Generative Photography: Scene-Consistent Camera Control for Realistic Text-to-Image Synthesis Supplementary Materials

Yu Yuan[1], Xijun Wang[1], Yichen Sheng[2], Prateek Chennuri[1], Xingguang Zhang[1], Stanley Chan[1]
[1]School of ECE, Purdue University [2]NVIDIA Research

A. Introduction

This supplementary material provides additional discussions and details on the construction of differential data (Section B), network design (Section C), evaluation metrics (Section D), and more visual results (Section E).

To better illustrate the continuity and effects of camera intrinsic setting control, we highly recommend readers view the **Videos/GIFs** provided in the project page: https://generative-photography.github.io/project/.

B. More Details of Building Differential Data

Our differential data pipeline dynamically generates training data by storing only base images and scene descriptions. Camera settings are sampled during training and simulated **on-the-fly** using physical principles, producing differential multi-frame data without pre-storing large video files. This also ensures continuous sampling of training data.

We provide below additional key considerations for constructing differential datasets for each type of camera setting, along with sample demonstrations.

B.1. Differential Data for Bokeh Rendering

As shown in Fig. 7, to enhance the prominence of the bokeh rendering effect, we impose the following two requirements on the base images: 1). The images should be nearly all-in-focus. 2). They should exhibit significant depth differences, allowing clear distinction between foreground and background.

We employ bokehMe [10] for realistic bokeh simulation. During this process, the value of the refocused disparity is consistently maintained at the depth of the foreground.

B.2. Differential Data for Focal Length

In the real world, obtaining a set of images of the same scene at multiple focal lengths is highly cumbersome, with a lack of perfect alignment between the images, and the achievable focal length range is limited [17]. In this paper, we reference the level-of-detail [8, 13] approach and compute the field-of-view (FoV) ratio of the desired focal length relative to the base image focal length. This ratio is then used for center cropping to approximate the actual continuous optical zoom process. In this subsection, we compare the performance of our method with that of actual optical zoom.

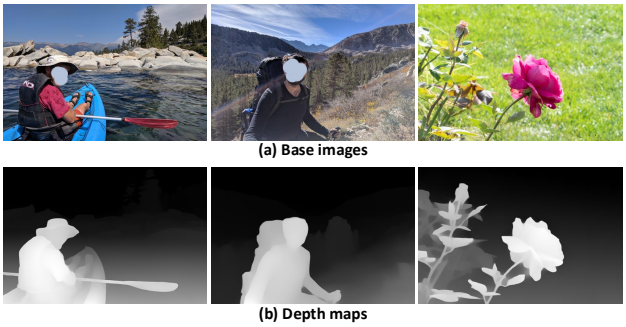

Figure 7. The first row shows examples of base images used for constructing bokeh rendering data, featuring prominent foregrounds and distinguishable backgrounds. The second row presents depth maps extracted using the Depth Anything [14, 15] model.

A camera's field-of-view (FoV) can be expressed in terms of the focal length f and the sensor dimensions (typically width w or height h). The formulas are as follows:

Horizontal FoV:

$$\text{FoV}_\text{h} = 2 \cdot \arctan\left(\frac{w}{2f}\right) \tag{1}$$

Vertical FoV:

$$\text{FoV}_\text{v} = 2 \cdot \arctan\left(\frac{h}{2f}\right) \tag{2}$$

Diagonal FoV:

$$\text{FoV}_\text{d} = 2 \cdot \arctan\left(\frac{\sqrt{w^2+h^2}}{2f}\right) \tag{3}$$

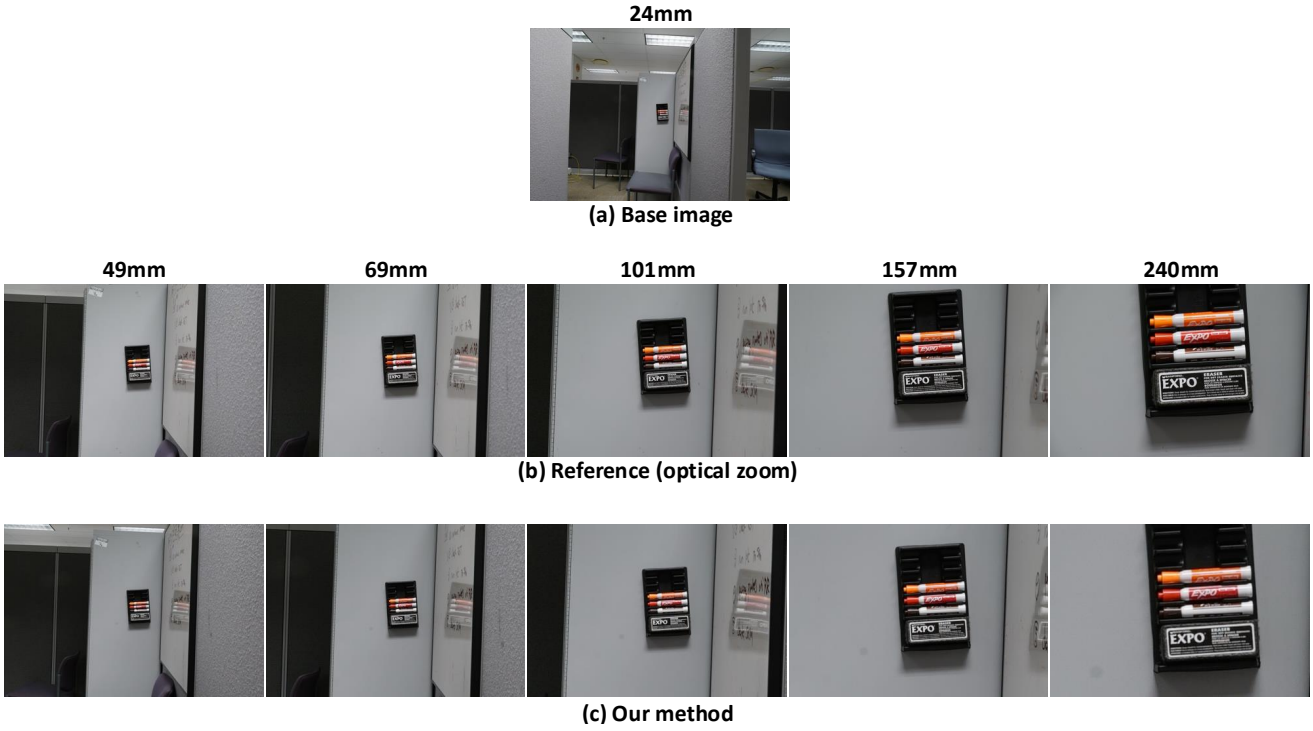

Figure 8. The comparison between the reference real focal lengths and our simulated results. Note that the real-world shooting data is derived from [17], and there may be slight misalignment between images of different resolutions due to shooting conditions. We observe that excessively high focal length simulation ratios can lead to a decline in image quality. Therefore, in this study, the focal length range is constrained to 24-70mm. Please zoom in for a more detailed comparison.

where w denotes the width of the sensor, h is the height of the sensor, and f represents the focal length.

Based on the aforementioned FoV calculation formula, we crop the central region of a high-resolution base image to simulate the corresponding view at larger focal lengths. Fig. 8 compares the optical zoom with the results generated by our cropping method. The real-world data for different focal lengths is from [17]. Our method demonstrates a high degree of consistency with the real data in terms of FoV. It is worth noting that due to the resolution and quality constraints of the base image, excessive cropping leads to significant loss of detail and quality. Therefore, in this work, we limit the focal length range to 24-70mm.

B.3. Differential Data for Shutter Speed

A realistic imaging model can be formulated as follows, similar to [2, 6, 11]. Consider a final LDR image, L, captured at an exposure time of t where the underlying HDR scene irradiance map is represented by H.

$$L = \texttt{ADC}\Big\{\xi \times \texttt{Clip}\big\{\text{Poisson}\big(t \times \text{QE} \times (H + \mu_{\text{dark}})\big)\big\} + N(0, \sigma^2_{\text{read}})\Big\}^{1/\gamma} \tag{4}$$

where ξ is the conversion gain, QE is the quantum efficiency, μ_{dark} is the dark current, and σ_{read} is the read noise standard deviation. Here, Poisson represents the Poisson distribution characterizing the photon arriving process and the dark current effect, and N represents the Gaussian distribution characterizing the sensor noise. $\texttt{ADC}\{\cdot\}$ is the analog-to-digital conversion and $\texttt{Clip}\{\cdot\}$ is the full well capacity induced saturation effect. We assume a linear camera response function for CMOS sensors and that the imperfections in the pixel array, ADC, and color filter array have been mitigated.

For the shutter speed control task, we selected base images with a high dynamic range and appropriate exposure to approximate H. By varying the parameter t in the formula 4, we simulate multiple frames corresponding to different shutter speeds.

B.4. Differential Data for Color Temperature

We employ an empirical approximation revised from [3] to map a given color temperature in Kelvin to corresponding RGB values, ensuring accurate and balanced color representation. The input kelvin is normalized by dividing by 100, resulting in temp. The conversion process is as follows:

For temp ≤ 66:

$$\begin{aligned}\mathbf{RGB} = (&255,\\ &\max\left(0, 99.47 \cdot \ln(\text{temp}) - 161.12\right),\\ &\max\left(0, 138.52 \cdot \ln(\text{temp} - 10) - 305.04\right))\end{aligned} \tag{5}$$

For $66 <$ temp ≤ 88:

$$\begin{aligned}\mathbf{RGB} = \big(&0.5 \cdot \left(255 + 329.70 \cdot (\text{temp} - 60)^{-0.1933}\right),\\ &0.5 \cdot \big(288.12 \cdot (\text{temp} - 60)^{-0.1155}\\ &+ 99.47 \cdot \ln(\text{temp}) - 161.12\ \big),\\ &0.5 \cdot (138.52 \cdot \ln(\text{temp} - 10) - 305.04 + 255)\big)\end{aligned} \tag{6}$$

For temp > 88:

$$\begin{aligned}\mathbf{RGB} = (&329.70 \cdot (\text{temp} - 60)^{-0.1933},\\ &288.12 \cdot (\text{temp} - 60)^{-0.1155},\\ &255)\end{aligned} \tag{7}$$

After computation, the RGB values are clipped to the range $[0, 255]$ to ensure valid color values. The resulting balanced RGB values are returned as a `float32` array, providing an accurate representation of the input temperature in RGB space.

C. More Details of Differential Camera Encoder

In the Differential Camera Encoder, an important aspect is the incorporation of the differences in camera setting scales. We extract the camera settings for F_r frames using the CLIP text encoder, compute the differences, and then reshape the result into an embedding of size $F_r \times C \times H \times W$.

In addition, this section will also provide more details on the coarse embedding and the embedding encoder.

C.1. Coarse Embedding

The input to the coarse embedding is solely the provided camera settings. Based on a simplified version of the physical simulation model, it outputs an embedding with a shape of $F_r \times C \times H \times W$.

For bokeh rendering, the input bokeh blur parameter is treated as an equivalent Gaussian blur kernel. A larger parameter indicates that the weight of each pixel in the output is lower, resulting in smaller global pixel embedding values.

As illustrated in Fig. 9, for focal length, we use mask to proxy the coarse embedding. Specifically, after calculating the field of view (FoV) ratio, we mask out regions of the original image resolution that should not be present.

For shutter speed, we roughly estimate the ratio between the target shutter time and the base shutter time (simplified as 0.2 second on average). This ratio is then used to compute the overall average brightness ratio of the image, which serves as the global coefficient for the coarse embedding.

Figure 9. We use a mask as the coarse embedding for focal length control. The black areas represent pixels around the edges of the frame that should not be displayed at the given focal length.

For color temperature, we estimate the ratio coefficients for the RGB channels based on the color temperature value, using a simplified version of the corresponding formula from Equation 5 to Equation 7. These coefficients are then used as the scaling factors for the coarse embedding.

C.2. Embedding Encoder

The embedding encoder takes both the coarse embedding and the differential information embedding as input. After encoding, it injects the information into the temporal attention layers of the foundation model in a hierarchical manner. Its internal structure is based on the T2I adapter [9], with additional temporal structures for multi-setting processing.

D. More Details of Proposed Metrics

D.1. Accuracy

To evaluate the accuracy of the camera physics in generated images, we first simulate the reference frames of the base image under multiple camera settings, using the same scene description and corresponding camera parameters for generation. We then calculate the overall trend of camera effects within the reference frames and the overall trend of camera effects within the generated multi-frame sequence. The Pearson correlation coefficient between these two trends is computed as an accuracy metric (CorrCoef). For each type of camera setting, we employ different methods to calculate the camera effects.

- For Bokeh: We compute the average blur level per frame using the Laplacian operator.
- For Focal Length: We first detect feature points using SIFT [7], then perform feature matching between adjacent frames using Brute-Force Matcher [1]. We calculate the similarity transformation matrix from the matched points and extract the scaling factor from the transformation matrix.
- For Shutter Speed: We compute the average brightness per frame.
- For Color Temperature: We compute the average color per frame.

D.2. Consistency

For the consistency between frames corresponding to different camera setting values, we compute the frame-to-frame

consistency using the Frame-wise Learned Perceptual Image Patch Similarity (LPIPS) [16]. Subsequently, we average the LPIPS scores of all adjacent frames to obtain the final score. An important nuance here is that a lower LPIPS score is not always preferable, as we require some variation in camera effects. Therefore, the LPIPS score should be compared to that of reference videos, with a closer match indicating better performance.

D.3. Following

We measure the prompt following of the generated frames by evaluating their alignment with the input prompts. Specifically, we use the CLIP [12] text and image encoders to obtain the features of the prompt and the generated frame, and then compute the cosine similarity between the two.

E. More Visual Results

In this section, we provide additional visual results and comparisons with other methods.

Fig. 10 to Fig. 13 illustrate the visual comparisons for bokeh rendering, focal length, shutter speed, and color temperature across various generative methods. Our approach demonstrates significant advantages in understanding camera physical parameters while maintaining scene consistency.

We strongly encourage readers to view the videos/GIFs we provide for more intuitive comparisons and additional case studies.

References

[1] Brute-Force Matcher. https://docs.opencv.org/4.x/dc/dc3/tutorial_py_matcher.html. 3

[2] Yiheng Chi, Xingguang Zhang, and Stanley H. Chan. HDR imaging with spatially varying signal-to-noise ratios. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5724–5734, 2023. 2

[3] Mark D. Fairchild. *Color Appearance Models*. 2013. 2

[4] 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. *International Conference on Learning Representations*, 2024. 5, 6, 7, 8

[5] 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. 5, 6, 7, 8

[6] Zhihao Li, Ming Lu, Xu Zhang, Xin Feng, M. Salman Asif, and Zhan Ma. Efficient visual computing with camera raw snapshots. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(7):4684–4701, 2024. 2

[7] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 3

[8] S.G. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, 1989. 1

[9] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, Ying Shan, and Xiaohu Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. *arXiv preprint arXiv:2302.08453*, 2023. 3

[10] Juewen Peng, Zhiguo Cao, Xianrui Luo, Hao Lu, Ke Xian, and Jianming Zhang. BokehMe: When neural rendering meets classical rendering. In *IEEE/CVF International Conference on Computer Vision and Pattern Recognition*, 2022. 1

[11] Xiangyu Qu, Yiheng Chi, and Stanley H. Chan. Spatially varying exposure with 2-by-2 multiplexing: Optimality and universality. *IEEE Transactions on Computational Imaging*, 10:261–276, 2024. 2

[12] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021. 4

[13] Andrew P. Witkin. Scale-space filtering. In *Readings in Computer Vision*, pages 329–332. 1987. 1

[14] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything: Unleashing the power of large-scale unlabeled data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 1

[15] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything v2. *arXiv preprint arXiv: 2406.09414*, 2024. 1

[16] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018. 4

[17] Xuaner Zhang, Qifeng Chen, Ren Ng, and Vladlen Koltun. Zoom to learn, learn to zoom. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 1, 2

Figure 10. Visual comparisons between different generative methods on camera bokeh rendering control. Both AnimateDiff [4] and CameraCtrl [5] have been fine-tuned/trained on our data.

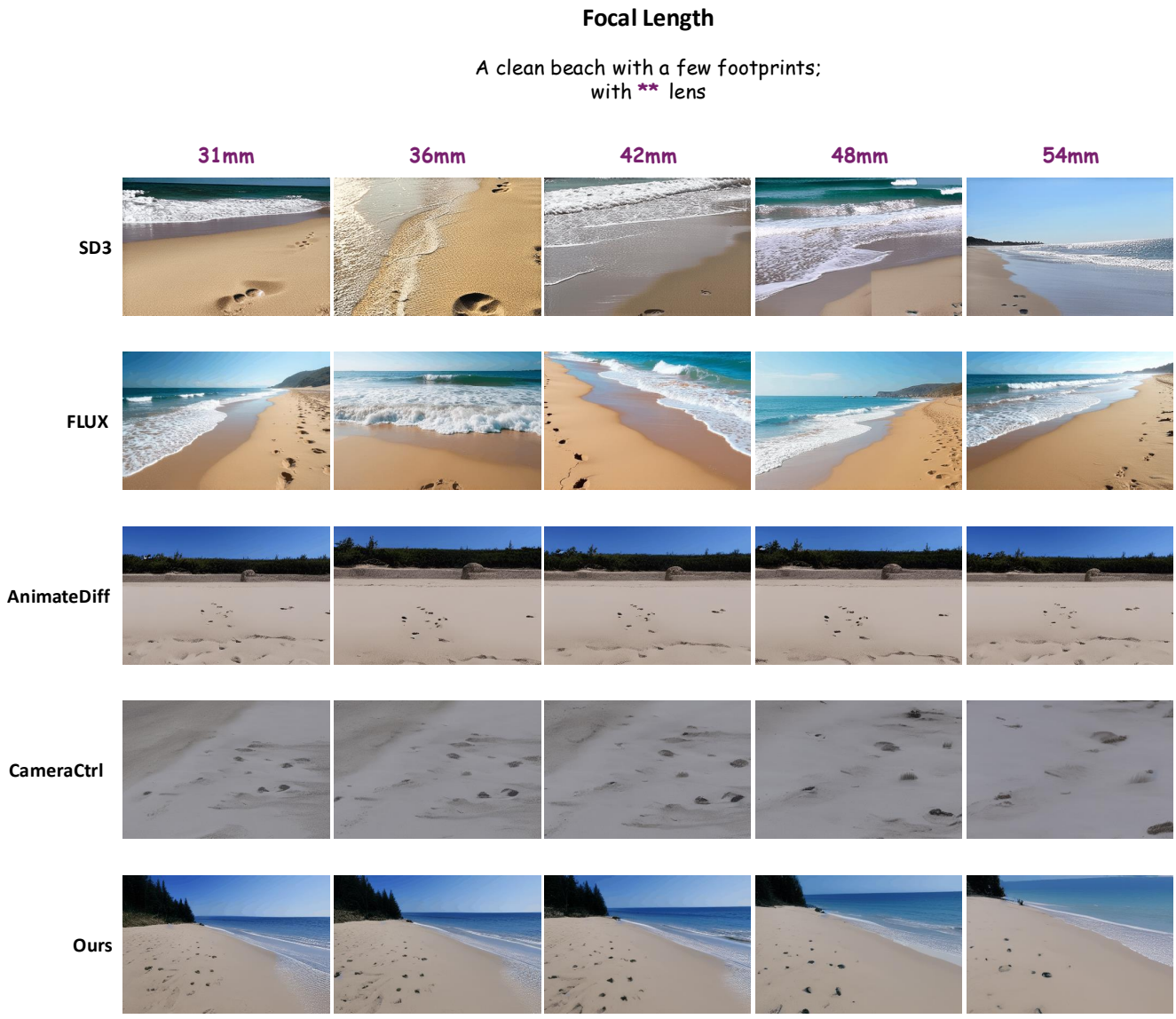

Figure 11. Visual comparisons between different generative methods on camera focal length control. Both AnimateDiff [4] and CameraCtrl [5] have been fine-tuned/trained on our data.

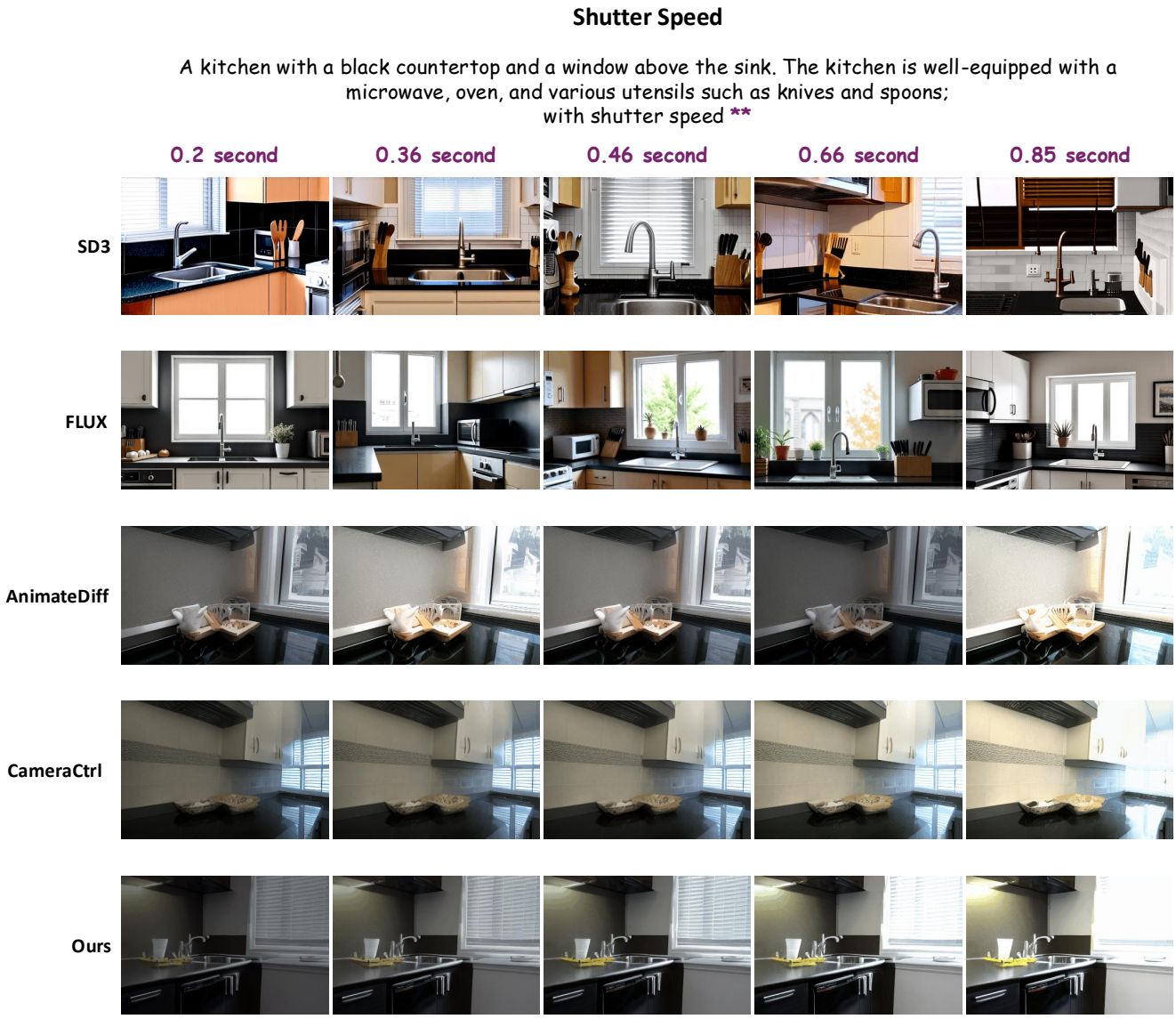

Figure 12. Visual comparisons between different generative methods on camera shutter speed control. Both AnimateDiff [4] and CameraCtrl [5] have been fine-tuned/trained on our data.

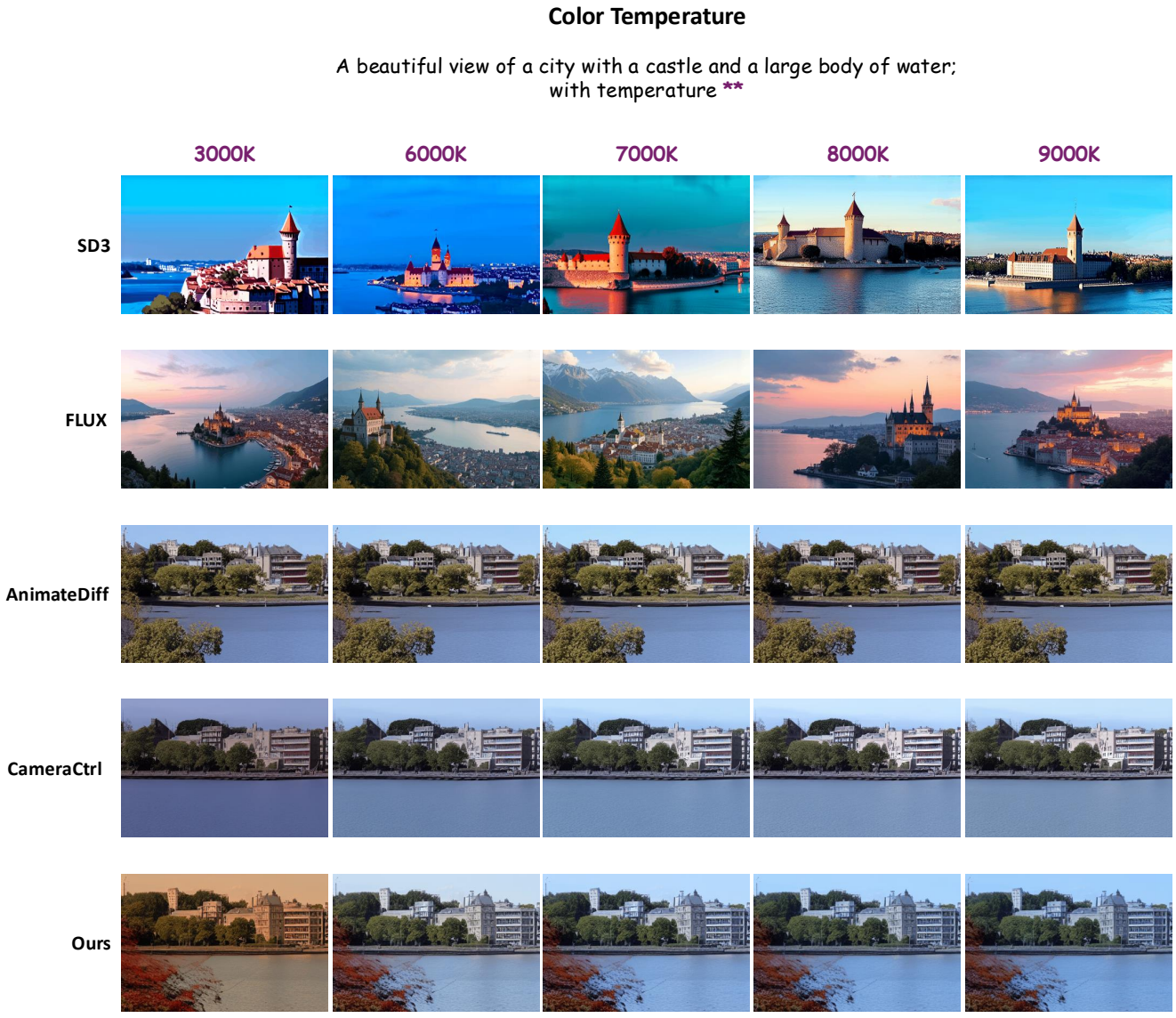

Figure 13. Visual comparisons between different generative methods on camera color temperature control. Both AnimateDiff [4] and CameraCtrl [5] have been fine-tuned/trained on our data.