

# SceneCrafter: Controllable Multi-View Driving Scene Editing

## (Supplementary Material)

Zehao Zhu<sup>1,2</sup>, Yuliang Zou<sup>1</sup>, Chiyu Max Jiang<sup>1</sup>, Bo Sun<sup>1</sup>, Vincent Casser<sup>1</sup>, Xiukun Huang<sup>1</sup>,  
Jiahao Wang<sup>3</sup>, Zhenpei Yang<sup>1</sup>, Ruiqi Gao<sup>4</sup>, Leonidas Guibas<sup>4</sup>, Mingxing Tan<sup>1</sup>, Dragomir Anguelov<sup>1</sup>  
<sup>1</sup>Waymo, <sup>2</sup>University of Texas at Austin, <sup>3</sup>Johns Hopkins University, <sup>4</sup>Google DeepMind

### A. More Technical Details

#### A.1. SceneCrafter Models

We trained a total of three diffusion models: the SceneCrafter teacher model for global edits (Teacher-G), the SceneCrafter teacher model for local edits (Teacher-L), and the SceneCrafter student model (Student). Tab. 1 provides an overview of the experimental settings for each of these models.

#### A.2. Self-attention Weights Manipulation

We show the algorithm details in **Algorithm 1**.  $DM$  is the diffusion model and  $M_t$  denotes the self attention weights of the source images in each intermediate U-Net layer.

---

#### Algorithm 1 Self-Attention Weights Manipulation

---

**Require:** source prompt  $\mathcal{P}$ , target prompt  $\mathcal{P}^*$

- 1: Sample  $z_T \sim \mathcal{N}(0, I)$
  - 2: Set  $z_T^* \leftarrow z_T$
  - 3: **for**  $t = T$  **to** 1 **do**
  - 4:   Compute  $(z_{t-1}, M_t) \leftarrow DM(z_t, \mathcal{P}, t)$
  - 5:   Compute  $z_{t-1}^* \leftarrow DM(z_t^*, \mathcal{P}^*, t, M_t)$
  - 6: **end for**
  - 7: **return**  $(z_0, z_0^*)$
- 

#### A.3. Inference Time

Our method takes approximately 10 seconds to generate or edit 8 multi-view images with  $512 \times 512$  size each on a single A100 GPU. For larger-scale operations, we run it on a cluster of 128 A100 GPUs, achieving an amortized inference time of just 0.1 seconds.

#### A.4. Multi-View Consistency

Our model is trained on 8 cameras whose fields of view overlap by several degrees between each neighboring pair. Therefore our metric is evaluated on a total of 16 image pairs per frame: 8 pairs by projecting from  $C_i$  to  $C_{i+1}$ , and 8 pairs by projecting from  $C_{i+1}$  to  $C_i$ . We illustrate

	Teacher-G	Teacher-L	Student
<b>Condition</b>			
Time of day	✓	✓	✓
Weather	✓	✓	✓
Agent boxes	✓	✗	✓
HD maps	✓	✓	✓
Foreground masks	✗	✓	✗
Raymaps	✓	✓	✓
Source image	✗	✗	✓
<b>Training</b>			
Batch size	128	128	128
Learning rate	0.0005	0.0005	0.0005
Training steps	100k	100k	200k
Training data	Real	Real	Synthetic
Masked training	✗	✓	✗
Pretrained model	CAT3D [1]	CAT3D [1]	Teacher-G
<b>Diffusion</b>			
Denoising Steps	50	50	50
Noise schedule	Linear	Linear	Linear
EMA	✓	✓	✓
Sampler	DDIM [4]	DDIM [4]	DDIM [4]
z-shape	$64 \times 64 \times 8$	$64 \times 64 \times 8$	$64 \times 64 \times 16$
<b>Misc</b>			
Generation ability	✓	✓	✓
Editing ability	✗	✗	✓

Table 1. **Experimental settings for the SceneCrafter model family.** *Teacher-G* refers to the SceneCrafter teacher model for global editing data generation, *Teacher-L* is the SceneCrafter teacher model for local editing data generation, and *Student* represents the SceneCrafter student model for scene editing.

this process in Fig. 1. For the projection, we utilize known camera intrinsic and extrinsic parameters, and assume a fixed baseline distance. While this is an approximation, the LPIPS metric is well documented to be sufficiently robust to minor spatial shifts and distortions [5]. Fig. 4 shows stitched 360° panoramic images of real and generated images with no visible seams or inconsistencies beyond projection error.

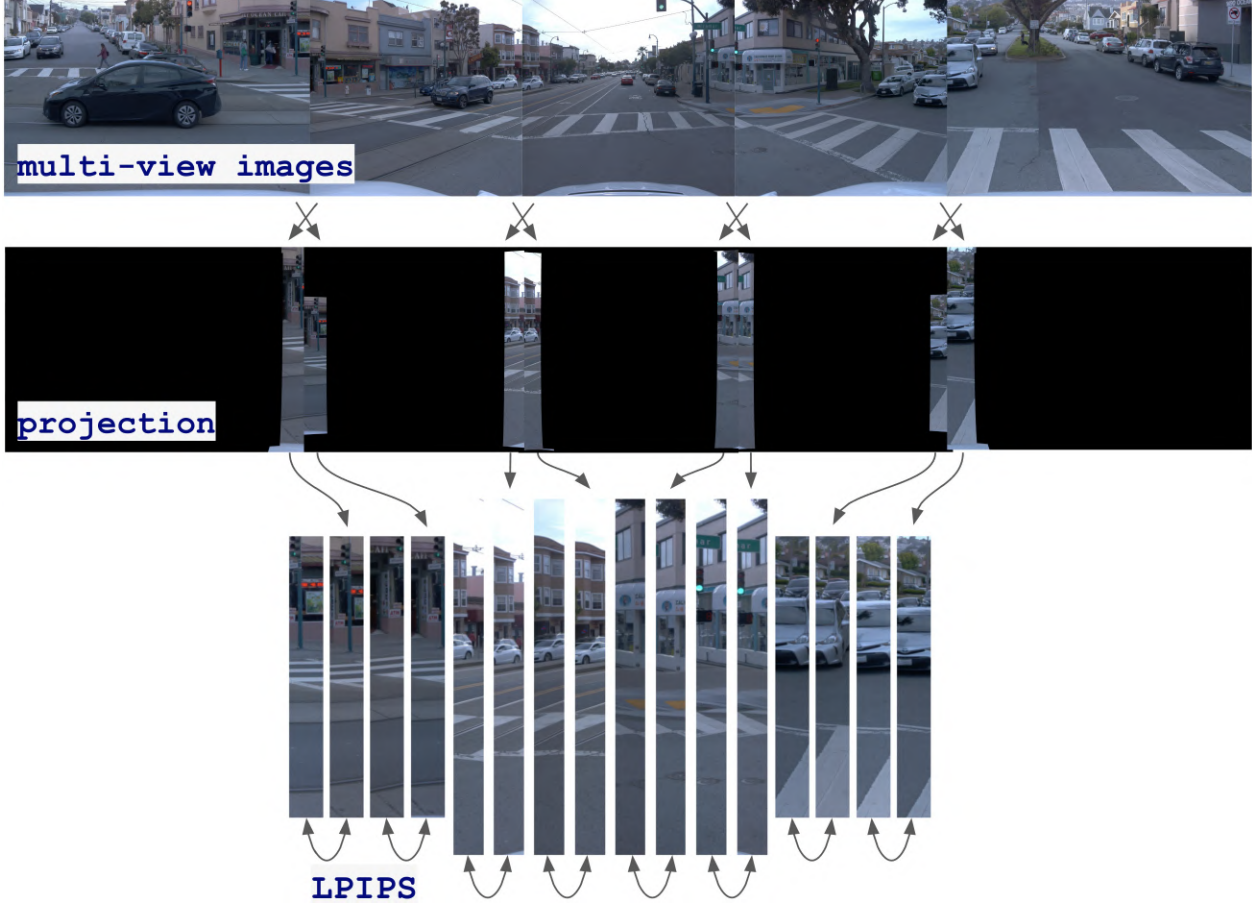


Figure 1. **Illustration of our multi-view consistency metric.** Starting from a multi-view image, we project each view into the two adjacent views. We are then able to define two pairs of image patches for each overlapping region and compare them using LPIPS to evaluate their consistency. For simplicity, we show an illustrative example with only 5 cameras.



Figure 2. Comparing segmentation results on real logs and synthetic images generated by SceneCrafter.

## B. Downstream Task Performance

The theme for this project is on simulation, specifically evaluating downstream models within a controllable simulated environment. Thus, we test off-the-shelf downstream models on our generated data without retraining them.

### B.1. Segmentation

We applied a Panoptic-DeepLab semantic segmentation model on real and conditionally generated camera images, and found its behavior to be similar (See Fig. 2). The KL divergence of aggregate class distributions of 26 classes across 8,000 images is only 0.01133, with the only minor variations being in vegetation, building and sky due to SceneCrafter being least conditioned on their outline. The



Figure 3. Comparing detection results on real logs and synthetic images generated by SceneCrafter.

predicted masks are sharp and semantically corresponding.

### B.2. Detection

We employed a monocular 3D detection model to compare predictions between real and conditionally generated front camera images. We find that the 3D detector holds up surprisingly well to the generated images as shown in Fig. 3. It is rare to find false-negative detections in the generated imagery.

### C. More Experimental Visualizations

We provide additional visualizations of time-of-day editing results in Fig. 5 and Fig. 6. By evenly sampling times between 7 PM and 8 PM, we apply all these times to edit the source images. The results demonstrate a seamless transition from day to night, while ensuring 1) geometric consistency between the edited outputs and the original source images, and 2) consistent editing across all results. This highlights the robustness and effectiveness of SceneCrafter.

Fig. 7 further showcases the weather editing results. Starting with multi-view images captured under a specific weather condition, we edit them to different weather conditions, including sunny, rainy, foggy, and snowy. Even when tackling challenging edits like snowy weather, which significantly differs from the original, SceneCrafter effectively preserves the scene’s original layout like trees and roads, while seamlessly adding realistic elements like snow covering roads or grass. This makes SceneCrafter a practical choice for simulating extreme weather conditions.

Fig. 8 illustrates the agent editing results. SceneCrafter allows for insertion or removal of arbitrary agents in the source images, controlled through specified agent boxes and their types (foreground for insertion and background for removal). The results demonstrate SceneCrafter’s ability to perform precise and dexterous scene manipulations, even for small vehicles. Additionally, the generated vehicles are highly realistic, appearing nearly indistinguishable from real ones and aligning perfectly with the lighting conditions of the original scenes.

We present more comparison results for global edits with SDEdit [3] and Prompt-to-Prompt [2] in Fig. 9 and Fig. 10. Our SceneCrafter excels at preserving fine-grained details from the source images, such as text and icons, while delivering realistic and precise edits. These scenes are also included as part of our user study data. For the first example of time-of-day editing, 9 out of 11 users rated SceneCrafter as having the best editing results, while 2 preferred P2P\*. For the first example of weather editing, all participants unanimously selected SceneCrafter as the best performer, showing its strong alignment with human preferences.

Fig. 11 offers a visual comparison of vehicle removal capabilities between our method and the 2D-Repaint and MV-Repaint baselines. 2D-Repaint relies on Stable Diffusion, and struggles to maintain view consistency in 3D editing task. While MV-Repaint leverages global editing teacher model, it lacks masked training priors, resulting in incomplete car removals. In contrast, our method, trained on alpha-blended pairs of “empty streets” and “populated streets,” demonstrates superior capability in handling complex scene editing with precision and consistency.



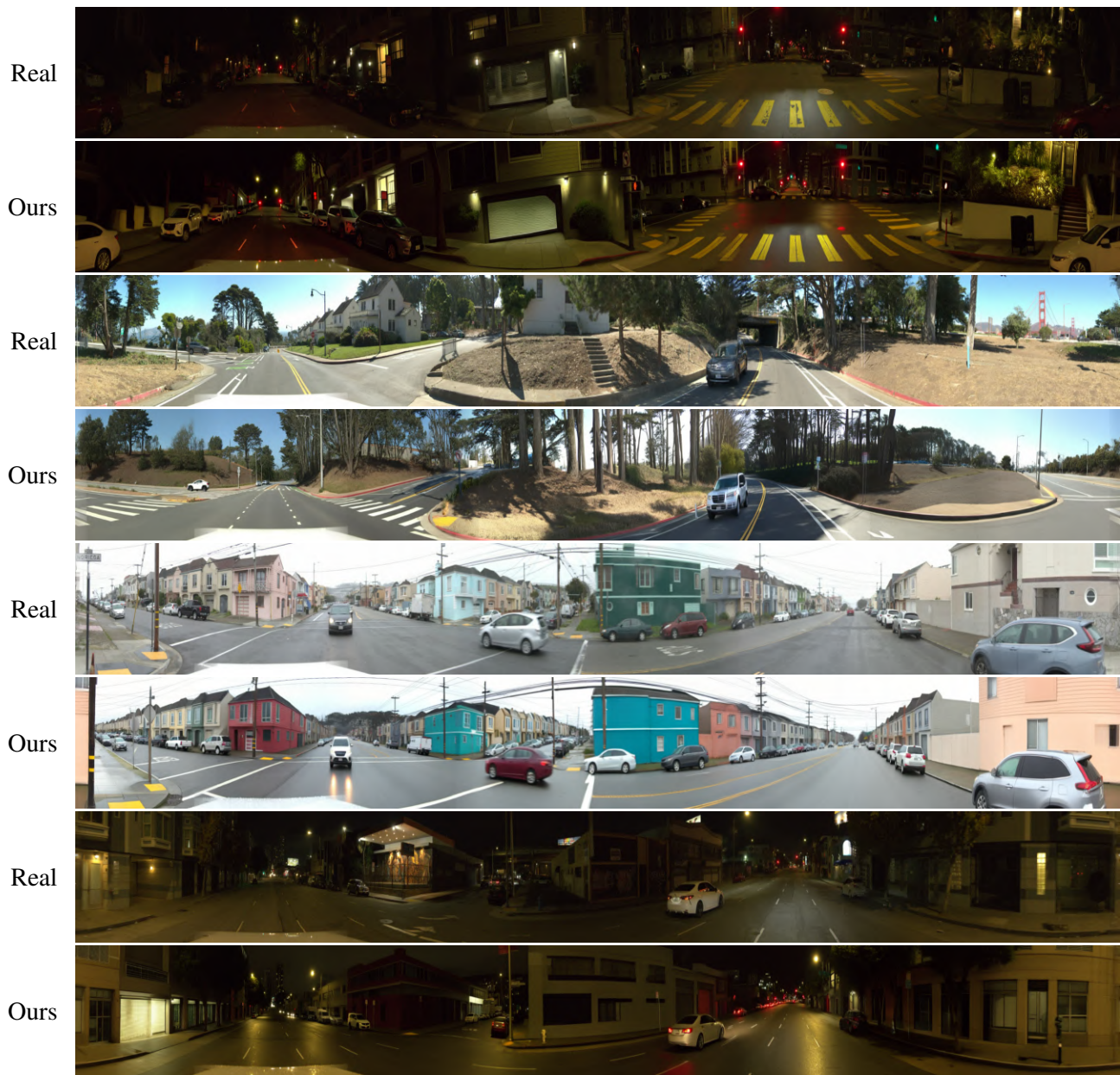


Figure 4. **Panoramic Images.** Here we show panoramic images generated by stitching individual camera views into one 360° surround view. Both real imagery as well as our method with conditioning produce no visible seams or inconsistencies beyond projection error.

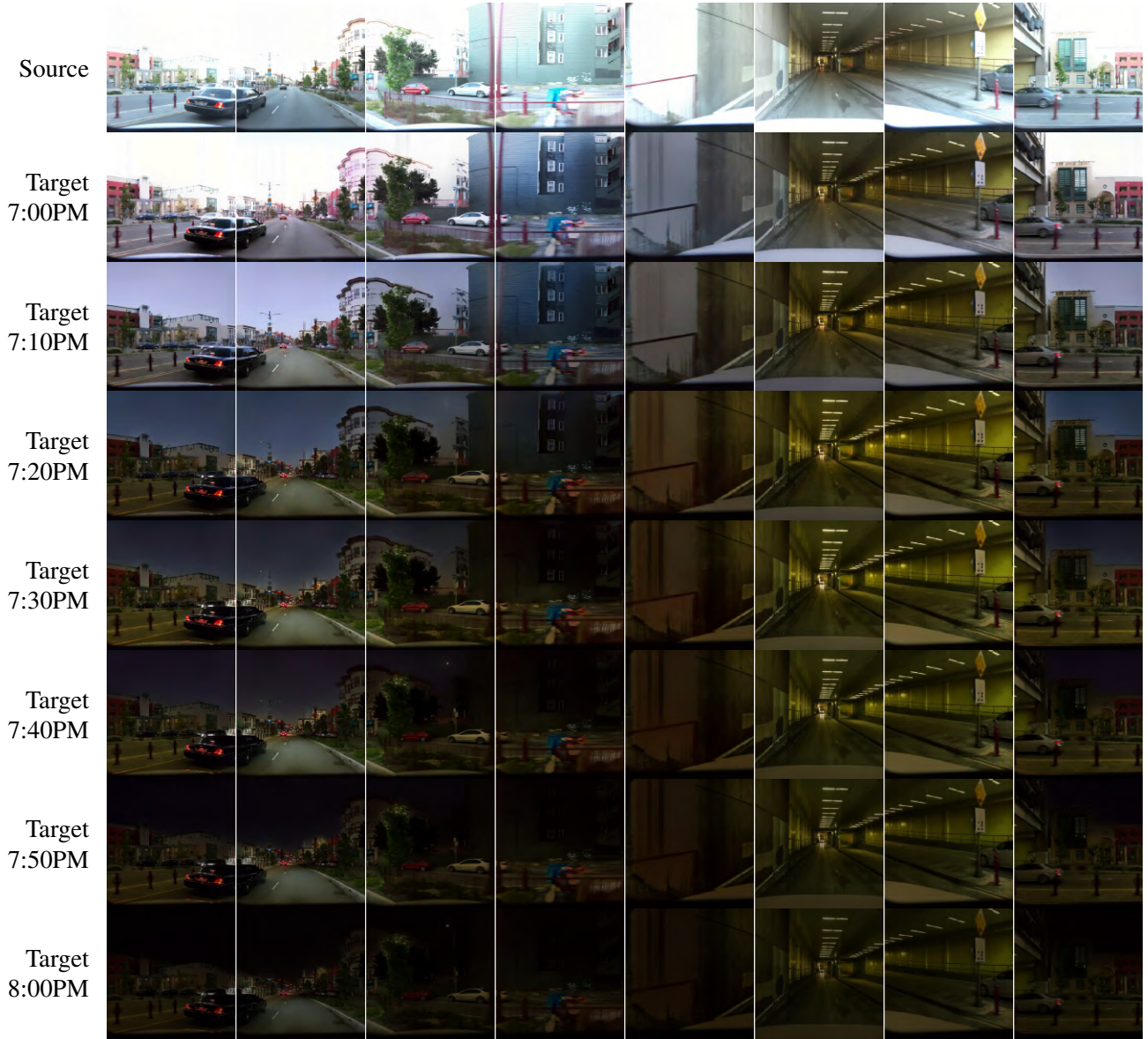


Figure 5. **More Visualizations on time of day editing.** We uniformly sample times between 7 PM and 8 PM to edit the source images (first row), effectively simulating day-to-night transitions.



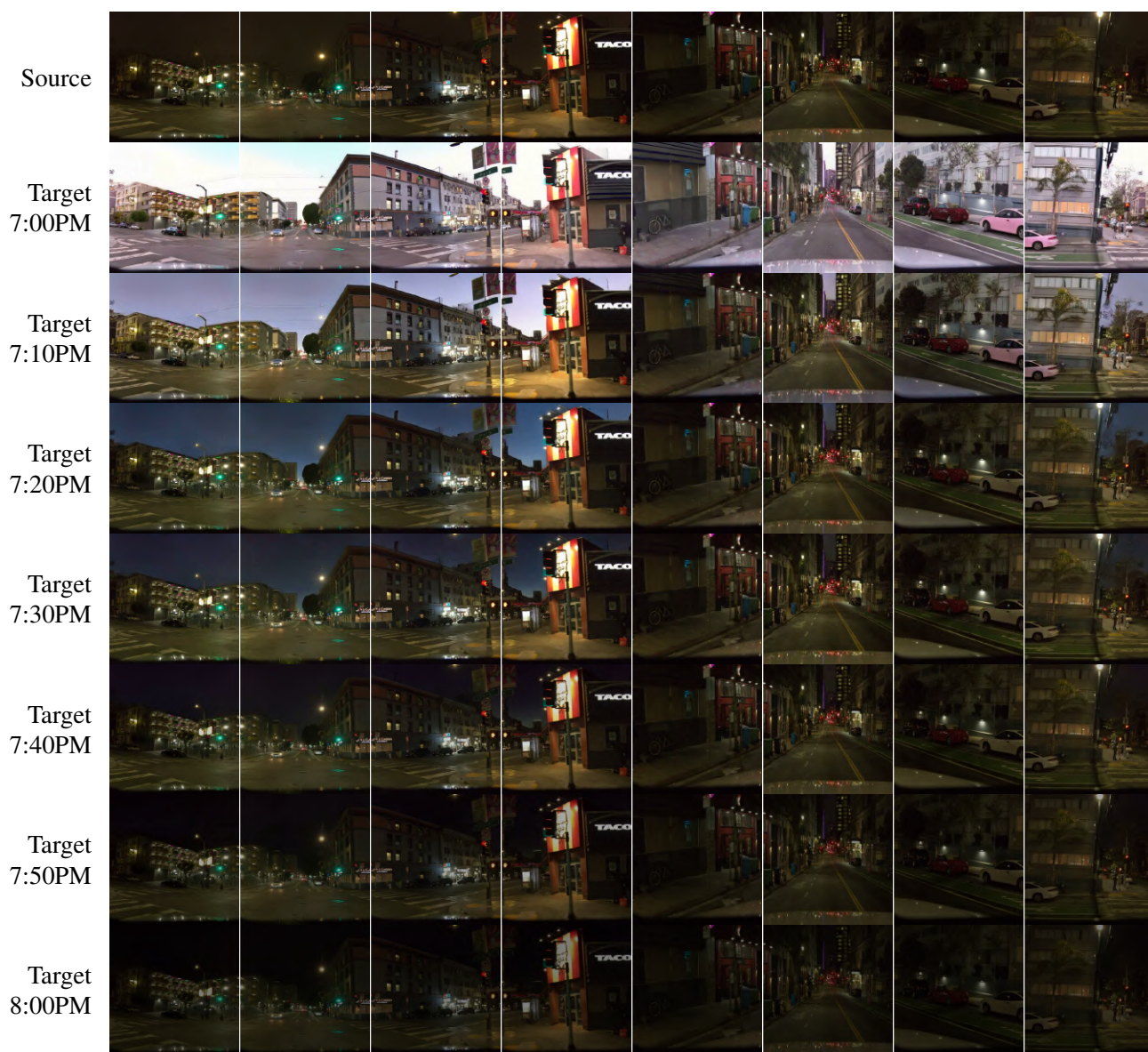


Figure 6. **More Visualizations on time of day editing.** We uniformly sample times between 7 PM and 8 PM to edit the source images (first row), effectively simulating day-to-night transitions.

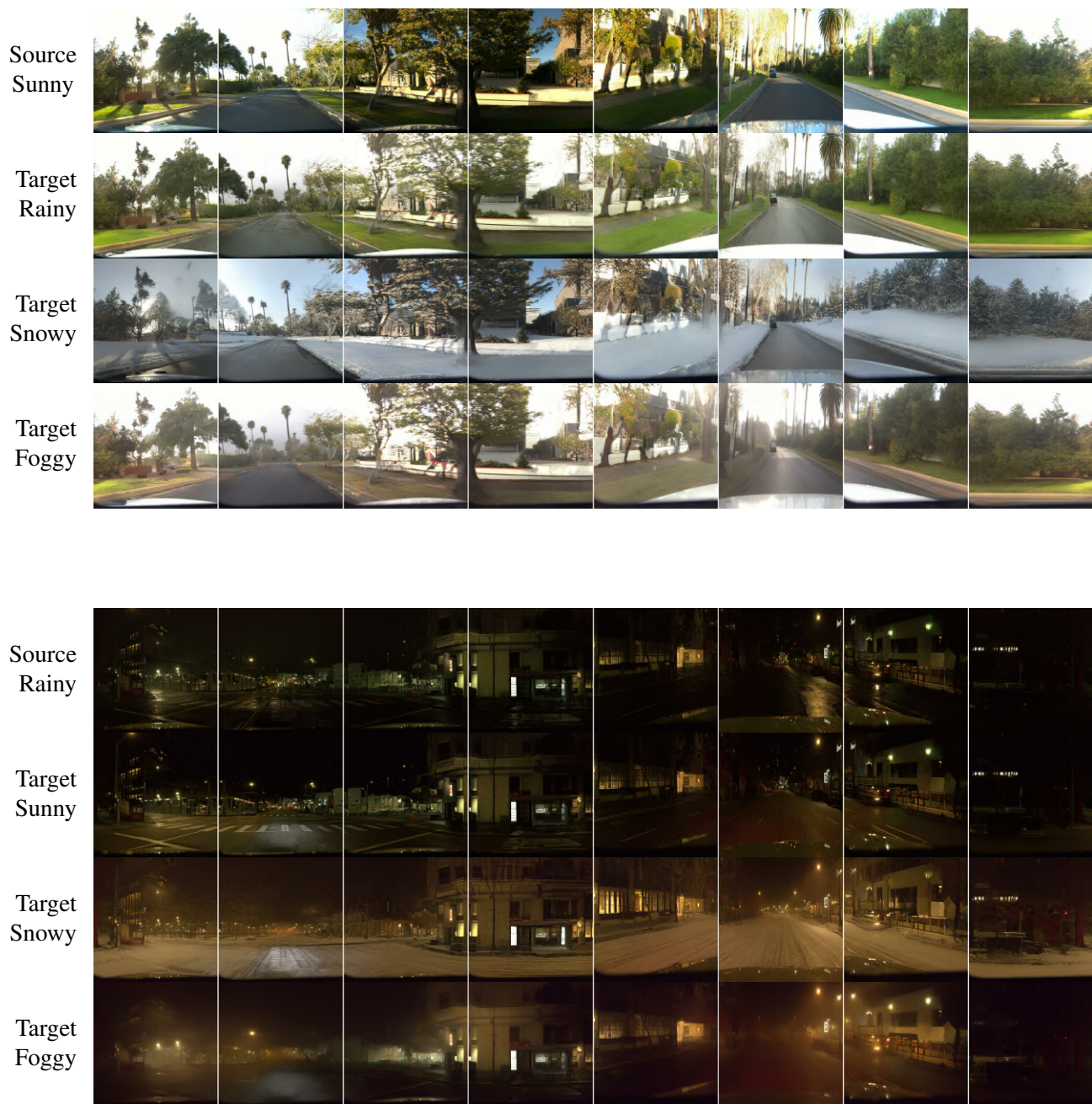


Figure 7. **More Visualizations on weather editing.** Given images captured under any specific weather (first row), our model transforms the scenes into other weather, including sunny, rainy, snowy, and foggy. The results maintain geometric consistency across all views while reflecting the intended weather effects.



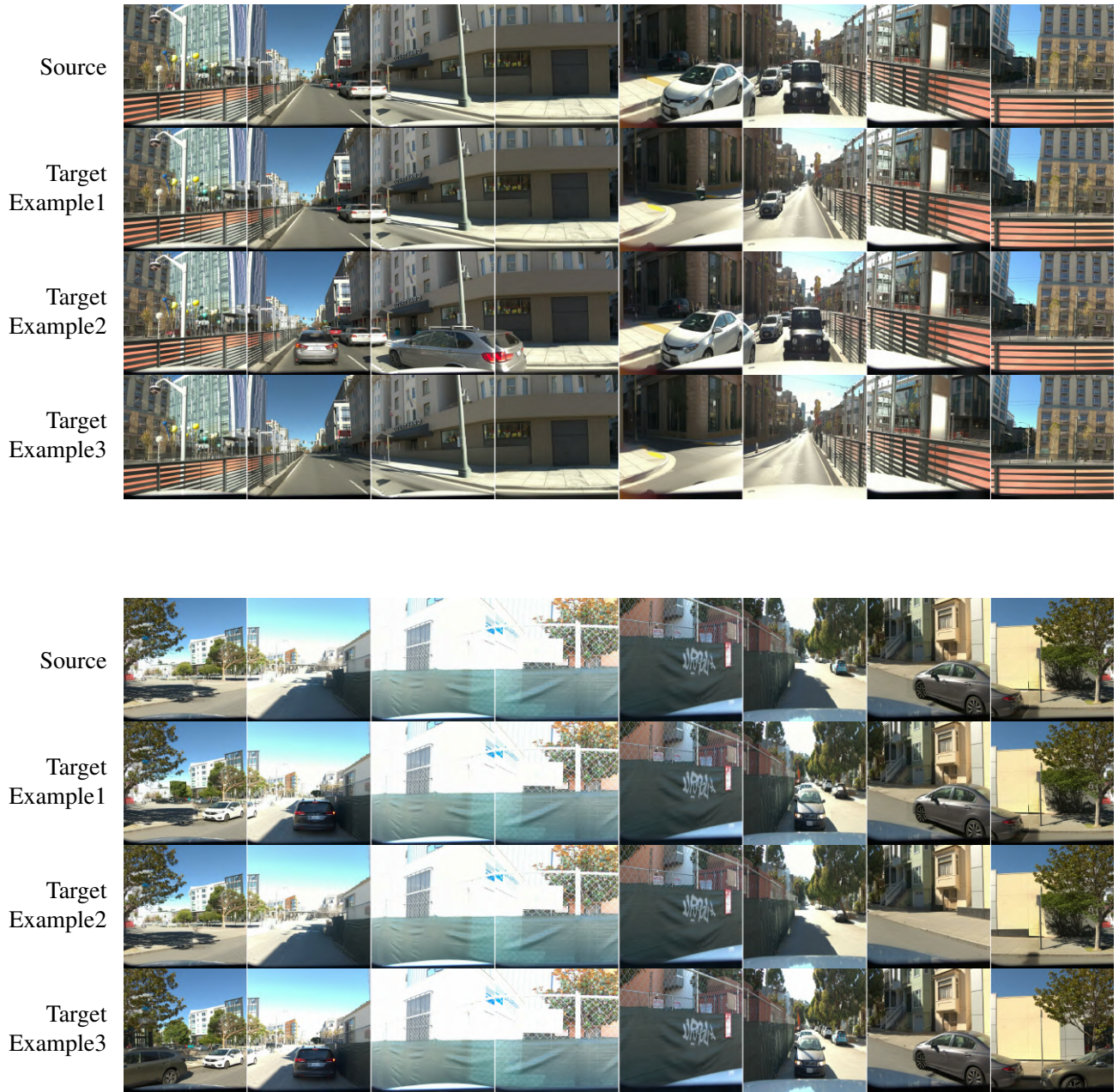


Figure 8. **More Visualizations on local editing.** SceneCrafter enables the insertion or removal of arbitrary agents within the source images. We demonstrate three editing examples conditioned on different agent boxes. SceneCrafter exhibits strong robustness across diverse agent box conditions, generating highly realistic vehicles or inpainted backgrounds.



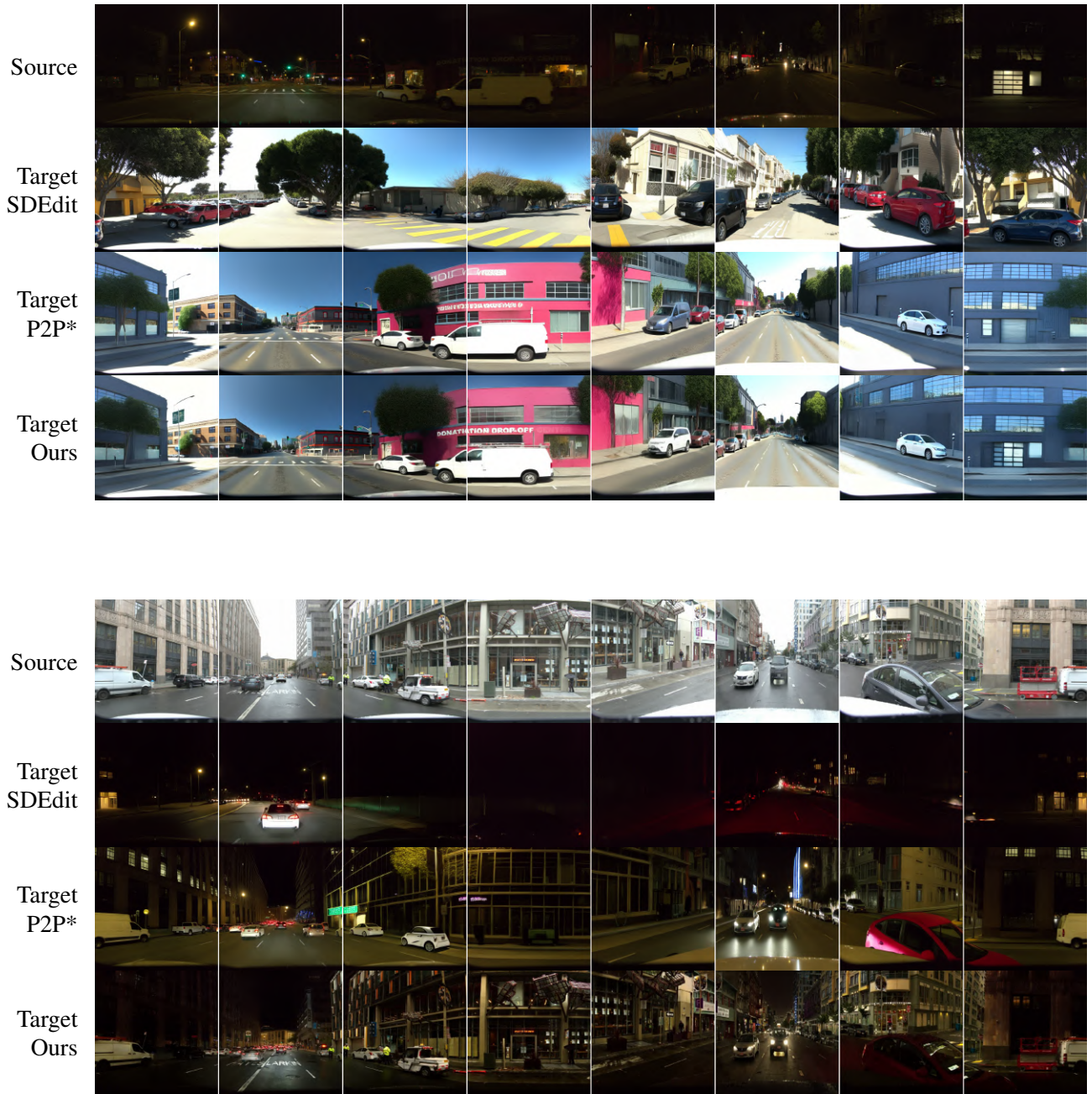


Figure 9. **Qualitative comparison with SDEdit and P2P\* baselines on time of day editing.** These scenes are included as part of our user study data. In the user study, 9 out of 11 participants rated SceneCrafter as having the best editing results for the first scene, with 2 preferred P2P\*. For the second scene, 10 out of 11 participants favored SceneCrafter and 1 preferred P2P\*.

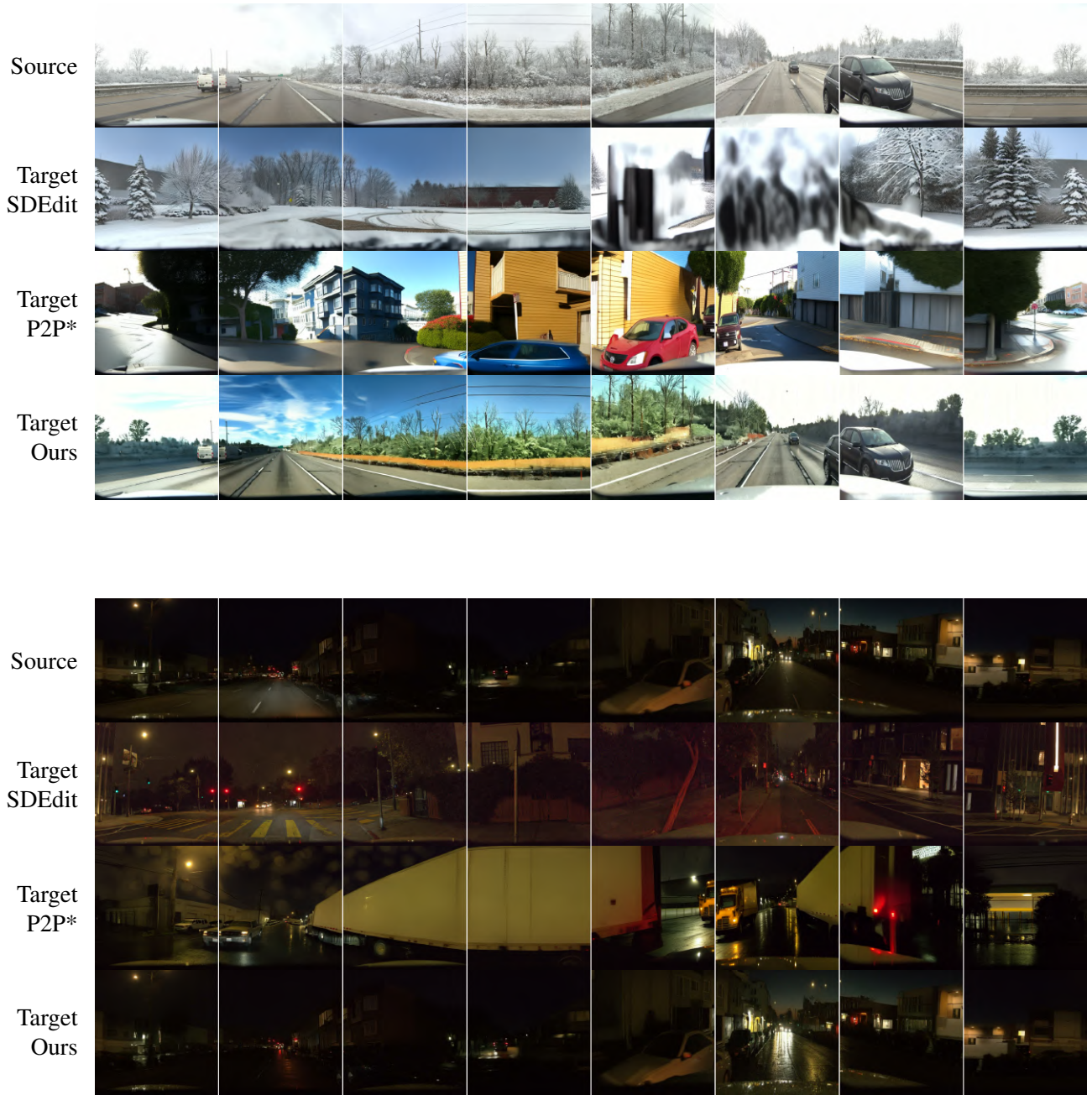


Figure 10. **Qualitative comparison with SDEdit and P2P\* baselines on weather editing.** In the user study, all participants preferred the editing results generated by our method for both scenes, demonstrating its strong alignment with human preferences.



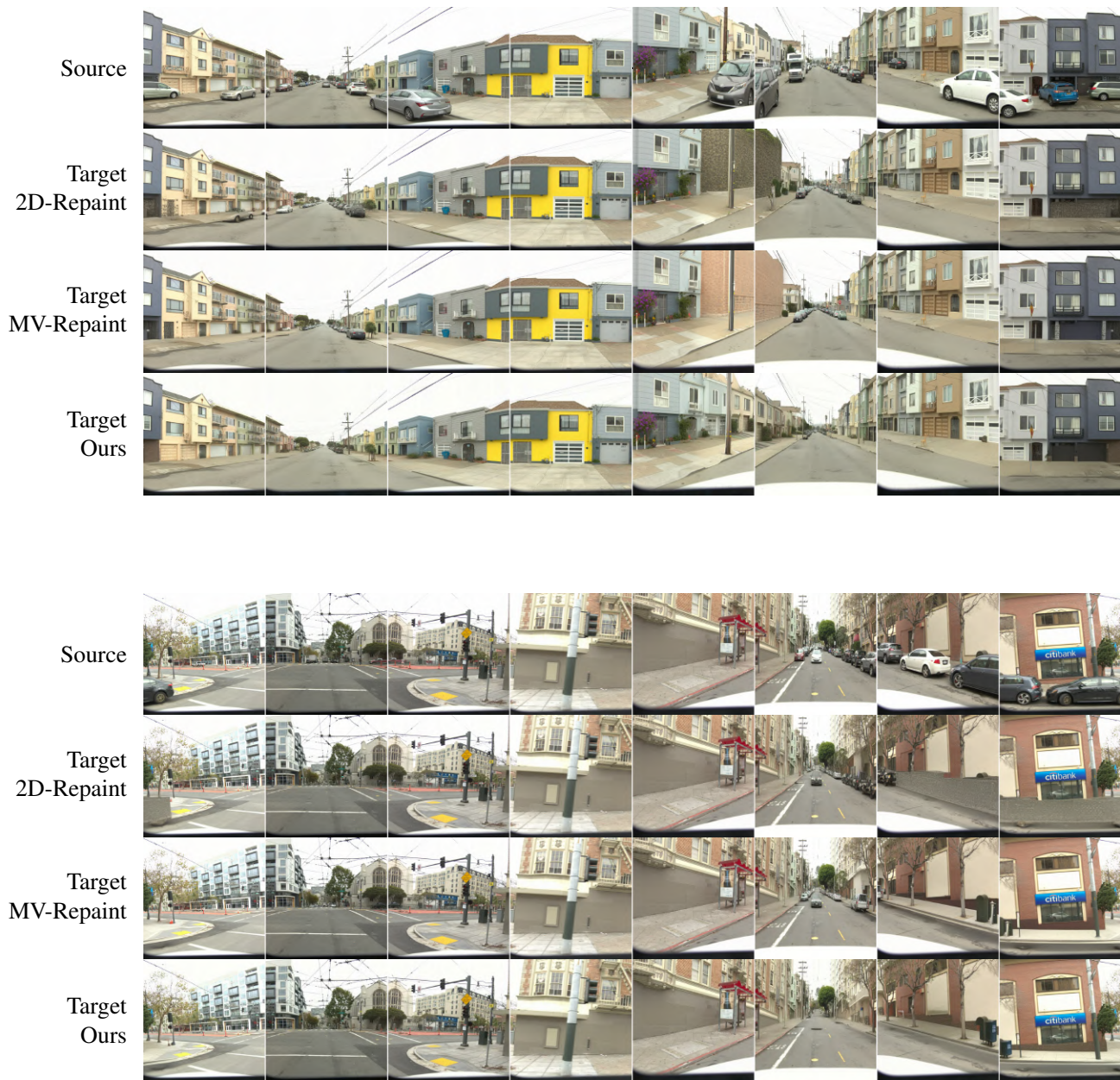


Figure 11. **Qualitative comparison with 2D-Repaint and MV-Repaint baselines on local editing.** Our method consistently achieves superior results in completely removing all vehicles from the scenes.

## References

- [1] Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul Srinivasan, Jonathan T Barron, and Ben Poole. Cat3d: Create anything in 3d with multi-view diffusion models. In *NeurIPS*, 2024.
- [2] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. In *ICLR*, 2023.
- [3] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *ICLR*, 2022.
- [4] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv:2010.02502*, 2020.
- [5] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.