Appendix

The appendix provides detailed supplementary material on the **DataDoP** dataset and **GenDoP** method. It outlines data availability, ensuring compliance with YouTube's policies and detailing our data sharing practices. The dataset construction process is described in detail, including shot collection, quality filtering, and semantic categorization using GPT-4o [2]. Additionally, we provide further dataset statistics. The appendix also explains the tokenization details of camera trajectory data for model processing. Finally, it includes information on the experimental setup, along with additional ablation studies

A. DataDoP Dataset

A.1. Data Availability Statement and Clarification

We are dedicated to upholding transparency and compliance in our data collection and sharing practices. Please take note of the following:

- **Publicly Available Data:** The data utilized in our studies is sourced from publicly available repositories. We do not access any exclusive or private data sources.
- **Data Sharing Policy:** Our data sharing policy is in line with established practices from previous works, such as [1]. Instead of providing raw data, we furnish YouTube video IDs essential for accessing the content.
- **Usage Rights:** The data we release is exclusively meant for research purposes. Any commercial use is not permitted under this agreement.
- **Compliance with YouTube Policies:** Our data collection and sharing practices strictly adhere to YouTube's data privacy and fair use policies. We ensure that user data and privacy rights are respected throughout the process.
- **Data License:** The data is distributed under Creative Commons Attribution 4.0 International License (CC BY 4.0).

Furthermore, the DataDoP dataset is provided solely for informational purposes. The copyright for the original video content remains with the respective owners. All DataDoP videos are sourced from the internet and are not owned by our institution. We disclaim responsibility for the content and interpretation of these videos. In relation to the future open-source version, researchers must agree not to reproduce, duplicate, sell, trade, resell, or exploit any portion of the videos or derived data for commercial purposes, and refrain from copying, publishing, or distributing any part of the DataDoP dataset.

A.2. Construction Details

Pre-processing. Our dataset construction involves a multi-stage curation process:

- **Shot Collection**: A curated collection of cinematographically significant films and documentaries forms the foundation of our dataset. Using PySceneDetect[1], we extract 43k initial shots through content-aware boundary detection. An optimized VSR pipeline[2] is employed to eliminate textual overlays while maintaining visual integrity. To enhance processing speed and reduce misclassification, we focus the check area on the lower 1/5 of the frame. Finally, we merge this dataset with a publicly available subset of MovieShots [4] to further diversify stylistic elements.
- **Quality Filtering**: We retained only shots with a duration between 10 and 20 seconds. Statistics can be seen in Fig. R1a. Then we exclude sequences with static frames and low-light conditions. For each shot, we calculate the pixel-wise similarity between all pairs of consecutive frames. The similarity between two frames F_1 and F_2 is defined as:

$$S(F_1, F_2) = \frac{\sum_{i,j} \mathbb{I}(F_1(i,j) = F_2(i,j))}{H \times W},$$

where $F_1(i,j)$ and $F_2(i,j)$ represent the pixel values at position (i,j) in frames F_1 and F_2, respectively, and H and W are the height and width of the frames. To identify static frames, we compute the average similarity $\overline{S}$ between all consecutive frame pairs in the shot. Specifically, for a shot with N frames, the average similarity is calculated as:

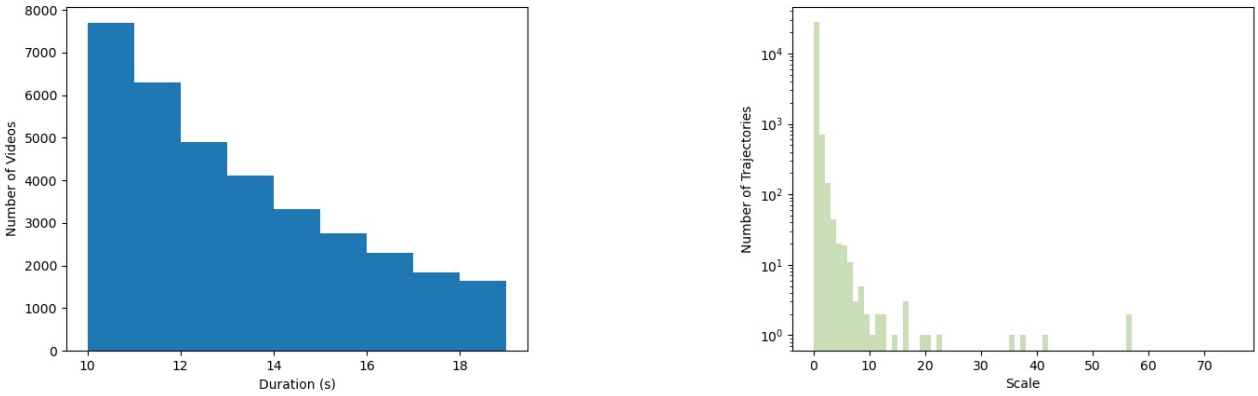
$$\overline{S} = \frac{1}{N-1} \sum_{k=1}^{N-1} S(F_k, F_{k+1}),$$

where F_k and F_{k+1} are consecutive frames in the shot, and $N-1$ is the number of consecutive frame pairs. If the average similarity $\overline{S}$ exceeds a threshold (e.g., $\overline{S} > 0.6$), the entire shot is considered static and excluded.
For each shot, the average brightness (mean gray value) for all frames is computed using the following formula:

$$\overline{B} = \frac{1}{N \times H \times W} \sum_{k=1}^{N} \sum_{i=1}^{H} \sum_{j=1}^{W} F_k(i,j),$$

where N is the total number of frames in the shot, H and W are the height and width of each frame, and $F_k(i,j)$ represents the pixel value at position (i,j) in frame k.
If the average brightness of a shot is below a predefined threshold (e.g., $\overline{B} < 15$), the shot is classified as too dark and excluded from the dataset.
- **Semantic Filtering**: We developed an automated categorization pipeline using GPT-4o. Following the definitions in Sec. 3.2, shots are classified into categories. Leveraging GPT-4o [2], we automate the categorization of shots into *Static*, *Free-Moving*, and *Tracking*. Shots

[1] https://github.com/Breakthrough/PySceneDetect
[2] https://github.com/YaoFANGUK/video-subtitle-remover



(a) Shot Length.

(b) Trajectory Scale.

Figure R1. **Dataset Statistics** in terms of video shot length and trajectory scale.

classified as *Object/Scene-Centric*, which are common in multi-view datasets, are not considered in this study. We then discard the *Static* and *Tracking* shots. The detailed process and examples are shown in Fig. R2.

Trajectory Extraction. We use MonST3R [8] to estimate the geometry of dynamic scenes, generating a time-varying dynamic point cloud along with per-frame camera poses and intrinsics in a feed-forward manner. This enables efficient video depth estimation and reconstruction [6]. Camera trajectories, along with the corresponding depth maps, are extracted for further processing. These trajectories undergo a series of steps including cleaning, smoothing, interpolation, and standardization into fixed-length sequences, ensuring their suitability for subsequent training.

- **Cleaning the Trajectories:** To clean the camera trajectories, we first extract the camera translations from the transformation matrices and compute the velocities between consecutive frames. A threshold is determined based on the 95th percentile of the velocity distribution, with an outlier exclusion factor α (set to 18.0). Frames with velocities exceeding this threshold are discarded. The remaining valid frames are then grouped into consecutive segments, ensuring that each segment contains at least 5 frames.
- **Smoothing the Trajectories:** After the trajectories have been cleaned, a Kalman [3] filter, based on a Constant Velocity model, is applied to smooth the valid frames within each segment. The smoothing process is performed using process and measurement noise standard deviations of 0.5 and 1.0, respectively. The smoothed segments are subsequently recombined with the original poses, resulting in a cleaned and smoothed camera trajectory. This smoothing step serves to reduce noise and enhance the stability and accuracy of the trajectory, facilitating more reliable analysis in subsequent stages.
- **Interpolation into Fixed-Length Sequences:** To ensure consistency across the trajectory data for downstream deep learning tasks, we standardize trajectories of varying lengths into fixed-length input sequences, addressing issues related to inconsistent time steps. First, spherical linear interpolation (SLERP) [5] is applied to the rotational component, while the translational component is interpolated linearly, ensuring smooth transitions between frames. The interpolated data is then padded to a fixed length of 120 frames, ensuring uniform time steps across all trajectory samples. This process guarantees that the input sequences are consistent in length and temporal structure, providing stable and reliable training data for deep learning models.

Motion Tagging. We present the distribution of translation and rotation combinations in Fig. R3. As shown, simpler motion combinations are more frequent, but motion tags still exhibit high diversity and complexity.

Caption Generation. In Fig. R4, we present the specific prompts and cases for generating two types of captions.

B. GenDoP Method

B.1. Camera Trajectory Tokenization Details

The camera trajectory tokenization process converts continuous camera parameters into discrete tokens. For the normalized parameters $\mathbf{R}_i^{\text{norm}}$ (rotation) and $\mathbf{t}_i^{\text{norm}}$ (translation) obtained after canonical normalization, the rotation matrix $\mathbf{R}_i^{\text{norm}}$ is converted into a unit quaternion $\mathbf{q} = (r_1', r_2', r_3', r_4')$, and the translation values $\mathbf{t}_i^{\text{norm}} = (t_1', t_2', t_3')$ are processed. This results in the combined vector $(r_1', r_2', r_3', r_4', t_1', t_2', t_3')$. Subsequently, the focal lengths f_x, f_y, and the scale factor s are also acquired. The process involves three key stages:

- **Rotation and Translation Normalization**: The rotation

[Prompt]
The images above are 16 frames sequentially extracted from a video, arranged in a 4x4 grid, from left to right and top to bottom. Answer the following questions based on the sequence of images.

Describe the trajectory of the camera's movement in the first paragraph. Provide a detailed and accurate description of how camera changes in position, orientation, angle, and distance throughout the scene.

In the second paragraph, classify the video into one of the following three categories:
1. The camera is **stationary** (with possible slight shaking), while the object change its position, or both may remain stationary.
2. The camera **moves** relative to the object/scene it is focused on, encompassing changes in position, orientation, angle, and distance, while the object/scene itself remains static (part of it may move slightly in place).
3. The camera **tracks** the movement of the objects as they changes positions and adjusts its focus accordingly.

Output 1/2/3 according to the above standard. Then, explain the reasoning behind the classification. If the camera is stationary, please make sure to choose option 1. If the camera tracks an object's motion from a fixed relative position, please choose option 3. For example, '2, In the video, the camera gradually approaches the object, causing it to become larger in the center of the screen.'

[Output] – Free-Moving
The sequence of images suggests that the camera initially starts with a close-up shot focused on the eyes and face of a character. As the frames progress, the camera appears to zoom out, revealing more of the character's face and upper body. The camera's position gradually shifts from a tight close-up to a medium shot, showing the character in more detail and eventually capturing a surrounding environment and additional figures in the background. The framing transitions smoothly, indicating a steady camera movement rather than sudden shifts.

2, The video is classified into this category because the initial focus is on the character's face, and as the sequence progresses, the camera moves back to encompass more of the scene and reveal other elements. This indicates that the camera is moving relative to the character to show a wider view, rather than the character staying stationary with a static camera or the camera following a moving character across a space.

[Output] – Static
Based on the sequence of images, the camera appears to stay relatively static, focusing on a subject who is using a communication device. There are no noticeable shifts in the camera's position relative to the subject, indicating that the camera does not move significantly. The subject seems to remain more or less in the same position throughout the frames, suggesting a stationary setup with only slight changes in the framing due to minor movements by the subject rather than the camera.

1, The video likely falls into the first category: "Camera is stationary (with possible slight shaking), and the object moves." This is because the camera remains fixed on the subject, who appears to be the one making subtle movements, such as talking or holding an object to their mouth, while the camera itself does not shift its position.

[Output] – Tracking
In the sequence of images, the camera follows the character closely, initially positioned behind her as she moves toward a doorway (frames 1-4). The camera then maintains its focus on the character from a rear perspective as she goes down a stairway (frames 5-8), showing a steady orientation and slight adjustments in angle to maintain the character in view. As the character exits into an open area, the camera's position shifts to a broader scene, capturing her from back and slightly to the side (frames 9-12), transitioning smoothly to a more distant angle that reveals more of the environment. The final frames (13-16) involve a dramatic upward tilt, panning to capture the collapsing buildings above from a low angle, indicating a significant change in camera orientation and distance from the character, while still emphasizing the surrounding chaos.

3, The video shows the camera following the character's movement throughout the scene. It adjusts its position and angle to maintain focus as she exits a building and enters an open space. The camera moves in relation to her, maintaining a dynamic perspective as the scene unfolds and dramatic events occur..

Figure R2. **Semantic Filtering.** Following the definitions in Sec. 3.2, shots are classified. Leveraging GPT-4o [2], we automate shot categorization into *Static*, *Free-Moving*, and *Tracking*. Shots categorized as *Object/Scene-Centric*, common in multi-view datasets, are not considered in films.

and translation components are tokenized as follows:

$$r_k = \frac{r'_k + 1}{2}, \quad k \in \{1, \ldots, 4\},$$

$$t_k = \frac{t'_k + 1}{2}, \quad k \in \{1, \ldots, 3\}.$$

This normalization maps values from the range $[-1, 1]$ to $[0, 1]$, while preserving the constraints on both rotation and translation.

- **Focal Length Adaptation**: Normalize focal lengths (f_x, f_y) relative to the principal points (c_x, c_y):

$$f_1 = \frac{f_x}{10c_x}, f_2 = \frac{f_y}{10c_y}.$$

Here, c_x and c_y typically represent half the image dimensions. The factor of 10 ensures that the focal length values remain within the range $(0, 1)$, accommodating typi-

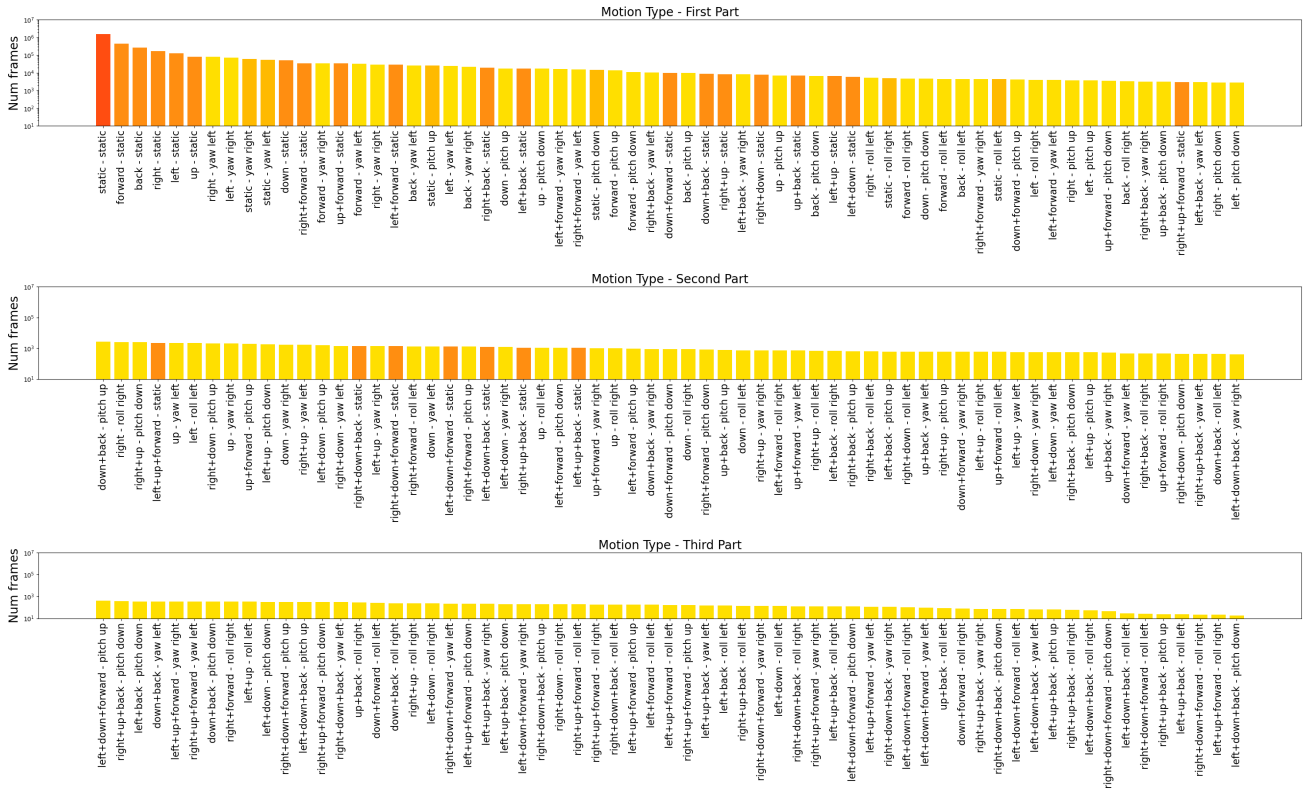


Figure R3. **Tag Distribution.** The distribution of Translation and Rotation combinations is shown in the figure. Different tag modes are represented by shades of yellow, ranging from deep to light: Static, Translation only, Rotation only, and both Translation and Rotation.

cal focal lengths.

- **Scale Parameter Transformation**: Apply logarithmic compression to the scale factor s:

$$s = \frac{\log_{10}(s') + 2}{4}.$$

This transformation enables a linear representation of multiplicative scale changes across three orders of magnitude ($0.01 \leq s' \leq 100$).

- **Parameter Clamping and Discretization**: All normalized parameters are clamped to the range $[0, 1]$ before discretization into N bins:

$$p^{token} = \lfloor p \cdot N \rfloor, \quad \forall p \in r_1, \ldots, t_3, f_1, f_2, s.$$

This process generates a compact 10-dimensional token that preserves the relative geometric relationships between parameters. The hyperparameter N controls the trade-off between quantization error and codebook size.

C. Experiments

C.1. Experiments Setting Details

We train GenDoP with a batch size of 16 using the AdamW optimizer, with a learning rate of 1e-5, $(\beta_1, \beta_2) = (0.9, 0.95)$, and a weight decay of 0.01. The KL loss weight is set to 1e-8. We use a gradient accumulation step of 1. Training is performed using bfloat16 mixed precision. The model converges with the best results at the 100th epoch.

C.2. Additional Qualitative Results

In the supplementary video available on our project website: https://kszpxxzmc.github.io/GenDoP/, we present additional cases with text prompts randomly generated by the LLM model [2]. These text prompts have never been seen in the training set, creating a certain gap compared to the captions in our dataset. As shown in the video, despite differences between the generated prompts and the training data, our model is still able to generate precise, high-quality, complex, and artistic trajectories.

Furthermore, we use TrajectoryCrafter [7] to showcase how our trajectory generation method can be applied to camera-controlled video generation. This allows for the creation of videos that align with the camera descriptions provided, ensuring the generated video sequences match the specific visual and motion criteria described by the camera control inputs.

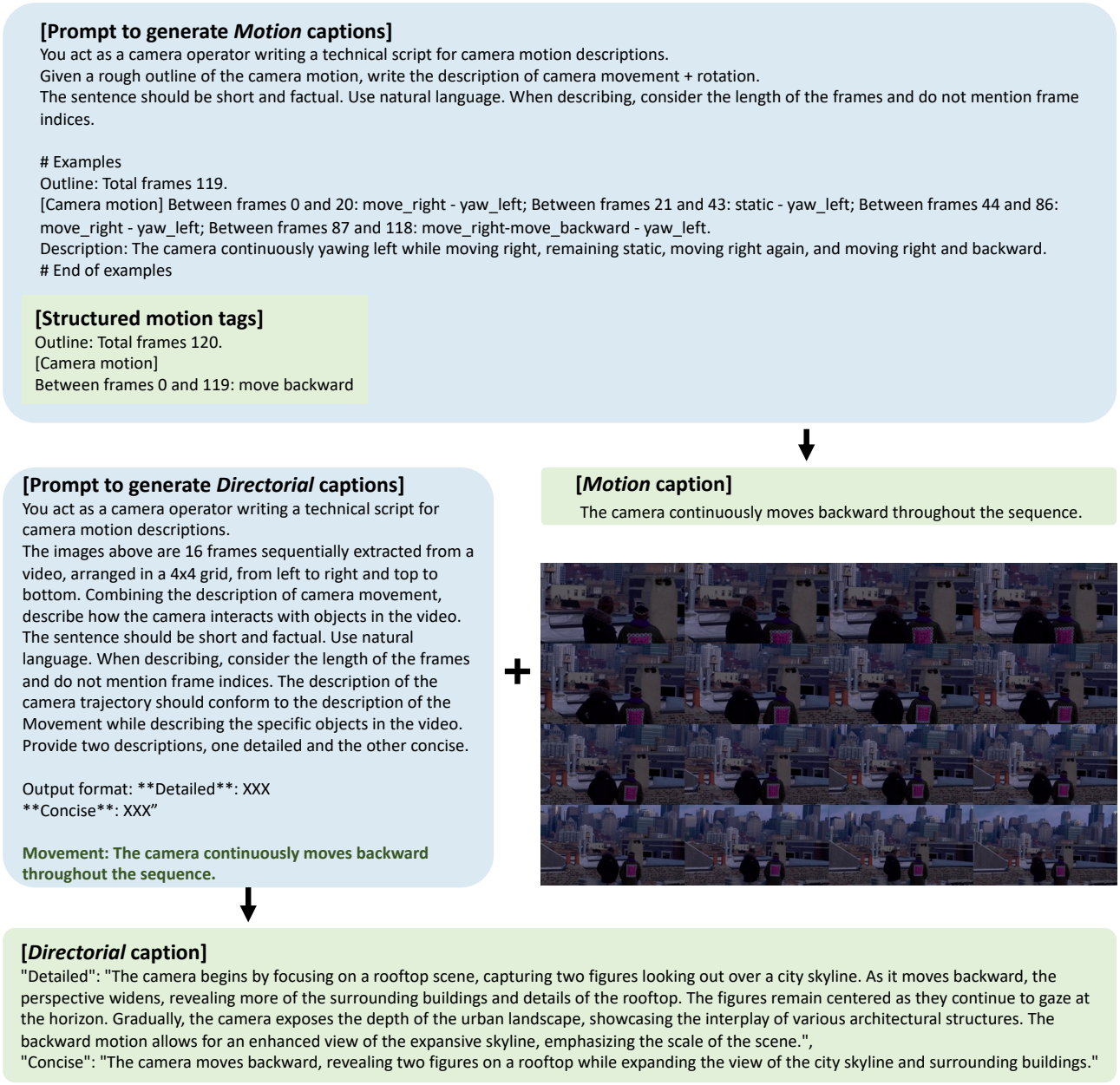


Figure R4. **Caption Generation.** We structure the motion tags by incorporating context, instructions, constraints, and examples, and then leverage GPT-4o to generate **Motion** captions that describe the camera motion alone. Next, we extract 16 evenly spaced frames from the shots to create a 4×4 grid, prompting GPT-4o to consider both the previous caption and the image sequence. This enables GPT-4o to generate **Directorial** captions that describe the camera movement, the interaction between the camera and scene, and the directorial intent.

C.3. Additional Ablation Studies

We conduct ablation experiments on several hyperparameters, as shown in Tab. S1, including the number of discrete bins, trajectory length, and model size. These parameters correspond to the discrete bin size B, the trajectory length, and the model size (as detailed in Sec. 5.1). Specifically, for the `small` size, the latent dimension is $L = 512$, with the backbone OPT Transformer consisting of 8 layers and 8 attention heads per layer. For the `base` size, the latent dimension is $L = 1024$, with 12 layers and 12 attention heads. For the `large` size, the latent dimension is $L = 1536$, with 16 layers and 24 attention heads.

The results indicate that optimal performance is achieved

Ablation		Text-Trajectory Alignment		Trajectory Quality	
		F1-Score↑	CLaTr-Score↑	Coverage↑	CLaTr-FID↓
Discrete bins	64	0.394	33.594	0.751	49.854
	128	**0.409**	35.824	0.851	24.748
	256	0.400	**36.179**	0.872	**22.714**
	512	0.391	35.201	0.882	23.633
	1024	0.393	34.277	**0.884**	24.979
Traj length	15	0.398	34.576	0.863	**22.238**
	30	**0.400**	**36.179**	**0.872**	22.714
	60	0.393	34.523	0.864	26.307
Model size	small	0.389	32.868	0.880	25.604
	base	**0.400**	**36.179**	0.872	22.714
	large	0.398	33.843	**0.888**	**20.474**

Table S1. **Ablation Study on Hyperparameters.** We conduct ablation experiments on several hyperparameters, including the number of discrete bins, trajectory length, and model size. These parameters correspond to the discrete bin size B, the trajectory length N, and the model size (as detailed in Sec. 5.1). The results show that the optimal performance is achieved when the number of discrete bins is set to 256, the trajectory length to 30, and the model size to `base`.

when the number of discrete bins is set to 256, the trajectory length to 30, and the model size to `base`. Notably, when the model size is set to *large*, although the performance in Text-texttt Alignment decreases, the Trajectory Quality improves. We speculate that this may be due to the larger model's tendency to overfit, learning better trajectory quality while failing to follow the text instructions effectively.

References

[1] Haoxuan Che, Xuanhua He, Quande Liu, Cheng Jin, and Hao Chen. Gamegen-x: Interactive open-world game video generation. *CoRR*, abs/2411.00769, 2024. 1

[2] Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Madry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codispoti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Giertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll L. Wainwright, Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont, Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey, Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei, Constantin Koumouzelis, and Dane Sherburn. Gpt-4o system card. *CoRR*, abs/2410.21276, 2024. 1, 3, 4

[3] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960. 2

[4] Anyi Rao, Jiaze Wang, Linning Xu, Xuekun Jiang, Qingqiu Huang, Bolei Zhou, and Dahua Lin. A unified framework for shot type classification based on subject centric lens. In *ECCV (11)*, pages 17–34. Springer, 2020. 1

[5] Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, 1985. 2

[6] Tong Wu, Jiaqi Wang, Xingang Pan, Xudong Xu, Christian Theobalt, Ziwei Liu, and Dahua Lin. Voxurf: Voxel-based efficient and accurate neural surface reconstruction. In *ICLR*. OpenReview.net, 2023. 2

[7] Mark YU, Wenbo Hu, Jinbo Xing, and Ying Shan. Trajectorycrafter: Redirecting camera trajectory for monocular videos via diffusion models, 2025. 4

[8] Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. *CoRR*, abs/2410.03825, 2024. 2