LIGHTMOTION: A LIGHT AND TUNING-FREE METHOD FOR SIMULATING CAMERA MOTION IN VIDEO GENERATION

Anonymous authors
Paper under double-blind review

A LLM USAGE STATEMENT

A large language model (LLM) was used for language polishing of the manuscript. In addition, the LLM was employed to assist in evaluating the generation quality, temporal coherence, and camera controllability of the generated videos, as reported in Table 2 of the main paper.

B DETAILED EXPERIMENTAL SETTINGS

B.1 CAMERA PARAMETER DEFINITIONS

Unlike traditional camera-controlled video generation models, our LightMotion eliminates the need for users to input technical camera parameters such as intrinsic, rotation, or translation matrices. Instead, we streamline the input parameters without requiring knowledge of camera geometry, thus lowering the barrier for non-professional users. Specifically, We only require users to input four camera parameters: x, y, z, and θ. By combining these parameters, we can simulate various camera motions in the real world.

Following previous work, Direct-A-Video Yang et al. (2024), we define the parameters as follows: x represents the X-pan ratio, defined as the total horizontal shift of the frame center from the first to the last frame related to the frame width, with $x > 0$ indicating the panning rightward. y denotes the Y-pan ratio, which indicates the total vertical shift of the frame center related to the frame height, with $y > 0$ indicating the panning downward. z refers to the Z-pan zooming ratio, defined as the scaling factor between the first and last frame, with $z > 0$ indicating zooming-in.

Different from Direct-A-Video, we additionally model the camera rotation and define relative parameters. We model rotation using point cloud projection theory, which primarily involves camera intrinsic parameters K, rotation matrices R^i, and depth information $d(u, v, 1)$. Here, we ignore $d(u, v, 1)$ (which will be discussed in the following section) and only consider the settings of K and R^i (rotation about the Y-Axis, the same applies to other cases):

$$K = \begin{pmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{pmatrix}, R^i = \begin{pmatrix} \cos\gamma^i & 0 & \sin\gamma^i \\ 0 & 1 & 0 \\ -\sin\gamma^i & 0 & \cos\gamma^i \end{pmatrix}. \tag{1}$$

Similar to pixel space, the camera's optical center (u_0, v_0) are positioned at the center of the latent space, $(\frac{h}{2}, \frac{w}{2})$. However, the focal lengths (f_x, f_y) in the latent space do not have physical significance. Through extensive experimentation, we found that $f_x = f_y = 15$ yields effective results in the latent space. Regarding the rotation matrix R^i for the i-th frame, relative angles γ^i are defined as $\frac{2\cdot\theta}{N} \cdot (i - N)$, with γ^i ranging from $-\theta$ to θ across N frames. Here, θ is the user-defined rotation parameter, and $\theta > 0$ indicates counterclockwise rotation.

B.2 CAMERA PARAMETER SETTINGS

Since not all methods support every type of camera motion, we define 16 distinct camera motion scenarios, including 8 panning, 4 zooming, and 4 rotation sequences, to assess the performance of each model on the respective motion types. Following, we will provide a detailed description of the parameter settings for these camera motions.

For panning, we define 4 motion types, including leftward, rightward, upward, and downward movements, each with two variations: small-scale and large-scale. Small-scale panning shifts the frame from first to last, covering 25% of the frame width, while large-scale panning covers 50%. Parameter settings are detailed in Table 1.

Camera Motion	Parameter Settings
Leftward (small-scale)	$x = -0.25, y = 0.00$
Leftward (large-scale)	$x = -0.50, y = 0.00$
Rightward (small-scale)	$x = 0.25, y = 0.00$
Rightward (large-scale)	$x = 0.50, y = 0.00$
Upward (small-scale)	$x = 0.00, y = -0.25$
Upward (large-scale)	$x = 0.00, y = -0.50$
Downward (small-scale)	$x = 0.00, y = 0.25$
Downward (large-scale)	$x = 0.00, y = 0.50$

Table 1: Camera panning parameter settings.

For zooming, we define two motion types: zooming-in and zooming-out, each having small-scale and large-scale variations. Small-scale zooming scales the frame from first to last, covering 24% of the frame size, while large-scale spanning 48%. Parameter settings are detailed in Table 2.

Camera Motion	Parameter Settings
Zooming-in (small-scale)	$z = 0.24$
Zooming-in (large-scale)	$z = 0.48$
Zooming-out (small-scale)	$z = -0.24$
Zooming-out (large-scale)	$z = -0.48$

Table 2: Camera zooming parameter settings.

For rotation, we also define two motion types: counterclockwise rotation and clockwise rotation, each having small-scale and large-scale variations. Small-scale rotation rotates the frame from first to last, ranging from $-\theta$ to θ where $\theta = 8$, while large-scale rotation uses $\theta = 16$. Parameter settings are detailed in Table 3.

Camera Motion	Parameter Settings
CCW. rotation (small-scale)	$\theta = 8$
CCW. rotation (large-scale)	$\theta = 16$
CW. rotation (small-scale)	$\theta = -8$
CW. rotation (large-scale)	$\theta = -16$

Table 3: Camera rotation parameter settings. "CCW." represents the counterclockwise while "CW." renotes the clockwise.

C Related Proofs

Theorem 1. Rotation of a 3D point cloud along the x, y, or z axis is inherently independent of depth.

Proof. Let $(u, v, 1)^T$ be the pixel coordinates in the original latent space, K the camera intrinsic matrix, and $(X_c, Y_c, d(u, v, 1))^T$ the spatial coordinates after point cloud projection, with $d(u, v, 1)$ representing the depth. Through the pin-hole camera model, we have:

$$d(u,v,1)\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K \cdot \begin{pmatrix} X_c \\ Y_c \\ d(u,v,1) \end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} X_c \\ Y_c \\ d(u,v,1) \end{pmatrix}, \tag{2}$$

where $f_x$and f_y are the focal lengths, and c_x and c_y are the coordinates of the camera's optical center.

Rearranging the above equations, we obtain:

$$\begin{cases} u = f_x \cdot \frac{X_c}{d(u,v,1)} + c_x \\ v = f_y \cdot \frac{Y_c}{d(u,v,1)} + c_y \end{cases} \Rightarrow \begin{cases} X_c = \frac{(u-c_x)}{f_x} \cdot d(u,v,1) \\ Y_c = \frac{(v-c_y}{f_y} \cdot d(u,v,1) \end{cases} . \tag{3}$$

According to the point cloud projection theory, by rotating the point cloud to another perspective using the rotation matrix R_y (taking the rotation around the Y-axis as an example, with the same principle applying to rotations around other axes), we obtain the following equation:

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = R_y \cdot \begin{pmatrix} X_c \\ Y_c \\ d(u,v,1) \end{pmatrix} = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \cdot \begin{pmatrix} X_c \\ Y_c \\ d(u,v,1) \end{pmatrix}. \tag{4}$$

Substituting Eq. (3) and simplifying, we have:

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = \begin{pmatrix} \cos\theta \cdot X_c + \sin\theta \cdot d(u,v,1) \\ Y_c \\ -\sin\theta \cdot X_c + \cos\theta \cdot d(u,v,1) \end{pmatrix} = \begin{pmatrix} \frac{\cos\theta \cdot d(u,v,1) \cdot (u-c_x)}{f_x} + \sin\theta \cdot d(u,v,1) \\ \frac{(v-c_y)}{f_y} \cdot d(u,v,1) \\ \frac{-\sin\theta \cdot d(u,v,1) \cdot (v-(y)}{f_y} + \cos\theta \cdot d(u,v,1) \end{pmatrix}. \tag{5}$$

Then, we can derive the following ratio relationship:

$$\begin{cases} \frac{X'}{Z'} &= \frac{\cos\theta \cdot f_y \cdot (u-c_x) + \sin\theta \cdot f_x \cdot f_y}{-\sin\theta \cdot f_x \cdot (v-c_y) + \cos\theta \cdot f_x \cdot f_y} \\ \frac{Y'}{Z'} &= \frac{(v-c_y)}{-\sin\theta \cdot (v-c_y) + \cos\theta \cdot f_y} \end{cases} . \tag{6}$$

On the other hand, the point cloud in the new perspective can be mapped to the new pixel coordinates $(u', v', 1)$ as in Eq. (2), satisfying the following relationship:

$$Z' \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = K \cdot \begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = \begin{pmatrix} f_x & 0 & C_x \\ 0 & f_y & C_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix}. \tag{7}$$

Substituting Eq. (6) and simplifying, we obtain:

$$\begin{cases} u' = f_x \cdot \frac{X'}{Z'} + c_x = \frac{\cos\theta \cdot f_y (u-c_x) + \sin\theta f_x \cdot f_y}{-\sin\theta \cdot (v-c_y) + \cos\theta \cdot f_y} + c_x \\ v' = f_y \cdot \frac{Y'}{Z'} + c_y = \frac{f_y \cdot (v-c_y)}{-\sin\theta (v-c_y) + \cos\theta \cdot f_y} + c_y \end{cases} . \tag{8}$$

The results show that projected pixel coordinates are independent of depth information $d(u, v, 1)$.

Proof End.

Theorem 2. By considering panning along the x and y axes, zooming along the z axis, and rotations around the x, y, and z axes, these six basic motions can approximate nearly all general camera movements in real-world scenarios.

Proof. Given an any camera pose $[R \mid T]$, we construct the rotation matrix R and translation matrix T in sequence.

(i) Rotation matrix. According to the Euler angle formulation, R can be decomposed into successive rotations along the x, y, and z axes. This process can be formulated as:

$$R = R_x(\alpha) \cdot R_y(\beta) \cdot R_z(\gamma), \tag{9}$$

where α, β, and γ are the Euler angles, which can be efficiently computed using an Euler angle solver. The matrices $R_x(\alpha)$, $R_y(\beta)$, and $R_z(\gamma)$ are identical to those used in our point cloud formulation.

(i) Translation matrix. Due to the linear additivity of translation, T can be decomposed into independent translations along the x, y, and z axes. This can be expressed as:

$$T = T_x(x) + T_y(y) + T_z(z), \tag{10}$$

where $T_x(x)$ and $T_y(y)$ can be covered by our panning mapping function $\mathcal{F}_{paning}$ along the x and y axes, respectively, while $T_z(z)$ can be covered by our zooming mapping function $\mathcal{F}_{zooming}$ along the z axis.

Proof End.

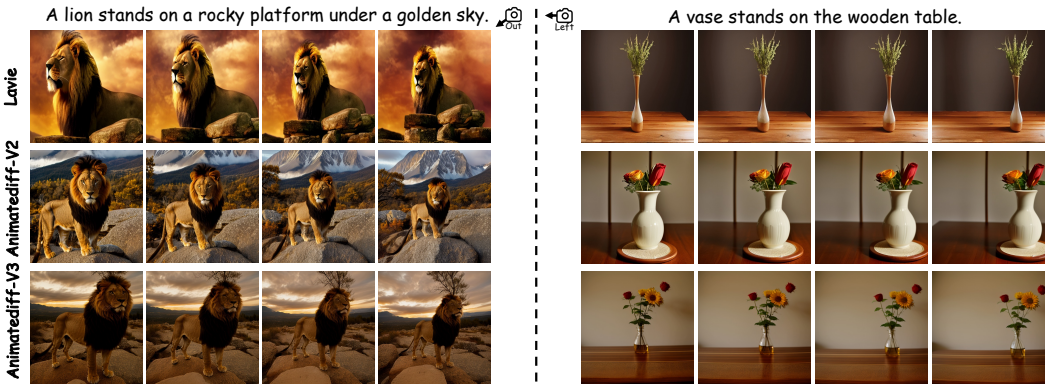

Figure 1: Our LightMotion can be seamlessly integrated into most existing frameworks

Figure 2: Our LightMotion can be integrated into ZeroScope with a resolution of 320×576.

D Compatibility with Different Framework

Besides Animatediff-V2 Guo et al. (2024), our method can also be integrated into other video generation frameworks such as Animatediff-V3 and LaVie Wang et al. (2024) at a resolution of 512×512, and into ZeroScope Sterling (2023) with a different resolution of 320×576. Additional qualitative results are presented in Figure 1 and Figure 2.

E Compatibility with Non-Linear Movements

In addition to the linear motions demonstrated in the main text, LightMotion also supports non-linear movements across frames by adjusting the coordinate mapping functions, as illustrated in Figure 3.

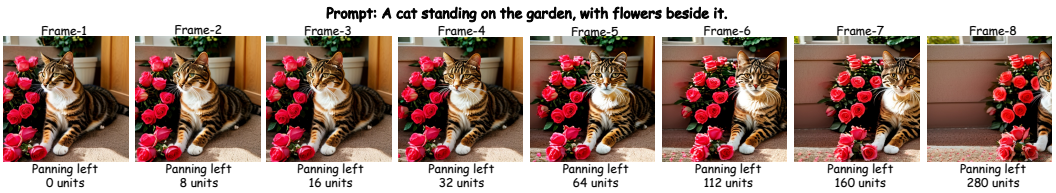

Figure 3: LightMotion supports non-linear movements by adjusting coordinate mappings.

F Additional Examples with Object Repetition

Building on the cases presented in the main paper, Figure 4 provides further examples that highlight the widespread nature of object repetition and inter-frame inconsistency.

G Various Camera Combinations

Furthermore, our LightMotion supports a wide variety of camera combinations, with additional visual results provided in Figure 5 and Figure 6, demonstrating its versatility.

Figure 4: Additional examples of object repetition and inter-frame inconsistency.

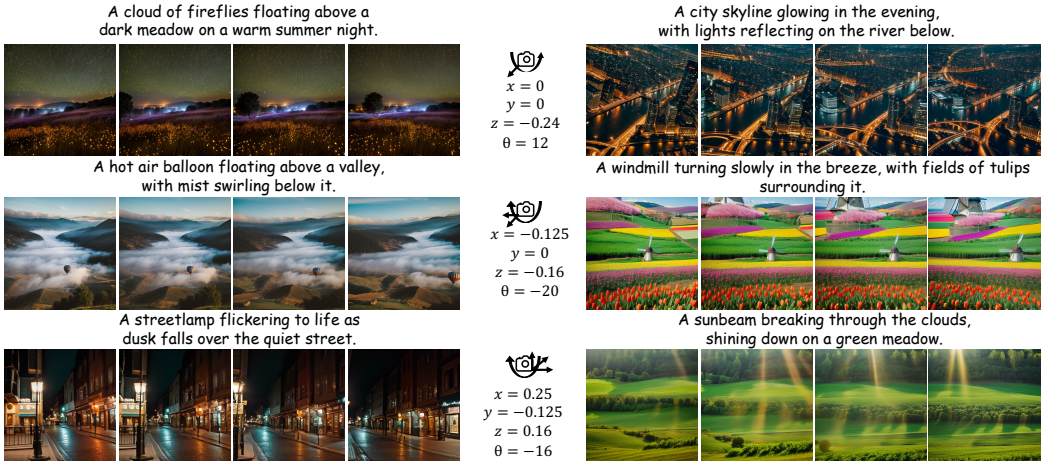

Figure 5: Additional visualization with camera motion using user-defined parameter combinations.

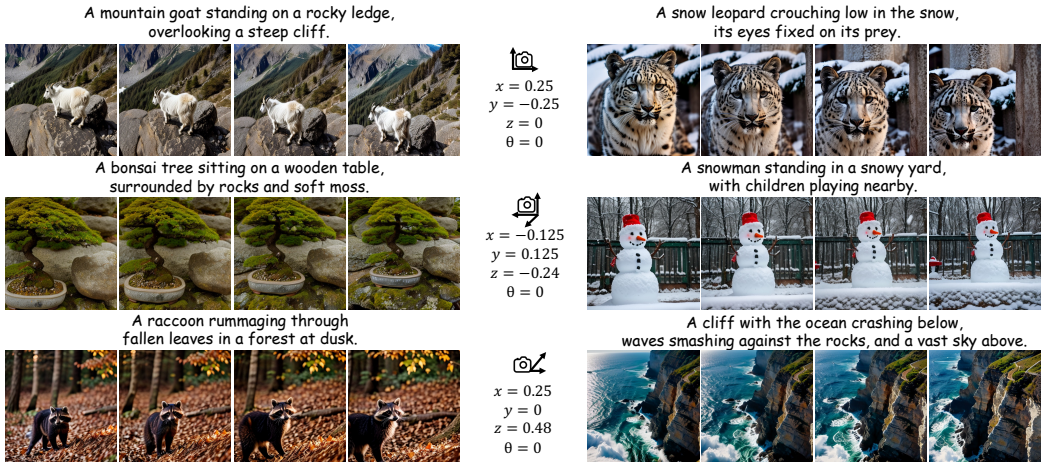

Figure 6: Additional visualization with camera motion using user-defined parameter combinations.

H Additional Qualitative Comparison

To highlight the superiority of LightMotion, we provide additional qualitative comparisons in Figure 7 and Figure 8, showing its performance across various camera motions.

I User Study

We further assess the preferences of users for various methods through a user study. Specifically, we design a questionnaire that includes 10 sets of videos generated by different methods. These

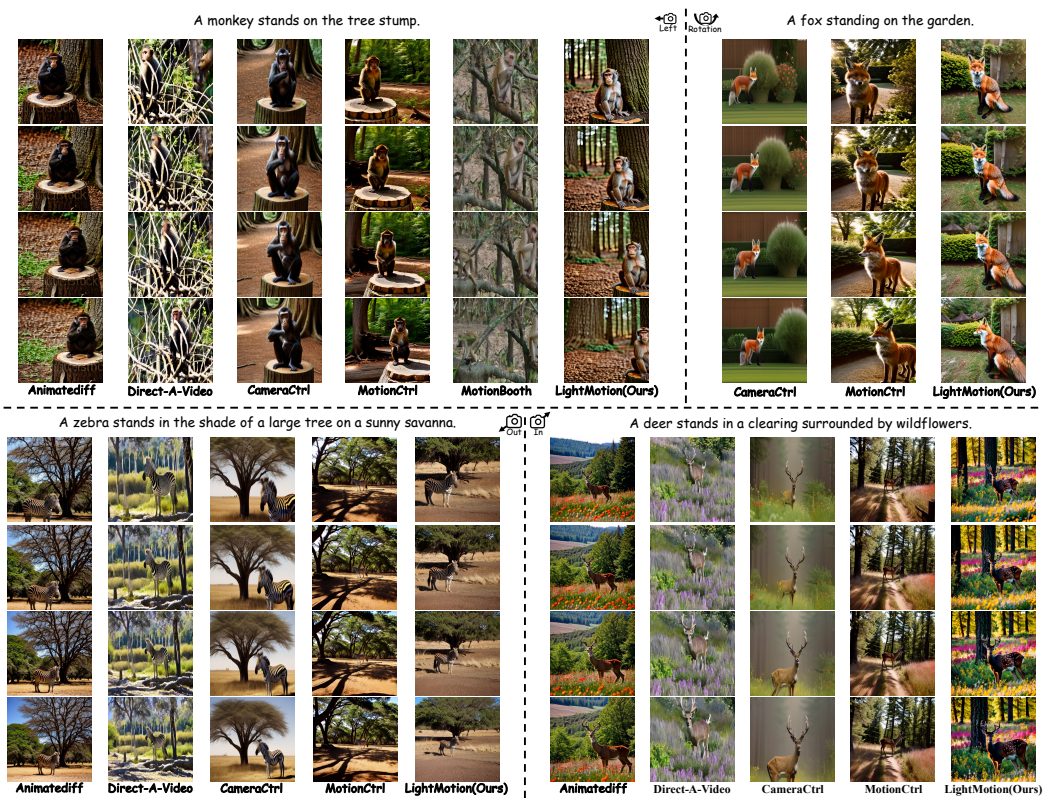

Figure 7: More qualitative comparisons with existing methods.

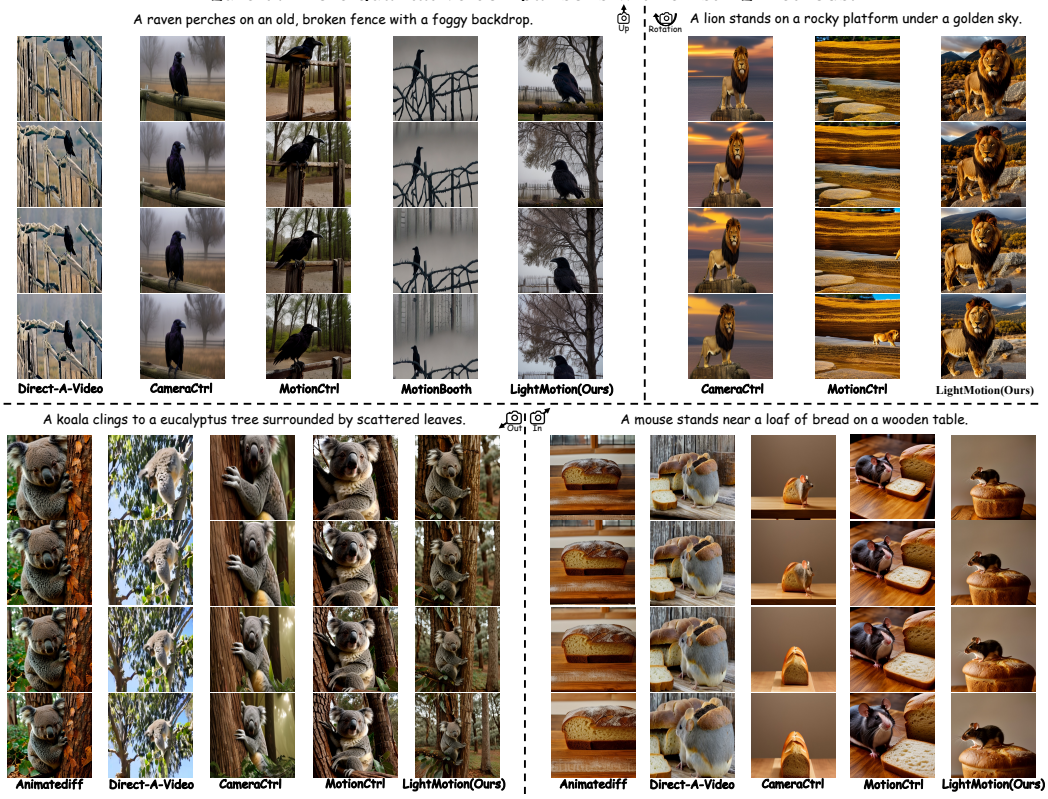

Figure 8: More qualitative comparisons with existing methods.

sets include two groups featuring camera panning, four groups focusing on camera zooming, and four groups highlighting camera rotation. Additionally, each generated video is accompanied by a relevant text description and the corresponding camera motion. In each set, the results from all methods are transformed into .gif files and presented on same page. We establish clear evaluation criteria for users, who score the videos on a scale of up to 100 in each set based on the following two aspects: (i) *Generation Quality*: This criterion evaluates the similarity between the generated video and its text description, as well as the aesthetic quality. (ii) *Camera Controllability*: This criterion assesses the alignment between the camera movements in the generated video and the specified camera motions. To ensure fairness, the names of all methods will be concealed, and the order of the generated results in each set will be randomized. Finally, 100 valid questionnaires were included in the analysis to evaluate user preferences.

J GPT-4o Evaluation

Additionally, the generated video samples for the user study will also be re-evaluated by GPT-4o. Similarly, we establish clear evaluation criteria for GPT-4o, which scores the videos on a scale of up to 100 in each set based on the following three aspects: (i) *Generation Quality*: This criterion evaluates the similarity between the generated video and its text description, as well as the aesthetic quality. (ii) *Coherence*: This criterion evaluates the semantic coherence between frames in the generated video. (iii) *Camera Controllability*: This criterion assesses the alignment between the camera movements in the generated video and the specified camera motions. To ensure robustness, we perform five repetitions and average the scores for each method.

References

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. In *ICLR*, 2024.

Spencer Sterling. Zeroscope, 2023.

Yaohui Wang, Xinyuan Chen, Xin Ma, Shangchen Zhou, Ziqi Huang, Yi Wang, Ceyuan Yang, Yinan He, Jiashuo Yu, Peiqing Yang, et al. Lavie: High-quality video generation with cascaded latent diffusion models. *IJCV*, 2024.

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 *SIGGRAPH*, 2024.