Supplementary File to
"Omni-3DEdit: Generalized Versatile 3D Editing in One-Pass"

This supplementary file provides the following materials:

- **Additional Training Data Details** (referring to Sec. S1);
- **Attention Visualization** (referring to Sec. S2);
- **Additional Testing Data Details** (referring to Sec. S3).

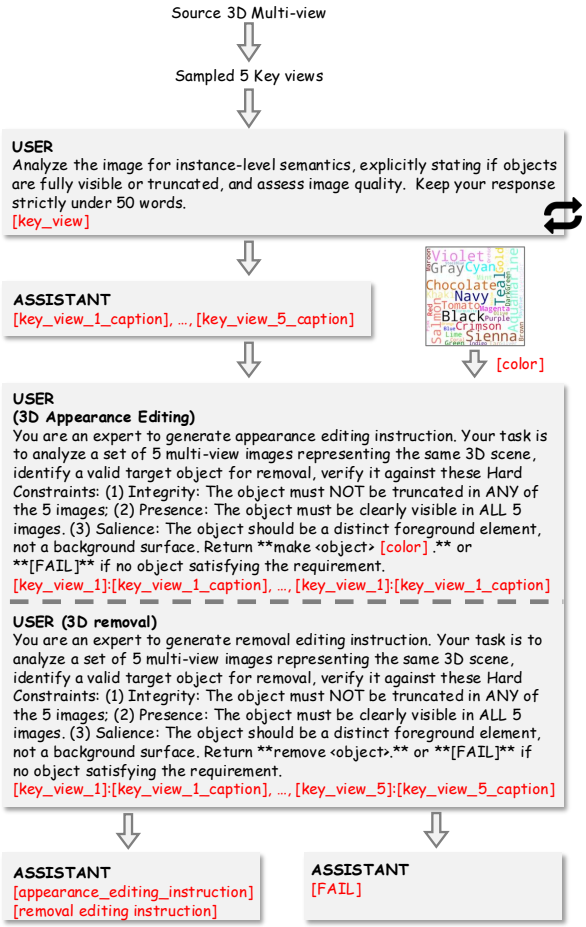

Figure S1. **System prompts for instruction generation.**

S1. Additional Training Data Details

In the main paper, we briefly described the four stages of our data curation pipeline. In this section, we provide more comprehensive implementation details.

Figure S2. **Comparison of per-view editing and consistent refinement.** Consistent refinement eliminates the minor inconsistencies that occur on per-view appearance editing.

Instruction Generation. We employ a locally deployed Qwen2.5-VL-32B [10] model together with the Gemini-2.5 Pro [3] API to generate appearance editing and object removal instructions based on the provided multi-view images. Given that processing an excessive number of views increases computational complexity and reduces inference speed, we randomly sample five views per scene as input for the model. Furthermore, to enhance multimodal reasoning capabilities, we first utilize VLM to describe the content of each image before determining the final editing instruction, which takes full advantage of the reasoning capability in the texture space. Regarding appearance editing, we observe that the model tends to generate repetitive colors ($e.g.$, frequently outputting "black"), resulting in limited diversity. To address this, we establish a pre-defined color bank and specify a randomly sampled color for each appearance editing directly, significantly improving the diversity of the generated appearance editing instructions. The complete system prompts are provided in Fig. S1.

Consistent Refinement. We observe that 3D appearance editing typically introduces minimal geometric deformation, while 3D object removal often results in backgrounds with smooth, regular geometry ($e.g.$, flat ground). Conse-

Prompt for vLLM-based Quality Filter

Role Definition
You are a strict 3D Computer Vision Quality Assurance Expert. Your mission is to evaluate the success of a 3D scene editing task by comparing source 3D multi-view inputs with edited 3D multi-view outputs, based on a specific instruction.

Evaluation Protocol
You must strictly follow this two-stage reasoning process before delivering a final verdict.

Stage 1: Per-View Independent Verification
Iterate through EACH view pair (Source View vs. Edited View) and verify:
1. **Instruction Adherence:** Does the specific view clearly reflect the requested edit? (e.g., Is the object removed? Did the color change?)
2. **Background Preservation:** Are the non-edited regions (background, other objects) identical to the source view? (Look for unwanted changes or hallucinations).
3. **Image Quality:** Is the edited view free from severe artifacts like blurring, holes, or in-painting noise?

Stage 2: Cross-View Consistency Check
Analyze the <edited 3D multi-view> images as a holistic 3D sequence:
1. **Geometric Consistency:** Does the edited object maintain a rigid 3D shape across all views? (Reject if the object shape morphs, stretches, or "drifts" between angles).
2. **Texture/Color Consistency:** Is the appearance uniform across different viewing angles? (e.g., The car shouldn't be red in View 1 but orange in View 4).

Failure Conditions (Automatic Fail)
Return `[FAIL]` immediately if ANY of the following are true:
- The edit is missing in one or more views.
- The object looks 3D-inconsistent (e.g., Janus problem, shape-shifting).
- Significant artifacts or background corruption are present.

Output Format
You must provide your analysis followed by the final decision:

Stage 1 Analysis: [Briefly confirm if each view matches the instruction and preserves background]
Stage 2 Analysis: [Briefly assess geometric and texture consistency across views]
Conclusion: [PASS] or [FAIL]

Figure S3. **System prompts for quality filter.** Firstly, evaluation is conducted on per source-edited views to ensure that the instruction has been successfully executed. Then, a subjectively multi-view consistent evaluation is conducted to filter cases with obvious inconsistency.

quently, even with per-frame editing, both appearance editing and removal tasks will introduce minor 3D inconsistencies manifesting as color variations or high-frequency texture artifacts. These discrepancies can be mitigated through techniques such as iterative 2D-3D-2D refinement [1, 2, 4, 6]. However, we eschew multi-round iterative refinement during our data generation phase, as this process is prohibitively time-consuming ($\sim$10 minutes per scene) and hinders scalable data curation. Instead, drawing inspiration from SDEdit [8], we randomly select an edited view to serve as the conditional view. For the remaining views, we add a small amount of noise and perform EDM denoising using the pre-trained SEVA model. This streamlined process requires $\sim$ 20 seconds per scene, significantly reducing the time overhead for data construction. In the inference (denosing) stage, SEVA denoises the noisy target view latent x_t with the EDM solver [5] as follows[1]:

$$x_t = x_{t+1} + \frac{\sigma_t - \sigma_{t+1}}{\sigma_{t+1}}(x_{t+1} - D_\theta(x_{t+1}, t+1; I)), \quad \text{(S1)}$$

where σ_t is the scheduled noise level at step $t \in [0, T]$. $D_\theta(x_t, t; I)$ is the estimated x_0 from x_t conditioned on the first frame I, which is defined as follows:

$$D_\theta(x_t, t; I) = x_t - (c^t_{skip} x_t + c^t_{out} F_\theta(c^t_{in} x_t, c^t_{noise}; I)), \quad \text{(S2)}$$

where c^t_{skip}, c^t_{in}, c^t_{out}, and c^t_{noise} are coefficients of the noise schedule in EDM.

Fig. S2 visualizes an example to demonstrate the effectiveness of our consistency refinement method. During the process of converting a white car to a red car by invoking Qwen-Image [9] on a per-view basis, local inconsistencies arise; for instance, the regions surrounding the two black circles in the second column are not turned red. Our consistency refinement successfully rectifies these discrepancies, yielding a highly consistent editing result.

Quality Filter. Fig. S3 provides a detailed illustration of the quality filtering process. We employ the Qwen2.5-VL-32B [10] to verify whether the edited images align with the editing instructions relative to the original images, ensuring that irrelevant regions remain largely unchanged and the image quality is not degraded by editing. Subsequently, we employ the VLM to subjectively assess the multi-view consistency, filtering out samples with significant discrepancies that cannot be rectified through consistency refinement.

Fig. S4 visualizes some examples to demonstrate data curation results. On the left, we display successful cases that pass all the four curation stages, including the final

[1]For simplicity, we omit the 2nd order correction. The noise level increases with the step t, which is opposite to the original paper [5].

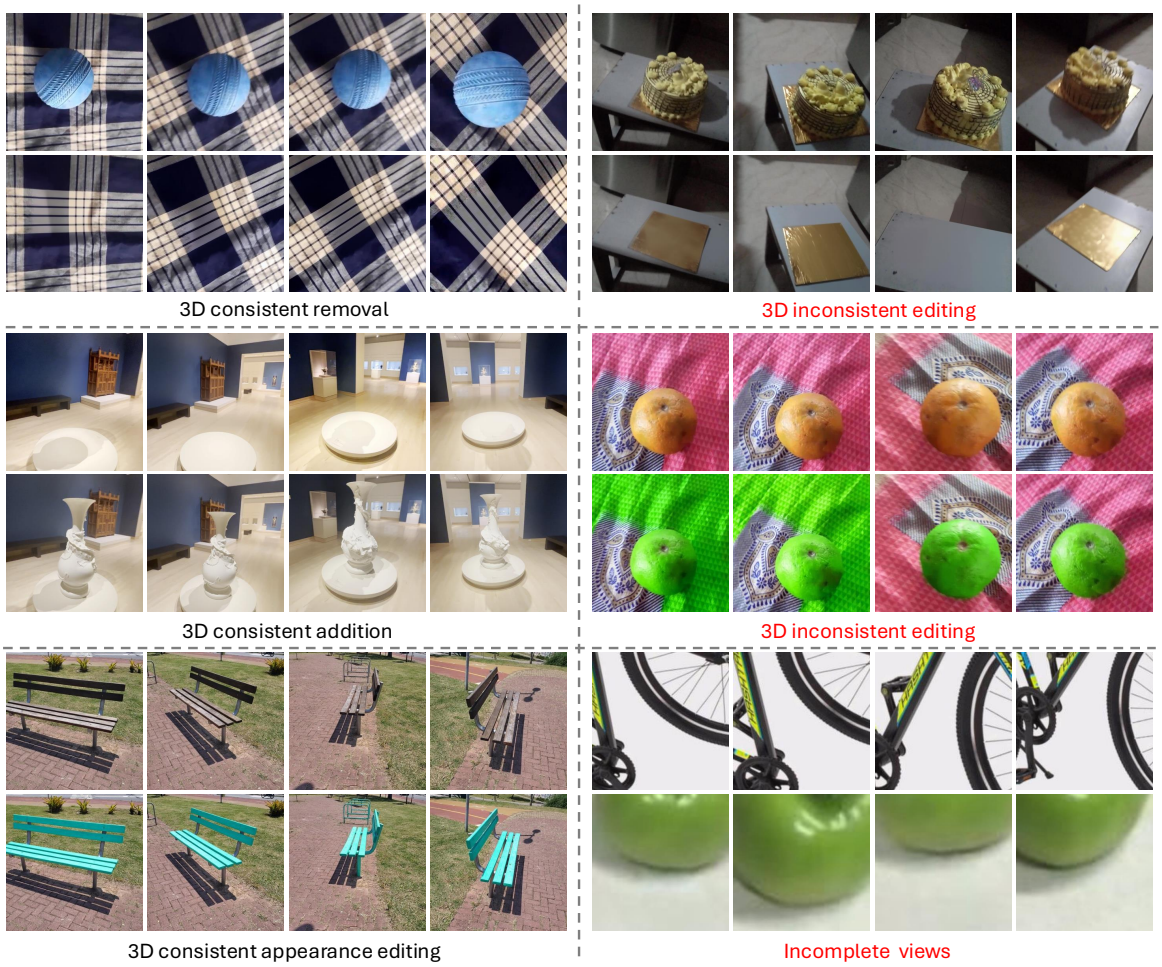

Figure S4. **Visualization of successful cases and failed cases.** We present several training data on the left and some failed cases on the right, which are caused by the inconsistent editing and incomplete views.

quality filter. On the right, we show some failure cases. In case 1, the cardboard at the base of the cake was not correctly removed in certain views. In case 2, large areas of the table-cloth were erroneously colored green during the process of changing the orange to green. These two cases exhibit significant erroneous regions that could not be corrected by consistency refinement, and they are consequently dropped during quality filtering. Additionally, case 3 is also discarded during the instruction generation phase due to dataset limitations or cropping issues, where the views do not contain the complete object and cannot be recognized by VLM.

Training Settings. We train Omni-3DEdit using the AdamW [7] optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a weight decay of 0.01. The training process utilizes a fixed learning rate of 1×10^{-4} and a batch size of 32 for a total of 4,000 training steps. Following the setting of SEVA [11] training setup, we employ the EpsWeight loss strategy but omit weighting from camera distance. Furthermore, we adopt an SNR shift of 2.4 to improve the signal-to-noise ratio scheduling. For classifier-free guidance (CFG), we apply a condition dropout rate (UCG rate) of 0.2 during

Table S1. **Training settings and hyperparameters.**

config	value
optimizer	AdamW
base learning rate	1e-4
weight decay	0.01
eps	1e-8
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$
batch size	32
learning rate schedule	ConstantLR
training steps	4K
loss weight	EpsWeight
snr shift	2.4
CFG	1.2
UCG rate	0.2

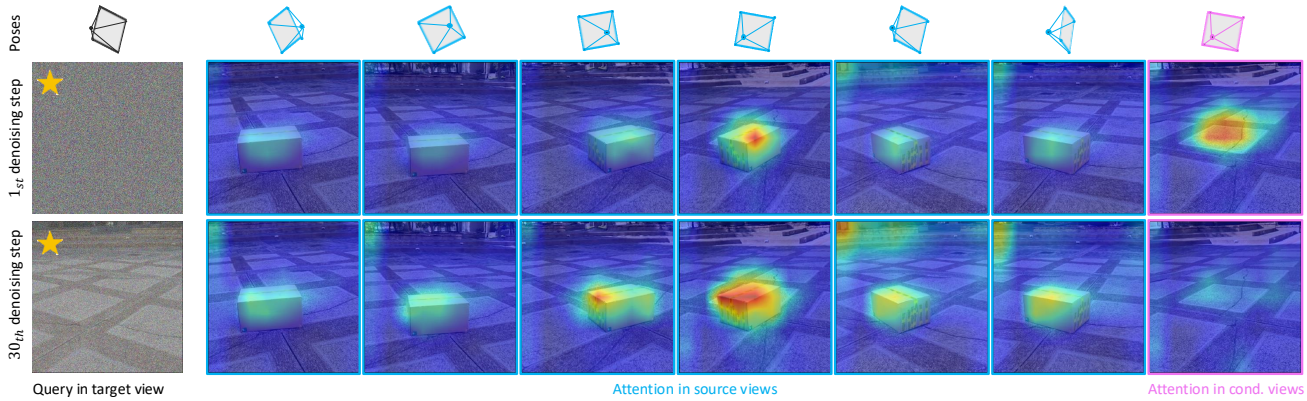

Figure S5. **Attention visualization of OmniNet Block during different denoising stages.** Omni-3DEdit can capture corresponding regions from source and conditional views based on the given camera poses. Pentagram is placed on the query regions.

Scene	Edit Instruction	Source Prompt	Target Prompt
Bear	"Turn it into a black bear."	"A bear in the park."	"A black bear in the park."
Bear	"Remove the bear."	"A bear in the park."	"A plain stone in the park."
Book	"Remove the book + add an apple."	"A table with a book."	"A table with an apple."
Spinnerf_1	"Remove the trash + add a poster."	"A trash on the ground."	"A poster."
Spinnerf_2	"Remove the box under tree."	"A box under the tree."	"A tree."
CO3D_Keyboard	"Make the keyboard green + add a bottle."	"A keyboard on the table."	"A green keyboard and a bottle on the table."
Garden	"Remove the vase."	"A vase in the garden."	"A wooden table in the garden."
Garden	"Make the table blue."	"A table."	"A blue table."
Spinnerf_7	"Remove the white object + add a plant."	"A kettle on bench."	"A plant on the bench."
Bicycle	"Make the bicycle golden."	"A bicycle on the glass."	"A golden bicycle on the glass."

Table S2. **The scene and instruction prompts in the datasets.**

training and use a guidance scale of 1.2 during inference.

S2. Attention Visualization

To probe how different views are leveraged during denoising, we visualize the attention distribution across distinct denoising stages. As illustrated in Fig. S5, we randomly select a target view and analyze the attention map corresponding to its top-left region within the multi-view attention block. At the initial denoising step (1^{st} step), the query region predominantly attends to areas exhibiting significant discrepancies between the reference and source views, $a.k.a.$, regions of boxes present and absent. Notably, the source view aligned with the reference view's camera pose garners the highest attention. This facilitates model's interpretation of the intended task (3D removal) and the target object. At the 30^{th} denoising step, the focused regions broaden beyond attending to the box's location across all source views, and it explicitly attends to regions in the original views that are geometrically coincident with the selected top-left position. This suggests that such source geometric textures are effectively preserved by Omni-3DEdit. These findings validate Omni-3DEdit's ability to utilize camera poses for resolving inter-view geometric relationships and precisely localizing relevant content from source views.

S3. Additional Test Data Details

We present more details of our manually constructed cases in Tab. S2. These cases encompass standard appearance editing instructions to align with prior methodologies, such as 'Turn it into a black bear'. Furthermore, we introduce more complex editing instructions, such as sequentially modifying an object's color before adding an additional object to the scene. As discussed in the main text, these composite instructions not only rely on the model's generalization capabilities but also necessitate high-fidelity results to support multi-round editing—a task that remains challenging for existing approaches.

Gemini Score. We employ evaluation criteria similar to those used in our quality filter, tasking Gemini-2.5 Pro [3] with assessing both the success of the multi-view editing and the maintenance of multi-view consistency. As shown in Fig. S6, we instruct Gemini to give a score from 1 to 5 by considering the editing instruction faithfulness, multi-view consistency, and visual quality comprehensively.

Prompt for Gemini Score

```
# Role
You are an expert evaluator for 3D-aware image editing tasks. Your goal is to assess the quality of edited multi-view images based on an editing instruction.

# Input Data
1. **Source Views**: Original images of the object.
2. **Edited Views**: Edited images of the object.
3. **Instruction**: The text description of the desired edit.

# Evaluation Criteria
Please analyze the results based on the following three dimensions:

1. **Editing Adherence**: Did the edit strictly follow the instruction? (e.g., correct color, object addition/removal).
2. **Multi-View Consistency (Critical)**: Does the edited views appear 3D geometrically consistent across all viewpoints?
3. **Image Quality**: Are the images clear, sharp, and free of artifacts?

# Output Format
You must structure your response strictly as follows. Provide a brief analysis for each dimension, followed by the final score.

### Analysis
1. **Editing Adherence**: [Provide your analysis here. Did the model follow the instruction?]
2. **Multi-View Consistency**: [Provide your analysis here. Are there any inconsistencies between views?]
3. **Image Quality**: [Provide your analysis here. Are there any artifacts?]

### Final Score
[Output a single integer from 1 to 5, where 1 is the worst and 5 is the best]
```

Figure S6. **System prompts for gemini score.**

References

[1] Minghao Chen, Iro Laina, and Andrea Vedaldi. Dge: Direct gaussian 3d editing by consistent multi-view editing. In *European Conference on Computer Vision*, pages 74–92. Springer, 2024.

[2] Yiwen Chen, Zilong Chen, Chi Zhang, Feng Wang, Xiaofeng Yang, Yikai Wang, Zhongang Cai, Lei Yang, Huaping Liu, and Guosheng Lin. Gaussianeditor: Swift and controllable 3d editing with gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21476–21485, 2024.

[3] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[4] Jiahua Dong and Yu-Xiong Wang. Vica-nerf: View-consistency-aware 3d editing of neural radiance fields. *Advances in Neural Information Processing Systems*, 36, 2024.

[5] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

[6] Ruihuang Li, Liyi Chen, Zhengqiang Zhang, Varun Jampani, Vishal M Patel, and Lei Zhang. Syncnoise: Geometrically consistent noise prediction for text-based 3d scene editing. *arXiv preprint arXiv:2406.17396*, 2024.

[7] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[8] 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. *arXiv preprint arXiv:2108.01073*, 2021.

[9] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report, 2025.

[10] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan.

Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[11] Jensen (Jinghao) Zhou, Hang Gao, Vikram Voleti, Aaryaman Vasishta, Chun-Han Yao, Mark Boss, Philip Torr, Christian Rupprecht, and Varun Jampani. Stable virtual camera: Generative view synthesis with diffusion models. *arXiv preprint arXiv:2503.14489*, 2025.