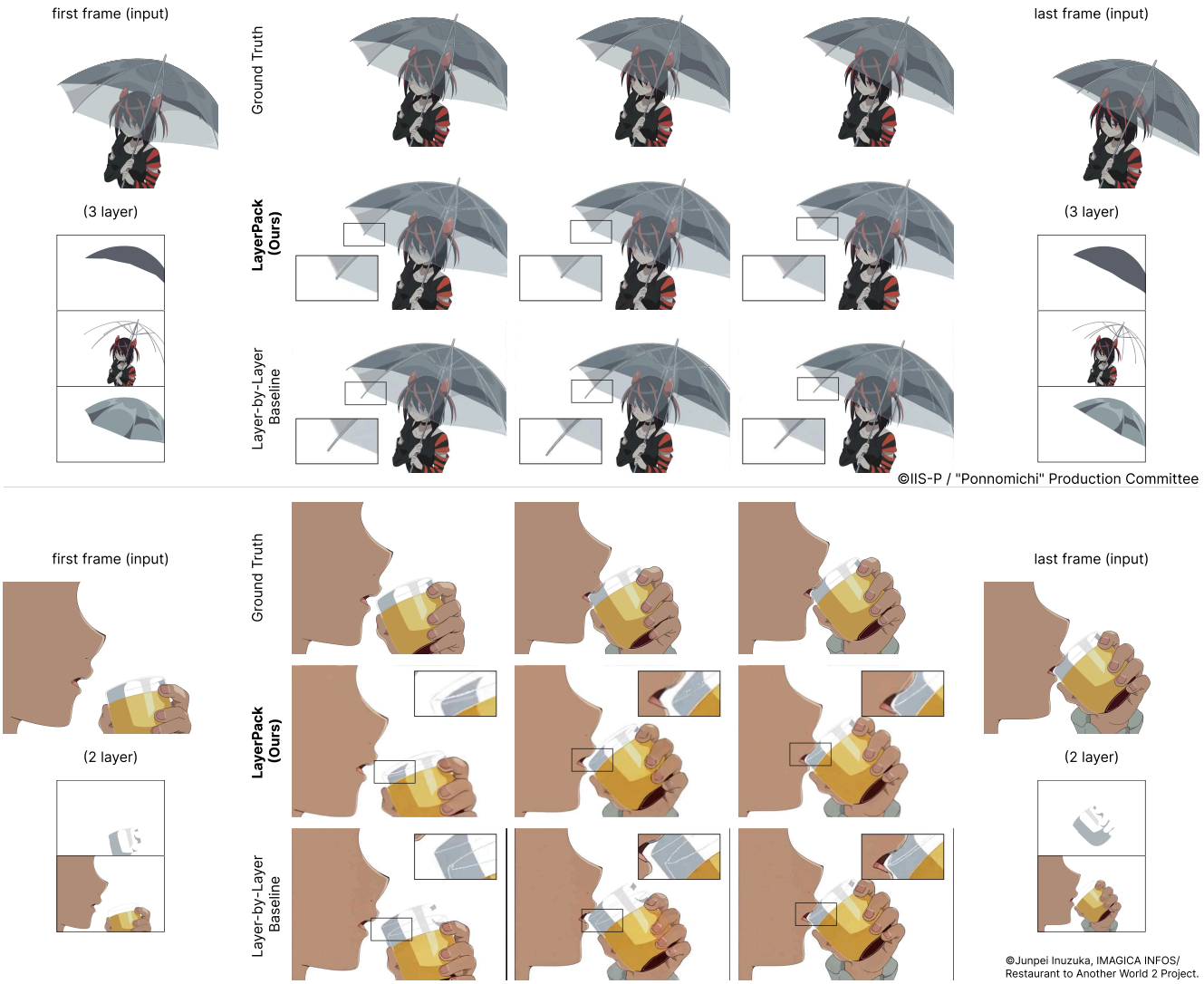

Figure 7: Qualitative Comparison 1. Comparison between the ground truth, the proposed method, and the layer-by-layer baseline. In the regions highlighted by boxes, our method produces synchronized layers, whereas the layer-by-layer baseline does not. Furthermore, our result is more similar to the ground truth frame.

rant, resulting in four variants depending on the occupied quadrant (TL, TR, BL, or BR).

Video Generation Settings. We use FramePack [21] as the backbone model to generate video inbetweening results conditioned on two keyframes. Experiments are conducted on six different scenes, and for each scene and packing pattern, we generate results using five different random seeds to account for stochastic variation.

Evaluation Settings. For each generated RGB frame $P_t \in \mathbb{R}^{3\times H\times W}$, we partition the image into four quadrants and evaluate video synchronization between regions that contain identical input content. For the **Vertical**, **Horizontal**, and **Cross** patterns, we compare the corresponding non-blank quadrant pairs within the same video. For the **Single** patterns, we compare non-blank quadrants across different videos to assess whether consistent motion is generated regardless of the quadrant location. We use Peak Signal-to-Noise Ratio (PSNR), Learned Perceptual Image Patch Similarity (LPIPS) [22] as evaluation metrics.

Results. Quantitative results are summarized in Table 1, and the distributions of PSNR and LPIPS across all test cases are shown in Figure 5. Statistical significance is evaluated using one-way ANOVA followed by Tukey's Honestly

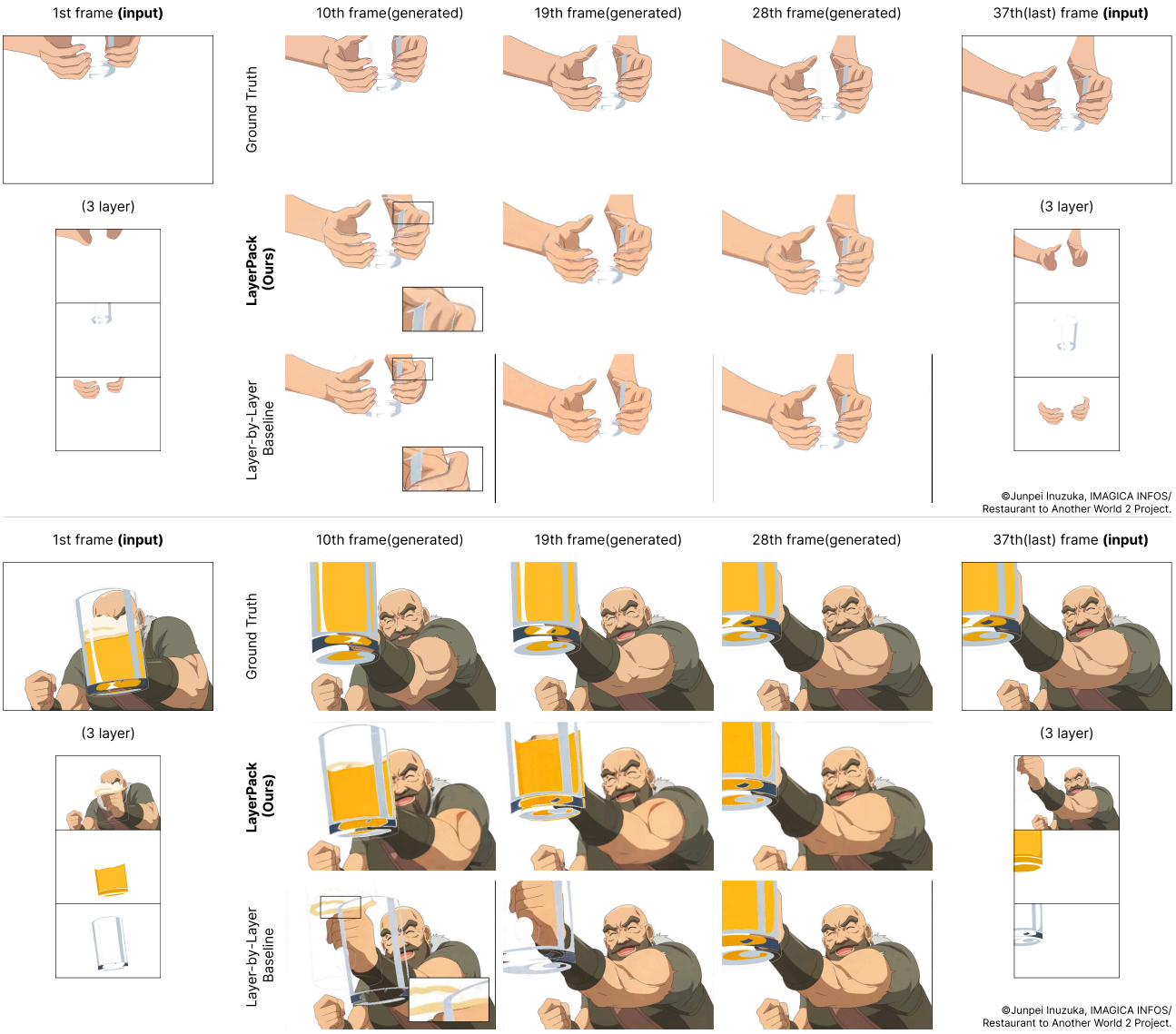

Figure 8: Qualitative Comparison 2. Comparison between the ground truth, the proposed method, and the layer-by-layer baseline. In the regions highlighted by boxes, our method produces synchronized layers, whereas the layer-by-layer baseline does not. Furthermore, our result is more similar to the ground truth frame.

Significant Difference (HSD) test.

ANOVA revealed a significant effect of the packing pattern on PSNR ($F = 112.35$, $p < 0.001$). Two-quadrant packing patterns (**Vertical** and **Horizontal**) achieved the highest PSNR, significantly outperforming all **Single** patterns ($p < 0.05$), with no significant difference between **Vertical** and **Horizontal**.

For LPIPS, ANOVA also showed a significant main effect ($F = 25.08$, $p < 0.001$). Two-quadrant packing patterns (**Vertical**, **Horizontal**, and **Cross**) achieved significantly lower LPIPS scores than **Single** patterns ($p < 0.01$), indicating stronger perceptual synchronization.

5.2. Evaluation on LayerPack

5.2.1 Evaluation Dataset

We evaluated our method using one-second anime-style videos where artists manually separated the foreground object into two or three layers. This data was provided with permission from a professional animation studio. From this real-world data, we prepared 40 different scenes where layers were effectively used. An example of the data is shown in Figure 6. This dataset contains a significant amount of data that overlaps with other layers, as well as data that shares shapes or boundaries with other layers.

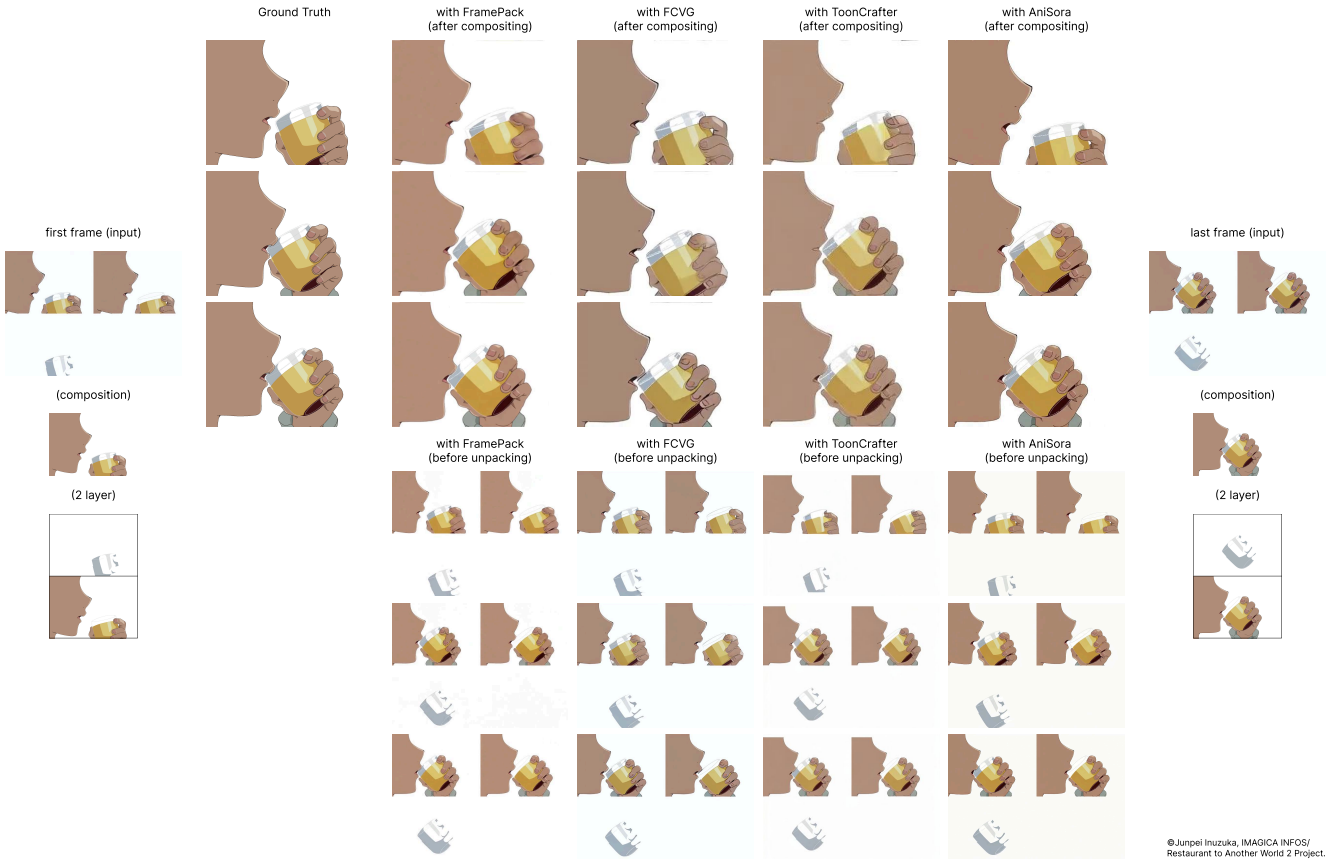

Figure 9: Validation test to demonstrate LayerPack's model-agnostic property. We demonstrate that the LayerPack approach is applicable to many inbetweening models in recent research.

Note that existing datasets could not be used in our evaluation. Most existing animation datasets lack layer information [10, 15] or lack images in a continuous sequence with layers [11]. The datasets used in LayerAnimate [18] were not publicly available.

5.2.2 Implementation

We evaluated our LayerPack method primarily using the pretrained model of FramePack[1] [21]. We chose FramePack as our backbone since it is one of the state-of-the-art inbetweening models and performs well for 2D animation data without additional fine-tuning with an animation dataset.

In our experiments, we apply bilinear super-resolution to the generated videos to restore them to the same resolution as the baseline method for a fair comparison.

[1]Note that our method, LayerPack, should not be confused with FramePack. Although the names are similar, the packing strategy and the underlying objectives are entirely different. The two methods are orthogonal and can be used together within the same pipeline.

5.2.3 Comparison

We construct a baseline by generating each layer sequence independently. Given the layer-separated keyframes

$$\{(X_0^{(l)}, X_T^{(l)}) \mid l \in \{0, 1, \ldots, L-1\}\}, \tag{6}$$

we run the same pretrained inbetweening model M L **times**, once for each layer:

$$\hat{X}_{0:T}^{(l)} = M\left(X_0^{(l)}, X_T^{(l)}; s\right), \quad l \in \{0, 1, \ldots, L-1\}, \tag{7}$$

where s denotes the random seed. For a fair comparison and to reduce stochastic variation, we use the **same seed** s (i.e., the same initial noise) for all L inference runs.

Similarly to the proposed method, we apply lumminance-key based matting to obtain alpha channels for each layer andthen composite the generated layers in the predefined order to obtain the final inbetween frames:

$$\hat{Y}_t = C\left(\hat{X}_t^{(0)}, \ldots, \hat{X}_t^{(L-1)}\right), \quad t \in \{0, 1, \ldots, T\}. \tag{8}$$

We also use FramePack [21] as the backbone model for this baseline method. This is because the comparison focuses not on differences in backbone architecture, but on the

distinction between *independent layer-wise generation* and *joint generation with LayerPack*.

We have not compared LayerAnimate [18], which handles layer-wise inbetweening. The objective of this research is to maintain cross-layer consistency, an issue LayerAnimate does not address. Furthermore, LayerAnimate cannot handle data where multiple layers overlap, therefore, it cannot utilize the evaluation dataset.

5.2.4 Quantitative Evaluation

For quantitative evaluation, we used Peak Signal-to-Noise Ratio (PSNR) and Learned Perceptual Image Patch Similarity (LPIPS) [22] as metrics for the generated results. These metrics are not intended to measure cross-layer consistency. To our knowledge, no metrics exist for evaluating cross-layer consistency, so we applied these metrics based on the following hypothesis: when the videos in each layer are synchronized temporally and spatially, the synthesized frames will have fewer unnatural parts and artifacts and exhibit higher perceptual fidelity. The lack of evaluation metrics is addressed in more detail in Section 6.

Our proposed method achieved scores better than both the baseline method, which composites outputs generated layer by layer, and the method that uses composite keyframes as input (Table 2) .

5.2.5 Qualitative Evaluation

The results of the qualitative evaluation are shown in Figure 7 and Figure 8. As highlighted in the boxes in these figures, the proposed method produces synchronized motion between independently generated layers, especially around object boundaries. Consistency was achieved in both temporal and spatial terms (see the supplementary video).

5.3. Validation of Model-Agnostic Property

We conducted additional experiments to validate LayerPack's model-agnostic property. For this purpose, we applied LayerPack to FramePack [21], FCVG [24], ToonCrafter [16], and AniSora [4]. We did not perform any additional training, as in other experiments. Figure 9 shows the results. For all methods, the composite results in the top row show spatial consistency, and there is no misalignment between the cup and its contents, as seen with the baseline method in the bottom of Figure 7.

6. Discussion

Failure cases One limitation of our method is that it may result in failures when objects move beyond the grid cell boundaries (Figure 10). In the first frame, the character is cropped off at both the top and bottom, and the two vertically stacked layers connect unnaturally, resulting in an unclear

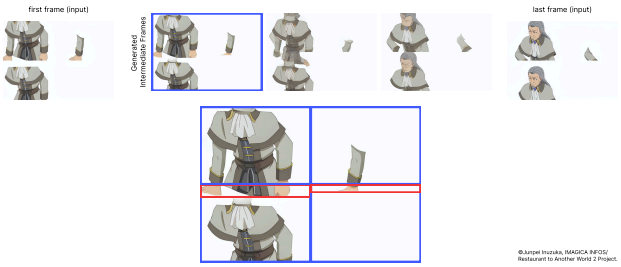

Figure 10: Failure case. The blue frame indicates the implicit boundary between layers, but in the generated intermediate frames, the object crosses this boundary (red frame).

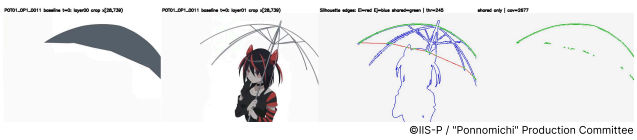

Figure 11: An example of edge-based cross-layer consistency evaluation. Detect boundaries and evaluate whether elements shared across multiple layers (green) are consistent.

boundary. This issue arises because the packing process divides the image into fixed grid cells, but the information about the boundaries between layers is not explicitly provided to the generative model. Consequently, when an object drawn near the boundary moves across it, the model may fail to maintain grid cell boundaries, leading to artifacts or unnatural transitions.

Evaluation metrics In this study, we conducted quantitative evaluations using PSNR and LPIPS metrics to assess cross-layer consistency. However, these metrics are intended to evaluate pairwise frame similarity and do not directly measure cross-layer consistency.

However, designing such metrics is difficult because cross-layer consistency involves complex relationships across multiple layers. These relationships include, but are not limited to, object interactions or tracking, overlay effects, temporal coherence, and structural connectivity across layers.

One possible direction is to define spatial consistency in terms of trackable correspondences across layers, such as points, edges, or regions. For example, a metric could be designed to evaluate consistency based on the hypothesis that edges appearing at similar spatial locations across multiple layers can be regarded as candidates for boundaries that should be shared among layers.(Figure 11) This evaluation does not assume that all edges should be shared between layers; it is performed only on the specific edges that should be shared. For temporal consistency evaluation, we attempted to design a metric based on optical-flow-based warping. However, most 2D hand-drawn animations are

largely produced using flood-fill-style coloring, resulting in large, flat-color regions with very limited texture or luminance variation, making the metric unstable for tracking temporal motion.

An important future work will be to explore the above methods further and consider ways to more directly assess consistency across layers.

Benchmark dataset limitations In this study, we used a real-world animation dataset from a professional animation studio. This dataset is valuable as it reflects actual production workflows, but it is not publicly available, so other researchers cannot replicate the evaluation under the same conditions.

In the field of animation, the absence of a common dataset is not only a problem of insufficient training data but also a barrier to fair comparison among different methods. To advance research in this area, it is crucial to establish publicly available datasets that accurately represent real-world production scenarios. Such datasets would enable researchers to benchmark their methods effectively and encourage innovation in layer-aware animation generation techniques.

Resolution limitation due to packing Our proposed method can handle varying numbers of layers. This is very useful in real-world applications where the number of layers varies by scene, and some scenes may have more than 10 layers.

However, because the generative model's output resolution remains constant, increasing the number of layers decreases the resolution of each grid cell.

FramePack [21] only supports 480p and AniSora [4] supports 480p and 720p. Therefore, the resolution of each grid cell is limited by the generative model's output resolution. For example, when using FramePack with 480p output resolution, if there are four layers, each grid cell can have a maximum resolution of 240p (2x2 grid). If there are nine layers, each grid cell can have a maximum resolution of 160p (3x3 grid). This limitation may lead to lower-quality outputs, especially when the number of layers is large. Super-resolution can help mitigate this issue, but it may not fully recover the lost details.

Investigating super-resolution methods specialized to LayerPack There is room for further investigation into super-resolution in the future. Our current implementation utilizes simple bilinear super-resolution; however, we anticipate that a reference-based super-resolution approach that uses the original resolution's keyframes as a reference may yield better results.

7. Conclusion

We proposed *LayerPack*, a packing method that enables existing video inbetweening models to handle layer-structured data without additional training or finetuning. Our results demonstrate that even when a single object is separated into multiple layers, as is common in data used in real anime production environments, the generated motion remains synchronized both temporally and spatially.

Acknowledgement

This work is based on results obtained from GENIAC (Generative AI Accelerator Challenge, a project to strengthen Japan's generative AI development capabilities), a project implemented by the Ministry of Economy, Trade and Industry (METI) and the New Energy and Industrial Technology Development Organization (NEDO). This work was also supported by JST, CRONOS, Japan Grant Number JPMJCS25K1.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *Commun. ACM*, 63(11):139–144, Oct. 2020. 2

[2] R. Huang, K. Cai, J. Han, X. Liang, R. Pei, G. Lu, S. Xu, W. Zhang, and H. Xu. LayerDiff: Exploring Text-guided Multi-layered Composable Image Synthesis via Layer-Collaborative Diffusion Model. arXiv, Mar. 2024. arXiv:2403.11929 [cs]. 2, 3

[3] S. Ji, H. Luo, X. Chen, Y. Tu, Y. Wang, and H. Zhao. LayerFlow: A Unified Model for Layer-aware Video Generation. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, SIGGRAPH Conference Papers '25, New York, NY, USA, 2025. Association for Computing Machinery. 2, 3

[4] Y. Jiang, B. Xu, S. Yang, M. Yin, J. Liu, C. Xu, S. Wang, Y. Wu, B. Zhu, X. Zhang, X. Zheng, J. Xu, Y. Zhang, J. Hou, and H. Sun. AniSora: Exploring the Frontiers of Animation Video Generation in the Sora Era. arXiv, May 2025. arXiv:2412.10255 [cs]. 2, 3, 4, 9, 10

[5] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. *CoRR*, abs/1312.6114, 2013. 2

[6] W. Kong, Q. Tian, Z. Zhang, R. Min, Z. Dai, J. Zhou, J. Xiong, X. Li, B. Wu, J. Zhang, K. Wu, Q. Lin, J. Yuan, Y. Long, A. Wang, A. Wang, C. Li, D. Huang, F. Yang, H. Tan, H. Wang, J. Song, J. Bai, J. Wu, J. Xue, J. Wang, K. Wang, M. Liu, P. Li, S. Li, W. Wang, W. Yu, X. Deng, Y. Li, Y. Chen, Y. Cui, Y. Peng, Z. Yu, Z. He, Z. Xu, Z. Zhou, Z. Xu, Y. Tao, Q. Lu, S. Liu, D. Zhou, H. Wang, Y. Yang, D. Wang, Y. Liu, J. Jiang, and C. Zhong. HunyuanVideo: A Systematic Framework For Large Video Generative Models. arXiv, Mar. 2025. arXiv:2412.03603 [cs]. 1, 2

[7] W. S. Peebles and S. Xie. Scalable Diffusion Models with Transformers. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4172–4182, 2022. 2

[8] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation. In *Computer Vision and Pattern Recognition*, 2016. 3

[9] F. Quattrini, V. Pippi, S. Cascianelli, and R. Cucchiara. Alfie: Democratising RGBA Image Generation with No $$$. In A. Del Bue, C. Canton, J. Pont-Tuset, and T. Tommasi, editors, *Computer Vision – ECCV 2024 Workshops*, pages 38–55, Cham, 2025. Springer Nature Switzerland. 2, 3, 4

[10] L. Siyao, S. Zhao, W. Yu, W. Sun, D. Metaxas, C. C. Loy, and Z. Liu. Deep Animation Video Interpolation in the Wild. pages 6583–6591, 06 2021. 2, 3, 8

[11] P.-D. Tudosiu, Y. Yang, S. Zhang, F. Chen, S. McDonagh, G. Lampouras, I. Iacobacci, and S. Parisot. MULAN: A Multi Layer Annotated Dataset for Controllable Text-to-Image Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22413–22422, June 2024. 3, 8

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention Is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. 2

[13] T. Wan, A. Wang, B. Ai, B. Wen, C. Mao, C.-W. Xie, D. Chen, F. Yu, H. Zhao, J. Yang, J. Zeng, J. Wang, J. Zhang, J. Zhou, J. Wang, J. Chen, K. Zhu, K. Zhao, K. Yan, L. Huang, M. Feng, N. Zhang, P. Li, P. Wu, R. Chu, R. Feng, S. Zhang, S. Sun, T. Fang, T. Wang, T. Gui, T. Weng, T. Shen, W. Lin, W. Wang, W. Wang, W. Zhou, W. Wang, W. Shen, W. Yu, X. Shi, X. Huang, X. Xu, Y. Kou, Y. Lv, Y. Li, Y. Liu, Y. Wang, Y. Zhang, Y. Huang, Y. Li, Y. Wu, Y. Liu, Y. Pan, Y. Zheng, Y. Hong, Y. Shi, Y. Feng, Z. Jiang, Z. Han, Z.-F. Wu, and Z. Liu. Wan: Open and Advanced Large-Scale Video Generative Models. arXiv, Apr. 2025. arXiv:2503.20314 [cs]. 1, 2

[14] L. Wang, Y. Li, Z. Chen, J.-H. Wang, Z. Zhang, H. Zhang, Z. Lin, and Y. Chen. TransPixeler: Advancing Text-to-Video Generation with Transparency. arXiv, Jan. 2025. arXiv:2501.03006 [cs]. 3, 4

[15] Y. Wu, X. Wang, G. Li, and Y. Shan. AnimeSR: learning real-world super-resolution models for animation videos. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. 2, 3, 8

[16] J. Xing, H. Liu, M. Xia, Y. Zhang, X. Wang, Y. Shan, and T.-T. Wong. ToonCrafter: Generative Cartoon Interpolation. *ACM Trans. Graph.*, 43(6), Nov. 2024. 2, 3, 4, 9

[17] J. Xing, M. Xia, Y. Zhang, H. Chen, W. Yu, H. Liu, G. Liu, X. Wang, and T.-T. Wong. DynamiCrafter: Animating Open-Domain Images with Video Diffusion Priors. pages 399–417, 10 2024. 2

[18] Y. Yang, L. Fan, Z. Lin, F. Wang, and Z. Zhang. LayerAnimate: Layer-level Control for Animation. arXiv, Mar. 2025. arXiv:2501.08295 [cs]. 2, 3, 4, 8, 9

[19] Z. Yang, J. Teng, W. Zheng, M. Ding, S. Huang, J. Xu, Y. Yang, W. Hong, X. Zhang, G. Feng, D. Yin, Y. Zhang, W. Wang, Y. Cheng, B. Xu, X. Gu, Y. Dong, and J. Tang. CogVideoX: Text-to-Video Diffusion Models with An Expert Transformer. arXiv, Mar. 2025. arXiv:2408.06072 [cs]. 1, 2

[20] L. Zhang and M. Agrawala. Transparent Image Layer Diffusion using Latent Transparency. *ACM Trans. Graph.*, 43(4), July 2024. 3, 4

[21] L. Zhang and M. Agrawala. Packing Input Frame Context in Next-Frame Prediction Models for Video Generation. arXiv, Apr. 2025. arXiv:2504.12626 [cs]. 2, 3, 4, 6, 8, 9, 10

[22] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018. 6, 9

[23] Z. Zheng, X. Peng, T. Yang, C. Shen, S. Li, H. Liu, Y. Zhou, T. Li, and Y. You. Open-Sora: Democratizing Efficient Video Production for All. arXiv, Dec. 2024. arXiv:2412.20404 [cs]. 1, 2

[24] T. Zhu, D. Ren, Q. Wang, X. Wu, and W. Zuo. Generative Inbetweening through Frame-wise Conditions-Driven Video Generation. 2024. 4, 9