

COMD: Training-free Camera Motion Transfer With Camera-Object Motion Disentanglement (Supplementary Material)

Anonymous Authors

1 OVERVIEW

In this supplementary material, more details about the proposed COMD and more experimental results are provided, including:

- More implementation details (Sec. 2);
- Solving Poisson Equation (Sec. 3);
- Temporal attention maps determines the video motion (Sec. 4)
- More Comparison Results (Sec. 5);
- More Experiments on the hyperparameters (Sec. 6);
- User Study (Sec. 7).

To see the generated results more clearly, we provide a [demo.html](#), which includes all the videos in the experiments.

2 MORE IMPLEMENTATION DETAILS

We conduct experiments based on AnimateDiff-v2 [5], we use DDIM [6] to accelerate the generation process with 25 denoising steps. Moreover, to decrease the computation cost, we employ the temporal attention maps in timestep $t = 15$ to represent the video motions in different timesteps as illustrated in Sec. 3.2 of the main paper. Furthermore, for few-shot camera motion disentanglement, we specified a video count of 5 and configured DBSCAN clustering [3] with an Eps-neighborhood of 4 and core points of 3.

3 SOLVING POISSON EQUATION

We complete the temporal attention map inside the object-moving region by solving a Poisson equation. The gradients within object-moving regions of the completed attention map are assumed to be zero and the boundary values should match those of the original attention map. We choose the parallel red-black ordering Gauss-Seidel iteration method to solve the Poisson equation. Initially, we label the pixels with red-black ordering, ensuring that each pixel and its neighboring pixels alternate between being labeled red and black. Next, while ensuring that the values of boundary nodes remain unchanged, we update the red and black pixels alternately until reaching a specified number of iterations or until the residual falls below a predefined threshold. The iteration process is illustrated by the pseudo code. This algorithm can be accelerated using parallel computing frameworks like CUDA.

4 TEMPORAL ATTENTION MAPS DETERMINES THE VIDEO MOTION

The foundation of our method comes from the observation that the temporal attention map determines the motions in the generated videos, including camera motions and object motions. To validate this, we conduct an experiment to swap the temporal attention maps between two videos, where one of them contains only camera motion while the other one contains only object motion. The results are shown in Fig. 1. It can be seen that after swapping the temporal

Algorithm 1 Solving Poisson Equation

```
1: function POISSON_SOLVING(u, f)
2:   u: RGB, f: gradient
3:   choose an initial guess  $u^{(0)}$ 
4:   while not converge do:
5:     for (i, j) is red node do:
6:        $u_{i,j}^{(k+1)} = \frac{1}{4}(f_{i,j} + u_{i+1,j}^{(k)} + u_{i-1,j}^{(k)} + u_{i,j+1}^{(k)} + u_{i,j-1}^{(k)})$ 
7:     end for
8:     for (i, j) is black node do:
9:        $u_{i,j}^{(k+1)} = \frac{1}{4}(f_{i,j} + u_{i+1,j}^{(k+1)} + u_{i-1,j}^{(k+1)} + u_{i,j+1}^{(k+1)} + u_{i,j-1}^{(k+1)})$ 
10:    end for
11:  end while
12: end function
```

attention map, the contents of the two videos are similar and the motions are totally swapped. The source videos of (a) keep the camera fixed while moving the bus from left to right and (b) keep the object fixed while zooming out the camera. After swapping the temporal attention maps, the second row of (a) keeps the bus fixed while zooming out the camera and (b) keeps the camera fixed, but a shadow of a bus moves from left to right. Therefore, the temporal attention maps determine both the camera and object motions and by swapping the temporal attention map, the motions can be transferred to a new video.

5 MORE COMPARISON RESULTS

Qualitative comparison. In this section, we show more results on one-shot and few-shot camera motion transfer results, where both one-shot and few-shot methods are employed to transfer zoom-in, zoom-out, pan-left, and pan-right camera motions. The qualitative comparison results are shown in Fig. 2 and 3. In the one-shot scenario, AnimateDiff+Lora [5] appears prone to overfitting to the provided video, whereas in the few-shot scenario, it tends to amalgamate features from the training videos, leading to inaccurate video generation in response to the given prompts. MotionCtrl [8] exhibits improved alignment with prompts in video generation; however, it may introduce shape distortions and logical inconsistencies in camera motion control. In contrast, our model achieves high-quality and high-diversity generation with only one-shot or few-shot data, without the need for training.

Quantitative comparison. To further validate the effectiveness of our model, we conduct quantitative comparisons on the four basic camera motions with one-shot and few-shot data. The comparison results are shown in Tab. 1. It shows that our model achieves the best FVD [7] and FID-V [1] scores, indicating the best generation quality and diversity of our model. Since AnimateDiff is overfitted to the training data, it has the minimum optical flow distance [4],

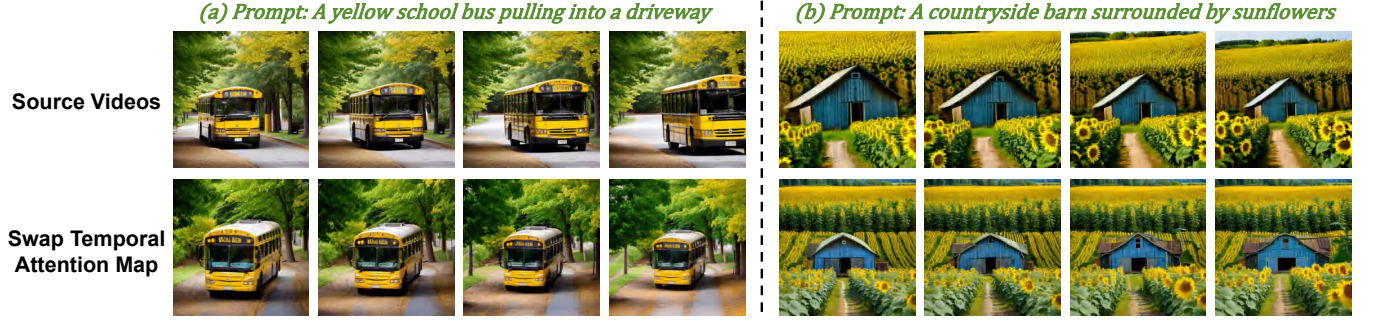


Figure 1: After swapping the temporal attention maps of the first-row videos, we have the second-row videos which swap the motions. The source videos of (a) moves the bus while fixing the camera, and (b) keeps the objects fixed while zooming out the camera. After swapping the temporal attention map, (a) keeps the bus fixed while zooming out the camera. And (b) fixed camera, but there is a shadow of a moving bus in the generated video (it is clearer in demo.html).

Table 1: Quantitative comparison results with the state-of-the-art methods on FVD, FID-V, and Optical Flow Distance. Note that AnimateDiff+Lora [5] overfits to the training data, thereby achieving the lowest flow distance. But FVD and FID-V demonstrate its worst generation diversity. In contrast, our model achieves the best FVD and FID-V, while also ensuring a good camera transfer accuracy compared to MotionCtrl [8].

Camera Motion		Pan Right			Pan Left			Zoom In			Zoom Out		
Data Scale	Method	FVD ↓	FID-V ↓	Flow Dis ↓	FVD ↓	FID-V ↓	Flow Dis ↓	FVD ↓	FID-V ↓	Flow Dis ↓	FVD ↓	FID-V ↓	Flow Dis ↓
One shot	Animatediff	382.4	4956.42	19.76	382.04	5939.96	15.22	482.58	6322.46	6.91	396.96	7767.33	5.78
	COMD (ours)	54.45	921.95	<u>37.92</u>	64.43	933.77	<u>35.64</u>	61.45	863.24	<u>12.11</u>	55.23	862.9	<u>6.93</u>
Largescale	MotionCtrl	95.83	1207.52	38.18	98.04	1196.54	55.25	80.58	935.08	13.12	80.12	928.41	7.88

(a) Comparison results on one-shot camera motion control. Bold and underline represent optimal and sub-optimal results, respectively.

Camera Motion		Pan Right			Pan Left			Zoom In			Zoom Out		
Data Scale	Method	FVD ↓	FID-V ↓	Flow Dis ↓	FVD ↓	FID-V ↓	Flow Dis ↓	FVD ↓	FID-V ↓	Flow Dis ↓	FVD ↓	FID-V ↓	Flow Dis ↓
Few shot	Animatediff	290.86	5198.78	25.61	268.29	4629.08	14.76	281.73	4333.26	5.72	251.44	3975.41	3.12
	COMD (ours)	55.94	1153.27	<u>35.98</u>	61.38	1092.09	<u>38.94</u>	51.97	847.08	<u>12.93</u>	52.90	910.76	<u>5.10</u>
Largescale	MotionCtrl	95.83	1207.52	38.18	98.04	1196.54	55.25	80.58	935.08	13.12	80.12	928.41	7.88

(b) Comparison results on few-shot camera motion control. Bold and underline represent optimal and sub-optimal results, respectively.

but it suffers from much worse FVD and FID-V. In summary, our model achieves the best FVD and FID-V, while also ensuring a good camera transfer accuracy compared to MotionCtrl. (Note that the header of Tab. 1(b) in the main paper should be "Pan Left" and "Zoom Out" and Tab. 1 here is the correct version)

Comparison on the computation cost. Moreover, we also compare the computation cost including computation time and GPU memory requirement between COMD, Animatediff+Lora [5] and MotionCtrl [2]. Since our model is a training-free method, we compute the time for disentangling the camera-object motions as our training time. To ensure the fairness of the experiment, we compute the time on the same NVIDIA A100 GPU. Meanwhile, we compare the GPU memory required for all the methods. The comparison results are shown in Tab. 2. It can be seen that our model can accomplish camera motion disentanglement in a few minutes while the other methods require a much longer training time. Moreover, both AnimateDiff+Lora and MotionCtrl require more than 30G GPU memory, while our model only needs 13G GPU

Table 2: Comparison on the computation resources. Our training-free COMD requires much less time to control the camera motions and it is the only method that is capable of running on a single NVIDIA 24G 3090/4090 GPU.

Method	Computation Time	GPU Memory
Animatediff+Lora [5]	≈10 hours	52G
MotionCtrl [8]	> 10 days	32G
One-shot COMD (Ours)	≈ 60 seconds	13G
Few-shot COMD (Ours)	≈150 seconds	13G

memory which is the only method that can be implemented on a single 24G 3090/4090 GPU.

Table 3: User Study from 28 volunteers in related research areas.

Method	Animatediff+Lora	MotionCtrl	COMD (Ours)
Percentage of Ranking First (%) ↑	0.65%	25.22%	74.13%
Average Rank ↓	2.93	1.80	1.27

6 MORE EXPERIMENTS ON THE HYPERPARAMETERS

In section 3.2 of the main paper, we propose that we can employ the temporal attention map in one intermediate step t to represent the motions in different timesteps. We find that the timestep t cannot either be too large or too small, since the temporal attention map in a too large t contains too much noise and the temporal attention map in a too small t contains little temporal information. To validate this, we conduct one-shot camera motion transfer experiments on different timesteps t , which are shown in Fig. 4. It can be seen that when t is too large ($t \geq 22$), the output videos suffer from heavy artifacts due to the noise in the temporal attention map. And when t is too small ($t \leq 3$), the generated video cannot be correctly generated since the temporal attention map contains too little motion information, which fails to guide the video generation process with accurate camera motions. The intermediate timesteps $5 < t < 20$ all generate good results. Therefore, we choose timestep $t = 15$ as our default hyperparameter.

7 USER STUDY

In this section, we conduct a user study to evaluate the effectiveness of our COMD. We have invited 28 volunteers in related research areas to rank the generated results from AnimateDiff+Lora [5], MotionCtrl [8] and our COMD considering the generation quality, diversity, and the camera transfer accuracy. Specifically, each volunteer ranked 20 sets of results, where each basic camera motion (pan left, pan right, zoom in, and zoom out) contains 5 videos. We compute the average ranking and the percentage of ranking first of the three methods, which is shown in Tab. 3. It can be seen that our model ranks first in **74.13%** situations and achieves the best average rank of **1.27**, demonstrating the superior performance of our COMD in camera motion transfer.

REFERENCES

- [1] Yogesh Balaji, Martin Renqiang Min, Bing Bai, Rama Chellappa, and Hans Peter Graf. 2019. Conditional GAN with Discriminative Filter Generation for Text-to-Video Synthesis. In *IJCAI*, Vol. 1. 2.
- [2] Tsai-Shien Chen, Chieh Hubert Lin, Hung-Yu Tseng, Tsung-Yi Lin, and Ming-Hsuan Yang. 2023. Motion-conditioned diffusion model for controllable video synthesis. *arXiv preprint arXiv:2304.14404* (2023).
- [3] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, Vol. 96. 226–231.
- [4] Gunnar Farnéback. 2003. Two-frame motion estimation based on polynomial expansion. In *Image Analysis: 13th Scandinavian Conference, SCIA 2003 Halmstad, Sweden, June 29–July 2, 2003 Proceedings 13*. Springer, 363–370.
- [5] Yuwei Guo, Ceyuan Yang, Anyi Rao, Yaohui Wang, Yu Qiao, Dahua Lin, and Bo Dai. 2023. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *arXiv preprint arXiv:2307.04725* (2023).
- [6] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502* (2020).
- [7] Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. 2018. Towards accurate generative models

of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717* (2018).

- [8] Zhouxia Wang, Ziyang Yuan, Xintao Wang, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. 2023. Motionctrl: A unified and flexible motion controller for video generation. *arXiv preprint arXiv:2312.03641* (2023).

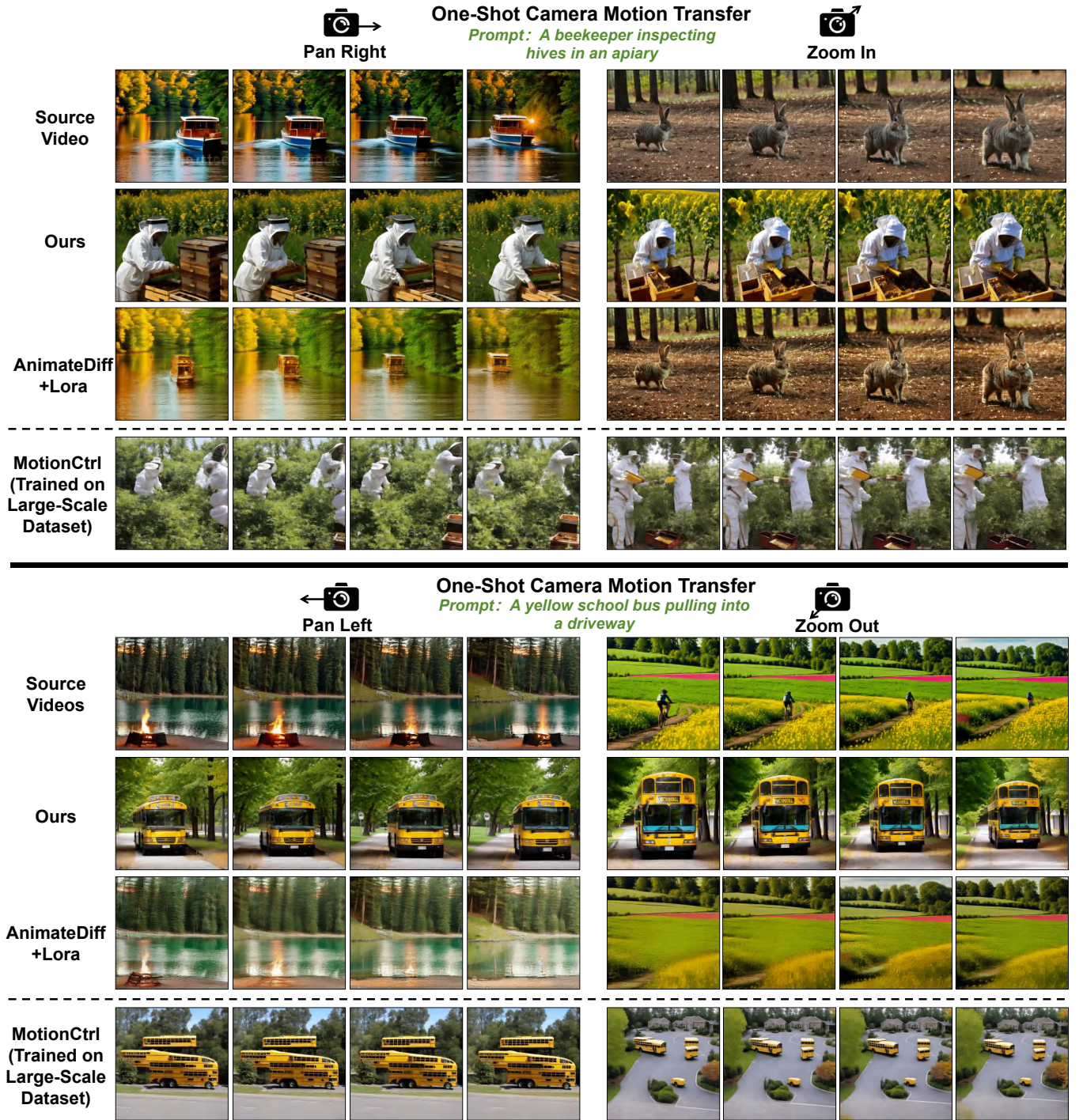


Figure 2: Comparison on one-shot camera motion transfer on four basic camera motions: pan right, pan left, zoom in and zoom out. AnimateDiff+Lora [5] overfits the training video. Even if MotionCtrl [8] is trained on a large-scale dataset, it still suffers from inaccurate camera motion control and some artifacts in the generated videos. In contrast, our model accurately transfers the camera motions while ensuring good generation quality and diversity.

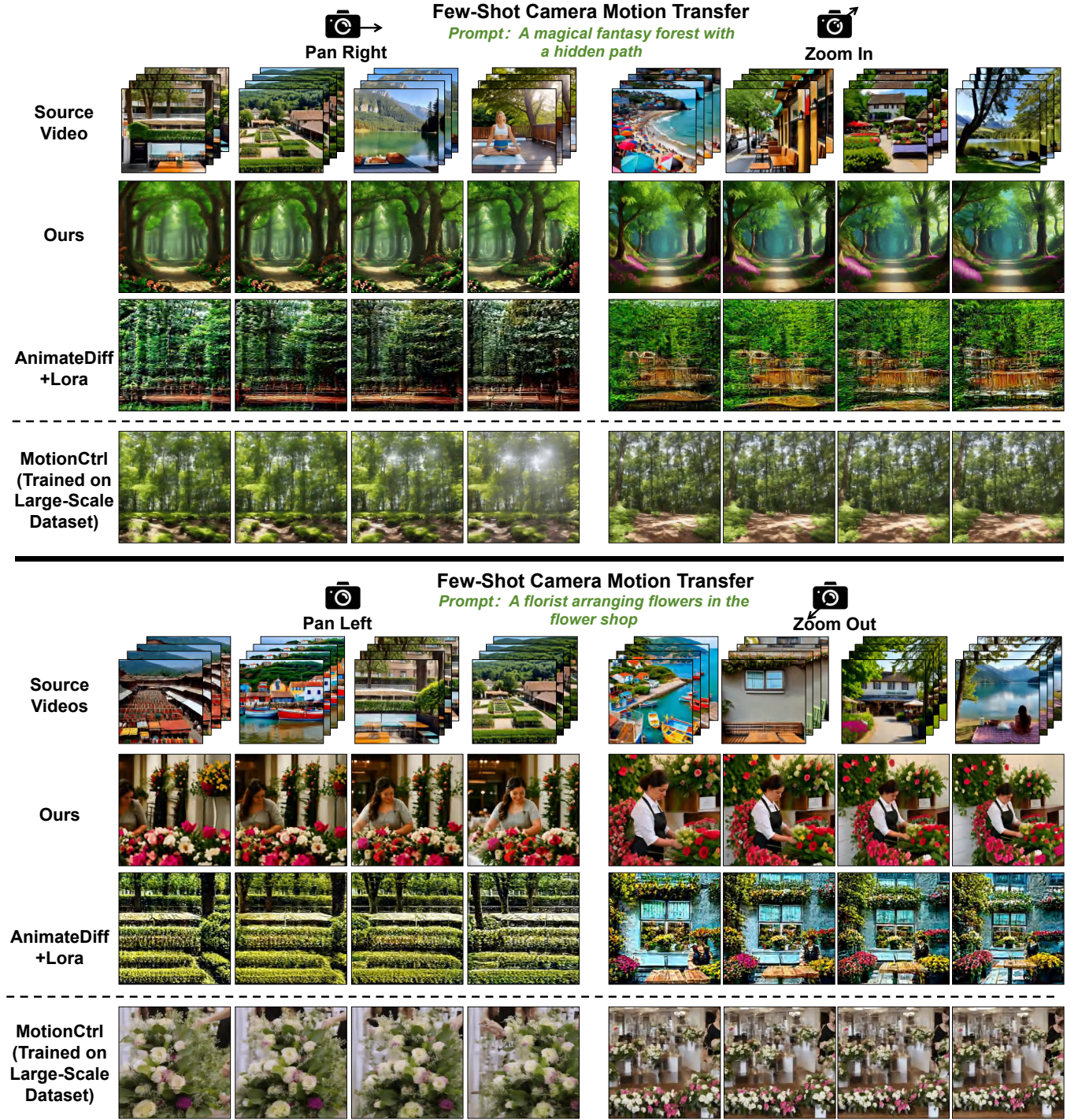


Figure 3: Comparison on few-shot camera motion transfer on four basic camera motions: pan right, pan left, zoom in and zoom out. AnimateDiff+Lora [5] overfits to the training videos, which generate videos with mixed features from the training data. Even if MotionCtrl [8] is trained on a large-scale dataset, it still suffers from inaccurate camera motion control and some artifacts in the generated videos. In contrast, our model accurately transferred the camera motions while ensuring good generation quality and diversity.

