Trajectory-guided Anime Video Synthesis via Effective Motion Learning

Jian Lin[1], Chengze Li[1], Haoyun Qin[2], Hanyuan Liu[3], Xueting Liu[1], Xin Ma[4], Cunjian Chen[4], Tien-Tsin Wong[4]

[1]Saint Francis University [2]University of Pennsylvania [3]The Chinese University of Hong Kong [4]Monash University

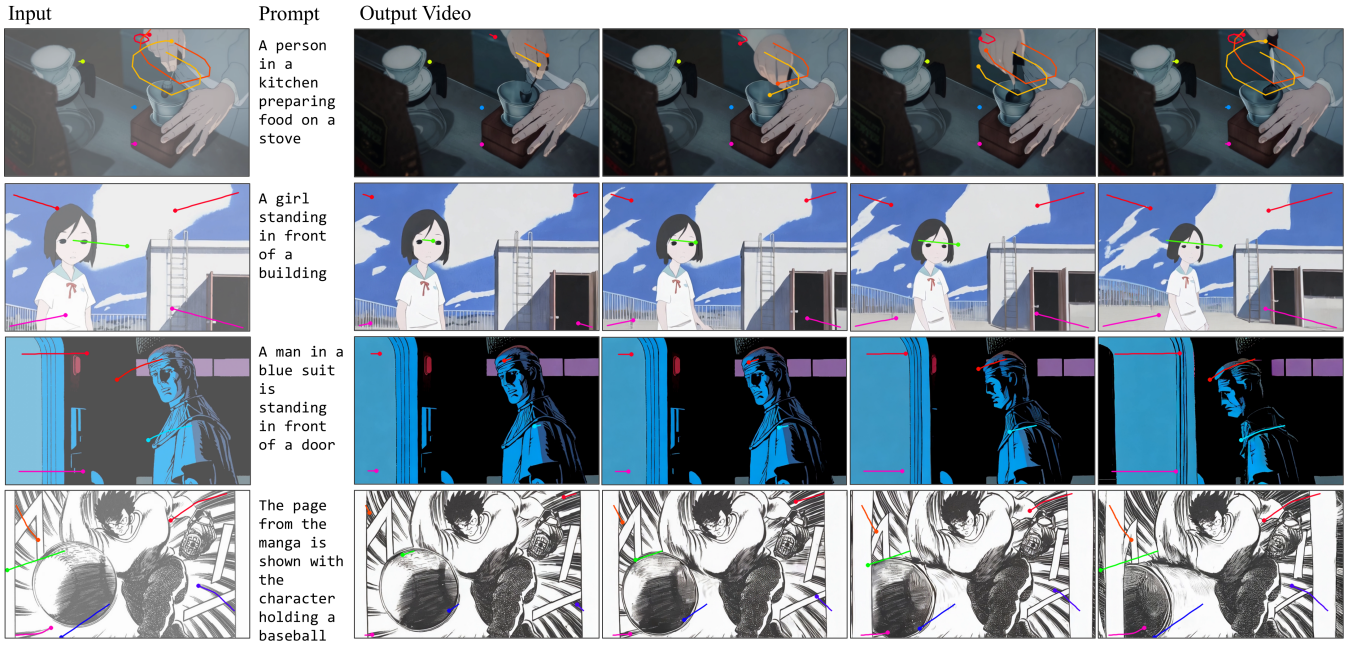


Figure 1: *Given a still frame of cartoon or comics, we will be able to animate it into a video clip guided by the user-specified motion trajectory and text prompts.*

Abstract

Cartoon and anime motion production is traditionally labor-intensive, requiring detailed animatics and extensive inbetweening from keyframes. To streamline this process, we propose a novel framework that synthesizes motion directly from a single colored keyframe, guided by user-provided trajectories. Addressing the limitations of prior methods, which struggle with anime due to reliance on optical flow estimators and models trained on natural videos, we introduce an efficient motion representation specifically adapted for anime, leveraging CoTracker to capture sparse frame-to-frame tracking effectively. To achieve our objective, we design a two-stage learning mechanism: the first stage predicts sparse motion from input frames and trajectories, generating a motion preview sequence via explicit warping; the second stage refines these previews into high-quality anime frames by fine-tuning ToonCrafter, an anime-specific video diffusion model. We train our framework on a novel animation video dataset comprising more than 500,000 clips. Experimental results demonstrate significant improvements in animating still frames, achieving better alignment with user-provided trajectories and more natural motion patterns while preserving anime stylization and visual quality. Our method also supports versatile applications, including motion manga generation and 2D vector graphic animations. The data and code will be released upon acceptance. For models, datasets and additional visual comparisons and ablation studies, visit our project page: https://animemotiontraj.github.io/.

CCS Concepts

• ***Applied computing*** → ***Fine arts;***

DOI: 10.2312/pg.20251258

1. Introduction

Cartoon motion production is traditionally a labor-intensive and time-consuming process. To translate the director's vision for subject, scene, and camera motion, animators must carefully illustrate the animatics and derive the rendering of colored keyframes from them, and then build up the inbetweening of these keyframes. Although this workflow has been a standard practice for decades, it involves a chain of tedious steps, including motion design, animatic rendering, and frame colorization, all of which require significant skill and effort from both directors and animators. In this work, we aim to synthesize animation motion directly from *one single colored keyframe*, guided by user-provided trajectories. These trajectories specify the movements of key regions of interest within the initial frame. This single-frame, trajectory-controlled setting targets the *ideation / animatic previsualization* phase: animators can rapidly explore alternative character and camera motions before investing effort in drawing additional keyframes for traditional two-keyframe inbetweening.

Prior works, such as DragNUWA [YWL*23], MotionCtrl [WYW*24] and Motion-I2V [SHW*24] have explored similar objectives. These methods primarily use optical flow to estimate scene and object motion, and sample user-specified motion trajectories from it. The trajectories are then encoded as feature embeddings to guide a fine-tuning of pre-trained video models for motion synthesis. However, we observe that these methods often fail to produce anime motion frames with visually pleasing subject actions, pose transitions, scene transitions, and detailed object dynamics and deformations. First, all of these methods rely on optical flow estimators to parse motion in anime for the downstream frame synthesis. However, optical flow quality cannot be guaranteed for animations, as the appearance and motion of anime often violate the texture, color, or gradient constraints that optical flow estimators depend on (Fig. 2 (c,e)). Moreover, these methods primarily rely on the generalization ability of existing image-to-video models, without explicitly interpreting and translating the motion hints into 2D correspondences, leading to unsatisfactory outputs. While Motion-I2V predicts explicit optical flow-based motion from the user trajectory and uses the predicted motion to warp the video diffusion latents, we find it difficult to adapt to the anime domain to provide stable and satisfactory motions.

In this work, we propose a novel framework to address these challenges. First, we introduce an efficient sparse motion representation that records displacements on a sparse array of keypoints. Using a high-quality point-based tracker such as CoTracker [KRG*25] which extracts motion from multi-frame correspondences, this representation effectively captures most frame-to-frame tracking and morphing profiles in anime. Motion reconstruction using this representation demonstrates significantly reduced distortions and improved structure preservation, as illustrated in Fig. 2. With this representation, we then develop our framework with a two-stage design. In the first stage, we propose a video diffusion model to predict the sparse motion representation directly from a set of predefined conditions, including the input frame, user trajectories, and text prompts. To support training, we also propose a large-scale anime motion dataset with a novel motion sampling procedure to approximate user trajectories from CoTracker estimations. Using the predicted sparse motion, we warp the initial frame to construct a motion preview sequence. Although this sequence may contain distortions and uncertainties, it serves as an explicit motion prior to lower the learning difficulties for later stages. In the second stage, we leverage the generative prior of an anime-specific video diffusion model, ToonCrafter [XLX*24], and fine-tune this model to interpret motion guidance implicitly from the user-provided trajectories and explicitly from the warped motion preview, to generate high-quality anime video clips. This approach enables trajectory-controlled generation of anime frames with improved details, realistic and trajectory-conforming motion, and robust handling of occlusions.

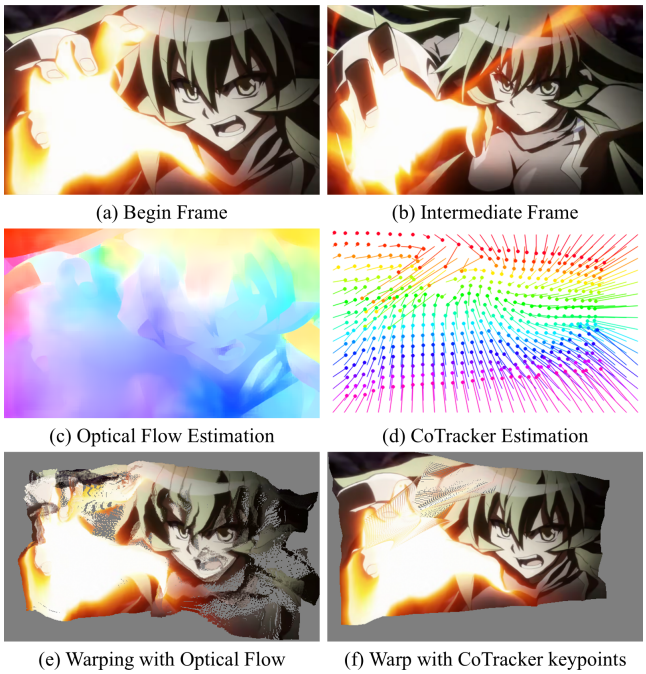


Figure 2: *Sparse motion tracking with CoTracker better estimates inter-frame anime correspondences to maintain the structural components when warping is performed.*

We evaluate our method extensively through both qualitative and quantitative experiments. The results demonstrate that explicit motion guidance and our proposed sparse representations allow our approach to achieve significantly improved performance. Specifically, it ensures better alignment with user-provided motion trajectories and more natural and intuitive motion patterns while preserving anime-specific stylization and frame-level visual quality (Fig. 1 and Fig. 6). In addition, we explore potential applications of our method, including motion generation for manga and vector graphics. These applications showcase the versatility of our approach to enable intuitive and semantically conforming animation production from a wide variety of media forms. We summarize our contributions as follows:

- We propose a novel sparse keypoint-based representation to provide efficient and higher-quality modeling of anime motion, compared with optical-flow based solutions;

- We produce a two-stage framework to encougage explicit learning of motions and leverage the generative prior of ToonCrafter, to assure high-quality video synthesis for anime;
- We propose a novel trajectory sampling and scene parsing approach with CoTracker, to form a new captioned anime clip dataset with over 500k clips;
- We demonstrate the generalization ability of our approach on various media such as manga and 2D vector graphics for potential applications in animation production and design.

2. Related Work

2.1. Anime Motion Understanding and Frame Synthesis

To parse motion in animations, dense optical flow tracking methods [SYLK18, TD20, XZC*22, HSZ*22, SHL*23] can be applied directly to animation frames. However, these methods often struggle on anime frames due to the domain gap in motion characteristics and appearances. The sparse detailing and lack of complex shading typical of anime introduce challenges for traditional optical flow approaches. To address this, various methods estimate or track anime motion using segmentation [ZLWH16, CPL21], outline [MGW24], geometry [DLKS18, SGX*23], or deformation [SBCv*11] correspondences. AnimeInterp [SZY*21] incorporates anime-specific constraints for optical flow estimation using a curated dataset, while Animerun [LLL*22] extends correlation estimation by adapting 3D animations. Despite these advancements, the ground truth motion provided by these methods remains limited in scale, making it insufficient to support large-scale motion prediction or frame synthesis models.

Recent advances in latent video diffusion models [BDK*23, CZC*24, CXH*23, YTZ*24] enable direct video synthesis from a driving image. However, these models often fail to produce satisfactory results for anime keyframes due to the domain gap. ToonCrafter [XLX*24] partially bridges this by fine-tuning on a large-scale anime dataset, allowing smooth motion transitions but requiring an additional ending keyframe. Animatediff also achieves video synthesis by leveraging pre-trained models with a motion adapter. While an anime-look LoRA enables anime synthesis, it is limited in generating diverse appearances and struggles with high-resolution outputs. Alternatively, anime frame generation can be approached as reference-based colorization [SZC*23, HZL24, MOW*25], but this requires per-frame sketch inputs. Video synthesis has also been extended to other non-photorealistic media like rough sketches [GVA*24] and vector graphics [WSML24].

2.2. Trajectory-based Video Synthesis

Trajectory-based video synthesis generates videos from a still image using point-based trajectories to define motion, enabling interactive control over an object's appearance and position. Traditional methods like ARAP [IMH05] and DragGAN [PTL*23] focus on appearance editing but lack natural temporal transitions. DragNUWA [YWL*23] first introduced a pipeline for interactive trajectory-guided video synthesis, including a strategy to sample user trajectories from optical flow and a video diffusion model for generating natural-looking and trajectory-following videos. Subsequent works have extended this by focusing on fine-grained motion region determination [WLG*24], adapting to large-scale video synthesis models [ZLL*25, WHF*25], integrating trajectory-based control for inbetweenings [WWZ*24], or providing separate controls for camera and object motion [WYW*24]. However, these methods rely on tracking estimators like optical flow or SIFT, which are often ill-suited to anime's unique motion and appearance. Furthermore, they lack explicit motion modeling, relying instead on the generative capacity of latent diffusion models, which limits their ability to produce visually pleasing results. PhysAnimator [XZJJ25] transforms user-specified motion into optical flow via physical simulation, but it is designed for specific motions and objects and does not consider multi-object interactions.

Another methodology closely related to our proposed approach is Motion-I2V [SHW*24], as both utilize a two-stage pipeline for motion prediction followed by frame synthesis. This approach first predicts 2D optical flow displacements by fine-tuning a latent video diffusion model. It then fine-tunes the Animatediff motion adapter [GYR*24], using warped latents from the predicted displacements as motion guidance to produce smooth videos. However, when tested in animations, its results were unsatisfactory (see Sect. 5). A key limitation lies in its reliance on optical flow estimators. Additionally, the Animatediff motion adapter struggles to generalize to anime-specific appearances due to the domain gap in its underlying text-to-image synthesis model, Stable Diffusion 1.5 [RBL*22]. Training the warping-guided Animatediff motion modules requires a large amount of training data, which is impractical given the scarcity of large-scale anime datasets. In contrast, our framework adopts a sparse motion representation that is both more efficient and less error-prone compared to the dense optical flow or latent motion transformations used in prior methods. Furthermore, by leveraging a large-scale pre-trained anime-specific model as our generative prior, we achieve higher-quality results without requiring an excessively large dataset. Importantly, our single-frame, trajectory-driven setup targets the earlier previsualization stage, offering animators a fast way to explore motion concepts without the need for extensive keyframe or inbetweening work.

3. Representation of Anime Motion

3.1. Sparse Anime Motion Representation

As outlined in the introduction, we propose to represent anime motion with a sparse motion representation $f \in \mathbb{R}^{L\times H_f\times W_f\times 2}$, which tracks the 2D displacements of predefined keypoints arranged in an evenly distributed $H_f \times W_f$ grid with the frame length L. In our following experiments, we set $H_f = 16$, $W_f = 28$, and $L = 16$. For any tracker point at position i, j (we omit the indices $i, j \in [0, H_f - 1] \times [0, W_f - 1]$ later for simplicity), we will be able to represent the motion as frame-wise displacements:

$$f = \Big((x^0, y^0), (x^1 - x^0, y^1 - y^0), ..., (x^{L-1} - x^{L-2}, y^{L-1} - y^{L-2})\Big), \tag{1}$$

where superscript denotes frame index and x and y denote tracker coordinates. With a reliable keypoint-based tracker such as CoTracker [KRG*25] to estimate the ground truth motion, frame reconstruction with f more closely resembles the original, preserving structure and reducing artifacts compared to optical flow-based methods, as shown in Fig. 2.

3.2. Trajectory Sampling

Besides motion representation, it is also critical to design the format of user-specified motion trajectories and also the trajectory sampling process from raw animation videos. Existing methods often sample (or approximate) trajectories by drawing random points from the optical flow map and tracing their locations. However, as previously discussed, optical flow estimators struggle with estimating anime motions, making it unstable and inaccurate for trajectory sampling. Motion-I2V [SHW*24] employs DOT [LMPS24] trackers to approximate user trajectories, but it lacks a clear explanation of the placement and tracking of tracker points to form them.

To address these issues, we propose a novel trajectory sampling approach by analyzing the most significant modes of motion on the sparse motion representation f, where motion is estimated using CoTracker. Compared with optical flow, we find that CoTracker better exploits the multi-frame correspondences to provide more precise and robust point-based tracking on anime. We illustrate the trajectory sampling process in Fig. 3. Specifically, we first determine the number of user trajectories, n_{traj}, by sampling from a truncated Gaussian distribution. The probability density is defined as $p(n) = d(n)/\sum_{i=n_{min}}^{n_{max}} d(i)$, where the raw density is $d(n) = e^{-|n-m|/\sigma}$. In our experiments, we set $m = 2$, $\sigma = 2$, and restrict n_{traj} to the range $[n_{min} = 1, n_{max} = 10]$. Once n_{traj} is determined, we extract the most significant motion modes by clustering the motion into n_{traj} groups via K-Means [HW79] clustering. We intentionally allow n_{traj}, and thus K, to be smaller than the actual number of moving objects. As a result, distinct motions can share a cluster. This design reduces the burden on the user to specify every moving part and encourages the model to infer plausible motion in unattended regions rather than overfitting to a fully specified set of trajectories. To ensure better locality and mitigate inadvertent grouping of far-apart motions, we introduce a position-aware motion feature $v \in \mathbb{R}^{W_f H_f \times 2L}$, defined as:

$$v = (x^0/16, y^0/16, x^1 - x^0, y^1 - y^0, \ldots, x^{L-1} - x^{L-2}, y^{L-1} - y^{L-2}), \tag{2}$$

where the coordinates of the starting point of any specific tracker point is scaled by $1/16$ to balance numerical scales. Within each cluster, we randomly select a trajectory with a probability proportional to its length relative to the total length of all trajectories in the group, ensuring that the selected trajectory is representative of the cluster and avoiding extremely short or uninformative samples (Fig. 3(d)). After sampling, we form the final sampled user trajectory set $j \in \mathbb{R}^{10 \times L \times 2}$. Each trajectory j_i is represented in frame-wise displacement format, which is the same as Eq. 1. If the total number of trajectories n_{traj} is less than 10, we apply zero padding for unused trajectory entries.

In this work, we simplify the representation j by uniformly resampling the trajectory along the temporal dimension, discarding the per-frame representation of speed and orientation. This reduces the need for users to provide fine-grained motion guidance, which can be tedious. By allowing rough scribbles, we enable previews of clip-level animatics with diverse motion configurations. Despite discarding precise frame timing, our model can still infer suitable motion, including nonlinear and exaggerated movements. This, along with the Gaussian-blurred dense trajectory map J (Sect. 4.1), deliberately removes fine timing signatures and empirically improves robustness to diverse, user-drawn trajectories at inference.

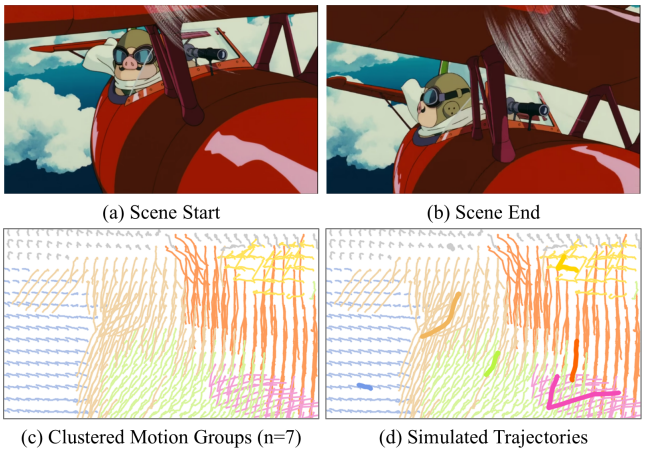


Figure 3: *The trajectory sampling process. To approximate user trajectories, we sample from motion groups that are created by clustering motion data from a sparse motion tracker.*

4. Methodology

We design our framework for trajectory-based animation synthesis in two stages. The first stage predicts sparse motion representations from a single input frame and user-provided trajectories, generating a motion preview sequence via explicit warping. In the second stage, this sequence is refined into high-quality anime videos leveraging ToonCrafter, an anime-specific video diffusion model to ensure enhanced visual quality. The overall illustration of the framework is depicted in Fig. 4.

4.1. Motion Prediction Stage

In the first stage of motion prediction, given an input still image $I \in \mathbb{R}^{H \times W \times 3}$ and the user-provided motion trajectory $j \in \mathbb{R}^{10 \times L \times 2}$, our goal is to predict subsequent frame motions in the form of the sparse motion representation f, as defined in Sect. 3.1. We use a 3D diffusion U-Net [CZC*24] to model and predict the sparse motion over a temporal length of $L = 16$ frames. Since the tracker grid operates at low resolution, we directly use the ground truth f as the diffusion target f_0 at $t = 0$, without applying latent compression. The forward diffusion process [HJA20] gradually adds noise to f_0 as:

$$f_t = \sqrt{\bar{\alpha}_t} f_0 + \sqrt{1 - \bar{\alpha}_t}\,\epsilon, \quad \epsilon \sim \mathcal{N}(0, I), \tag{3}$$

where α_t and $\bar{\alpha}_t$ are diffusion scheduling constants. Here, subscripts represent diffusion timesteps.

To integrate user-provided motion trajectories and other priors into the diffusion pipeline, we design a fusion mechanism that categorizes conditioning inputs into two types: those injected via cross-attention [RBL*22] and those concatenated directly to f_t

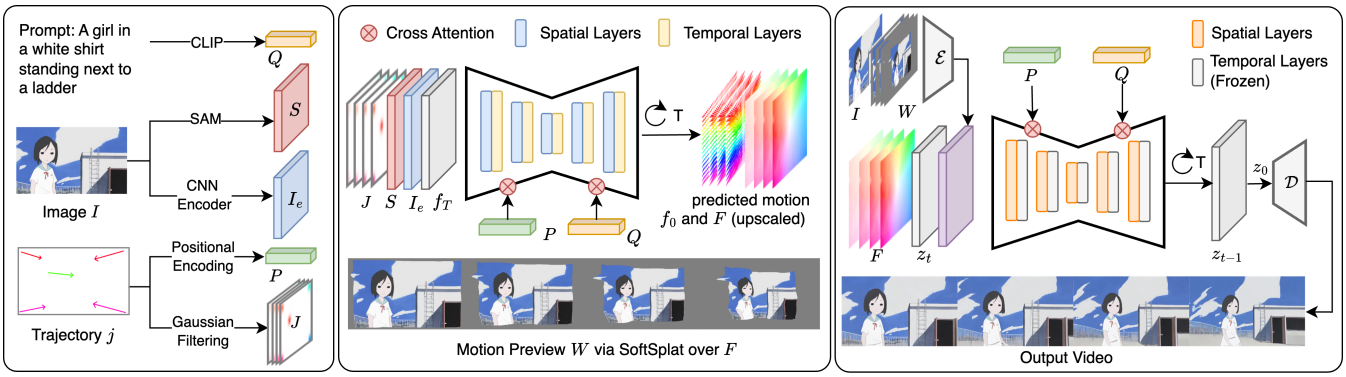


Figure 4: *Methodology overview. The left block shows the computed conditions, the middle block represents the first stage of motion prediction, and the right block depicts the second stage of anime frame synthesis.*

along the channel dimension as inputs to the 3D U-Net. Regarding cross-attention conditions, we first compute the CLIP embedding [RKH*21] Q of the input text prompt. In addition to that, we also propose an improved encoding P to convert the user-specified trajectory j to better distinguish individual trajectories and capture precise positional information:

Positional Trajectory Representation ($P \in \mathbb{R}^{1024}$): For each trajectory $j_i \in \mathbb{R}^{L\times 2}$, we apply a transformation $\mathbf{T}$ to the displacements starting from the second frame, where $\mathbf{T}$ is defined as:

$$\mathbf{T}(x) = \text{sign}(x) \cdot \sqrt{|x|/n}, \tag{4}$$

applied to the x- and y-direction displacements. Here, $n = 16$ is a normalization factor that compresses the value distribution of the displacements, regularizing the model to predict more stable temporal displacements. Without this transformation, we observe significant instability in the prediction of motion, which we hypothesize is caused by the large numerical scale of displacement values, making optimization more difficult. To encode the transformed displacements, we compute their Fourier positional embeddings [MST*21, LLW*23] and use a learnable linear MLP to project the sum of the positional embeddings and the raw features into a 1024-dimensional representation P, to emphasize position awareness in user trajectories for subsequent motion learning.

For conditions that will be concatenated with the input f_t along the channel dimension, we compute the following:

First Frame Features ($I_e \in \mathbb{R}^{L\times H_f\times W_f\times 256}$): We extract features from the first frame I using a learnable encoder with a structure similar to the basic CNN encoder described in [KRG*25]. The outputs of the first four convolutional layers are reshaped to a uniform size of $H_f \times W_f$ and concatenated along the channel dimension. This combined feature is then replicated on the temporal dimension for L times.

SAM Features ($S \in \mathbb{R}^{L\times H_f\times W_f\times 256}$): We extract the SAM [KMR*23] features from the input image I. The features are then rescaled to $H_f \times W_f$, and replicated similarly across the temporal dimension.

DragNUWA Trajectory Map ($J \in \mathbb{R}^{L\times H_f\times W_f\times 2}$): We incorporate a DragNUWA compatible trajectory condition into the model using the method specified in [YWL*23]. Specifically, the user trajectory is drawn as discrete points on a map and smoothed with a Gaussian kernel of size $K = 11$ to form the dense trajectory map.

For training objectives, we find that directly predicting ground truth motion f_0 yields more stable training and motion displacement predictions compared to the commonly used $\epsilon-$ noise prediction. We denote the diffusion U-Net as $f_\theta(P, Q, I_e, S, J, f_t)$ and define the training objective as:

$$\mathcal{L}_f = \|f_0 - f_\theta(P, Q, I_e, S, J, f_t)\|_2^2. \tag{5}$$

During inference, we compute the inverse of the displacement transformation $\mathbf{T}$ (Eq. 4) on the predicted motion $\hat{f}$ to recover the absolute displacements, which are then upscaled into a dense correspondence map F. Using F, we warp the initial frame I into a motion preview sequence $W \in \mathbb{R}^{L\times H\times W\times 3}$ by Softmax splatting [NL20]. Although this splatted preview can exhibit minor holes or local distortions due to the sparse motion representation, it provides sufficient guidance for the second stage to inpaint and regularize these imperfections, yielding smooth and visually clear videos.

During our experiments in the construction of W, we sometimes observe temporal inconsistency in the predicted motion, leading to distorted results. This issue arises because the ground truth CoTracker tracking is not inherently smooth due to the complex motion patterns of animations. Additionally, occlusions can introduce outlier tracking points that mislead the model during training. Moreover, our dataset of approximately 500k video clips may still be insufficient to fully generalize the motion prediction model. To address this, we propose a regularization technique to improve the temporal stability in motion predictions. Specifically, we generate multiple predictions under the same input conditions but initialize them with different noise values $f_T \sim \mathcal{N}(0, I)$, resulting in K different motion predictions. The final displacements are computed as their averages. In our experiments, we set $K = 4$, but users can adjust this smoothing parameter. Increasing K results in smoother motion with fewer distortions, though it can slightly reduce the ad-

herence of the model to user-provided trajectories; quantitative effects of varying K are provided in the supplementary.

4.2. Frame Synthesis Stage

We design the second stage as a frame synthesis module. In this stage, we use the motion preview sequence W as a condition to guide a video diffusion model to refine artifacts and distortions within W. This process helps produce smooth-motion videos with enhanced realism and improved completeness in both appearance and motion. Since the scale of our dataset is insufficient to train a tailored video diffusion model from scratch, we build upon pre-trained video diffusion models as a strong foundation. We selected ToonCrafter [XLX*24] as our backbone model due to its extensive prior knowledge of anime and cartoon-specific characteristics, acquired through fine-tuning on curated anime datasets. Furthermore, ToonCrafter inherits its architecture from VideoCrafter [CZC*24], which is well suited to incorporate additional user conditions. Although we considered more advanced models, such as DiT [YTZ*24, KTZ*25], these alternatives posed challenges in fine-tuning and adapting them to the conditional synthesis of anime videos with a limited amount of data. The discrepancy between cartoons/anime and natural videos makes the motion preview conditioned fine-tuning of these larger models very challenging, leading to model collapse, or undesired results.

For fine-tuning of the ToonCrafter model, we freeze the temporal layers of its 3D U-Net and focus on regularizing appearance through the spatial layers of the warped motion preview W. This strategy is inspired by the original ToonCrafter implementation, which suggests that training temporal layers with limited data may result in motion corruption. To construct the model, we utilize the pre-trained ToonCrafter 3D U-Net as the backbone, with additional conditions integrated during fine-tuning. Specifically, given a 3-channel video $W \in \mathbb{R}^{L\times H\times W\times 3}$ with a video length of $L = 16$, we encode it using the ToonCrafter VAE $\mathcal{E}$ to obtain the latent representation $\mathbf{z} = \mathcal{E}(\mathbf{x}), \mathbf{z} \in \mathbb{R}^{L\times h\times w\times 4}$. The noise latent $\mathbf{z}_t$ at time t is then computed using the forward diffusion process (Eq. 3).

To incorporate user-specified conditions, we concatenate the predicted upscaled flow map $F \in \mathbb{R}^{L\times H\times W\times 2}$ and the motion preview $W \in \mathbb{R}^{L\times H\times W\times 3}$ as additional inputs to the 3D diffusion U-Net. Both F and W are encoded by the VAE encoder $\mathcal{E}$ and concatenated with the noise latent $\mathbf{z}_t$ along the channel dimension. The flow map F provides implicit motion guidance by indicating motion directions, while the motion preview W offers rough appearance hints via warped frames. This reduces the learning difficulty for the spatial layers by focusing only on refining errors in W, such as occlusions, rough estimations, or inaccuracies in the sparse flow f predicted during the first stage. Finally, we inject the positional trajectory representations P (defined in Sect. 4.1) and the CLIP-encoded text embeddings Q into the model via cross-attention. These conditions enable the model to align the generated motion with user-provided trajectories while preserving animation coherence and visual quality. With the settings above, we will be able to write down the second-stage loss as:

$$\mathcal{L}_z = \|\epsilon - \epsilon_\theta(P, Q, I, W, F)\|_2^2. \tag{6}$$

Last, since the ToonCrafter VAE requires bidirectional inputs to decode the predicted latent $\mathbf{z}_0$ into the final video output, but we only have the first frame as ground truth, we use a standard DynamiCrafter [XXZ*25] VAE decoder to decode the last frame latent $\mathbf{z}_0^{L-1}$. This decoded last frame, combined with the first frame ground truth, serves as the bidirectional reference for decoding the full video sequence.

5. Experiments

5.1. Experimental Setup

Dataset To form our dataset, we collected 99,884 diverse animation videos from the Sakugabooru [sT25] video collection. Using CoTracker [KRG*25], we performed keypoint-based motion tracking and cutscene detection on a predefined grid of array points. The cutscenes were separated according to the continuity of the visibility of the tracker points, rather than relying on the commonly used PySceneDetect [Cas24], for a higher precision. For more details on the data preparation process, please refer to the supplementary material. Ultimately, we collected 533,344 single-cutscene video clips, with 2,500 clips held out for validation and 3,000 clips for testing. To annotate each clip, captions for the middle frame were generated using BLIP-2 [LLSH23], following the same methodology as ToonCrafter. We have the dataset and the annotations uploaded on the project website.

Baseline Setup For our comparative evaluation, we select three baseline models: DragNUWA [YWL*23], Motion-I2V [SHW*24], and MotionCtrl [WYW*24]. To disentangle the contributions of our proposed dataset from those of the model architecture, we establish two experimental settings for each baseline. The first setting utilizes the official pretrained models, whereas the second involves fine-tuning them on our dataset. For the fine-tuning of DragNUWA and MotionCtrl, we sample user trajectory inputs by generating motion conditions using our proposed trajectory sampling technique. In the specific case of MotionCtrl, this involves injecting its OMCM module into our ToonCrafter model to serve as an implicit motion condition and jointly training the module with the main diffusion model.

5.2. Comparison with Exsiting Methods

Quantitative Evaluation We first perform a quantitative evaluation using FVD [UvSK*19] and LPIPS [ZIE*18] to assess proximity to the anime video frames and perceptual quality. Additionally, since frame synthesis with guidance from ground truth trajectories can be seen as a reconstruction task, we evaluate the average PSNR and SSIM [HZ10] to measure alignment with trajectory conditions. To assess the appearance quality of individual frames, we also calculate the FID metric [HRU*17] by comparing generated frames against ground truth. We used the test set of 3000 new video clips and show the results in Tab. 1.

The results highlight two main insights. First, the efficacy of our dataset is demonstrated by the performance improvements when DragNUWA and MotionCtrl are finetuned on it, validating its value for capturing cartoon and anime motion. Second, under the same data conditions, our method surpasses the finetuned counterparts

Table 1: *Quantitative comparison with baseline models. ↓ = lower is better, and ↑ = higher is better. Bold indicates best value.*

Method	LPIPS ↓	FVD ↓	PSNR ↑	SSIM ↑	FID ↓
DragNUWA					
Vanilla	0.54	1675.8	12.4	0.53	10.3
Finetuned	0.49	1219.4	13.5	0.58	9.5
MotionCtrl					
Vanilla	0.69	1891.4	10.7	0.60	42.7
Finetuned	0.48	1264.2	13.8	0.61	13.4
ToonCrafter	0.48	1232.3	14.3	0.60	11.3
Motion-I2V					
Vanilla	0.64	1766.9	11.0	0.41	30.0
Finetuned	0.65	1608.8	11.1	0.42	27.4
Ours	**0.44**	**1119.6**	**15.1**	**0.63**	**7.1**

across all metrics. To validate that these gains are due to our framework's architecture and not just the choice of a pretrained generator (ToonCrafter), we created a variant of MotionCtrl by substituting its video synthesis module with ToonCrafter's and finetuning it on our dataset. This hybrid configuration improved on the original MotionCtrl but did not match our method's performance. Overall, the statistics show the superiority of our two-stage framework, which decouples sparse motion prediction from frame synthesis, proving more effective than end-to-end approaches.

In contrast to other baselines, Motion-I2V failed to adapt to the target domain. The pretrained model lacked generalization, and finetuning did not rectify this. We believe the AnimateDiff component of Motion-I2V is less suited for domain adaptation in this context, limiting its ability to learn the specialized motion patterns for anime synthesis.

Visual Comparison on Trajectory-based Anime Synthesis We present a visual comparison of our method against key competitors, with further video results on our supplementary webpage. We encourage readers to view the videos, as they best illustrate temporal dynamics and artifacts. All comparisons use test images unseen during training. We overlay the uniformly replayed progression on the results to demonstrate trajectory adherence, judged by whether the driven regions' motion follows the spatial course of each curve over time while preserving appearance. Note that our model focuses on producing harmonious, deformation-aware, and clean anime motion; exact displacement amplitudes may be moderated when strict following would introduce distortions or incoherent poses.

Without fine-tuning on our anime dataset, the baseline competitor models struggle significantly (Fig. 7 top). DragNUWA fails to generate coherent motion, and MotionCtrl exhibits severe content drift, reflected in their weaker quantitative scores. For a more equitable comparison, we evaluate the fine-tuned versions of the baseline models. Even after fine-tuning, DragNUWA often produces visual artifacts like facial distortions (Fig. 6), which we hypothesize is due to a lack of explicit and robust temporal modeling. On the other hand, MotionCtrl better preserves structures but tends to produce dull motions that simply pan objects without deformations or pose changes (Fig. 7 bottom, Fig. 8 top), and sometimes still leads to distortions. We believe these methods are limited by their reliance on implicit trajectory guidance, which may not be enough to translate motion hints into plausible object motion and deformation. Additionally, these models lack semantic awareness; in comparison, our model leverages SAM features to better isolate subjects during prediction. For Motion-I2V, it may maintain structure and produce anime-like motion if the input image is close to natural photos; however, appearance and color cannot be maintained over time, and the overall scene composition may gradually change, a known limitation. Furthermore, its performance depends heavily on the generalization of its underlying AnimateDiff model, which we find challenging to adapt to diverse anime styles.

Motion Manga and Vector Graphics We also evaluate our model's generalization ability on non-photorealistic media, such as vector-like graphics, manga and storyboard sketches. When applied to vector-like graphics, our method produces more natural motion, such as simulating the propulsive pushes of a jellyfish moving forward (Fig. 7 bottom). In contrast, the other methods either simply pan the jellyfish or fail to generalize on the input vector graphics. For manga inputs, the special effects of the so-called *speed line* effects are widely used with parallel straight or curved lines to convey motion, energy, or intensity in a scene (as depicted near the arm of the boy in Fig. 8). These lines illustrate the direction and speed of movement to emphasize dramatic actions or emotions. By providing trajectories aligned with the speed line directions, our model is able to produce depiction of dynamic scenes while faithfully captures the stylistic elements of manga, such as artist-drawn scribbles and screentones. In comparison, DragNUWA provides naive motions that merely pan the underlying region in the specified directions and does not have any motion or expression changes on the character's face. MotionCtrl and Motion-I2V, on the other hand, fail to generalize to manga-style appearances.

Finally, we demonstrate our model's utility in a challenging preproduction scenario using a hand-drawn storyboard sketch (Fig. 8 bottom). Traditionally, artists must mentally conceptualize motion, camera paths, and composition to manually draw a sequence of rough sketches, which is a labor-intensive process. Our approach streamlines this workflow significantly. By simply providing a motion trajectory, an artist can rapidly generate dynamic previews for different creative ideas. The resulting animation can serve as a direct visual reference for subsequent frames, reducing the cognitive load and time spent iterating on the storyboard. In stark contrast, the baseline models fail to produce any meaningful frames from this challenging and sparse input, underscoring the unique generalization capability of our method.

5.3. Ablation Studies

We conducted ablation studies to evaluate the importance of our framework design. First, suppose that we remove the first stage to predict explicit motion but rely solely on user-specified trajectories to guide anime synthesis, the framework becomes equivalent to the MotionCtrl design, with performance already reflected in Tab. 1. This demonstrates the importance of explicit motion prediction for anime synthesis. In addition, we evaluated alternative

Table 2: *Ablation study results for our framework.*

Setup	LPIPS ↓	FVD ↓	PSNR ↑	SSIM ↑	FID ↓
Predict f as [LTSH24]	0.47	1184.2	14.5	0.61	8.9
Predict f as Optical flow	0.48	1364.1	14.4	0.60	8.4
Stage 1 w/o P	0.45	1154.3	15.0	**0.63**	**7.0**
Stage 1 w/o S	0.45	1158.7	14.5	0.62	8.7
Ours Full	**0.44**	**1119.6**	**15.1**	**0.63**	7.1

configurations for the first stage of motion prediction. We experimented with two prominent motion representations: (a) 4D motion volumes, following the methodology of [LTSH24], and (b) dense optical flow. For these experiments, the motion preview W is produced by warping the input frame I based on the predicted motion representation. To facilitate optical flow comparison, we establish a ground truth by fine-tuning the GMFlow model [XZC*22] on the AnimeRun dataset [LLL*22]. We demonstrate the results in Tab. 2, and the supplementary website provides additional qualitative examples. In total, both alternative configurations failed to match the performance of our sparse CoTracker representation. This highlights a fundamental limitation of dense methods in this domain: the sparse textures inherent to cartoon and animation styles cannot reliably support the calculation of a dense correspondence field without introducing significant errors.

We also evaluate the designs of the building blocks of our methods. This includes an ablation study on: (a) the positional trajectory embedding P; and (b) the SAM feature S. We find that removing the positional trajectory embedding causes overly exaggerated motion, though the frame appearances usually remain intact. In addition, removing SAM feature introduces distortions and visual artifacts, compromising the structural integrity of the animated regions.

5.4. User Study

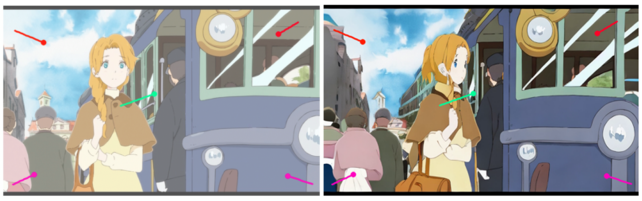
To assess the perceptual quality and trajectory adherence of our generated animations, we conducted a user study comparing our method against three leading competitors: MotionCtrl, DragNUWA, and Motion-I2V. For the study, we recruited 16 participants and presented them with 23 sets of results. Participants evaluated complete animations, not static frames. To ensure a fair comparison, each set contained time-synced videos generated from the same input trajectory by all four methods. The videos were presented in a randomized order to mitigate presentation bias. Participants were asked to rank the animations from best (1) to worst (4) based on overall visual quality and how well the motion followed the intended path. The results, summarized in Tab. 3, show that our method achieved the highest average rank, indicating a clear user preference in terms of visual fidelity and motion control. Refer to the supplemetary material for additional comparisons.

5.5. Limitations

While our method effectively produces smooth motion with pose changes and deformations, we observe certain limitations in how

Table 3: *User study results comparing our method against baseline models. Lower average rank indicates better performance.*

	Ours	MotionCtrl	DragNUWA	Motion-I2V
Rank ↓	**1.33**	2.36	2.71	3.60



(a) Input and trajectory (b) Last synthesized frame

Figure 5: *Limitation of our proposed framework. The intended motion that directs the girl moving by the bus is misinterpreted as pose changes. Prompt: "A girl is walking by a bus".*

the refinement stage handles user-specified trajectories. Specifically, the refinement stage may overly recover and hallucinate details, which can cause the movement of specific regions to deviate from the exact distance and direction specified by the trajectory (the second row of Fig. 1). This reflects a trade-off between achieving high visual quality and maintaining strict adherence to user-provided trajectories, as discussed in Sect. 4.1.

Additionally, our method may not always clearly interpret the intent behind a user-specified trajectory. For example, when a trajectory is placed on a subject, the system can misinterpret a simple translation or panning motion as a hint for deformation or pose changes, as shown in Fig. 5. To address this, users may need to provide multiple parallel trajectories on the same object to indicate that the motion is primarily positional. We consider it as a future work, to allow the system to distinguish between different types of motion guidance.

6. Conclusion

We propose a two-stage framework for trajectory-guided anime video synthesis, combining explicit motion prediction with a refinement stage using video diffusion models. The key innovation of our method lies in an efficient sparse motion representation and estimation technique, to produce smooth, high-quality anime video clips while preserving stylistic elements. Experiments show that our approach outperforms existing methods in both motion fidelity and visual quality on a wide variety of 2D media.

Acknowledgements

The work described in this paper was fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. UGC/FDS11/E02/23).

References

[BDK*23] BLATTMANN A., DOCKHORN T., KULAL S., MENDELEVITCH D., KILIAN M., LORENZ D., LEVI Y., ENGLISH Z., VOLETI

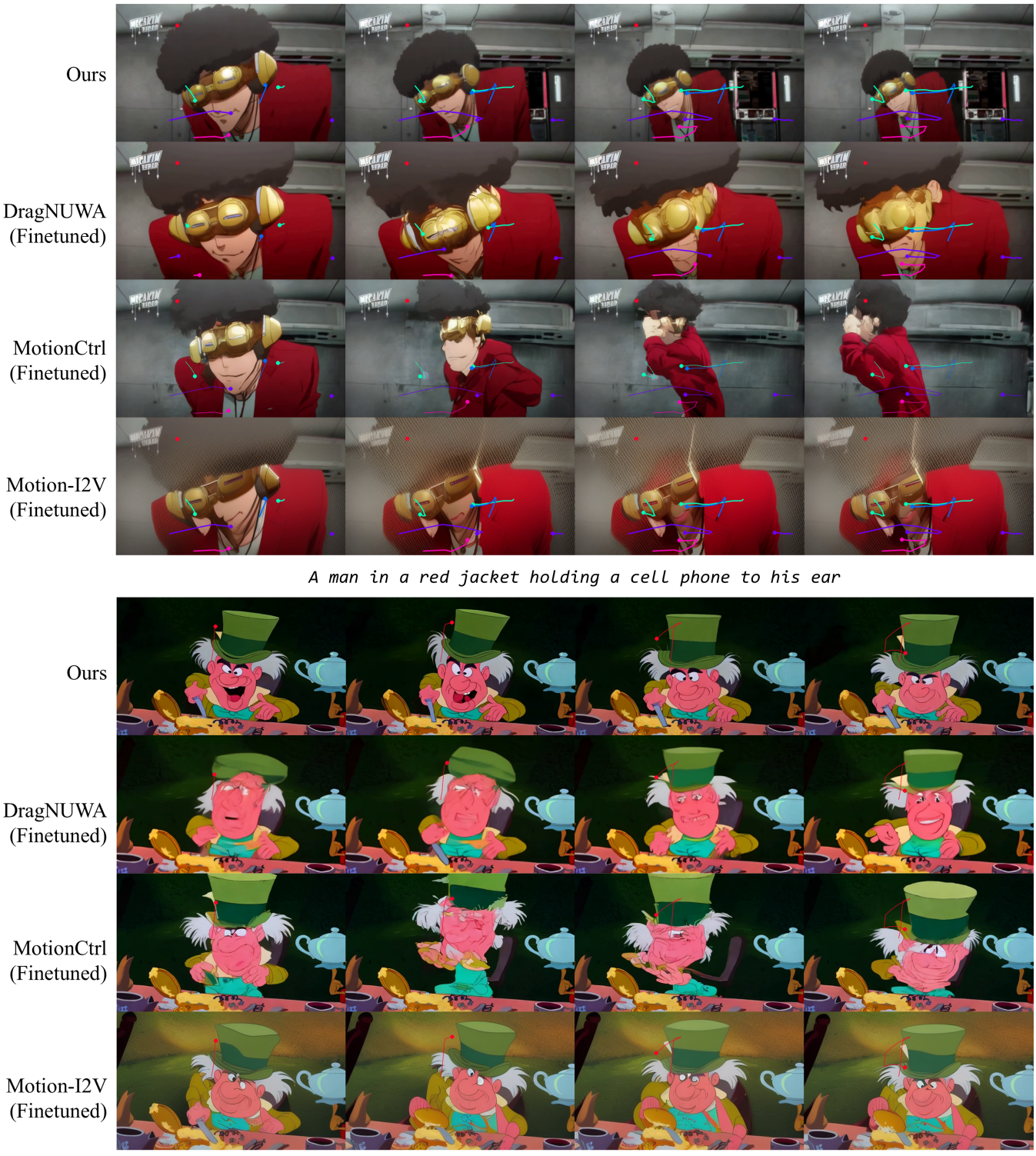


Figure 6: *Visual comparison of trajectory-guided motion synthesis. Results: (top) using sampled trajectory from ground truth; (bottom) using manually drawn trajectory from human testers. Columns are results over the 16-frame clip from 25% to 100% progression. Colored curves are the uniform-speed playback of input trajectories according to frame; colors only distinguish different paths.*

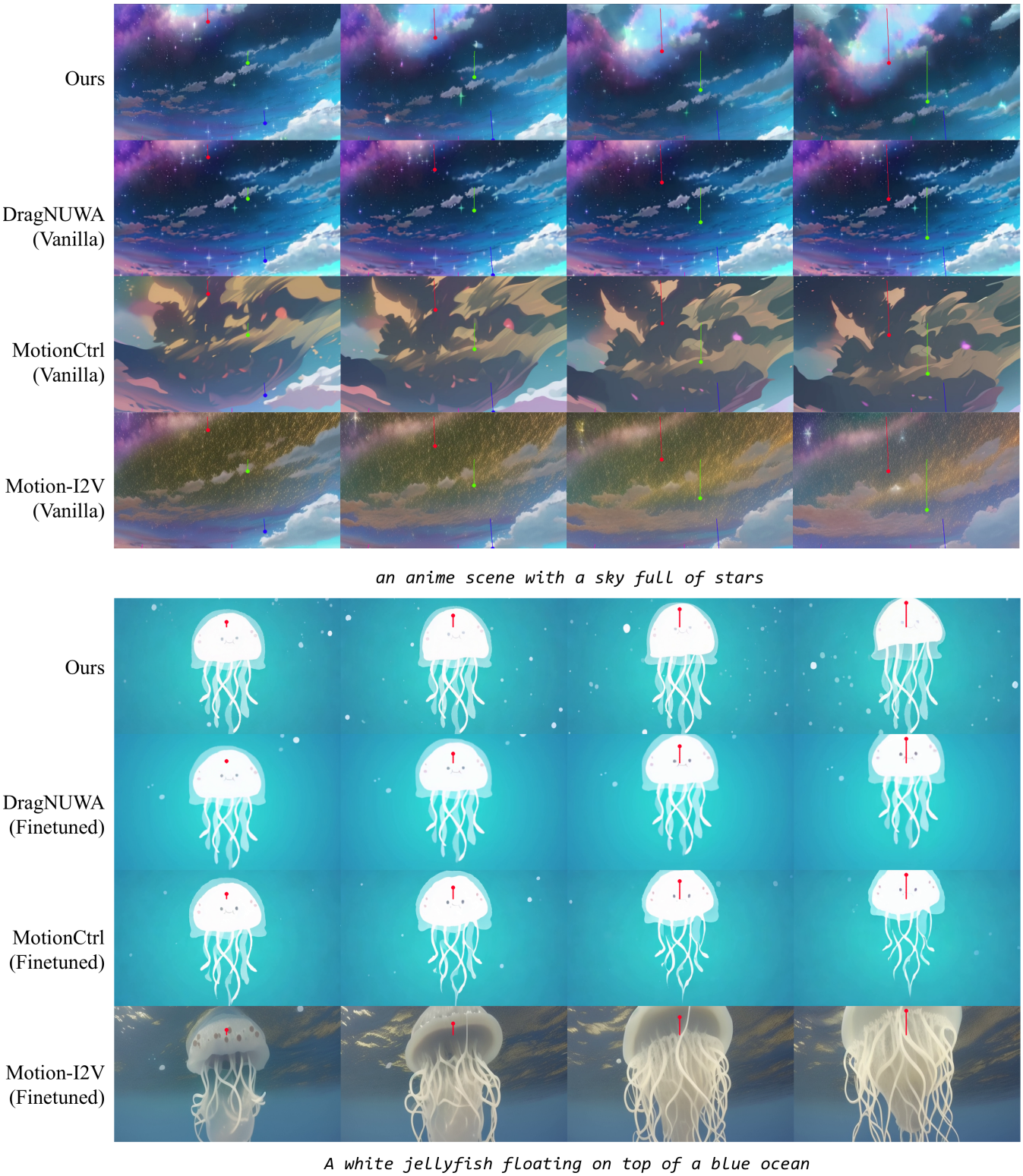


Figure 7: *Visual comparison with vanilla models (top) and results on vector graphics (bottom). All trajectories are provided from human testers.*

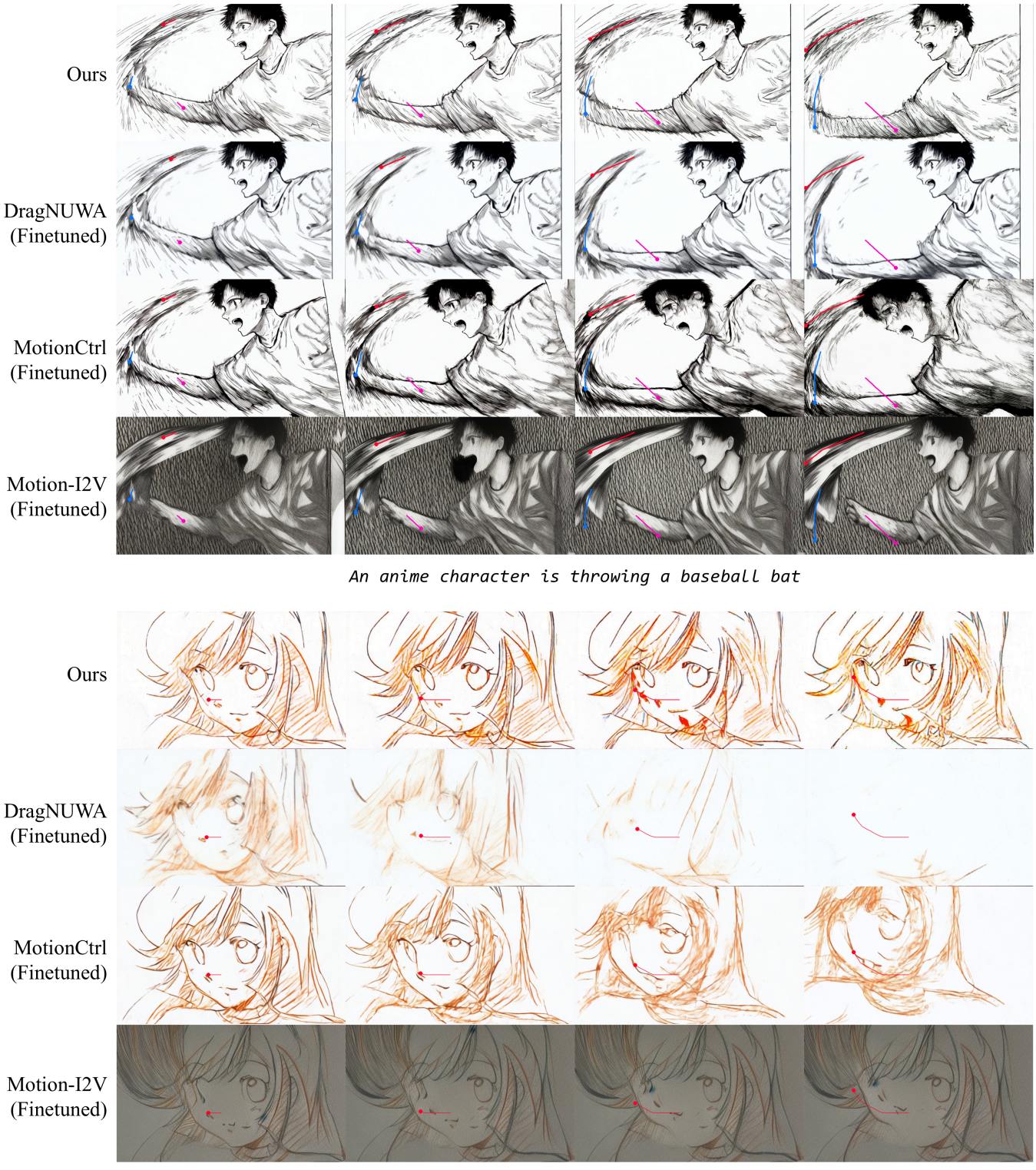


Figure 8: *Visual comparison on trajectory-based motion synthesis on manga frames and storyboard sketches. All trajectories are provided from human testers.*

V., LETTS A., JAMPANI V., ROMBACH R.: Stable video diffusion: Scaling latent video diffusion models to large datasets, 2023. URL: https://arxiv.org/abs/2311.15127, arXiv:2311.15127. 3

[Cas24] CASTELLANO B.: Pyscenedetect, mar 2024. Software available at https://github.com/Breakthrough/PySceneDetect/. URL: https://github.com/Breakthrough/PySceneDetect/. 6

[CPL21] CASEY E., PÉREZ V., LI Z.: The animation transformer: Visual correspondence via segment matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (October 2021), pp. 11323–11332. 3

[CXH*23] CHEN H., XIA M., HE Y., ZHANG Y., CUN X., YANG S., XING J., LIU Y., CHEN Q., WANG X., WENG C., SHAN Y.: Videocrafter1: Open diffusion models for high-quality video generation, 2023. arXiv:2310.19512. 3

[CZC*24] CHEN H., ZHANG Y., CUN X., XIA M., WANG X., WENG C., SHAN Y.: Videocrafter2: Overcoming data limitations for high-quality video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2024), pp. 7310–7320. 3, 4, 6

[DLKS18] DVOROŽNÁK M., LI W., KIM V. G., SÝKORA D.: Toonsynth: example-based synthesis of hand-colored cartoon animations. *ACM Trans. Graph. 37*, 4 (July 2018). URL: https://doi.org/10.1145/3197517.3201326, doi:10.1145/3197517.3201326. 3

[GVA*24] GAL R., VINKER Y., ALALUF Y., BERMANO A., COHEN-OR D., SHAMIR A., CHECHIK G.: Breathing life into sketches using text-to-video priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2024), pp. 4325–4336. 3

[GYR*24] GUO Y., YANG C., RAO A., LIANG Z., WANG Y., QIAO Y., AGRAWALA M., LIN D., DAI B.: Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. In *International Conference on Learning Representations (ICLR)* (2024). 3

[HJA20] HO J., JAIN A., ABBEEL P.: Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual* (2020), Larochelle H., Ranzato M., Hadsell R., Balcan M., Lin H., (Eds.). URL: https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html. 4

[HRU*17] 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. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, 2017), NIPS'17, Curran Associates Inc., p. 6629–6640. 6

[HSZ*22] HUANG Z., SHI X., ZHANG C., WANG Q., CHEUNG K. C., QIN H., DAI J., LI H.: FlowFormer: A transformer architecture for optical flow. *ECCV* (2022). 3

[HW79] HARTIGAN J. A., WONG M. A.: A k-means clustering algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics) 28*, 1 (1979), 100–108. 4

[HZ10] HORÉ A., ZIOU D.: Image quality metrics: Psnr vs. ssim. In *2010 20th International Conference on Pattern Recognition* (2010), pp. 2366–2369. doi:10.1109/ICPR.2010.579. 6

[HZL24] HUANG Z., ZHANG M., LIAO J.: Lvcd: Reference-based lineart video colorization with diffusion models. *ACM Trans. Graph. 43*, 6 (Nov. 2024). URL: https://doi.org/10.1145/3687910, doi:10.1145/3687910. 3

[IMH05] IGARASHI T., MOSCOVICH T., HUGHES J. F.: As-rigid-as-possible shape manipulation. *ACM transactions on Graphics (TOG) 24*, 3 (2005), 1134–1141. 3

[KMR*23] KIRILLOV A., MINTUN E., RAVI N., MAO H., ROLLAND C., GUSTAFSON L., XIAO T., WHITEHEAD S., BERG A. C., LO W.-Y., ET AL.: Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2023), pp. 4015–4026. 5

[KRG*25] KARAEV N., ROCCO I., GRAHAM B., NEVEROVA N., VEDALDI A., RUPPRECHT C.: Cotracker: It is better to track together. In *European Conference on Computer Vision (ECCV)* (2025), Springer, pp. 18–35. 2, 3, 5, 6

[KTZ*25] KONG W., TIAN Q., ZHANG Z., MIN R., DAI Z., ZHOU J., XIONG J., LI X., WU B., ZHANG J., WU K., LIN Q., YUAN J., LONG Y., WANG A., WANG A., LI C., HUANG D., YANG F., TAN H., WANG H., SONG J., BAI J., WU J., XUE J., WANG J., WANG K., LIU M., LI P., LI S., WANG W., YU W., DENG X., LI Y., CHEN Y., CUI Y., PENG Y., YU Z., HE Z., XU Z., ZHOU Z., XU Z., TAO Y., LU Q., LIU S., ZHOU D., WANG H., YANG Y., WANG D., LIU Y., JIANG J., ZHONG C.: Hunyuanvideo: A systematic framework for large video generative models, 2025. URL: https://arxiv.org/abs/2412.03603, arXiv:2412.03603. 6

[LLL*22] LI S., LI Y., LI B., DONG C., LIU Z., LOY C. C.: Animerun: 2d animation visual correspondence from open source 3d movies. *Advances in Neural Information Processing Systems 35* (2022), 18996–19007. 3, 8

[LLSH23] LI J., LI D., SAVARESE S., HOI S. C. H.: BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *Proceedings of the 40th International Conference on Machine Learning (ICML)* (23–29 Jul 2023), Krause A., Brunskill E., Cho K., Engelhardt B., Sabato S., Scarlett J., (Eds.), vol. 202 of *Proceedings of Machine Learning Research*, PMLR, pp. 19730–19742. 6

[LLW*23] LI Y., LIU H., WU Q., MU F., YANG J., GAO J., LI C., LEE Y. J.: GLIGEN: Open-set grounded text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2023), pp. 22511–22521. 5

[LMPS24] LE MOING G., PONCE J., SCHMID C.: Dense optical tracking: Connecting the dots. In *CVPR* (2024). 4

[LTSH24] LI Z., TUCKER R., SNAVELY N., HOLYNSKI A.: Generative image dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2024). 8

[MGW24] MO H., GAO C., WANG R.: Joint stroke tracing and correspondence for 2d animation. *ACM Transactions on Graphics 43*, 3 (2024), 1–17. 3

[MOW*25] MENG Y., OUYANG H., WANG H., WANG Q., WANG W., CHENG K. L., LIU Z., SHEN Y., QU H.: Anidoc: Animation creation made easier. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)* (June 2025), pp. 18187–18197. 3

[MST*21] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: NeRF: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM 65*, 1 (2021), 99–106. 5

[NL20] NIKLAUS S., LIU F.: Softmax splatting for video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), pp. 5437–5446. 5

[PTL*23] PAN X., TEWARI A., LEIMKÜHLER T., LIU L., MEKA A., THEOBALT C.: Drag your gan: Interactive point-based manipulation on the generative image manifold. In *ACM SIGGRAPH 2023 conference proceedings* (2023), pp. 1–11. 3

[RBL*22] ROMBACH R., BLATTMANN A., LORENZ D., ESSER P., OMMER B.: High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (Los Alamitos, CA, USA, jun 2022), IEEE Computer Society, pp. 10674–10685. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR52688.2022.01042, doi:10.1109/CVPR52688.2022.01042. 3, 4

[RKH*21] RADFORD A., KIM J. W., HALLACY C., RAMESH A., GOH G., AGARWAL S., SASTRY G., ASKELL A., MISHKIN P., CLARK

J., KRUEGER G., SUTSKEVER I.: Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning* (18–24 Jul 2021), Meila M., Zhang T., (Eds.), vol. 139 of *Proceedings of Machine Learning Research*, PMLR, pp. 8748–8763. URL: https://proceedings.mlr.press/v139/radford21a.html. 5

[SBCv*11] SÝKORA D., BEN-CHEN M., ČADÍK M., WHITED B., SIMMONS M.: Textoons: practical texture mapping for hand-drawn cartoon animations. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2011), NPAR '11, Association for Computing Machinery, p. 75–84. URL: https://doi.org/10.1145/2024676.2024689, doi:10.1145/2024676.2024689. 3

[SGX*23] SIYAO L., GU T., XIAO W., DING H., LIU Z., LOY C. C.: Deep geometrized cartoon line inbetweening. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)* (2023), pp. 7257–7266. doi:10.1109/ICCV51070.2023.00670. 3

[SHL*23] SHI X., HUANG Z., LI D., ZHANG M., CHEUNG K. C., SEE S., QIN H., DAI J., LI H.: Flowformer++: Masked cost volume autoencoding for pretraining optical flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 1599–1610. 3

[SHW*24] SHI X., HUANG Z., WANG F.-Y., BIAN W., LI D., ZHANG Y., ZHANG M., CHEUNG K. C., SEE S., QIN H., DAI J., LI H.: Motion-i2v: Consistent and controllable image-to-video generation with explicit motion modeling. In *ACM SIGGRAPH 2024 Conference Papers* (New York, NY, USA, 2024), SIGGRAPH '24, Association for Computing Machinery. URL: https://doi.org/10.1145/3641519.3657497, doi:10.1145/3641519.3657497. 2, 3, 4, 6

[sT25] SAKUGABOORU TEAM: sakugabooru. https://sakugabooru.com/, 2025. A booru dedicated to sakuga videos and images. Serving 157,282 posts as of retrieval. Powered by Moebooru 6.0.0. URL: https://sakugabooru.com/. 6

[SYLK18] SUN D., YANG X., LIU M.-Y., KAUTZ J.: PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. 3

[SZC*23] SHI M., ZHANG J.-Q., CHEN S.-Y., GAO L., LAI Y.-K., ZHANG F.-L.: Reference-based deep line art video colorization. *IEEE Transactions on Visualization and Computer Graphics 29*, 6 (2023), 2965–2979. doi:10.1109/TVCG.2022.3146000. 3

[SZY*21] SIYAO L., ZHAO S., YU W., SUN W., METAXAS D., LOY C. C., LIU Z.: Deep animation video interpolation in the wild. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2021), pp. 6587–6595. 3

[TD20] TEED Z., DENG J.: RAFT: Recurrent all-pairs field transforms for optical flow. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II* (2020), Springer, pp. 402–419. 3

[UvSK*19] UNTERTHINER T., VAN STEENKISTE S., KURACH K., MARINIER R., MICHALSKI M., GELLY S.: Towards accurate generative models of video: A new metric & challenges, 2019. URL: https://arxiv.org/abs/1812.01717, arXiv:1812.01717. 6

[WHF*25] WANG A., HUANG H., FANG Z., YANG Y., MA C.: ATI: Any trajectory instruction for controllable video generation. *arXiv preprint arXiv:2505.22944* (2025). 3

[WLG*24] WU W., LI Z., GU Y., ZHAO R., HE Y., ZHANG D. J., SHOU M. Z., LI Y., GAO T., ZHANG D.: Draganything: Motion control for anything using entity representation. In *European Conference on Computer Vision* (2024), Springer, pp. 331–348. 3

[WSML24] WU R., SU W., MA K., LIAO J.: Aniclipart: Clipart animation with text-to-video priors. *International Journal of Computer Vision* (2024), 1–17. 3

[WWZ*24] WANG W., WANG Q., ZHENG K., OUYANG H., CHEN Z., GONG B., CHEN H., SHEN Y., SHEN C.: Framer: Interactive video interpolation. *arXiv preprint https://arxiv.org/abs/2410.18978* (2024). 3

[WYW*24] WANG Z., YUAN Z., WANG X., LI Y., CHEN T., XIA M., LUO P., SHAN Y.: Motionctrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers* (New York, NY, USA, 2024), SIGGRAPH '24, Association for Computing Machinery. doi:10.1145/3641519.3657518. 2, 3, 6

[XLX*24] XING J., LIU H., XIA M., ZHANG Y., WANG X., SHAN Y., WONG T.-T.: Tooncrafter: Generative cartoon interpolation. *ACM Transactions on Graphics (TOG) 43*, 6 (2024), 1–11. 2, 3, 6

[XXZ*25] XING J., XIA M., ZHANG Y., CHEN H., YU W., LIU H., LIU G., WANG X., SHAN Y., WONG T.-T.: Dynamicrafter: Animating open-domain images with video diffusion priors. In *European Conference on Computer Vision* (2025), Springer, pp. 399–417. 6

[XZC*22] XU H., ZHANG J., CAI J., REZATOFIGHI H., TAO D.: Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 8121–8130. 3, 8

[XZJJ25] XIE T., ZHAO Y., JIANG Y., JIANG C.: Physanimator: Physics-guided generative cartoon animation. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)* (June 2025), pp. 10793–10804. 3

[YTZ*24] YANG Z., TENG J., ZHENG W., DING M., HUANG S., XU J., YANG Y., HONG W., ZHANG X., FENG G., ET AL.: Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072* (2024). 3, 6

[YWL*23] YIN S., WU C., LIANG J., SHI J., LI H., MING G., DUAN N.: Dragnuwa: Fine-grained control in video generation by integrating text, image, and trajectory. *arXiv preprint arXiv:2308.08089* (2023). 2, 3, 5, 6

[ZIE*18] ZHANG R., ISOLA P., EFROS A. A., SHECHTMAN E., WANG O.: The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR* (2018). 6

[ZLL*25] ZHANG Z., LIAO J., LI M., DAI Z., QIU B., ZHU S., QIN L., WANG W.: Tora: Trajectory-oriented diffusion transformer for video generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference* (2025), pp. 2063–2073. 3

[ZLWH16] ZHU H., LIU X., WONG T.-T., HENG P.-A.: Globally optimal toon tracking. *ACM Transactions on Graphics (SIGGRAPH 2016 issue) 35*, 4 (July 2016), 75:1–75:10. 3