A. Additional Analysis

A.1. Visual Encoder Analysis

Visual encoder comparison. As summarized in Tab. A1, we benchmark a diverse collection of visual foundation models spanning two supervision paradigms. **Semantic-aligned** encoders, such as DeiT III [60], CLIP [46], and DINOv2 [43], are trained on image-level objectives to learn high-level semantic representations. For instance, CLIP aligns vision and language through contrastive learning, while DINOv2 learns robust self-supervised features from massive unlabeled datasets. In contrast, **spatial-aligned** models leverage pixel-level supervision to capture fine-grained geometry and detail. This category includes models for monocular depth estimation (Depth Anything [79, 80]), semantic segmentation (SAM [31, 47]), dense 3D reconstruction (DUSt3R [66] and MASt3R [33]), and self-supervised image reconstruction (MAE [18]). SD-VAE [48], the compact and efficient variational autoencoder from the Stable Diffusion [48] pipeline, also falls into this category.

Encoder	Supervision	Dataset	Arch.
Semantic-aligned			
DeIT III [60]	Classification	ImageNet-22k	ViT-B/16
CLIP [46]	Language	WIT-400M	ViT-B/16
DINOv2 [43]	Image feature	LVD-142M	ViT-B/14
Spatial-aligned			
Depth Any. [79]	Depth	MIX-14	ViT-B/14
Depth Any. V2 [80]	Depth	MIX-13	ViT-B/14
SAM [31]	Segmentation	SA-1B	ViT-B/16
SAM 2 [47]	Segmentation	SA-V	Hiera [49]
DUSt3R [66]	Point regression	MIX-8	ViT-L/16
MASt3R [33]	Point matching	MIX-14	ViT-L/16
MAE [18]	Pixel	ImageNet-1k	ViT-B/16
SD-VAE [48]	Pixel	OpenImages	CNN

Table A1. **Overview of investigated visual foundation models.**

As shown in Fig. 7 of the main paper and detailed in Tab. A2, our evaluation reveals key insights into foundation model effectiveness for 3D reconstruction. Spatial-aligned encoders consistently outperform semantic-aligned counterparts across all metrics, achieving higher PSNR (22.68-23.39 vs. 21.84-22.47) and lower LPIPS (0.203-0.224 vs. 0.227-0.250). This performance gap indicates that pixel-level supervision provides richer geometric priors than semantic-level training for 3D reconstruction. We also observe that model size does not correlate with reconstruction quality. For instance, while large models like MASt3R [33] (303M) achieve a top-tier PSNR of 23.29 , they do so at significant computational cost (73ms). In contrast, SD-VAE achieves comparable PSNR while delivering superior SSIM and LPIPS scores with a model nearly 9× smaller (34M) and 30% faster inference (51ms). **SD-VAE emerges as the Pareto-optimal choice, delivering the best reconstruction quality and computational efficiency among all evaluated encoders.** Cross-domain evaluation demonstrates SD-VAE's strong generalization: it not only excels on RealEstate10K but also maintains robust performance on challenging out-of-distribution datasets including outdoor ACID scenes and object-centric DTU views. Based on its superior accuracy, parameter efficiency, and cross-domain transfer, we adopt SD-VAE as our default visual encoder.

Encoder Adaptation Analysis. To assess the potential for further performance gains, we evaluate the effect of finetuning the SD-VAE encoder, with results detailed in Tab. A3. Finetuning yields consistent improvements across all scenarios, with the gains being most pronounced in cross-domain generalization. On the challenging DTU dataset, for instance, performance improves across all metrics (+0.25 PSNR, +0.011 SSIM, -0.008 LPIPS). These results demonstrate that encoder adaptation can enhance reconstruction quality, particularly for out-of-distribution scenarios. However, we choose to keep the encoder frozen to pre-

Encoder	Res.	Params.	Time (s)†	RealEstate10K			RealEstate10K → ACID			RealEstate10K → DTU			Overall		
				PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Semantic-aligned															
DeIT III [60]	256	86 M	0.053	24.82	0.832	0.156	26.26	0.789	0.186	14.44	0.481	0.393	21.84	0.701	0.245
CLIP [46]	256	86 M	0.051	25.14	0.839	0.151	26.44	0.791	0.184	14.24	0.475	0.415	21.94	0.702	0.250
DINOv2 [43]	224	86 M	0.051	25.74	0.856	0.140	26.90	0.809	0.173	14.77	0.502	0.368	22.47	0.722	0.227
Spatial-aligned															
Depth Any. [79]	224	86 M	0.056	26.05	0.863	0.136	27.20	0.817	0.168	14.78	0.491	0.368	22.68	0.724	0.224
Depth Any. V2 [80]	224	86 M	0.054	25.93	0.861	0.137	27.12	0.815	0.169	15.07	0.509	0.360	22.71	0.728	0.222
SAM [31]	256	89 M	0.065	26.39	0.869	0.130	27.79	0.831	0.158	14.95	0.521	0.343	23.04	0.740	0.210
SAM 2 [47]	256	68 M	0.054	26.12	0.865	0.135	27.59	0.826	0.163	14.88	0.505	0.361	22.86	0.732	0.220
DUSt3R [66]	256	303 M	0.073	26.56	0.873	0.129	27.73	0.833	0.158	15.21	0.527	0.342	23.17	0.744	0.210
MASt3R [33]	256	303 M	0.073	26.63	0.876	0.127	27.91	0.837	0.156	15.32	0.552	0.331	23.29	0.755	0.205
MAE [18]	256	86 M	0.051	26.65	0.874	0.127	28.04	0.838	0.154	15.24	0.526	0.344	23.31	0.746	0.208
SD-VAE [48]	256	34 M	0.051	26.50	0.872	0.129	28.16	0.842	0.153	15.21	0.567	0.326	23.29	0.760	0.203

Table A2. **Performance comparison across various visual encoders.** †: Inference time of the entire network.

Finetuning	RealEstate10K			RealEstate10K → ACID			RealEstate10K → DTU		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
✗	26.68	0.875	0.126	28.29	0.844	0.152	15.38	0.587	0.317
✓	**26.81** (+0.13)	**0.878** (+0.003)	**0.125** (-0.001)	**28.39** (+0.10)	**0.846** (+0.002)	**0.151** (-0.001)	**15.63** (+0.25)	**0.598** (+0.011)	**0.309** (-0.008)

Table A3. **Effect of finetuning the SD-VAE encoder.**

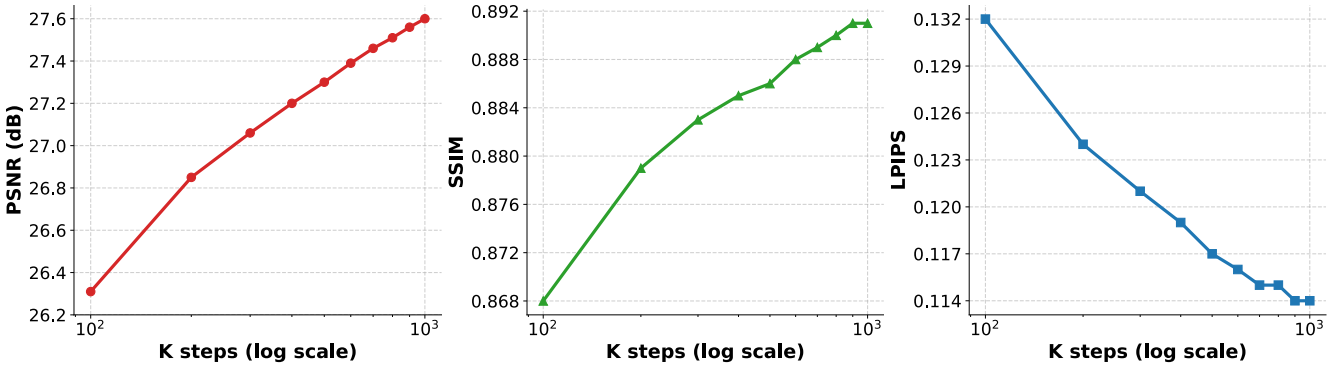

Figure A1. **Performance scaling during pre-training.** H3R exhibits power-law scaling on RealEstate10K, where performance improves linearly with the logarithm of training steps, similar to scaling patterns observed in large language models [20, 29, 61]. PSNR shows no signs of saturation even at 1 million training steps, indicating potential for further improvement.

serve pretrained representations and avoid dataset-specific overfitting. While this prioritizes generalization over peak performance, encoder adaptation remains a promising avenue for future work.

A.2. Training Dynamics

Performance scaling during pre-training. We analyze the scaling behavior of our model during pre-training on RealEstate10K, as shown in Fig. A1. The results reveal a predictable power-law scaling relationship, where performance across all three metrics improves near-linearly with the logarithm of training steps. This scaling behavior follows the well-documented power-law relationship observed in large language models [20, 29], suggesting that 3D reconstruction models benefit from similar scaling principles. Crucially, even after 1 million training steps, the PSNR curve does not yet show signs of saturation, indicating potential for further improvement. The steady, predictable scaling suggests that our 3D reconstruction model benefits from the same fundamental scaling laws that govern other foundation models, highlighting the value of increased computational investment.

Effect of Exponential Moving Average (EMA). We evaluate the impact of applying EMA to model parameters during training, with results presented in Tab. A4. EMA stabilizes training by maintaining exponentially weighted averages of model parameters, effectively reducing parameter noise and promoting convergence to more generalizable solutions. This technique has proven effective across various vision tasks, particularly in image generation [48] and restoration [67]. Our results demonstrate consistent improvements across all metrics: PSNR increases by 0.18 , SSIM by 0.002, and LPIPS decreases by 0.002. Given these consistent improvements, we adopt EMA by default in our final model.

EMA	PSNR↑	SSIM↑	LPIPS↓
✗	27.42	0.889	0.116
✓	**27.60** (+0.18)	**0.891** (+0.002)	**0.114** (-0.002)

Table A4. **Effect of EMA on RealEstate10K.**

A.3. Input Analysis

Effect of camera pose normalization. We analyze the effect of camera pose normalization, a common technique in 3D reconstruction [41, 85] where input poses are transformed to a canonical coordinate frame defined by their mean pose. As shown in Tab. A5, normalization exhibits contrasting effects across datasets. While normalization yields marginal improvements on large-scale, forward-facing scenes like RealEstate10K and ACID, it significantly degrades performance on the object-centric DTU dataset, causing a -0.39 drop in PSNR. We attribute this discrepancy stems from different camera motion characteristics. In large scenes, normalization stabilizes training by bounding the coordinate space. For object-centric settings, however, poses are already tightly clustered around the object, and normalization can amplify small but meaningful po-

Camera pose normalization	RealEstate10K PSNR↑	RealEstate10K SSIM↑	RealEstate10K LPIPS↓	RealEstate10K → ACID PSNR↑	RealEstate10K → ACID SSIM↑	RealEstate10K → ACID LPIPS↓	RealEstate10K → DTU PSNR↑	RealEstate10K → DTU SSIM↑	RealEstate10K → DTU LPIPS↓
✗	27.05	0.882	0.121	28.55	0.849	0.145	**15.49**	**0.587**	**0.314**
✓	**27.08** (+0.03)	**0.883** (+0.001)	**0.120** (-0.001)	**28.61** (+0.06)	**0.852** (+0.003)	**0.144** (-0.001)	15.10 (-0.39)	0.565 (-0.022)	0.319 (+0.005)

Table A5. **Effect of camera pose normalization.** While camera normalization yields modest improvements on scene-level datasets, it significantly degrades generalization performance on the object-level DTU dataset.

Encoder	Res.	RealEstate10K PSNR↑	RealEstate10K SSIM↑	RealEstate10K LPIPS↓	RealEstate10K → ACID PSNR↑	RealEstate10K → ACID SSIM↑	RealEstate10K → ACID LPIPS↓	RealEstate10K → DTU PSNR↑	RealEstate10K → DTU SSIM↑	RealEstate10K → DTU LPIPS↓
DINOv2 [43]	224	25.74	0.856	0.140	26.90	0.809	0.173	14.77	0.502	0.368
Depth Any. [79]	224	26.05	0.863	0.136	27.20	0.817	0.168	14.78	0.491	0.368
Depth Any. V2 [80]	224	25.93	0.861	0.137	27.12	0.815	0.169	15.07	0.509	0.360
CLIP [46]	224	24.90	0.832	0.155	26.20	0.782	0.189	13.91	0.448	0.437
	256	25.14	0.839	0.151	26.44	0.791	0.184	14.24	0.475	0.415
DUSt3R [66]	224	26.44	0.871	0.130	27.63	0.830	0.160	15.24	0.530	0.346
	256	26.56	0.873	0.129	27.73	0.833	0.158	15.21	0.527	0.342
MASt3R [33]	224	26.46	0.872	0.130	27.79	0.834	0.158	15.32	0.541	0.335
	256	26.63	0.876	0.127	27.91	0.837	0.156	15.32	0.552	0.331

Table A6. **Effect of input resolution.**

Method	RealEstate10K (2 views) PSNR↑	RealEstate10K (2 views) SSIM↑	RealEstate10K (2 views) LPIPS↓	RealEstate10K (4 views) PSNR↑	RealEstate10K (4 views) SSIM↑	RealEstate10K (4 views) LPIPS↓	RealEstate10K (6 views) PSNR↑	RealEstate10K (6 views) SSIM↑	RealEstate10K (6 views) LPIPS↓	RealEstate10K (8 views) PSNR↑	RealEstate10K (8 views) SSIM↑	RealEstate10K (8 views) LPIPS↓
MVSplat	26.36	0.868	0.129	22.20	0.820	0.118	20.95	0.803	0.203	20.35	0.792	0.214
H3R-α (Ours)	**27.46** (+1.10)	**0.889** (+0.021)	**0.115** (-0.014)	**29.28** (+7.08)	**0.920** (+0.100)	**0.090** (-0.028)	**29.97** (+9.02)	**0.930** (+0.127)	**0.083** (-0.120)	**30.24** (+9.89)	**0.934** (+0.142)	**0.080** (-0.134)

Table A7. **Performance comparison across varying number of input views.**

sitional variations, disrupting the geometric cues essential for precise reconstruction. Given our emphasis on cross-domain generalization, we omit pose normalization in our final model.

Effect of input resolution. We investigate the impact of input resolution on reconstruction performance by comparing models trained at 224×224 versus 256×256 resolution, as shown in Tab. A6. Higher resolution yields only modest improvements: +0.17 PSNR for MASt3R, +0.12 PSNR for DUSt3R, and +0.24 PSNR for CLIP on RealEstate10K. The results suggest that input resolution is less critical for reconstruction quality compared to architectural choice and training method. We adopt 256×256 as our default resolution primarily for consistency with standard practice [4, 7].

Effect of number of input views. We evaluate our model's adaptability with respect to the number of input views, comparing H3R-α against MVSplat on RealEstate10K. As detailed in Tab. A7, the two methods exhibit opposite scaling behaviors. H3R-α demonstrates robustness with the number of input views, with PSNR climbing from 27.46 (2 views) to 30.24 (8 views). Conversely, MVSplat degrades as more views are added, dropping from 22.20 (4 views) to 20.35 (8 views). These results highlight our framework's effective multi-view aggregation capabilities.

Effect of view overlap. We evaluate our method's robustness to varying view overlap, comparing H3R against MVSplat in Tab. A8. H3R consistently outperforms the baseline across all tested overlap ranges. The performance gain is most pronounced under challenging low-overlap scenarios [0.60,0.65], where our method achieves a substantial 1.73 dB improvement in PSNR. As the view overlap increases, providing richer geometric cues, this advantage gradually narrows to 1.07 in high-overlap conditions [0.95,1.00]. This trend demonstrates H3R's robustness across varying geometric constraints, with particularly strong performance in challenging scenarios where existing methods often fail.

A.4. Further Comparison

Comparison with GS-LRM. We compare our H3R-β against the state-of-the-art GS-LRM [85], as shown in Tab. A9. Our method demonstrates remarkable efficiency, utilizing only 30% of the trainable parameters (91M vs. 300M) and 20% of the training cost (37 vs. 192 GPU-days). This efficiency does not compromise quality; in fact,

Overlap	MVSplat			H3R (ours)			Improvement		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	ΔPSNR↑	ΔSSIM↑	ΔLPIPS↓
$[0.60, 0.65)$	24.41	0.841	0.151	26.14	0.877	0.127	+1.73	+0.036	-0.024
$[0.65, 0.70)$	25.00	0.852	0.144	26.44	0.882	0.124	+1.44	+0.030	-0.020
$[0.70, 0.75)$	25.68	0.862	0.133	27.00	0.887	0.116	+1.32	+0.025	-0.017
$[0.75, 0.80)$	25.74	0.864	0.131	27.01	0.888	0.115	+1.27	+0.024	-0.016
$[0.80, 0.85)$	26.15	0.871	0.129	27.33	0.893	0.113	+1.18	+0.022	-0.016
$[0.85, 0.90)$	26.18	0.872	0.129	27.39	0.894	0.114	+1.21	+0.022	-0.015
$[0.90, 0.95)$	25.92	0.863	0.134	27.12	0.886	0.118	+1.20	+0.023	-0.016
$[0.95, 1.00)$	27.86	0.881	0.117	28.93	0.899	0.105	+1.07	+0.018	-0.012

Table A8. **Performance comparison across varying overlaps.**

our method achieves superior perceptual quality with better SSIM (0.897 vs. 0.892) and LPIPS (0.110 vs. 0.114) scores. This combination of efficiency and quality makes our model more practical for widespread adoption.

Method	#Trainable Params.	#GPU days 4090 / A100	RealEstate10K PSNR↑	SSIM↑	LPIPS↓
GS-LRM	300M	0 / 192	**28.10**	0.892	0.114
H3R-β	91M	30 / 7	28.03	**0.897**	**0.110**

Table A9. **Comparison with GS-LRM [85].** We achieve comparable performance with **30%** trainable parameters and **20%** training cost.

B. 3D Gaussian Parameterization

3D Gaussians provide an explicit and flexible representation for 3D scenes, and their parameterization is crucial for model performance. To ensure reproducibility, we provide detailed specifications for each Gaussian parameter. The specific configurations for each parameter are provided in Tab. B1.

Parameter	Activation	Channel
Center	None	3
Scale	Sigmoid	3
Rotation	L2 norm	4
Opacity	Sigmoid	1
Color	ReLU	3

Table B1. **3D Gaussian parameterization.**

Ray distance. We uniformly sample 128 depth hypotheses $\{d_i\}_{i=1}^{128}$ in inverse depth space between the near and far planes. The model output is transformed into a probability distribution ω over these hypotheses using softmax activation and the ray distance t is computed as the weighted sum:

$$\omega = \text{softmax}(\mathbf{G}_{\text{distance}}), \tag{14}$$

$$t = \sum_{i=1}^{128} \omega_i \cdot d_i, \tag{15}$$

where $\mathbf{G}_{\text{distance}}$ is the depth head output. The near and far planes are dataset-specific. For scene-level datasets such as RealEstate10K and ACID, we set the planes to 1 and 100, respectively. For the object-level DTU dataset, we adopt the configuration from MVSplat [7] with near and far planes of 2.215 and 4.525, respectively. Our pilot experiments demonstrate that employing multiple hypotheses yields slight performance improvements over the two-hypothesis approach used in GS-LRM [85].

Scale. Following pixelSplat [4], we parameterize Gaussian scales in image space instead of world space. The scale head output $\mathbf{G}_{\text{scale}}$ is mapped to a predefined scale range in pixel space $[s_{\text{min}}, s_{\text{max}}]$ using sigmoid activation:

$$\omega = \sigma(\mathbf{G}_{\text{scale}}), \tag{16}$$

$$s_{\text{pixel}} = (1 - \omega)s_{\text{min}} + \omega s_{\text{max}}. \tag{17}$$

We then compute the world-space scale s_{world} as:

$$s_{\text{world}} = s_{\text{pixel}} \cdot p_{\text{world}} \cdot t, \tag{18}$$

where p_{world} is the pixel size in world space. This scaling approach maintains proper perspective by ensuring that distant Gaussians have appropriate screen-space sizes. For both scene-level and object-level datasets, we set the pixel-space scale range to $s_{\text{min}} = 0.5$ and $s_{\text{max}} = 15.0$.

Opacity. The opacity of each Gaussian is transformed to the range $(0, 1)$ using sigmoid activation.

Rotation. As in [85], we predict unnormalized quaternions and apply L2-normalization to obtain unit quaternions.

RGB. For simplicity, we predict the zero-order Spherical Harmonics (SH) coefficients. We apply ReLU activation to ensure non-negative color values.

Center. Rather than predicting the Gaussian center directly, we derive it from the ray distance and camera parameters.

For each pixel, the ray origin ray_o and direction ray_d are computed from the known camera parameters. The Gaussian center xyz is then determined by:

$$xyz = \text{ray}_o + t \cdot \text{ray}_d. \tag{19}$$

C. Implementation Details

C.1. Datasets

We train and evaluate our method on two large-scale datasets: RealEstate10K [88] and ACID [35]. RealEstate10K contains 67,477 training scenes and 7,289 test scenes of diverse indoor and outdoor environments from YouTube, while ACID comprises 11,075 training scenes and 1,972 test scenes of natural landscapes captured by drones. For both datasets, camera poses are estimated using Structure-from-Motion (SfM) [51]. We follow the official train/test splits and evaluation protocol of pixelSplat [4], where two input context views are used to synthesize three novel views for each test scene. To evaluate cross-dataset generalization, we perform zero-shot evaluation on the object-centric DTU dataset [24]. Following the setup in [7], we evaluate on 16 validation scenes, rendering four novel views for each scene. We evaluate rendering quality with three standard metrics: PSNR, SSIM [70], and LPIPS [86].

C.2. Model Details

Our camera-aware Transformer comprises 12 layers with hidden dimensions of 512 and MLP hidden dimensions of 1536, employing Pre-LayerNorm, QK-Norm [19], and SwiGLU [52] activation.

C.3. Training Details

We initialize the visual encoder from publicly available checkpoints and freeze its parameters throughout training. Unless otherwise specified, we adopt the hyperparameters from MVSplat [7]. Following [4, 7], we apply random horizontal flipping for data augmentation. The pixel gradient loss weight is empirically set to 1.0. We employ Bfloat16 mixed-precision training and cache visual features to accelerate training. Detailed training settings for the RealEstate10K and ACID datasets are provided in Tab. D1 and Tab. D2, respectively.

H3R: Pre-training (256×256, 2 views) We pre-train the H3R model with two context views at 256×256 resolution. The model is trained for 1M steps on RealEstate10K and 400K steps on ACID. Training requires seven days and three days, respectively, on 4 NVIDIA RTX 4090 GPUs. Following pixelSplat [4], the maximum frame distance is linearly increased from 25 to 45 over the initial 150K steps and then held constant.

H3R-α: Multi-view (256×256, 2-8 views) We finetune base model with random 2-8 context views at 256×256 resolution. The model is trained for 30K steps on RealEstate10K and 90K steps on ACID. Training takes about 15 hours on 4 NVIDIA A6000 GPUs and 28 hours on 8 NVIDIA RTX4090 GPUs. During finetuning, we randomly include target camera poses as input with probability 0.5 for each training sample.

H3R-β: High-resolution (512×512, 2 views) We finetune base model with two context views at 512×512 resolution. The model is trained for 80K steps on RealEstate10K and 20K steps on ACID. Training takes about 42 and 11 hours on 4 NVIDIA A100 GPUs, respectively. The maximum frame distance between context views is fixed at 45.

D. Additional Visualizations

We present additional qualitative results on RealEstate10K in Figs. D1 to D3. Collectively, these studies illustrate that incorporating target pose, more input views, and higher resolution inputs directly contributes to substantial gains in structural integrity, detail preservation, and overall photorealism.

config	H3R	H3R-α	H3R-β
peak learning rate	1e-4	5e-5	5e-5
min learning rate	5e-5	-	-
warm-up steps	3,000	0	0
LR schedule	cosine decay 150k steps, then constant	constant	constant
optimizer	Adam		
betas	(0.9, 0.999)		
weight decay	0		
gradient clip	0.5		
total batch size	16		
EMA decay	0.999		
trainable parameters	90.9 M		
training steps	1,000,000	30,000	80,000
GPU	4 × RTX 4090	4 × A100-80GB	4 × A100-80GB
training time	7.4 days	15 hours	42 hours

Table D1. **Training settings for RealEstate10K.**

config	H3R	H3R-α	H3R-β
peak learning rate	1e-4	5e-5	5e-5
min learning rate	5e-5	-	-
warm-up steps	3,000	0	0
LR schedule	cosine decay 150k steps, then constant	constant	constant
optimizer	Adam		
betas	(0.9, 0.999)		
weight decay	0		
gradient clip	0.5		
total batch size	16		
EMA decay	0.999		
trainable parameters	90.9 M		
training steps	400,000	90,000	20,000
GPU	4 × RTX 4090	8 × RTX 4090	4 × A100-80GB
training time	3 days	28 hours	11 hours

Table D2. **Training settings for ACID**.

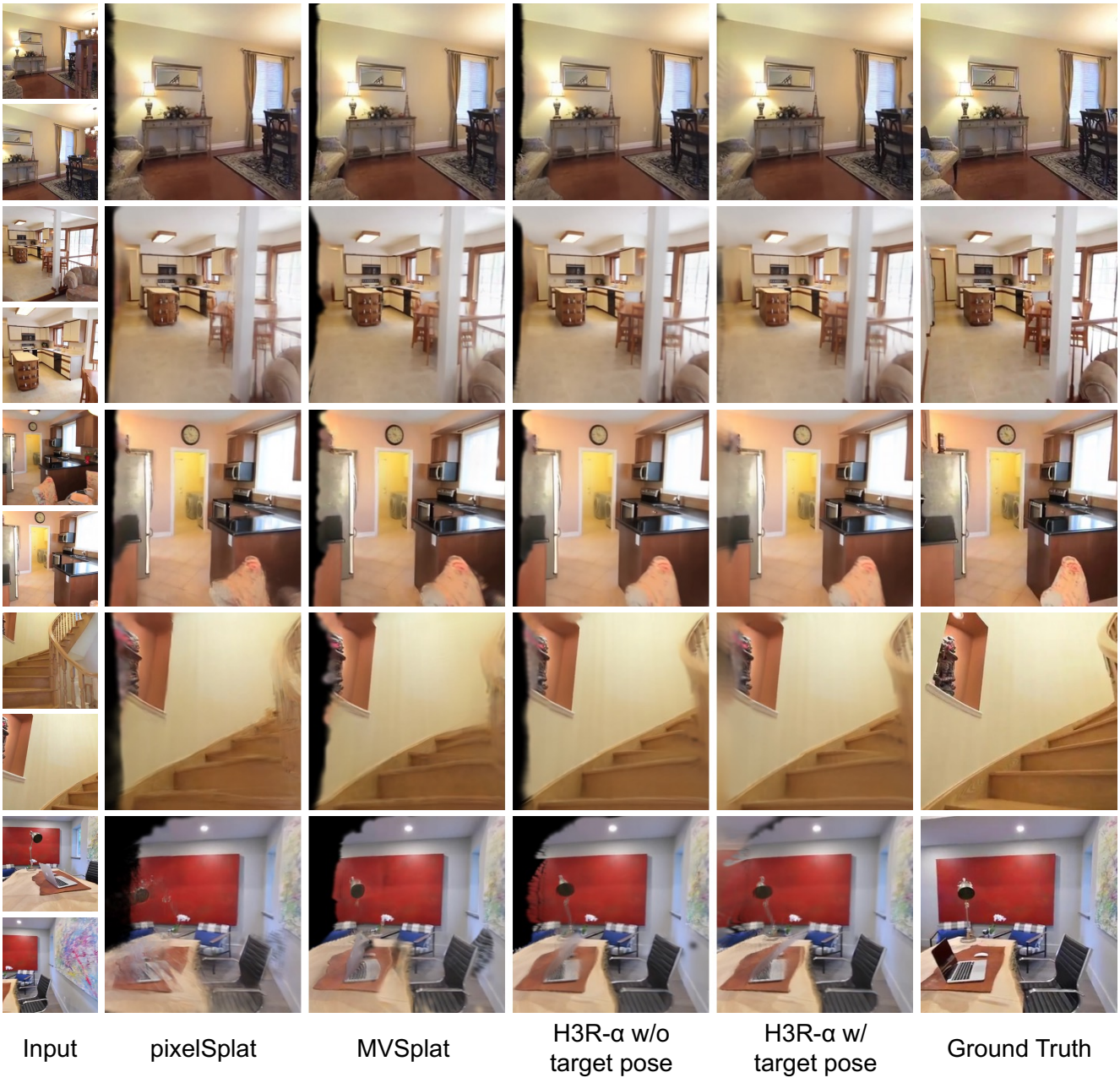

Figure D1. **Impact of target camera poses on RealEstate10K.** Our method (H3R-α) leverages target camera poses to generate more complete and view-aligned Gaussian splats, improving geometric coherence while mitigating artifacts, particularly in unobserved regions.

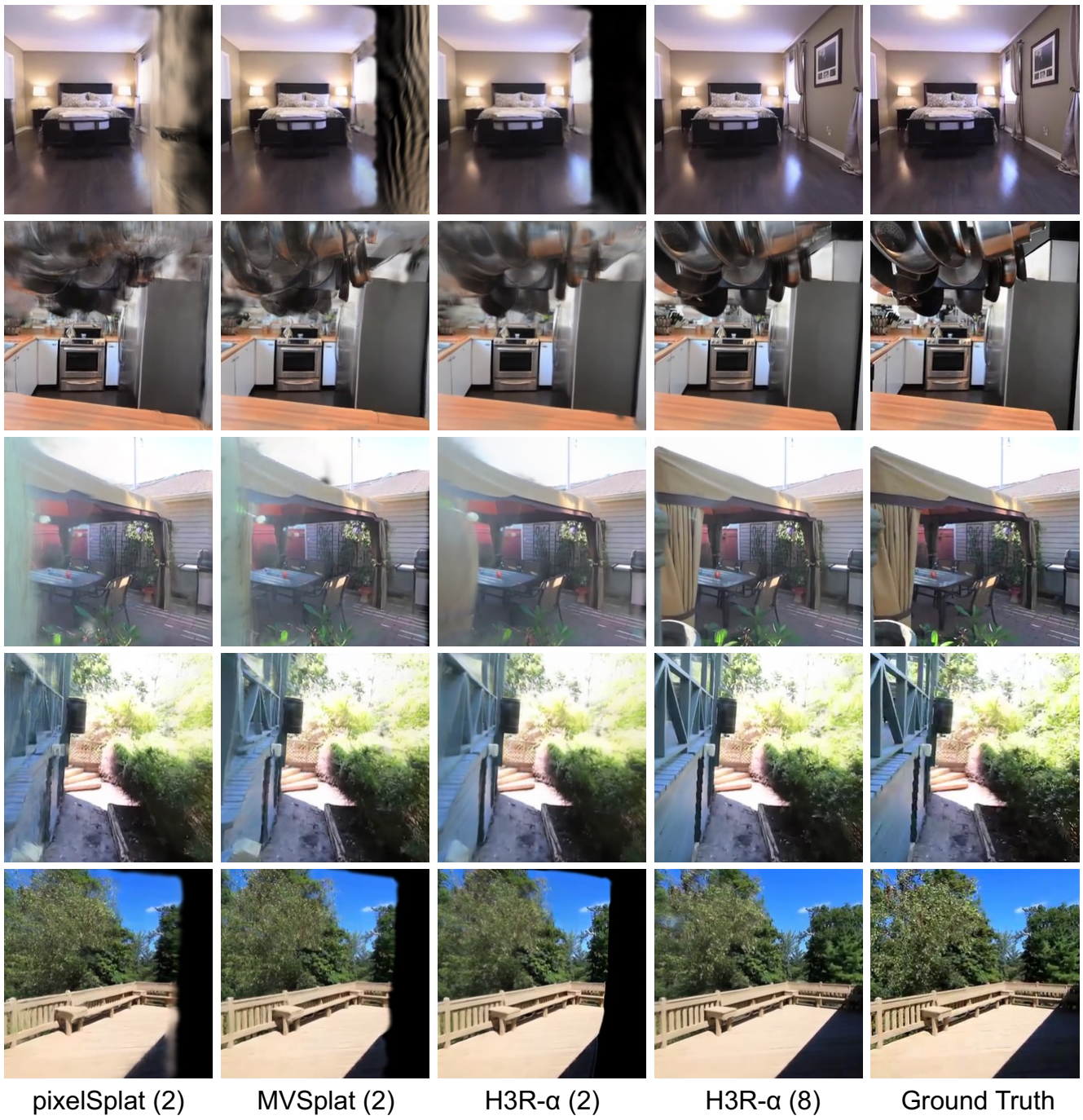

Figure D2. **Impact of the number of input views on RealEstate10K.** Increasing input views from two to eight enhances geometric completeness and visual fidelity, particularly for scene boundaries and specular surfaces.

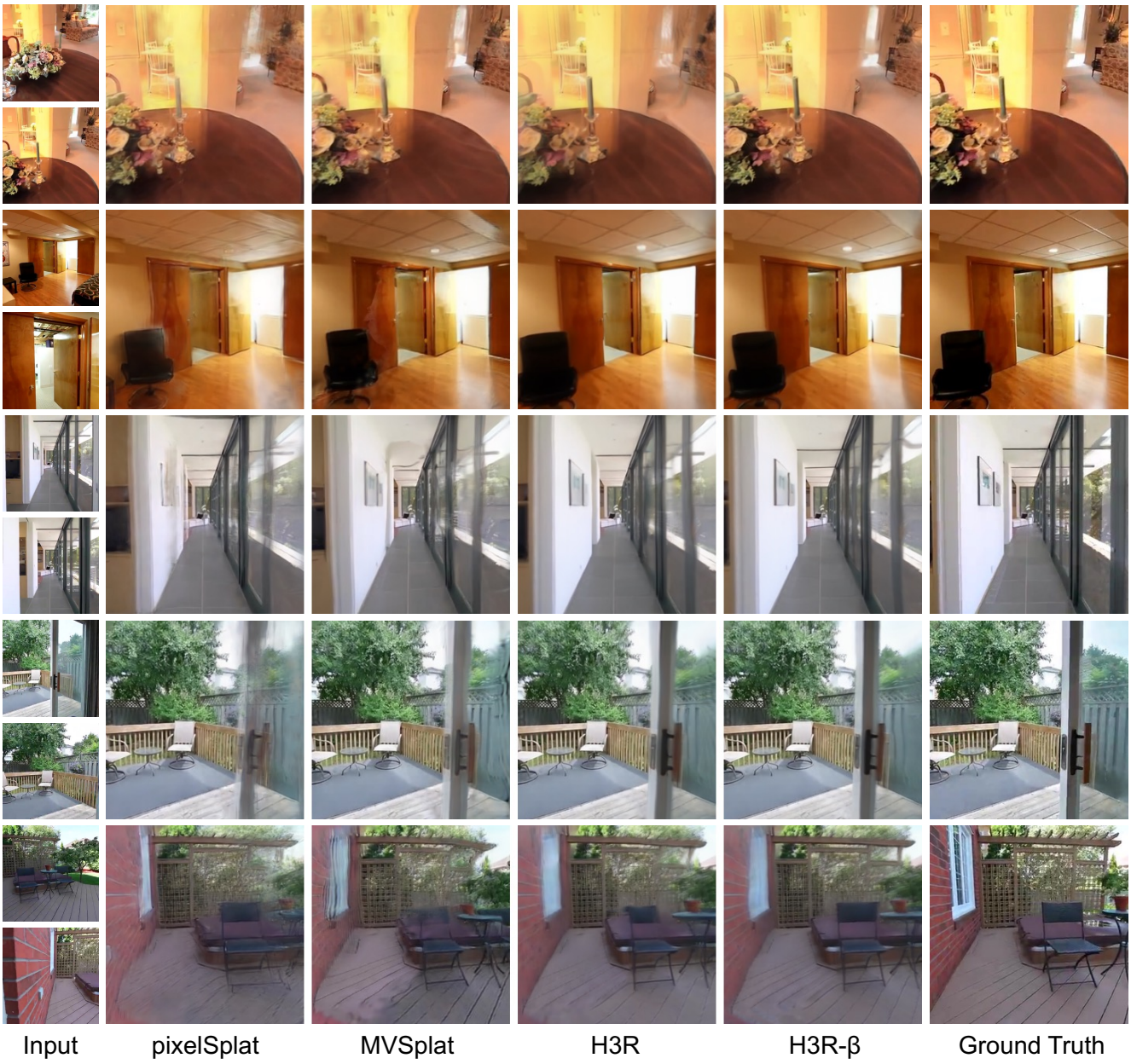

Figure D3. **Impact of input resolution on RealEstate10K.** Using 512×512 inputs, our H3R-β achieves more accurate geometry and photorealistic texture than recent methods, particularly for sharp edges and complex surfaces.