FloVD: Optical Flow Meets Video Diffusion Model for Enhanced Camera-Controlled Video Synthesis

Wonjoon Jin[1,2†] Qi Dai[2] Chong Luo[2] Seung-Hwan Baek[1] Sunghyun Cho[1]

[1]POSTECH [2]Microsoft Research Asia

Abstract

This paper presents FloVD, a novel optical-flow-based video diffusion model for camera-controllable video generation. FloVD leverages optical flow maps to represent motions of the camera and moving objects. This approach offers two key benefits. Since optical flow can be directly estimated from videos, our approach allows for the use of arbitrary training videos without ground-truth camera parameters. Moreover, as background optical flow encodes 3D correlation across different viewpoints, our method enables detailed camera control by leveraging the background motion. To synthesize natural object motion while supporting detailed camera control, our framework adopts a two-stage video synthesis pipeline consisting of optical flow generation and flow-conditioned video synthesis. Extensive experiments demonstrate the superiority of our method over previous approaches in terms of accurate camera control and natural object motion synthesis.

1. Introduction

Video diffusion models have made significant strides in generating high-quality videos by leveraging large-scale datasets [3–5, 12, 14, 22, 26, 31, 39]. However, they often lack the ability to incorporate user-defined controls, particularly in terms of camera movement and perspective. This limitation restricts the practical applications of video diffusion models, where precise control over camera parameters is crucial for various tasks such as film production, virtual reality, and interactive simulations.

Recently, several approaches have introduced camera controllability to video diffusion models. One line of methods uses either text descriptions or user-drawn strokes that describe background motion as conditional inputs to represent camera motion [6, 22, 25, 33, 40]. However, these methods support only limited camera controllability, such as zoom and pan, during video generation.

More sophisticated camera control has been achieved by directly using camera parameters as inputs [2, 11, 18, 32–35, 38, 41, 42]. In particular, recent methods embed input camera parameters using the Plücker embedding scheme [27], which involves embedding ray origins and directions, and feed them into video diffusion models [2, 11, 35, 42]. While these approaches offer more detailed control, they require a training dataset that includes ground-truth camera parameters for every video frame. Acquiring such datasets is challenging, leading to the use of restricted datasets that primarily consist of static scenes, such as RealEstate10K [43]. Consequently, they suffer from limited generalization capability, producing videos with unnatural object motions and inaccurate camera control (Fig. 1).

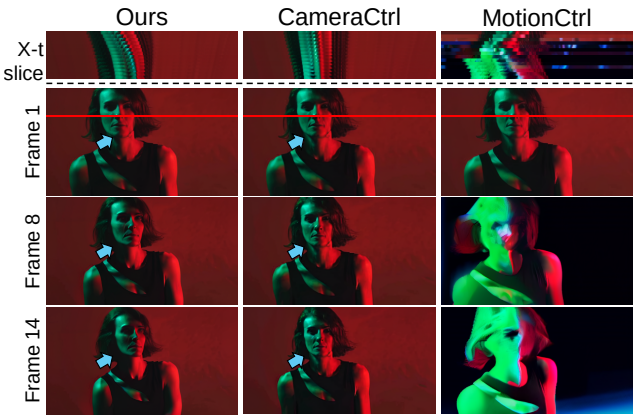

Figure 1. Synthesized video frames with 'zoom-out' camera motion. X-t slice reveals pixel value changes along the red line. Our method shows natural object motion and accurate camera control, while CameraCtrl [11] produces a foreground object without motions, and MotionCtrl [33] produces artifacts.

To enable realistic video synthesis with natural object motions and accurate camera control, our key idea is using *optical flow* as conditional input to a video diffusion model. This approach provides two key benefits. First, since optical flow can be directly estimated from videos, our method eliminates the need for using datasets with ground-truth camera parameters. This flexibility enables our model to utilize arbitrary training videos. Second, since background motion of optical flow encodes 3D correlations across different viewpoints, our method enables detailed camera con-

[†]Work done during an internship at Microsoft Research Asia.

trol by leveraging the background motion. As a result, our approach facilitates natural object motion synthesis and precise camera control (Fig. 1).

Based on the key idea, this paper presents *FloVD*, a novel camera-controllable video generation framework that leverages optical flow. Given a single image and camera parameters, FloVD synthesizes future frames with a desired camera trajectory. To this end, our framework employs a two-stage video synthesis pipeline: optical flow generation and flow-conditioned video synthesis. We generate optical flow maps representing motions of the camera and moving objects from the input image and camera parameters. This flow maps are then fed into a flow-conditioned video synthesis model to generate the final output video.

To synthesize natural object motion while supporting detailed camera control, FloVD divides the optical flow generation stage into two sub-problems: camera flow generation and object flow generation. First, we convert input camera parameters into optical flow of the background motion by using 3D structure estimated from the input image. Next, an object motion synthesis model is introduced to generate the optical flow for object motions based on the input image. By combining both the background and object motion flows, we obtain the final optical flow maps.

Our contributions are summarized as follows:

- We present a novel camera-controllable video generation framework that leverages optical flow, allowing our method to utilize arbitrary training videos without ground-truth camera parameters.
- To achieve detailed camera control and high-quality video synthesis, we adopt a two-stage video synthesis pipeline, flow generation and flow-conditioned video synthesis.
- Extensive evaluation demonstrates the effectiveness of our method, showcasing its ability to produce high-quality videos with accurate camera control and natural object motion.

2. Related Work

2.1. Camera-Controllable Video Synthesis

Following the tremendous success of video diffusion models, numerous efforts have been made to integrate camera controllability into the video generation process. MovieGen [22] uses text descriptions that describe camera motions to control camera motion. MCDiff [6], DragNUWA [40], and MotionI2V [25] enable camera control through user-provided strokes, manipulating background motion to adjust camera movement. AnimateDiff [9] and Direct-a-video [38] enable camera control by training models on an augmented video datasets that contain simple camera movements, such as translation. While these methods offer basic camera control with high-level instructions such as zoom and pan, they lack the detailed control capability for user-defined specific camera motions.

Recent methods have demonstrated detailed control of camera movement by using desired camera parameters as conditional input. To this end, these approaches train models on datasets that provide ground-truth camera parameters for every video frame. MotionCtrl [33] directly projects the camera extrinsic parameters onto the intermediate features of a diffusion model, while CameraCtrl [11] leverages the Plücker embedding scheme [27] to encode the camera origin and ray directions as conditioning input. CamCo [35] and CamI2V [42] enhance camera control through an epipolar attention mechanism across video frames. VD3D [2] enables camera motion control within the video synthesis process of transformer-based video diffusion models.

While these methods support detailed camera control, they are primarily trained on restricted datasets [43] due to the requirement for camera parameters during training. This limitation degrades the generalization ability and leads to unnatural object motion synthesis. In contrast, our model can be trained on arbitrary videos by leveraging optical flow maps as input, which can be robustly estimated using recent optical flow estimation models [28].

2.2. Object Motion Synthesis

Besides camera motion control, controlling object motion during video generation has been practiced either by utilizing user-provided inputs or by learning the distribution of object motion from data. Several studies employ user-stroke input [6, 10, 25, 40] or modulation of attention maps during diffusion model's denoising process [38] to enable desired control of object motion. Another research direction focuses on synthesizing object motion by learning the diverse motions present in video datasets [7, 8, 15, 19, 30]. Given a single image, these methods synthesize flow maps to represent natural object motion and use them to animate the input image. However, these methods are primarily aimed at object motion synthesis and do not support detailed camera motion control during video generation.

3. FloVD Framework

Fig. 2 presents an overview of FloVD. Our framework takes an image I^s and camera parameters $\mathcal{C} = \{C_t\}_{t=1}^{T}$ as input where t is a video frame index, and T is the number of video frames. C_t is defined as a set of extrinsic and intrinsic camera parameters. Given the input conditions, our framework synthesizes a video $\mathcal{I} = \{I_t\}_{t=1}^{T}$ that starts from I^s as the first frame and follows the input camera trajectory, where I_t is the t-th video frame.

FloVD consists of two stages. First, the flow generation stage synthesizes two sets of optical-flow maps that represent camera and object motions using 3D warping and an object motion synthesis model (OMSM), respectively. We refer to these optical flow maps as camera flow maps

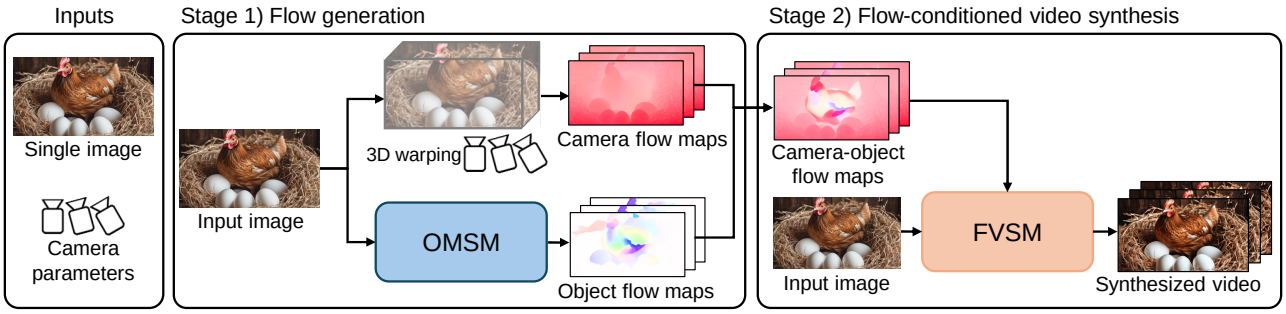

Figure 2. Overview of FloVD. Given an image and camera parameters, our framework synthesizes a video frames following the input camera trajectory. To this end, we synthesize two sets of optical flow maps that represent camera and object motions. Then, two optical flow maps are integrated and fed into the flow-conditioned video synthesis model, enabling camera-controllable video generation.

and object flow maps. These flow maps are integrated to form a single set of optical flow maps, which we refer to as camera-object flow maps. In the subsequent stage, a flow-conditioned video synthesis model (FVSM) synthesizes a video using the input image I^s and the camera-object flow maps. In the following, we describe each stage in detail.

3.1. Flow Generation

Camera flow generation. In the flow generation stage, we first generate camera flow maps $\mathcal{F}^c = \{f_t^c\}_{t=1}^T$ where f_t^c is an optical flow map from the first frame to the t-th frame. To generate camera flow maps reflecting the 3D structure in the input image I^s, we estimate a depth map d^s from I^s using an off-the-shelf single-image 3D estimation network [37]. Using the estimated depth map, we unproject each pixel coordinate x^s in the input image I^s into the 3D space. Then, for each t, we warp the unprojected coordinates and project them back to the 2D plane using C_t to obtain the warped coordinate x_t. Finally, we construct the camera flow map f_t^c by computing displacement vectors from x^s to x_t.

Object flow synthesis. In this stage, we also generate object flow maps $\mathcal{F}^o = \{f_t^o\}_{t=1}^T$ that represent object motions independent of background motions. To this end, we develop OMSM based on the latent video diffusion model [3]. Specifically, as shown in Fig. 3(a), OMSM consists of a denoising U-Net, and an encoder and decoder of the latent video diffusion model's variational autoencoder (VAE). In OMSM, the input image I^s is first encoded by the VAE encoder. Then, the denoising U-Net takes a concatenation of the encoded input image and a noisy latent feature volume as input, and iteratively denoises the latent feature volume to synthesize latent object motion flow maps. Finally, the VAE decoder decodes the synthesized result and produces object flow maps $\mathcal{F}^o$.

Inspired by Marigold [17], we utilize the VAE decoder of the latent video diffusion model, which is trained on RGB images, for decoding object flow maps without any architectural changes or fine-tuning. Specifically, from the three

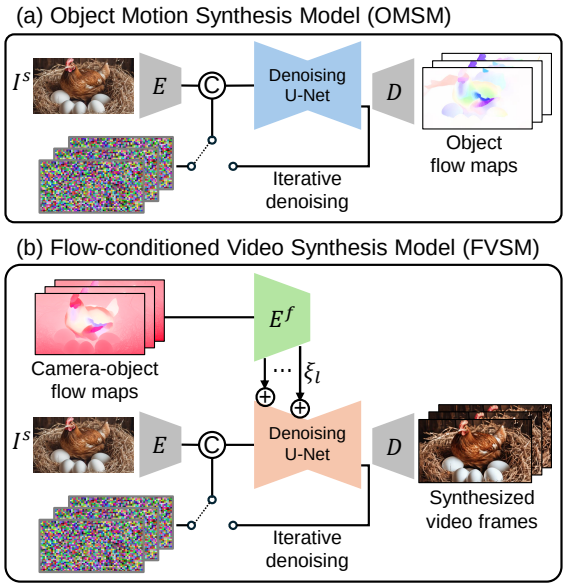

Figure 3. Network architectures of OMSM and FVSM.

output channels of the VAE decoder corresponding to RGB, we use only the first two channels for the x and y components of an object flow map. Our training process also involves the VAE encoder. Like the VAE decoder, we use the VAE encoder of the latent video diffusion model without any architectural changes or fine-tuning. For the three input channels of the VAE encoder, we feed the x and y components of an object flow map, along with their average $(x+y)/2$. We verified that the optical flow map can be reconstructed from the encoded latent feature with a negligible error without any modification of the VAE.

Flow integration. Once the camera and object flow maps, $\mathcal{F}^c$ and $\mathcal{F}^o$, are generated, we obtain camera-object flow maps $\mathcal{F} = \{f_t\}_{t=1}^T$ by combining them. The integration is performed as follows.

First, we estimate a binary mask M^{obj} from the input image I^s using an off-the-shelf segmentation model [23], which indicates pixels corresponding to moving objects. We use a single binary mask M^{obj} for t, as all flow maps are

forward directional optical flow maps from the first frame to the t-th frame. Based on M^{obj}, we combine f_t^c and f_t^o. Specifically, for each pixel x specified by M^{obj}, we compute its displaced position x' using the object motion in f_t^o as $x' = x + f_{t,x}^o$, where $f_{t,x}^o$ is the optical flow vector in f_t^o at pixel x. Next, we transform x' using the camera parameter C_t and the depth map d^s, obtaining x'_t, which represents the displaced position of x at the t-th frame due to both camera and object motions. We then compute the flow vector $f'_{t,x} = x'_t - x$. Finally, we derive the camera-object flow map f_t as:

$$f_{t,x} = (1 - M_x^{obj}) \cdot f_{t,x}^c + M_x^{obj} \cdot f'_{t,x}, \tag{1}$$

where M_x^{obj} is the binary value of M^{obj} at pixel x.

It is important to note that physically valid integration of camera and object flows requires object motion information along the z-axis (orthogonal to the image plane), which is not captured in the object flow maps. Thus, our integration process does not produce physically accurate camera-motion flow maps. However, we experimentally found that our framework can still synthesize videos with natural object motions. This is made possible by the flow-conditioned video synthesis model (FVSM), which is trained on natural-looking videos, ensuring realistic object motions, even for input noisy camera-motion flow maps.

Our OMSM is trained to generate non-zero flow vectors only for dynamic objects. Therefore, we do not necessarily need to use the mask M^{obj} in Eq. (1); instead, we can transform the entire object motion flow maps using the camera parameters. However, we found empirically that using the mask M^{obj} improves video synthesis quality by removing incorrectly synthesized flow vectors in static regions of f_t^o.

3.2. Flow-Conditioned Video Synthesis

The flow-conditioned video synthesis stage synthesizes a video $\mathcal{I}$ using the input image I^s and the camera-object flow maps $\mathcal{F}$ as conditions. To achieve this, our framework utilizes FVSM, which extends the latent video diffusion model [3] by incorporating an additional flow encoder inspired by the T2I-Adapter architecture [20]. Specifically, as shown in Fig. 3(b), our model consists of a flow encoder E^f, a denoising U-Net, and a VAE encoder and decoder.

The flow encoder takes the input camera-object flow maps $\mathcal{F}$, and computes multi-level flow embeddings:

$$\{\xi_{(t,l)}\}_{t=1,l=1}^{T,L} = E^f(\mathcal{F}), \tag{2}$$

where $\xi_{(t,l)}$ is a flow embedding of the t-th frame I_t at level l. Each flow embedding has the resolution of its corresponding layer's latent feature in the denoising U-Net. The denoising U-Net takes a concatenation of the encoded input image and a noisy latent feature volume as input, iteratively denoises the latent feature volume of the video. Additionally, the denoising U-Net also takes the multi-level flow embeddings by adding each of them to the feature at each layer of the denoising U-Net. Finally, the synthesized video frames are obtained by decoding the denoised latent feature volume using the VAE decoder. More details on the network architecture can be found in the supplemental document.

4. FloVD Training

FloVD utilizes two diffusion models: OMSM and FVSM, which are trained separately. As discussed in Sec. 1, both models can be effectively trained using a wide range of videos with dynamic object motions without requiring ground-truth camera parameters, thanks to our optical-flow-based representation. In the following, we explain our datasets and training strategies for our models.

4.1. Training Datasets

For training these diffusion models, we primarily use an internal dataset containing 500K video clips, and its subset of video clips without camera motions. We refer to these as the full dataset and the curated dataset, respectively. The full dataset contains scenes similar to those in the Pexels dataset [1]. The curated dataset contains around 100K video clips. For training OMSM, we use both datasets, while for training FVSM, we use only the full dataset.

Training the diffusion models in our framework requires optical flow maps for each video clip. We estimate the optical flow maps using an off-the-shelf estimator [28], and use them as the ground-truth object flow maps for OMSM, and the camera-object flow maps for FVSM.

The curated dataset is generated through the following process. For each video clip in the full dataset, we first detect the static background region from the first frame using an off-the-shelf semantic segmentation model [23]. Next, we compute the average magnitude of the optical flow vectors for all video frames within the background region. If this average magnitude is smaller than a specified threshold, we consider the video clip to have no camera motion and include it in the curated dataset.

4.2. Training Object Motion Synthesis

OMSM is trained in two stages. The first stage initializes the model with the parameters of a pre-trained video diffusion model, and trains the model on the full dataset. The second stage fine-tunes the model using the curated dataset without camera motions. During training, we update only the parameters of the denoising U-Net, while fixing the parameters of the VAE encoder and decoder. We train the model via denoising score matching [16].

The two-stage approach helps overcome the domain difference between the video synthesis task of the pretrained model and the object motion synthesis task, allowing for effective learning of object motion synthesis from the small-

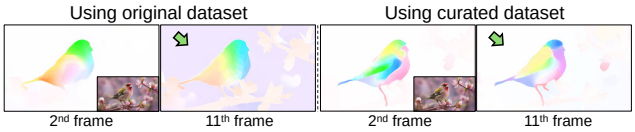

Figure 4. Object flow maps synthesized by OMSM trained on the full dataset (left), and the curated dataset (right). White indicates optical flow vectors with no motion.

scale curated dataset. Fig. 4 shows an example of synthesized motions after the first and second training stages. After the first stage, OMSM is effectively trained to synthesize object motion flow maps, but they exhibit camera motions in the background. After the second stage, the model can successfully synthesize object motion flow maps with minimal camera motions.

4.3. Training Flow-Conditioned Video Synthesis

We initialize FVSM using the parameters of a pretrained video diffusion model [3]. Then, we train only the flow encoder while fixing the other components. Similar to OMSM, we train FVSM via denoising score matching [16]. While the optical flow maps are directly estimated from video datasets in the training time, the camera-object flow maps used in the inference time are synthesized through 3D warping and OMSM. Nevertheless, both optical flow maps contain camera and object motions in the form of flow vectors, enabling the FVSM to effectively produce natural videos with the desired camera motion.

5. Experiments

5.1. Implementation Details

FloVD synthesizes 14 video frames at once, following Stable Video Diffusion [3]. We use a resolution of 320×576 for both video frames and optical flow maps. FVSM is trained for 50K iterations with 16 video clips and their optical flow maps per training batch. OMSM is trained on the full dataset for 100K iterations and then fine-tuned using the curated dataset for 50K iterations, with 8 optical flow maps per training batch. Inspired by T2I-Adapter [20], we use a quadratic timestep sampling strategy (QTS) in training FVSM for better camera controllability (Tab. 4). For stable training and inference of FloVD, we adaptively normalize optical flow maps based on statistics computed from the training dataset, following Li et al. [19]. Refer to the supplemental document for more implementation details.

5.2. Evaluation Protocol

Camera controllability. We employ the evaluation protocol of previous methods [11, 42] for the camera controllability. Specifically, for an input image and camera parameters, we first synthesize a video. We then estimate camera parameters from the synthesized video using GLOMAP [21], and compare the estimated camera parameters against the input parameters to evaluate how faithfully the synthesized video follows the input camera parameters. For the evaluation dataset, we sampled 1,000 video clips and their associated camera parameters from the test set of RealEstate10K [43].

To evaluate estimated camera parameters against input ones, we measure the mean rotation error (mRotErr), mean translation error (mTransErr), and mean error in camera extrinsic matrices (mCamMC), which are defined as:

$$\begin{aligned} \text{mRotErr} &= \frac{1}{T}\sum_{t=1}^{T} \cos^{-1}\frac{\text{tr}(\hat{R}_t R_t^T)-1}{2}, \\ \text{mTransErr} &= \frac{1}{T}\sum_{t=1}^{T} \|\hat{\tau}_t - \tau_t\|, \quad \text{and} \\ \text{mCamMC} &= \frac{1}{T}\sum_{t=1}^{T} \|[\hat{R}_t|\hat{\tau}_t] - [R_t|\tau_t]\|_2, \end{aligned} \tag{3}$$

where T is the number of video frames. $\hat{R}_t$ and $\hat{\tau}_t$ are the camera rotation matrix and translation vector estimated from the t-th synthesized video frame, and R_t are τ_t are their corresponding input rotation matrix and translation vector, respectively.

Video synthesis quality. We evaluate the video synthesis quality in terms of (1) sample quality and (2) object motion synthesis quality. To evaluate the sample quality, we first construct a benchmark dataset using 1,500 real-world videos randomly sampled from the Pexels dataset [1], which we refer to as 'Pexels-random.' To evaluate model's capability of diverse object motion synthesis, we construct three benchmark video datasets with small, medium, and large object motions, each containing 500 video clips with minimal camera motions to avoid potential bias caused by camera motion. The datasets are categorized based on the average magnitudes of the optical flow vectors of moving objects: smaller than 20 pixels (Pexels-small), between 20 and 40 pixels (Pexels-medium), and more than 40 pixels (Pexels-large). More details on the benchmark datasets can be found in the supplemental document.

To evaluate the video synthesis quality, we synthesize videos using the first frames of videos in the aforementioned benchmark datasets and compare these synthesized videos with the datasets. While our method's video synthesis quality is minimally affected by these parameters, previous methods that synthesize video frames directly from them might be more influenced. To account for this, we utilize seven types of camera trajectories during video synthesis: translation to the left, right, up, and down, as well as zoom-in, zoom-out, and no camera motion ('stop'). Consequently, for all models, we generate seven videos for each video included in the benchmark datasets. Finally, we evaluate the video synthesis performance of a given method through the Frechet Video Distance (FVD) [29], Frechet Image Distance (FID) [13], and Inception Score (IS) [24].

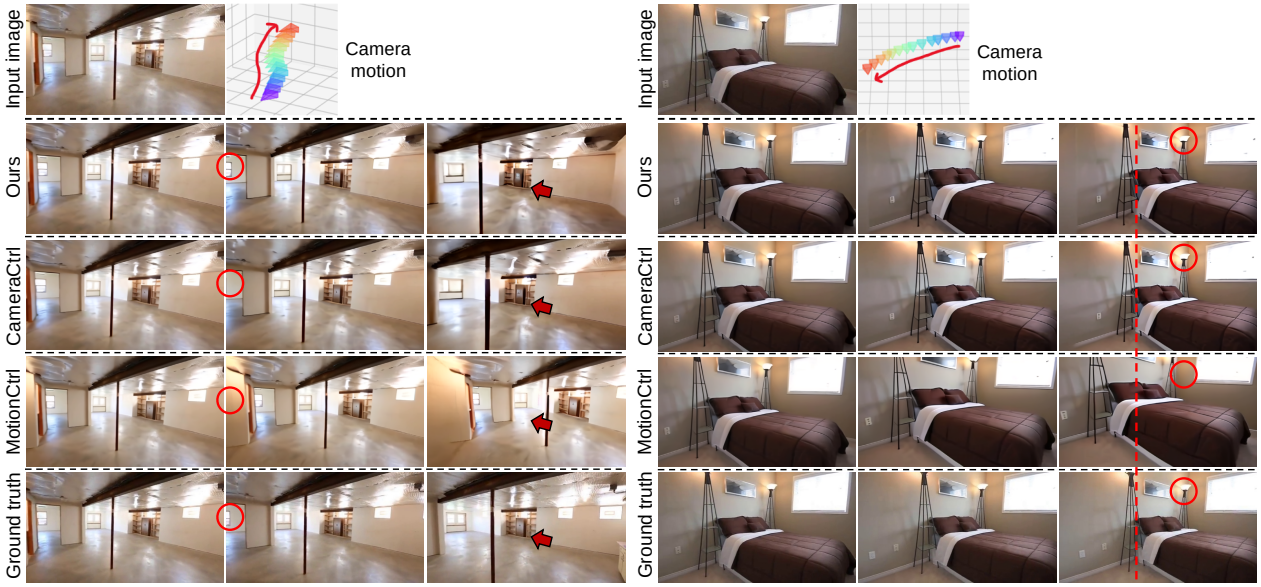

Figure 5. Qualitative comparison of camera control using the RealEstate10K test dataset [43]. MotionCtrl [33] often fails to follow the input camera parameters. Notably, our method shows accurate camera control results despite not using camera parameters in training.

	Training Data	mRotErr (°)	mTransErr	mCamMC
MotionCtrl	RE10K	5.90	0.1610	0.1719
CameraCtrl	RE10K	1.44	0.0927	0.0945
Ours	RE10K	**1.43**	**0.0869**	**0.0887**
Ours	Internal	1.79	0.0994	0.1018
Ours w/ OMSM	RE10K	1.52	0.0971	0.0989
Ours w/ OMSM	Internal	1.88	0.1042	0.1066

Table 1. Quantitative evaluation of camera controllability using the RealEstate10K test dataset [43]. Our method shows superior camera control performance against previous methods [11, 33], even without using camera parameters in training.

5.3. Comparison

We compare our method with recent camera-controllable video synthesis methods, MotionCtrl [33] and CameraCtrl [11], both of which support detailed camera control by taking camera parameters as input. Additional comparisons with other methods that support basic camera movements can be found in the supplemental document.

Camera controllability. We first compare the camera controllability of our method against MotionCtrl [33] and CameraCtrl [11]. Both MotionCtrl and CameraCtrl were trained on RealEstate10K [43], which provides no object motions but a wider range of camera motions than our full dataset. For a comprehensive comparison, we evaluate four versions of our model. Specifically, we train FVSM on either our internal dataset or RealEstate10K, but without utilizing the ground-truth camera parameters available in RealEstate10K. We also include variants of our model with and without OMSM as RealEstate10K contains only static scenes without moving objects. In this evaluation, OMSM is trained using our internal dataset.

As shown in Fig. 5, MotionCtrl [33] produces video frames that do not accurately follow the input camera trajectories due to its suboptimal camera parameter embedding scheme. On the other hand, both CameraCtrl [11] and ours accurately reflect the input camera parameters, and produce video frames that closely resemble the ground-truth frames.

As reported in Tab. 1, our model trained on RealEstate10K [43] outperforms both MotionCtrl and CameraCtrl across all metrics. Moreover, our other models show comparable performances to CameraCtrl, while using the internal dataset and incorporating OMSM slightly increase errors due to domain differences and object motions. These results prove the effectiveness of our camera control scheme based on optical flow.

Video synthesis quality. We also compare the video synthesis quality of our method with previous ones [11, 33]. Fig. 6 shows a qualitative comparison, including X-t slices to visualize pixel value changes over time, computed from the positions marked by the red lines. In this comparison, we synthesize videos using camera parameters without any motion, mainly to compare the video synthesis quality. CameraCtrl [11] produces results with no object motions, as shown in its X-t slices. MotionCtrl [33] produces artifacts with inconsistent foreground and background regions, as marked by blue arrows. These artifacts result from the limited generalization capability, since MotionCtrl updates certain pre-trained parameters of the video diffusion model during training. Unlike these methods, our method produces high-quality videos with natural object motions.

The superior performance of our method is also evidenced by the quantitative comparison in Tab. 2. For the Pexels-random dataset, our method reports better sample quality against the previous methods [11, 33]. These results prove that our method does not harm the video synthesis

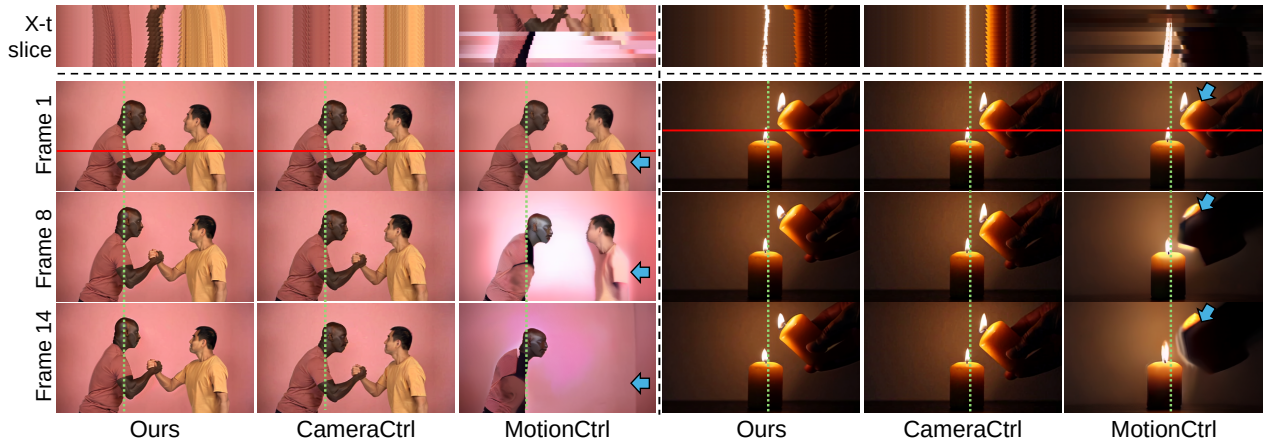

Figure 6. Qualitative comparison of video synthesis quality. Video frames are synthesized with 'stop' camera motion. X-t slice reveals how pixel value changes over time along the horizontal red line. MotionCtrl [33] often fails to follow input camera trajectory and synthesizes video frames with artifacts, due to the lack of generalization capability. CameraCtrl [11] frequently synthesizes motionless object in generated videos. Our method synthesizes video frames with natural object motion while supporting precise camera control.

	Pexels-random			Pexels-small (< 20)			Pexels-med. (< 40)			Pexels-large (≥ 40)		
	FVD	FID	IS	FVD	FID	IS	FVD	FID	IS	FVD	FID	IS
MotionCtrl	93.54	36.06	11.19	235.39	28.94	10.53	214.47	25.76	10.74	**188.89**	25.84	11.71
CameraCtrl	151.06	40.69	11.23	226.40	25.57	10.63	328.71	24.54	10.76	340.36	25.53	11.68
Ours	**91.55**	**35.66**	**11.63**	**220.65**	**22.49**	**11.58**	**183.14**	**20.71**	**11.68**	207.39	**21.12**	**12.95**

Table 2. Quantitative evaluation of video synthesis quality using the Pexels dataset [1]. Our method shows superior video synthesis performance against previous methods [11, 33].

quality of the pre-trained video diffusion model, compared to the previous ones.

Our method also achieves better performances for the benchmark datasets of object motion synthesis quality (Pexels-small, Pexels-medium, and Pexels-large), as reported in Tab. 2. While CameraCtrl exhibits significantly degraded quality for large object motions (Pexels-large), our method achieves substantially better results for all three benchmark datasets. MotionCtrl often fails to follow input camera parameters, synthesizing videos where the viewpoint remains close to the input image. This may lead to good FVD scores, as the synthesized videos align well with the minimal camera movement present in most benchmark videos. However, as shown in Fig. 6, MotionCtrl often produces visual artifacts in the synthesized videos. More visual examples for these artifacts can be found in the additional qualitative comparison of the supplemental document. Furthermore, our method, which employs a timestep sampling strategy from the EDM framework [16], outperforms previous methods across all metrics, as demonstrated in Tab. 3 and Tab. S1 of the supplemental document.

5.4. Further Analysis

Ablation study. We conduct an ablation study to verify the effect of our main components: OMSM, and training with a wide range of real-world videos, both of which are made possible by our optical-flow-based framework. We utilize our models trained with two different timestep sampling strategies for in-depth analysis under different settings. In Fig. 7, the baseline model indicates a variant of our model that has no OMSM (i.e., only FVSM) and is trained on RealEstate10K [43]. '+ OMSM' indicates a variant model with OMSM, while its FVSM is still trained on RealEstate10K. OMSM is trained using our full and curated datasets. '+ large-scale data' is our final model where both OMSM and FVSM are trained using our datasets.

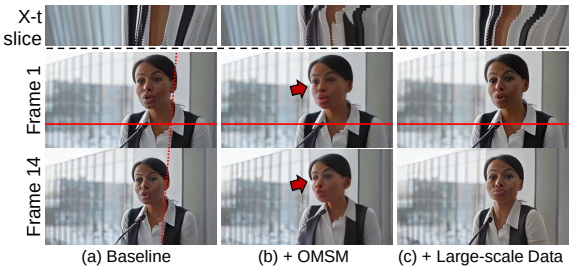

Figure 7. Qualitative ablation with 'zoom-out' camera motion. X-t slice reveals pixel value changes along the horizontal red line.

	Training Data	Timestep Strategy	Pexels-random FVD (↓)	FID (↓)	IS (↑)
Baseline	RE10K	QTS	157.99	39.61	11.19
+ OMSM	RE10K	QTS	104.92	36.33	11.51
+ large-scale data	Internal	QTS	**91.55**	**35.66**	**11.63**
Baseline	RE10K	EDM	148.42	39.92	11.23
+ OMSM	RE10K	EDM	95.31	36.90	11.43
+ large-scale data	Internal	EDM	**80.74**	**35.65**	**11.73**

Table 3. Ablation study of our main components with the evaluation of video synthesis quality using the Pexels-random dataset.

As shown in Fig. 7(a), the baseline model does not synthesize noticeable object motions. Introducing OMSM enables our framework to generate object motion, but it also occasionally produces artifacts for moving objects as shown in Fig. 7(b). Our final model produces natural-looking object motions without noticeable artifacts (Fig. 7(c)). Tab. 3 also shows similar trends that introducing each component to our framework consistently improves evaluation metrics

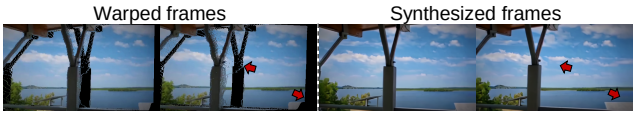

Figure 8. Explicit camera control. Our model can follow the warped frames while handling artifacts such as holes, which are caused by imperfect 3D structure estimation.

	mRotErr (°)	mTransErr	mCamMC
Ours w/ EDM	1.70	0.0983	0.1010
Ours w/ QTS	1.43	0.0869	0.0887

Table 4. Timestep sampling strategies for camera controllability.

for Pexels-random. Additional quantitative ablation study can be found in Tab. S1 of the supplemental document.

Flow-conditioned video synthesis. Our method generates camera flow maps using the 3D structure estimated from an input image, then feeds them to FVSM. To better understand our framework, Fig. 8 presents visualizations of warped images using the estimated 3D structure and camera parameters, alongside their associated video synthesis results produced by FVSM. As shown in the figure, 3D-based image warping may introduce distortions and holes, yet still provides realistic-looking images. This result indicates that leveraging the 3D structure can serve as a powerful hint for camera-controllable video synthesis. Fig. 8 also shows that our flow-conditioned video synthesis successfully produces realistic-looking results that closely resemble the warping results, but without artifacts such as distortions and holes.

Timestep sampling strategy. As stated in Sec. 5.1, we adopt the quadratic timestep sampling strategy (QTS) for better camera controllability, instead of the timestep sampling strategy of the EDM framework [16] used in Stable Video Diffusion [3]. Our model using QTS shows slightly compromised video synthesis quality compared to our model using EDM (Tab. 3). Nevertheless, our model with QTS still demonstrates better performance than the previous methods (Tab. 2). Moreover, QTS enhances the camera controllability of FVSM (Tab. 4). This improved camera controllability results from increased chances of training with highly noised data, allowing our model to effectively leverage flow conditions for structural content generation.

5.5. Applications

Temporally-consistent video editing. Our framework using FVSM enables temporally-consistent video editing at no extra cost. A video can be edited as follows. First, we obtain optical flow maps from the video by using an off-the-shelf optical flow estimator [28]. We then edit the first frame of the input video using an off-the-shelf image editing tool, e.g., InfEdit [36]. We synthesize a video by using FVSM with the edited first frame and estimated optical flow maps as inputs. As shown in Fig. 9, this simple strategy using our framework can produce temporally-consistent video editing results, thanks to our optical-flow-based approach.

Figure 9. Temporally-consistent video editing.

Figure 10. Video synthesis results with the dolly zoom effect.

Cinematic camera control. Thanks to its 3D-awareness, our framework enables complex camera controls such as the dolly zoom. The dolly zoom is a camera technique where the camera moves forward while simultaneously adjusting the zoom in the opposite direction. To achieve this, we synthesize videos with control over both the camera's intrinsic and extrinsic parameters. Fig. 10 shows a synthesized video with the dolly zoom effect, where the subject remains a similar size while the background appears to converge inward. Notably, our framework achieves this without being trained on video datasets containing camera intrinsic parameters that vary across frames.

6. Conclusion

This paper proposed FloVD, a novel optical-flow-based video diffusion model for camera controllable video generation. Since existing methods require a training dataset with ground-truth camera parameters, they are mainly trained on the restricted datasets that primarily consist of static scenes, leading to video synthesis with unnatural object motion. Unlike previous methods, our method leverages optical flow maps to represent both camera and object motions, enabling the use of arbitrary training videos without ground-truth camera parameters. Moreover, our method facilitates detailed camera control by leveraging background motions of optical flow, which encodes 3D correlation across different viewpoints. Our extensive experiments demonstrate that FloVD provides realistic video synthesis with natural object motion and accurate camera control.

Limitations. Our method is not free from limitations. Errors from both the object motion synthesis model and the semantic segmentation model may result in unnatural object motion in the synthesized videos. The estimation error of segmentation model can be alleviated through user interaction by providing point prompts for object regions. Our future work will involve a seamless integration of camera and object motions to synthesize more natural videos.

Ethical considerations. FloVD is purely a research project. Currently, we have no plans to incorporate FloVD into a product or expand access to the public. We will also put Microsoft AI principles into practice when further developing the models. In our research paper, we account for the ethical concerns associated with video generation research. To mitigate issues associated with training data, we have implemented a rigorous filtering process to purge our training data of inappropriate content, such as explicit imagery and offensive language, to minimize the likelihood of generating inappropriate content.

References

[1] Pexels, royalty-free stock footage website. https://www.pexels.com. Accessed: 2024-09-30. 4, 5, 7

[2] Sherwin Bahmani, Ivan Skorokhodov, Aliaksandr Siarohin, Willi Menapace, Guocheng Qian, Michael Vasilkovsky, Hsin-Ying Lee, Chaoyang Wang, Jiaxu Zou, Andrea Tagliasacchi, et al. Vd3d: Taming large video diffusion transformers for 3d camera control. *arXiv preprint arXiv:2407.12781*, 2024. 1, 2

[3] 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 preprint arXiv:2311.15127*, 2023. 1, 3, 4, 5, 8

[4] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *CVPR*, pages 22563–22575, 2023.

[5] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. 1

[6] Tsai-Shien Chen, Chieh Hubert Lin, Hung-Yu Tseng, Tsung-Yi Lin, and Ming-Hsuan Yang. Motion-conditioned diffusion model for controllable video synthesis. *arXiv preprint arXiv:2304.14404*, 2023. 1, 2

[7] Michael Dorkenwald, Timo Milbich, Andreas Blattmann, Robin Rombach, Konstantinos G Derpanis, and Bjorn Ommer. Stochastic image-to-video synthesis using cinns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3742–3753, 2021. 2

[8] Yuki Endo, Yoshihiro Kanamori, and Shigeru Kuriyama. Animating landscape: self-supervised learning of decoupled motion and appearance for single-image video synthesis. *arXiv preprint arXiv:1910.07192*, 2019. 2

[9] Yuwei Guo, Ceyuan Yang, Anyi Rao, Zhengyang Liang, Yaohui Wang, Yu Qiao, Maneesh Agrawala, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *arXiv preprint arXiv:2307.04725*, 2023. 2

[10] Zekun Hao, Xun Huang, and Serge Belongie. Controllable video generation with sparse trajectories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7854–7863, 2018. 2

[11] Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for text-to-video generation. *arXiv preprint arXiv:2404.02101*, 2024. 1, 2, 5, 6, 7

[12] Yingqing He, Tianyu Yang, Yong Zhang, Ying Shan, and Qifeng Chen. Latent video diffusion models for high-fidelity long video generation. *arXiv preprint arXiv:2211.13221*, 2022. 1

[13] 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. *Advances in neural information processing systems*, 30, 2017. 5

[14] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646, 2022. 1

[15] Aleksander Holynski, Brian L Curless, Steven M Seitz, and Richard Szeliski. Animating pictures with eulerian motion fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5810–5819, 2021. 2

[16] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022. 4, 5, 7, 8

[17] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9492–9502, 2024. 3

[18] Zhengfei Kuang, Shengqu Cai, Hao He, Yinghao Xu, Hongsheng Li, Leonidas Guibas, and Gordon Wetzstein. Collaborative video diffusion: Consistent multi-video generation with camera control. *arXiv preprint arXiv:2405.17414*, 2024. 1

[19] Zhengqi Li, Richard Tucker, Noah Snavely, and Aleksander Holynski. Generative image dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24142–24153, 2024. 2, 5

[20] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, and Ying Shan. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4296–4304, 2024. 4, 5

[21] Linfei Pan, Dániel Baráth, Marc Pollefeys, and Johannes L Schönberger. Global structure-from-motion revisited. In *European Conference on Computer Vision (ECCV)*, 2024. 5

[22] Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, et al. Movie gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024. 1, 2

[23] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman

Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 3, 4

[24] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016. 5

[25] Xiaoyu Shi, Zhaoyang Huang, Fu-Yun Wang, Weikang Bian, Dasong Li, Yi Zhang, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, et al. Motion-i2v: Consistent and controllable image-to-video generation with explicit motion modeling. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 1, 2

[26] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022. 1

[27] Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. *Advances in Neural Information Processing Systems*, 34: 19313–19325, 2021. 1, 2

[28] Zachary Teed and Jia Deng. 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 16*, pages 402–419. Springer, 2020. 2, 4, 8

[29] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018. 5

[30] Vikram Voleti, Alexia Jolicoeur-Martineau, and Chris Pal. Mcvd-masked conditional video diffusion for prediction, generation, and interpolation. *Advances in neural information processing systems*, 35:23371–23385, 2022. 2

[31] Yanhui Wang, Jianmin Bao, Wenming Weng, Ruoyu Feng, Dacheng Yin, Tao Yang, Jingxu Zhang, Qi Dai, Zhiyuan Zhao, Chunyu Wang, et al. Microcinema: A divide-and-conquer approach for text-to-video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8414–8424, 2024. 1

[32] Zhenzhi Wang, Yixuan Li, Yanhong Zeng, Youqing Fang, Yuwei Guo, Wenran Liu, Jing Tan, Kai Chen, Tianfan Xue, Bo Dai, et al. Humanvid: Demystifying training data for camera-controllable human image animation. *arXiv preprint arXiv:2407.17438*, 2024. 1

[33] 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 *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 1, 2, 6, 7

[34] Dejia Xu, Yifan Jiang, Chen Huang, Liangchen Song, Thorsten Gernoth, Liangliang Cao, Zhangyang Wang, and Hao Tang. Cavia: Camera-controllable multi-view video diffusion with view-integrated attention. *arXiv preprint arXiv:2410.10774*, 2024.

[35] Dejia Xu, Weili Nie, Chao Liu, Sifei Liu, Jan Kautz, Zhangyang Wang, and Arash Vahdat. Camco: Camera-controllable 3d-consistent image-to-video generation. *arXiv preprint arXiv:2406.02509*, 2024. 1, 2

[36] Sihan Xu, Yidong Huang, Jiayi Pan, Ziqiao Ma, and Joyce Chai. Inversion-free image editing with natural language. *arXiv preprint arXiv:2312.04965*, 2023. 8

[37] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv preprint arXiv:2406.09414*, 2024. 3

[38] Shiyuan Yang, Liang Hou, Haibin Huang, Chongyang Ma, Pengfei Wan, Di Zhang, Xiaodong Chen, and Jing Liao. Direct-a-video: Customized video generation with user-directed camera movement and object motion. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–12, 2024. 1, 2

[39] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. 1

[40] Shengming Yin, Chenfei Wu, Jian Liang, Jie Shi, Houqiang Li, Gong Ming, and Nan Duan. Dragnuwa: Fine-grained control in video generation by integrating text, image, and trajectory. *arXiv preprint arXiv:2308.08089*, 2023. 1, 2

[41] David Junhao Zhang, Roni Paiss, Shiran Zada, Nikhil Karnad, David E Jacobs, Yael Pritch, Inbar Mosseri, Mike Zheng Shou, Neal Wadhwa, and Nataniel Ruiz. Recapture: Generative video camera controls for user-provided videos using masked video fine-tuning. *arXiv preprint arXiv:2411.05003*, 2024. 1

[42] Guangcong Zheng, Teng Li, Rui Jiang, Yehao Lu, Tao Wu, and Xi Li. Cami2v: Camera-controlled image-to-video diffusion model. *arXiv preprint arXiv:2410.15957*, 2024. 1, 2, 5

[43] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018. 1, 2, 5, 6, 7