Seeing Fast and Slow: Learning the Flow of Time in Videos

Yen-Siang Wu[1,2], **Rundong Luo**[1], **Jingsen Zhu**[1], **Tao Tu**[1], **Ali Farhadi**[3], **Matthew Wallingford**[3], **Yu-Chiang Frank Wang**[2], **Steve Marschner**[1], **Wei-Chiu Ma**[1]

[1]Cornell University, [2]National Taiwan University, [3]University of Washington

Abstract

How can we tell whether a video has been sped up or slowed down? How can we generate videos at different speeds? Although videos have been central to modern computer vision research, little attention has been paid to perceiving and controlling the passage of time. In this paper, we study time as a learnable visual concept and develop models for reasoning about and manipulating the flow of time in videos.

We first exploit the multimodal cues and temporal structure naturally present in videos to learn, in a self-supervised manner, to detect speed changes and estimate playback speed. We then show that these learned temporal reasoning models enable us to curate the largest slow-motion video dataset to date from noisy in-the-wild sources. Such slow-motion footage, typically filmed by high-speed cameras, contains substantially richer temporal detail than standard videos. Using this data, we further develop models capable of temporal control, including speed-conditioned video generation, which produces motion at specified playback speed, and temporal super-resolution, which tranforms low-FPS, blurry videos into high-FPS sequences with fine-grained temporal details. Our findings highlight time as a manipulable, perceptual dimension in video learning, opening doors to temporally controllable video generation, temporal forensics detection, and potentially richer world-models that understand how events unfold over time.

Project Page: https://seeing-fast-and-slow.github.io/

1 Introduction

Humans exhibit a strong intuition for the flow of time. Even without explicit cues, we often sense when a video is sped up or slowed down: surface ripples propagating too quickly, objects falling at unnatural rates, or human motion that feels sluggish. This intuition reflects an implicit understanding of the speed at which events unfold in the physical world [46]. Despite remarkable progress in video understanding and generation, current models lack such temporal reasoning [31, 32, 44]. When asked to predict playback speed, modern vision–language models (VLMs) [12, 65] frequently hallucinate or give incorrect predictions. When prompted to generate content at a particular speed, existing generative models [14, 49] often ignore instructions or generate video with imprecise timing. These issues highlight a fundamental limitation in current models: the inability to reason about time.

This limitation is not particularly surprising. Modern models primarily learn from videos captured at standard frame rates (*e.g.*, 24–60 fps) [11, 54], and therefore never learn the notion of temporal speed. In effect, we are asking the model to infer time's variability from observing a single cadence. We argue that for models to understand the flow of time and to control temporal dynamics, they must be exposed to data spanning the

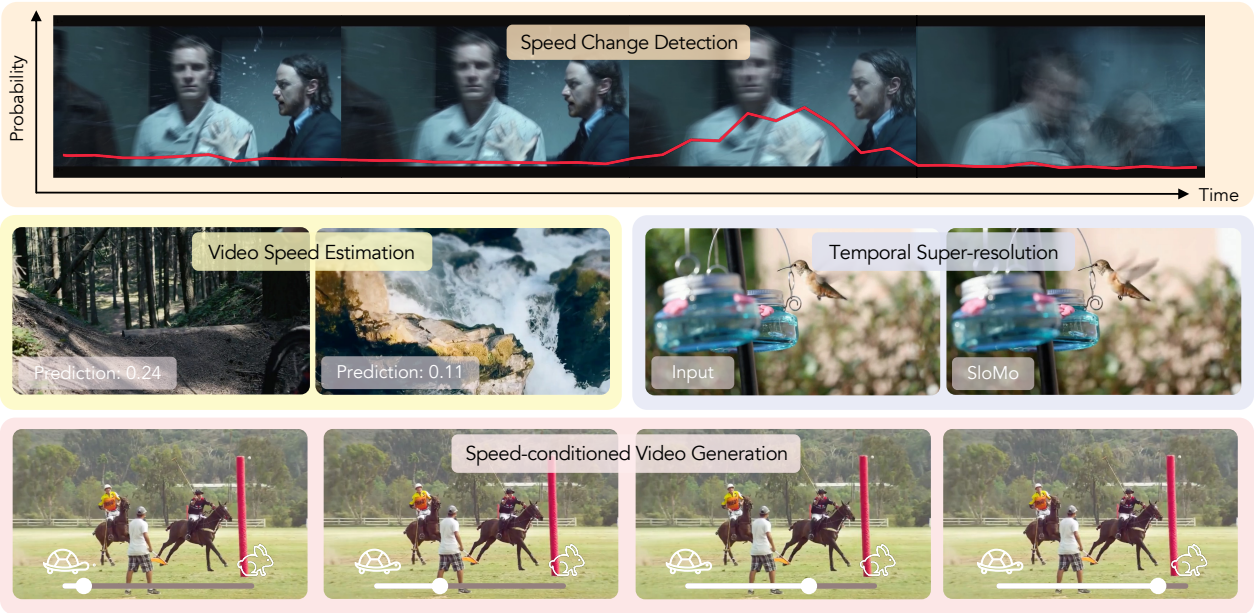

Figure 1 We develop models towards **perceiving and manipulating the flow of time**, including (a) *speed-change detection*, which locates moments where playback speed shifts; (b) *video speed estimation*, which infers how much a video has been sped up or slowed down; (c) *extreme temporal super-resolution*, converting low-FPS, blurry videos into their high-FPS, clear counterparts; and (d) *speed-conditioned video generation*, synthesizing the same event at user-specified temporal speeds. For more details on these results, please refer to section 5 and the project page. *Please open in Adobe Acrobat Reader to view the embedded animations.*

continuum of temporal speeds.

Obtaining such data at scale is not trivial. Although there are vast untapped stores of slow-motion videos online, the associated metadata required to train models, such as the frame rate, playback speed, and time stamps, are often incomplete. A natural solution would be to have humans manually annotate the videos. However, naively labeling is prohibitively time consuming and imprecise. Therefore, we look to other directions for providing temporal supervision for video understanding and generation models.

In this paper, we show that temporal cues and multi-modal signals in videos can be leveraged to infer playback speed without explicit labels. Our first key insight builds on the principle of *time–frequency scaling*: when a video's playback speed changes, its audio pitch shifts. Thus, by locating regions where the pitch shifts (Fig. 2), we can obtain free cross-modal supervision to train models that visually detect speed changes.

Our second insight exploits the *equivariance* of speed estimation under temporal rescaling: if we downsample a video by a certain factor, its perceived speed should decrease proportionally by the same factor. By resampling each clip and enforcing this proportional relationship in the model's predictions, we use temporal resampling as a powerful self-supervised training signal. Together, these complementary cues enable models to infer the lapse of time directly from visual dynamics, even in the absence of clean or reliable temporal labels.

We apply our temporal speed models to in-the-wild videos collected from online platforms. Using these models, we automatically annotate and curate a large-scale slow-motion dataset spanning diverse activities, motion patterns, and temporal scales. These slow-motion videos, typically captured at high frame rates (*e.g.*, 1000+ FPS) and played back at standard frame rates (*e.g.*, 30 FPS), preserve fine temporal details and exhibit reduced motion blur, allowing machines to perceive and learn how real-world dynamics actually unfold.

Next, using our newly constructed dataset, we develop models capable of manipulating the flow of time in videos. We focus on two tasks: (i) speed-conditioned video generation, which equips models with the ability to render physical motion at different temporal speeds; and (ii) temporal super-resolution, which converts videos into higher frame-rate sequences. Both tasks are challenging: standard videos often lack fine-grained motion

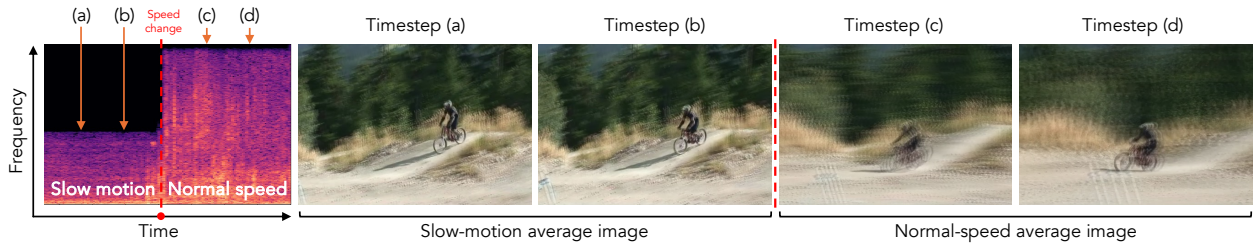

Figure 2 Audio signal naturally encodes cues for speed change detection. When a video's playback speed changes, its audio pitch shifts accordingly – speeding up raises the pitch, while slowing down lowers it. This provides **free cross-modal supervision** for detecting speed changes by locating moments of pitch variation. Left: Audio spectrogram (**used only during training**). Right: Average image at each selected timestep, obtained by averaging nearby frames. Higher playback speed leads to larger content changes and therefore greater blur.

cues due to motion blur, and synthesizing motion at precise speeds requires accurate temporal modeling. We show that finetuning existing models on our slow-motion dataset enables them to generate lifelike dynamics at controllable playback speeds and to interpolate frames at higher effective frame rates.

Since the tasks explored in this work are relatively new, there are no established benchmarks or standard evaluation protocols. To ensure a fair and meaningful assessment, we carefully curate dedicated evaluation datasets and conduct comprehensive human perceptual studies. Our experiments show that the proposed models perform strongly across both understanding and generation tasks, achieving 92% accuracy on speed-change detection, near-human accuracy on playback-speed estimation, 80.3% human-preference win rate over the base model for temporal super-resolution.

2 Related Works

Time as a learnable visual concept. Humans exhibit a remarkably malleable perception of time when watching videos. Studies in cognitive science [46] show that people tend to overestimate the duration of an action when it is presented in slow motion. From a computer vision perspective, learning the concept of time has long intrigued researchers. For instance, earlier works on arrow-of-time (AoT) [20, 41, 59] sought to determine if a video is playing forward or backward. Building on this line of research, recent studies have extended large multimodal models with AoT awareness [1, 61] as well as broader temporal reasoning capabilities [2, 15, 17, 57].

In the context of video representation learning, temporal information has also been widely exploited as a self-supervised signal [6, 16, 30, 36, 51, 62]. In particular, pace classification has emerged as an effective pretext task for improving temporal understanding, where videos are augmented with different temporal sampling rates and models are trained to classify them as slow, normal, or fast [6, 51, 62]. Extending this idea, SpeedNet [6] demonstrates that the learned pace classifier can further be leveraged to estimate "speediness" scores across a video, enabling adaptive video speedups.

More recently, Pulse-of-Motion [19] proposed a video FPS predictor to evaluate the speed correctness of major video generation models. However, their speed-perception model is trained primarily on standard frame-rate videos ($\leq$ 60 fps), with only 8 videos at 120 fps (UVG [35]) and 118 videos at 240 fps (Adobe240fps [47]). As a result, their notion of speed is limited to a narrow range (1/2, 1, 2, 4$\times$) and oftentimes inaccurate, which we will show in Sec. 5.2. In contrast, our method is trained on videos spanning a wide range of speeds and uses a dedicated self-supervised objective, allowing it to predict speed across a wide range of temporal scales.

Manipulating the continuum of time. Understanding and altering the temporal dimension of videos is a fundamental problem in computer vision. Frame interpolation addresses this by synthesizing intermediate frames between two existing inputs, effectively increasing a video's temporal resolution. Earlier approaches to interpolation primarily relied on kernels [33, 39, 60] or optical flow estimation [24, 42] to model motion, whereas recent generative approaches leverage the rich priors of image or video diffusion models to better handle complex motion and improve realism [14, 18, 25, 53].

Beyond interpolation, prior work on video generation has explored motion control techniques [7, 9, 50, 58] to

Dataset	#Clips	#Videos	#Frames	Max raw FPS	Content
Adobe240fps [47]	118	118	80K	240	Urban
YouTube240 [26]	1,014	-	296K	240	Unknown
NfS [29]	100	100	383K	240	Generic
X4K1000FPS [43]	175*	175	875K	1,000	Urban
SportsSloMo [10]	8,498*	259	1.2M	240	Sports
SloMo-44K (Ours)	44,632	18,235	18M	10,000+	Generic

Table 1 Comparison with existing high-frame-rate datasets. Our dataset is the largest generic slow-motion collection to date, containing over 70× more videos and 150× more frames than previous datasets. *Counts are based on full-length clips rather than short, fixed-duration trimmed segments.

explicitly manipulate temporal dynamics. More recently, video editing approaches like BulletTime [56] and SpaceTimePilot [23] have enabled joint editing of camera motion and temporal progression, offering flexible spatio-temporal control.

In this work, we focus on playback speed-conditioned video generation, ranging from normal speed (1.0×) to extreme slow motion (0.01×). Unlike editing-centric settings [23, 56] that merely rescale the temporal axis of an existing video, our generation-centric task requires the model to internalize the real-world progression speed of physical motion and accurately fabricate it at the requested speed (see supp. for detailed comparisons). Furthermore, the broad conditioning range requires our model to generate dynamics at an exceptionally high temporal resolution. By leveraging our proposed SloMo-44K, which spans diverse temporal scales and real-world motion patterns, our model achieves high-fidelity video generation across a wide range of speeds, outperforming existing approaches trained only on low-frame-rate or synthetic data (Sec. 5.3).

Slow-motion video datasets. Large-scale video generation datasets such as WebVid-10M [5], Panda-70M [11], and OpenVid-1M [38] have driven recent advances in video generation. However, these datasets primarily contain standard-speed footage (24–60 fps) and thus lack the dense temporal continuity required to model realistic motion dynamics. In contrast, slow-motion videos, recorded at hundreds or even thousands of fps, offer much finer temporal details that normal-speed videos cannot capture. Existing slow-motion datasets, however, remain limited in either scale, frame rate, or scene diversity. For example, Adobe240fps [47], YouTube240 [26], and NfS [29] each contain only a few hundred videos at 240 fps; SportsSloMo [10] includes more clips but focuses solely on sports scenes; and X4K1000FPS [43] consists of 1000 fps self-recorded videos yet remains small in size. Consequently, these datasets primarily serve as benchmarks for video frame interpolation rather than as large-scale resources for model training. In contrast, our proposed SloMo-44K is the first large-scale and general-purpose slow-motion dataset, offering a foundation for learning fine-grained dynamics.

3 Perceiving the Flow of Time

We aim to develop computational visual systems that can perceive and manipulate the flow of time. In this section, we focus on how models understand and perceive time, answering fundamental questions such as *Has this video been sped up or slowed down? If so, when, and by how much?* These abilities form the foundation for deeper temporal reasoning and pave the way for fine-grained temporal control, which we explore in Sec. 4.

As most of the metadata associated with public slow-motion videos, such as playback speed, is often noisy or incomplete, data for models to learn from is lacking. In the following section, we detail how models can learn to detect speed changes and infer the speed of time through self-supervision. These models then curate the largest slow-motion video dataset to date for visually understanding and controlling the flow of time.

3.1 Learning to Detect Temporal Speed Changes

We begin with detecting temporal speed changes in videos. Our goal is to locate times in the video where the playback speed changes, transitioning between normal, fast, and slow motion. Such effects are common in modern videos and are often used to emphasize key moments or create dramatic effects. If a model can

reliably detect these changes, it not only reveals an understanding of temporal continuity, but also provides a foundation for estimating absolute speed and calibrating temporal labels in large-scale video datasets.

Manually annotating speed changes is laborious and difficult to scale, and using optical flow as a proxy measurement is often noisy and unreliable (see Sec. 5). We instead leverage cues encoded in the audio of videos. Specifically, our key idea is to use the principle of **time–frequency scaling** to detect speed changes. For videos that contain original audio tracks (*e.g.*, not dubbed or overlaid with background music), there exists a natural coupling between visual motion and sound. When such videos are sped up (compressed in time), their corresponding audio frequencies shift upward, resulting in a higher pitch, and vice versa when slowed down. These salient audio cues (pitch shift) allows us to effectively locate speed-change events. An illustration is shown in Fig. 2.

Using this strategy, we automatically collect over 8K speed-change labels. We then finetune a *visual* speed-change detector based on VideoMAEv2 [52] using a standard binary cross-entropy loss. Notably, during inference, the detector *operates solely on visual input* and is agnostic to *audio* content.

3.2 Learning to Infer the Speed of Time

Once we can detect speed changes within videos, a natural follow-up question arises: By how much has the video been sped up or slowed down?

As there is no current reliable source of supervision, we rely on a self-supervised objective. Our key insight is that the speed estimation model should be **equivariant** to temporal resampling, *i.e.,* if we speed up a video by a factor of k, the predicted speed should scale by k. By enforcing this proportional relationship between the model's input and output, we can transform temporal resampling into a powerful self-supervised training signal (figure 3). Formally, let $\mathbf{V}$ be a video clip sampled from a video of duration T, and $\mathbf{V}^k$ be the k-times accelerated version of $\mathbf{V}$, where $k \sim \mathcal{N}(1, \frac{T}{2})$. Denote f_θ as the playback speed estimator. We train the model f_θ using:

$$\mathcal{L} = \left[\log f_\theta(\mathbf{V}^k) - \log(k \cdot f_\theta(\mathbf{V}))\right]^2. \qquad (1)$$

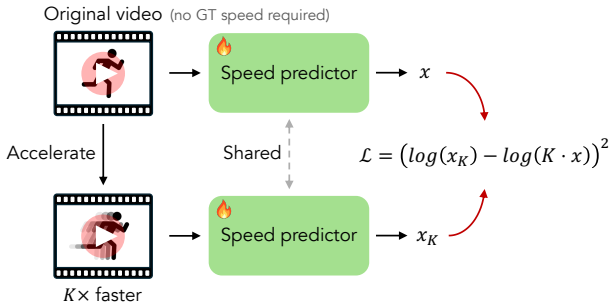

Figure 3 Learning to predict speed. Our speed estimator is trained with both self-supervised and supervised objectives. For videos without ground-truth speed, we enforce temporal consistency by subsampling the video by a factor of K, feeding both original and accelerated clips into the model, and constraining their predicted speeds to differ by K. For videos with known frame rates, we directly regress the playback speed.

Calibration. While the self-supervised loss allows the model to predict playback speed purely from motion cues without relying on ground-truth labels, the estimation is sometimes only correct up to a scale. To anchor predictions to absolute playback rates, we additionally incorporate a small set of videos with known ground-truth playback speeds. For example, we use videos from the Adobe240FPS dataset [47], where the playback speed is precisely determined. In addition to the self-supervised loss, we train the network with small amounts of supervised data in log space.

Iterative prediction. With calibration, our model can already produce relatively accurate speed estimations. However, in practice, we find that our model tends to underestimate when encountering extremely slow-motion videos. We hypothesize that this is because the motion difference in ultra-slow videos are often very subtle due to resolution constraints. Subsampling frames before inference further makes it challenging. To mitigate the issue, we adopt an iterative prediction approach. If a video is initially predicted to have a playback speed of x, we accelerate the video accordingly to bring it closer to normal speed, and then re-estimate its playback speed to obtain a more precise prediction. Repeating this process iteratively produces progressively refined predictions. The intuition is that speed differences become more discernible when the playback speed approaches one, where more training data is also concentrated, allowing the model to estimate speed more reliably. We empirically unroll the estimator three times. Please refer to the supp. material for more details and ablation study.

Method	$\rho \uparrow$	$r_s \uparrow$	RMSE$\downarrow$	$e^{\text{RMSE}} \downarrow$
Human expert	**0.880**	**0.783**	**0.492**	**1.636**
Optical flow	0.385	0.354	-	-
VideoLLM [12]	0.426	0.308	1.568	4.796
SpeedNet [6]	0.476	0.331	1.261	3.529
Pulse-of-Motion* [19]	0.508	0.525	1.181	3.258
Ours	**0.735**	**0.706**	**0.649**	**1.913**

Table 2 Video speed prediction results. We report the Pearson and Spearman correlation coefficients (ρ, r_s), root mean squared error (RMSE), and e^{RMSE}, all computed in log space. Because optical-flow magnitude is used only to rank relative speed, its RMSE is omitted. Our method significantly narrows the gap between machine and human performance. *Results are obtained using the authors' publicly available checkpoint.

3.3 Annotating the Speed of Time

With reliable speed understanding models in hand, we now leverage them to transform in-the-wild slow-motion videos into a high-quality annotated dataset.

Sources. We source videos from YouTube, Vimeo, and Flickr using queries such as "high frame rate," "high-speed camera," and "slow motion," along with their synonyms. Following [7, 34], we then use TransNetv2 [45] to segment videos into shots and an OCR model [63] to remove video clips with excessive text overlay. To further improve quality, we filter out CGI or screen recordings using Qwen2.5-VL [4] and discard low-quality samples based on video quality assessment [21].

Speed annotation. As many videos have unspecified or heterogeneous playback speed, we leverage our speed change detector to segment videos into clips with homogeneous playback speeds. We then run our speed estimator to annotate the playback speed for each clip. Finally, we densely caption them with InternVL3 [65]. We refer the readers to the supp. material for more details.

SloMo-44K. Our resulting dataset contains 44,632 slow-motion video clips with a total of 18 million frames, making it the largest of its kind. It spans diverse real-world scenes and motion patterns, and covers a broad range of video durations. A comparison with previous datasets is shown in Tab. 1. For more statistics, please see the supp. material.

4 Manipulating the Flow of Time

Having learned to detect speed changes and infer the speed of time in videos, we now turn to the problem of manipulating it. In this section, we demonstrate how models can learn to adjust the dynamics of generated content or to slow down existing videos with fine-grained precision.

4.1 Speed-conditioned Video Generation

Given an image, a text prompt, and a target playback speed, our goal is to develop a model that can generate dynamic visual content unfolding at the desired temporal rate and show physical dynamics (*e.g.*, fluid dynamics, shattering, vibration, etc) perceptible only at that temporal scale.

Intuitively, for any text-to-video model, a potential way to control playback speed would be to incorporate motion-related text modifiers (*e.g.*, `ultra-slow`, `slightly-slow`) into the prompt. However, as we show in Sec. 5.3, textual conditioning alone is insufficient for generating videos at the desired speed, often failing to make any noticeable difference. We conjecture that this is because physical motion cues are entangled with other semantic factors in the text prompt, resulting in a weak and easily overloaded conditioning signal.

With this in mind, we propose to leverage our speed-annotated dataset to train a video model with *explicit speed control*. We build model upon Wan2.1-I2V [49] with two key modifications. Given a target speed, we first discretize it into logarithmically spaced buckets that represent different temporal speeds. Empirically, these buckets span from extremely slow (0.01×) to normal speed (1.0×), covering the range in which most

Figure 4 Speed-conditioned video generation. Given a text prompt and an image as input (first column), our model learns to generate videos of different speed. The first row shows the average image of the video, while the second row visualizes the optical flow between the first and second frame. When the speed control is set to slow, the average image should still be crisp (because the scene does not change much) and the magnitude of optical flow is small (and thus the color is faint). As the speed of the video increases, the average image becomes blurrier and the magnitude of optical flow increases (the color become more saturated). For more results, see the project page.

SloMo-44K videos fall (see supp. for analysis). Then we encode the bucket id with sinusoidal positional embedding ϕ, apply a multilayer perceptron MLP_θ, and add it to the timestep embedding [3]:

$$\texttt{Bucket_ID} = \left\lfloor \frac{\log(\texttt{speed}) - \log(0.01)}{\log(1) - \log(0.01)} \cdot N_{\text{buckets}} \right\rfloor,$$
$$\texttt{time_emb} \leftarrow \texttt{time_emb} + \text{MLP}_\theta \cdot \phi(\texttt{Bucket_ID}). \tag{2}$$

This encourages the model to align its denoising schedule with the temporal speed of the video. Empirically, we set $N_{\text{buckets}} = 10$.

To further enhance the control over temporal speed, we modulate the latent features with playback speed through frame-wise conditioning [3]:

$$\texttt{latent}[i] \leftarrow \texttt{latent}[i] + \text{MLP}_\psi(\phi(i \cdot \texttt{speed})), \tag{3}$$

where $\texttt{latent}[i]$ denotes the latent feature at temporal index i.

We follow the image-to-video setup of Wan2.1-I2V [49]. During training, we optimize both the linear projection layers and the LoRA adapters applied to the transformer backbone. The resulting model supports controllable slow-motion synthesis under text, image, and speed conditions.

4.2 Extreme Temporal Super-Resolution

Finally, we apply our data to the task of temporal super-resolution, where the goal is to transform low-FPS, blurry videos into their high-FPS, clear counterparts. While previous approaches typically assume clean and sharp input frames [14, 42, 53], we focus on a more practical and challenging setting in which the input may have motion blur. Low-FPS videos naturally exhibit stronger motion blur due to longer exposure times, making the reconstruction of high velocity details significantly harder.

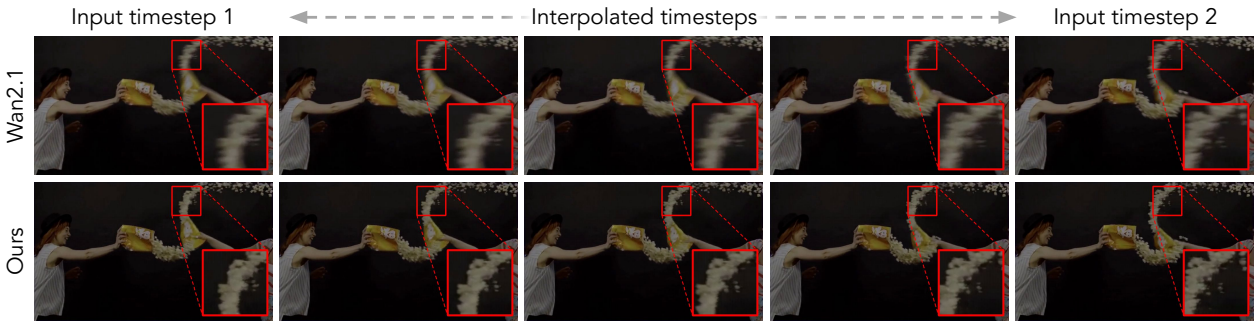

Figure 5 Temporal super-resolution qualitative results. We compare against the strongest baseline, Wan2.1, under the *Blurred-input* setting. Given blurry input frames, the baseline produces blurry intermediate frames, whereas our method generates sharp frames with smooth motion, closely resembling footage captured by a high-speed camera. Inset zooms are provided for clearer visual comparison.

Our key idea is to use our large-scale slow-motion dataset to synthesize a wide range of low-FPS, motion blurred videos for training. Following [8, 37, 47], we generate synthetic blurry inputs by averaging a temporally centered window of 8 frames, followed by temporal subsampling to mimic low-frame-rate capture.

We build our model upon Wan2.1-VACE [27] for its flexible conditioning capability. The VACE framework takes a reference video and an arbitrary binary mask that specifies the frames or regions to be generated. We fine-tune LoRA adapters on top of the pretrained base model. By training on these paired data, the model can jointly perform motion deblurring and frame interpolation, faithfully capturing real-world shooting conditions. Notably, it also achieves superior performance even in the traditional setting with clean, non-blurry inputs. Due to computational resource constraints, we focus on 8-times upsampling.

5 Experiments

In this section, we evaluate how well our proposed techniques perceive and manipulate the flow of time. We first benchmark our speed-change detector and playback-speed estimator, and then showcase that models trained on our dataset exhibit improved speed controllability and temporal super-resolution capabilities.

5.1 Speed Change Detection

Data. We evaluate our model on the test split of our curated speed-change data. Because these labels are automatically derived from audio cues and may contain false positives, we have four annotators re-label them and keep only samples where human annotations and audio-based labels agree. We sample 2-second clips from test videos. If the speed change occurs between $\frac{1}{3}$ and $\frac{2}{3}$ of the clip, we treat it as a positive sample; if the speed change occurs elsewhere, or if no speed change is present, we treat it as a negative sample.

Baselines. We compare against a SOTA VideoLLM, Gemini 2.5 [12], as well as a flow-based detector. For the latter, we adopt SEA-RAFT [55] to compute flow magnitudes between adjacent frames, smooth the magnitudes with a moving window of size 5, and classify a clip as positive if the magnitude within its middle third changes by more than a threshold determined based on a held-out set.

Results. Our model achieves the highest test accuracy (92.4%), outperforming both Gemini 2.5 (59.5%) and the flow-based baseline (80.4%). We further apply our model to an iconic movie scene in *X-Men* with time-freeze effects (first row of figure 1). We visualize model predictions alongside movie frames at corresponding timesteps. To show motion intensity at each moment, we blend each frame with its neighbors. Here, time accelerates from slow motion to normal speed, causing initially sharp frames to become blurry. Our model accurately identifies this transition moment, demonstrating its potential for video forensics.

Method	DAVIS				SloMo-44K-Test			
	FloLPIPS↓	LPIPS↓	FID↓	FVD↓	FloLPIPS↓	LPIPS↓	FID↓	FVD↓
FILM [42]	<u>0.252</u>	**0.200**	<u>20.7</u>	711.3	<u>0.087</u>	<u>0.066</u>	**10.7**	<u>257.6</u>
LDMVFI [14]	0.307	0.251	31.6	916.0	0.139	0.113	22.7	443.0
GI [53]	0.353	0.234	21.4	<u>552.1</u>	0.124	0.093	14.9	503.6
Wan2.1 [27]	0.316	0.260	24.7	634.5	0.108	0.078	16.6	594.0
Ours	**0.242**	<u>0.203</u>	**18.2**	**394.0**	**0.078**	**0.061**	<u>10.9</u>	**182.2**

Table 3 Temporal super-resolution from clear inputs. Following prior work on frame interpolation [14, 42, 53], we set aside the motion blur typical of low-FPS capture and evaluate under *clean* inputs. Specifically, we obtain low-FPS inputs by 8× temporal subsampling and task models with reconstructing the original videos. Finetuned on SloMo-44K, our model outperforms prior methods on *video-based* metrics (*i.e.*, FloLPIPS, FVD) and matches or achieves the best results on *image-based* metrics (*i.e.*, LPIPS, FID). Best results are in **bold** and second-best are <u>underlined</u>.

Method	FloLPIPS↓	LPIPS↓	FID↓	FVD↓
FILM [42]	0.099	0.080	20.3	250.0
LDMVFI [14]	0.136	0.117	31.4	340.4
GI [53]	0.123	0.108	26.1	439.4
Wan2.1 [27]	0.126	0.095	28.5	436.1
Ours	**0.067**	**0.058**	**12.4**	**134.3**

Table 4 Temporal super-resolution results on motion-blurred videos (SloMo-44K-Test). Our method achieves the best performance across all metrics, demonstrating robust temporal super-resolution under motion blur, a common artifact in real-world low-FPS videos.

5.2 Playback Speed Estimation

Data. We collect 111 videos with verified playback speeds by reviewing online sources that explicitly state the playback rate in the video title or description.

Metrics. We evaluate the quality of speed predictors using the following metrics: (1) Pearson correlation coefficient ρ, which measures the linear relationship between predictions and ground truth; (2) Spearman's rank correlation r_s, which evaluates the correctness of their ordinal relationships; and (3) root mean squared error (RMSE). We transform playback speeds to log space to ensure scale invariance. For interpretability, we also report e^{RMSE}, which reflects the average multiplicative deviation between predictions and ground truth in linear space.

Baselines. We consider four baselines: Gemini 2.5 [12], SpeedNet [6], Pulse-of-Motion [19], optical flow magnitude, and expert human judgment. For Gemini, we increase its sampling rate to 8 FPS to better capture temporal cues and prompt it to estimate playback speed. For the flow-based baseline, we use SEA-RAFT [55] to compute the optical flow between consecutive frames and then aggregate the flow magnitude by averaging it across the entire video. Since flow magnitude is only used as a relative, but not an absolute, measure of speed, we do not report absolute error metrics (*e.g.*, RMSE). For human evaluation, we design an interface that allows annotators to interactively adjust playback rate until the perceived motion feels like the real-world speed. In practice, it takes on average ∼40 seconds to annotate the speed of a video.

Results. As shown in table 2, our method significantly outperforms Gemini, SpeedNet, and the optical-flow baseline, and substantially narrows the gap between human and model performance.

5.3 Speed-conditioned Video Generation

Setup. We set the playback speeds of our model directly via control values of 1, 4, 7, and 10, which corresponds to increasingly faster motion. We then utilize 56 image–prompt pairs from the test split of SloMo-44K, spanning diverse scenes and motion patterns, for evaluation.

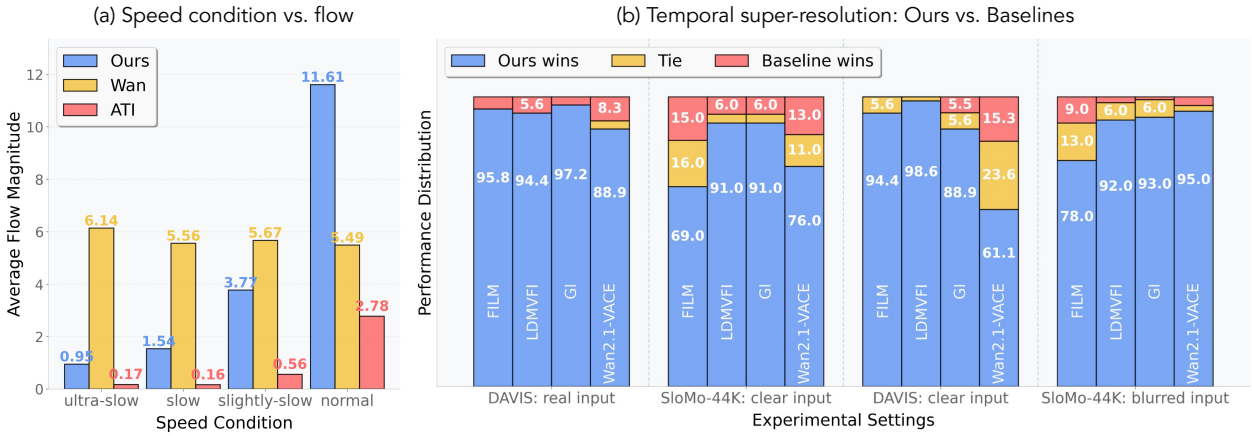

Figure 6 (a) Speed condition vs. optical flow. Our speed control strongly correlates with the average optical flow magnitude of videos generated under different speed conditions, whereas the baselines yield similar motion magnitudes for slow and ultra-slow speeds. **(b) User study on temporal super-resolution.** The results show that human users consistently prefer videos generated by our model (blue) over those generated by baselines (red) across all settings.

Method	FID↓	FVD↓
ATI [50]	73.4	1473.5
Wan2.1 [49]	72.2	1266.8
Ours	**68.4**	**1114.1**

Table 5 Speed-controlled video generation results. We compare our video model trained on SloMo-44K with the pretrained Wan2.1 [49] and the trajectory-based model, ATI [50]. Our model demonstrates superior quality in slow-motion generation.

Training data	FID↓	FVD↓
Standard videos	72.4	1392.9
SloMo-44K	**68.4**	**1114.1**

Table 6 Speed-conditioned video generation with different training data. We train our video generation model on either SloMo-44K or standard-FPS datasets. Results show that training on SloMo-44K yields superior slow-motion generation quality, confirming the importance of high-frame-rate data.

Metrics. To measure speed controllability, we compute the average optical flow magnitude for each speed condition and examine whether the resulting magnitudes follow the expected speed ordering. To assess video generation quality, we compute FID [22] and FVD [48] on all 48 text-image–speed triplets from the evaluation set that have corresponding ground-truth videos.

Baselines. We compare against the base model, Wan2.1 [49], and a trajectory-based motion control model, ATI [50]. Since Wan2.1 does not natively support playback speed control, we approximate different speeds using prompt modifiers (*e.g.*, "ultra slow-motion," "slow-motion," "slightly slow-motion," and "normal"). For the trajectory-based baseline, we first generate a normal-speed video using Wan2.1 and extract its dense tracklets using CoTracker3 [28]. We then linearly interpolate these tracklets to obtain motion trajectories that are 1/4/16/64 times slower and provide them to ATI as trajectory conditions.

Results. We report FID and FVD in table 5, and optical flow analysis in figure 6(a). Our model achieves higher slow-motion generation quality while exhibiting superior speed controllability. As shown qualitatively in figure 4, videos generated with smaller playback-speed values appear slower than those produced with larger values, indicating that the conditioning signal is effectively learned.

5.4 Extreme Temporal Super-Resolution

Setup. Evaluation is performed on both DAVIS [40] and our SloMo-44K under three settings: (1) *Clear-input*: on both datasets, we subsample every 8th frame and have the model interpolate frames in between; (2) *Blurred-input*: we synthesize motion blur in low-FPS test videos following the same procedure described in section 4.2, then have the models reconstruct the original clear SloMo-44K videos; (3) *Real-input*: Since DAVIS videos are normal speed and already contain strong motion blur, we have the models convert them to

Method	IP	$\rho \uparrow$	$r_s \uparrow$	RMSE $\downarrow$
VideoLLM [12]		0.426	0.308	1.568
	✓	**0.552**	**0.479**	**1.221**
Ours		0.680	0.684	0.917
	✓	**0.735**	**0.706**	**0.649**

Table 7 Ablation on iterative prediction (IP). We conduct an ablation study on the IP mechanism. The results show that the inclusion of IP consistently improves performance for both VideoLLM and our self-supervised model, demonstrating its robustness in enhancing speed prediction accuracy.

Training data	IP	$\rho \uparrow$	$r_s \uparrow$	RMSE $\downarrow$
Standard Videos		0.242	0.223	1.232
	✓	0.632	0.598	0.737
SloMo-44K		0.680	0.684	0.917
	✓	**0.735**	**0.706**	**0.649**

Table 8 Playback-speed prediction using different training data. We train the speed predictor using either SloMo-44K or only standard datasets (Adobe240fps [47] and normal-speed videos). The results show that exposing the model to a broad range of temporal scales improves its speed prediction capability.

8× slow motion, testing their ability to enhance real-world, low-FPS footage. Since ground-truth slow motion is unavailable, we conduct a user study for evaluation.

Baselines. We benchmark our method against four interpolation models: FILM [42], LDMVFI [14], Generative Inbetween [53], and vanilla Wan2.1-VACE [27]. For models that generate only a single intermediate frame, we recursively apply them to achieve 8× temporal super-resolution.

Metrics. We report standard temporal super-resolution metrics (*e.g.*, LPIPS [64], FID [22], FloLPIPS [13], and FVD [48]) along with human preference rates. For the user study, human raters are presented with two videos in each trial, one from our model and the other from a baseline, and asked to select the better one.

Results. We show the quantitative comparison under the *Clean-input* setting in table 3, those under the *Blurred-input* setting in table 4, the user study results across all settings in figure 6(b), and qualitative results in figure 5. Our model consistently outperforms all baselines and achieves over 90% user preference in the *Real-input* setting (figure 6(b)). These results demonstrate our model's superior ability to model real-world dynamics at high temporal resolution.

5.5 Analysis

In this section, we conduct ablation studies to examine the impact of iterative prediction on speed estimation and to assess the importance of using SloMo-44K to train both speed predictors and speed-conditioned video generation models.

Importance of Iterative prediction. The ablation study in table 7 shows that IP enhances the accuracy of both our model and VideoLLM, confirming the technique's robustness across models.

Importance of SloMo-44K. Is SloMo-44K necessary for training our proposed models? To answer this question, we first compare our speed predictor (trained on SloMo-44K) with a baseline trained solely on standard datasets (Adobe240fps [47] and normal-speed videos). The result in table 8 shows that models exposed to a wider variety of speeds outperform those trained only on standard videos. We further evaluate our speed-conditioned model against a baseline trained on videos artificially slowed from standard-FPS footage, following [23]. As shown in table 6 and figure 7, the baseline exhibits stuttering artifacts, whereas our model faithfully synthesizes slow-motion dynamics.

(a) Trained on SloMo-44K (b) Trained on standard videos

Figure 7 Speed-controlled video generation with different training data. Training on standard-FPS videos with artificial slowdowns leads to stuttering artifacts, while training on SloMo-44K results in realistic slow-motion dynamics. *Videos are embedded. Please view with Adobe Acrobat Reader.*

6 Discussion

In this work, we showcase how we can leverage the multimodal cues, temporal structure inherently present in videos, and the vast amount of slow motion videos available on the internet to develop models that can perceive and manipulate the flow of time. While we achieve state-of-the-art performance across both understanding and generation tasks, we still encounter several limitations. For instance, our speed understanding models can be misled when videos contain limited motion cues or when people deliberately move slowly; our generation models also rely on a pretrained Wan backbone, leaving room for improvement through architectural innovations or full fine-tuning. We believe this work establishes time as a manipulable dimension in video learning and opens new directions for temporal forensics and richer world models that capture how events unfold.

Acknowledgment

The research is partially supported by a gift from Ai2, NVIDIA Academic Grant, DARPA TIAMAT program No. HR00112490422, and the National Science and Technology Council via grants NSTC 114-2634-F-002-006 and NSTC 114-2640-E-002-006. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of DARPA. We also thank Xuyi Meng, Chuanruo Ning, and Guangzhao He for their valuable contributions and discussions, and the National Center for High-performance Computing (NCHC) for providing computational and storage resources.

Author Contributions

Yen-Siang Wu led the project, designed and curated the SloMo-44K dataset, proposed the core methodology, and conducted the primary experiments and manuscript drafting. **Rundong Luo** helped data curation, contributed to the design, training, and evaluation of the video generation models, performed the literature review and manuscript drafting. **Jingsen Zhu** executed the preliminary experiments that informed the development of the proposed approach. **Tao Tu** managed the user studies, prepared the visual demonstrations, and assisted in proofreading.

References

[1] Alisson Azzolini, Junjie Bai, Hannah Brandon, Jiaxin Cao, Prithvijit Chattopadhyay, Huayu Chen, Jinju Chu, Yin Cui, Jenna Diamond, Yifan Ding, et al. Cosmos-reason1: From physical common sense to embodied reasoning. *arXiv:2503.15558*, 2025.

[2] Piyush Bagad, Makarand Tapaswi, and Cees GM Snoek. Test of time: Instilling video-language models with a sense of time. In *CVPR*, 2023.

[3] Sherwin Bahmani, Ivan Skorokhodov, Guocheng Qian, Aliaksandr Siarohin, Willi Menapace, Andrea Tagliasacchi, David B Lindell, and Sergey Tulyakov. Ac3d: Analyzing and improving 3d camera control in video diffusion transformers. In *CVPR*, 2025.

[4] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv:2502.13923*, 2025.

[5] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *ICCV*, 2021.

[6] Sagie Benaim, Ariel Ephrat, Oran Lang, Inbar Mosseri, William T Freeman, Michael Rubinstein, Michal Irani, and Tali Dekel. Speednet: Learning the speediness in videos. In *CVPR*, 2020.

[7] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv:2311.15127*, 2023.

[8] Tim Brooks and Jonathan T Barron. Learning to synthesize motion blur. In *CVPR*, 2019.

[9] Ryan Burgert, Yuancheng Xu, Wenqi Xian, Oliver Pilarski, Pascal Clausen, Mingming He, Li Ma, Yitong Deng, Lingxiao Li, Mohsen Mousavi, et al. Go-with-the-flow: Motion-controllable video diffusion models using real-time warped noise. In *CVPR*, 2025.

[10] Jiaben Chen and Huaizu Jiang. Sportsslomo: A new benchmark and baselines for human-centric video frame interpolation. *arXiv:2308.16876*, 2023.

[11] Tsai-Shien Chen, Aliaksandr Siarohin, Willi Menapace, Ekaterina Deyneka, Hsiang-wei Chao, Byung Eun Jeon, Yuwei Fang, Hsin-Ying Lee, Jian Ren, Ming-Hsuan Yang, et al. Panda-70m: Captioning 70m videos with multiple cross-modality teachers. In *CVPR*, 2024.

[12] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv:2507.06261*, 2025.

[13] Duolikun Danier, Fan Zhang, and David Bull. Flolpips: A bespoke video quality metric for frame interpolation. In *Picture Coding Symposium*, 2022.

[14] Duolikun Danier, Fan Zhang, and David Bull. Ldmvfi: Video frame interpolation with latent diffusion models. In *AAAI*, 2024.

[15] Xi Ding and Lei Wang. Do language models understand time? In *ACM WWW*, 2025.

[16] Michael Dorkenwald, Fanyi Xiao, Biagio Brattoli, Joseph Tighe, and Davide Modolo. Scvrl: Shuffled contrastive video representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4132–4141, 2022.

[17] Yang Du, Yuqi Liu, and Qin Jin. Reversed in time: A novel temporal-emphasized benchmark for cross-modal video-text retrieval. In *ACM MM*, 2024.

[18] Haiwen Feng, Zheng Ding, Zhihao Xia, Simon Niklaus, Victoria Abrevaya, Michael J Black, and Xuaner Zhang. Explorative inbetweening of time and space. In *ECCV*, 2024.

[19] Xiangbo Gao, Mingyang Wu, Siyuan Yang, Jiongze Yu, Pardis Taghavi, Fangzhou Lin, and Zhengzhong Tu. The pulse of motion: Measuring physical frame rate from visual dynamics. *arXiv preprint arXiv:2603.14375*, 2026.

[20] Amir Ghodrati, Efstratios Gavves, and Cees GM Snoek. Video time: Properties, encoders and evaluation. *arXiv preprint arXiv:1807.06980*, 2018.

[21] Chenlong He, Qi Zheng, Ruoxi Zhu, Xiaoyang Zeng, Yibo Fan, and Zhengzhong Tu. Cover: A comprehensive video quality evaluator. In *CVPRW*, 2024.

[22] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017.

[23] Zhening Huang, Hyeonho Jeong, Xuelin Chen, Yulia Gryaditskaya, Tuanfeng Y Wang, Joan Lasenby, and Chun-Hao Huang. Spacetimepilot: Generative rendering of dynamic scenes across space and time. *arXiv:2512.25075*, 2025.

[24] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In *ECCV*, 2022.

[25] Siddhant Jain, Daniel Watson, Eric Tabellion, Ben Poole, Janne Kontkanen, et al. Video interpolation with diffusion models. In *CVPR*, 2024.

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

[27] Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. Vace: All-in-one video creation and editing. In *ICCV*, 2025.

[28] Nikita Karaev, Yuri Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. In *ICCV*, 2025.

[29] Hamed Kiani Galoogahi, Ashton Fagg, Chen Huang, Deva Ramanan, and Simon Lucey. Need for speed: A benchmark for higher frame rate object tracking. In *ICCV*, 2017.

[30] Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE international conference on computer vision*, pages 667–676, 2017.

[31] KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. Videochat: Chat-centric video understanding. *arXiv:2305.06355*, 2023.

[32] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. In *EMNLP*, 2024.

[33] Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *ICCV*, 2017.

[34] Rundong Luo, Matthew Wallingford, Ali Farhadi, Noah Snavely, and Wei-Chiu Ma. Beyond the frame: Generating 360° panoramic videos from perspective videos. In *ICCV*, 2025.

[35] Alexandre Mercat, Marko Viitanen, and Jarno Vanne. Uvg dataset: 50/120fps 4k sequences for video codec analysis and development. In *Proceedings of the 11th ACM multimedia systems conference*, pages 297–302, 2020.

[36] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European conference on computer vision*, pages 527–544. Springer, 2016.

[37] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *CVPR*, 2017.

[38] Kepan Nan, Rui Xie, Penghao Zhou, Tiehan Fan, Zhenheng Yang, Zhijie Chen, Xiang Li, Jian Yang, and Ying Tai. Openvid-1m: A large-scale high-quality dataset for text-to-video generation. *arXiv:2407.02371*, 2024.

[39] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *CVPR*, 2017.

[40] 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 *CVPR*, 2016.

[41] Lyndsey C Pickup, Zheng Pan, Donglai Wei, YiChang Shih, Changshui Zhang, Andrew Zisserman, Bernhard Scholkopf, and William T Freeman. Seeing the arrow of time. In *CVPR*, 2014.

[42] Fitsum Reda, Janne Kontkanen, Eric Tabellion, Deqing Sun, Caroline Pantofaru, and Brian Curless. Film: Frame interpolation for large motion. In *ECCV*, 2022.

[43] Hyeonjun Sim, Jihyong Oh, and Munchurl Kim. Xvfi: extreme video frame interpolation. In *ICCV*, 2021.

[44] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, et al. Moviechat: From dense token to sparse memory for long video understanding. In *CVPR*, 2024.

[45] Tomás Soucek and Jakub Lokoc. Transnet v2: An effective deep network architecture for fast shot transition detection. In *ACM MM*, 2024.

[46] Verena Steinhof, Anna Schroeger, Roman Liepelt, and Laura Sperl. Time and video speed perception: a comprehensive investigation of the relation between estimated video speed, clip duration and original duration: V. steinhof et al. *Cognitive Research: Principles and Implications*, 2025.

[47] Shuochen Su, Mauricio Delbracio, Jue Wang, Guillermo Sapiro, Wolfgang Heidrich, and Oliver Wang. Deep video deblurring for hand-held cameras. In *CVPR*, 2017.

[48] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. Fvd: A new metric for video generation. In *ICLRW*, 2019.

[49] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv:2503.20314*, 2025.

[50] Angtian Wang, Haibin Huang, Zhiyuan Fang, Yiding Yang, and Chongyang Ma. ATI: Any trajectory instruction for controllable video generation. *arXiv:2505.22944*, 2025.

[51] Jiangliu Wang, Jianbo Jiao, and Yun-Hui Liu. Self-supervised video representation learning by pace prediction. In *ECCV*, 2020.

[52] Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. Videomae v2: Scaling video masked autoencoders with dual masking. In *CVPR*, 2023.

[53] Xiaojuan Wang, Boyang Zhou, Brian Curless, Ira Kemelmacher-Shlizerman, Aleksander Holynski, and Steven M Seitz. Generative inbetweening: Adapting image-to-video models for keyframe interpolation. In *ICLR*, 2024.

[54] Yi Wang, Yinan He, Yizhuo Li, Kunchang Li, Jiashuo Yu, Xin Ma, Xinhao Li, Guo Chen, Xinyuan Chen, Yaohui Wang, et al. Internvid: A large-scale video-text dataset for multimodal understanding and generation. *arXiv:2307.06942*, 2023.

[55] Yihan Wang, Lahav Lipson, and Jia Deng. Sea-raft: Simple, efficient, accurate raft for optical flow. In *ECCV*, 2024.

[56] Yiming Wang, Qihang Zhang, Shengqu Cai, Tong Wu, Jan Ackermann, Zhengfei Kuang, Yang Zheng, Frano Rajič, Siyu Tang, and Gordon Wetzstein. Bullettime: Decoupled control of time and camera pose for video generation. *arXiv:2512.05076*, 2025.

[57] Zhenhailong Wang, Ansel Blume, Sha Li, Genglin Liu, Jaemin Cho, Zineng Tang, Mohit Bansal, and Heng Ji. Paxion: Patching action knowledge in video-language foundation models. In *NeurIPS*, 2023.

[58] Zhouxia Wang, Ziyang Yuan, Xintao Wang, Yaowei Li, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. Motionctrl: A unified and flexible motion controller for video generation. In *SIGGRAPH*, 2024.

[59] Donglai Wei, Joseph J Lim, Andrew Zisserman, and William T Freeman. Learning and using the arrow of time. In *CVPR*, 2018.

[60] Xiaoyu Xiang, Yapeng Tian, Yulun Zhang, Yun Fu, Jan P Allebach, and Chenliang Xu. Zooming slow-mo: Fast and accurate one-stage space-time video super-resolution. In *CVPR*, 2020.

[61] Zihui Xue, Mi Luo, and Kristen Grauman. Seeing the arrow of time in large multimodal models. *arXiv preprint arXiv:2506.03340*, 2025.

[62] Yuan Yao, Chang Liu, Dezhao Luo, Yu Zhou, and Qixiang Ye. Video playback rate perception for self-supervised spatio-temporal representation learning. In *CVPR*, 2020.

[63] Maoyuan Ye, Jing Zhang, Shanshan Zhao, Juhua Liu, Bo Du, and Dacheng Tao. Dptext-detr: Towards better scene text detection with dynamic points in transformer. In *AAAI*, 2023.

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

[65] Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv:2504.10479*, 2025.