VIAFormer: Voxel-Image Alignment Transformer for High-Fidelity Voxel Refinement

Supplementary Material

8. More Results

To further demonstrate the robustness and versatility of our method, this section provides additional qualitative results. We first evaluate on real-world captures: As shown in Fig. 9, VIAFormer effectively completes large missing regions such as sofa undersides (first two rows), yielding watertight meshes suitable for downstream applications. However, the model does not complete all cases; for example, the third sofa lacks a reconstructed base. We speculate this arises because a portion of our training data contains objects with naturally open bottoms, causing the model to mistakenly associate excessively large missing regions with such intentional openings. For synthetic and VFM-derived noise, Figs. 12 and 13 present additional results of our method.

Furthermore, we conduct an experiment to evaluate the model's capacity for half-space completion. The model is tasked with inferring the missing half of a shape, given the VFM-estimated image index ($\text{pos}_{I,i}$), four corresponding camera views ($\{I_i\}_{i=1}^{4}$), and one half of the ground-truth voxel grid (v_{gt}). The results, shown in Fig. 14, indicate that our model is capable of reconstructing the missing geometry based on these inputs.

9. Model Architecture Details

This section provides detailed diagrams for the model architectures discussed in Sec. 4. Fig. 15 illustrates the complete architecture of our proposed VIAFormer model. For the ablation studies, Fig. 16 details the baseline models used for comparison: (a) the geometry-only 24-Layer Self-Attention model, and (b) the 24-Layer Cross-Attention model with its various conditioning adapters.

10. Data Process Details

This section expands upon the data generation process from Sec. 4.5, with a visual overview provided in Fig. 10.

10.1. VFM Noise

We employ Pi3 [40], a powerful vision foundation model (VFM), to simulate point clouds that mimic real-world sensor data, which are typically coarse, noisy, and incomplete in occluded regions. For each sample in the dataset, we first exclude the bottom view from the set of rendered images, as such viewpoints are rarely feasible in practical capture scenarios. The remaining multi-view images are then fed into Pi3 to predict per-view point clouds in the world coordinate system, along with confidence probabilities. We

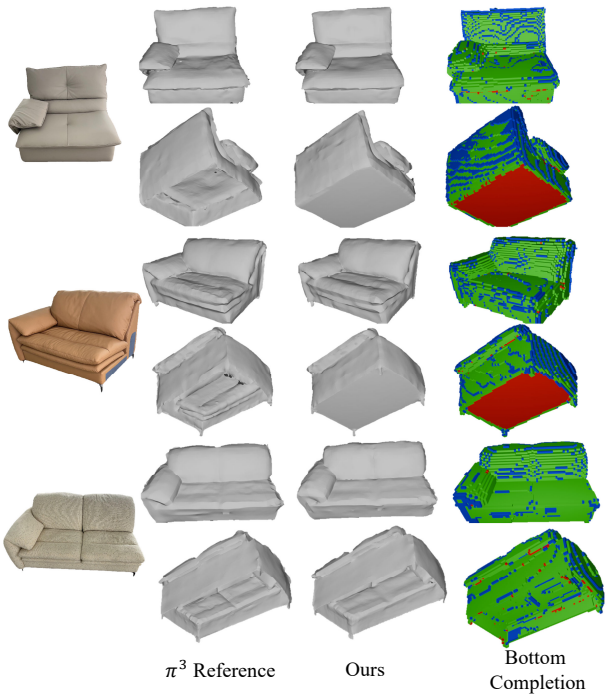


Figure 9. **Visualizing the Effect of VIAFormer's Refinement in Real-world 3D Sofa Creation Pipeline.** The "Pi3 reference" and "Ours" columns compare the final meshes produced by Trellis [44] from the original and our refined voxels, respectively. The rightmost column displays the direct voxel modifications performed by our model (red: added, green: keep, blue: removed).

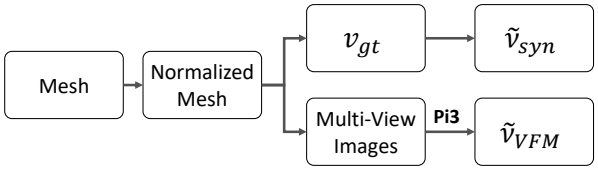


Figure 10. **Data process pipeline.** To simulate real-world noise and enhance model robustness, we generate two distinct types of voxel inputs: VFM Noise and Synthetic Corruptions.

apply a confidence threshold of $p > 0.2$ to filter out low-confidence points, effectively removing background clutter. The filtered point clouds from all views are fused to obtain a coarse reconstruction of the object's geometry. Since Pi3 produces outputs with arbitrary scale and orientation, we normalize the fused point cloud to unit space and align it using the ground-truth pose of the first view, thereby transforming the relative reconstruction into an absolute world coordinate frame. Finally, the normalized point cloud is converted into a voxel grid at resolution 64 using Open3D's

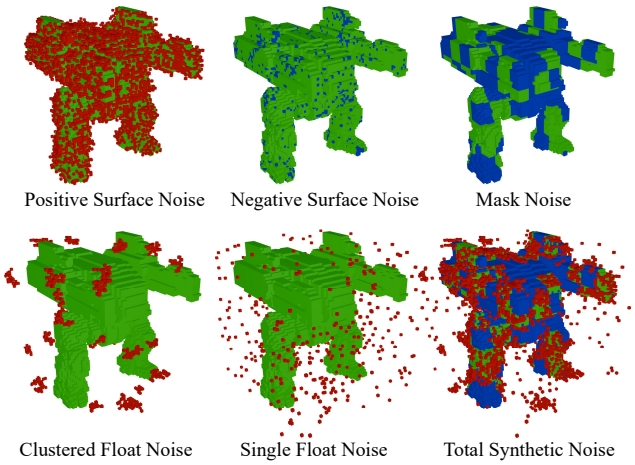


Figure 11. **Synthetic noise generation modules.** A clean model (green) is corrupted with various noise types (additions in red, removals in blue). Top row: Positive Surface, Negative Surface, and Mask Noise. Bottom row: Clustered Float, Single Float, and the combined Total Synthetic Noise used for training.

`create_from_point_cloud_within_bounds` function. The resulting voxel representation is intentionally noisy and incomplete, forming a training pair with the clean ground-truth voxel grid, which is consistent with our voxel refinement objective.

10.2. Synthetic Corruptions

To complement the realistic VFM noise and further improve model robustness, we introduce a diverse synthetic corruption pipeline. We argue that existing approaches like block-wise masking used in PatchComplete [26] are too limited for general-purpose refinement. Therefore, our pipeline explicitly adds surface noise and volumetric artifacts (floaters) to better prepare the model for a wider range of real-world degradations.

As shown in Fig. 11, we design our synthetic corruptions as composable noise modules. Each module takes an occupancy grid $v \in \{0,1\}^{R\times R\times R}$ as input, along with specific control parameters, and generates an additive noise score map $n \in \mathbb{R}^{R\times R\times R}$. The noise scores from all applied modules are summed and added to the original grid, which is then binarized to produce the final corrupted grid $\tilde{v}$:

$$\tilde{v} = \text{clamp}\left(v + \sum n, 0, 1\right). \tag{9}$$

1) SDF-based Single-Voxel Noise. This module generates isolated noise voxels based on their distance to the object's surface, calculated via a Signed Distance Function (SDF). Voxels within a specified SDF range (a "shell") become candidates. Each candidate is then selected with a probability p, randomly sampled from a predefined list (see Table 3). This process is used to create both:

- **Positive Noise** (e.g., *Surface Noise*, *Far-field Noise*), which adds spurious voxels outside the object surface.
- **Negative Noise** (e.g., *Surface Erosion*), which creates holes by removing voxels just inside the surface.

2) SDF-based Clustered Noise. To simulate more spatially coherent artifacts, this module extends the SDF-based approach. Instead of adding or removing single voxels, it treats each stochastically selected point as a seed. From each seed, a cluster of k connected voxels is grown, where k is randomly sampled from a list. This generates larger, contiguous artifacts such as *Clustered Floaters* (detached fragments) or *Clustered Erosion* (larger cavities).

3) Coarse-level Masking. This module simulates large-scale, structural incompleteness by removing entire blocks of voxels. It achieves this by generating a low-resolution binary mask, upscaling it to the full grid resolution, and applying a large negative score to remove the voxels in the masked-out regions.

4) Half-Space Removal. Distinct from our main noise pipeline, Half-Space Removal is a special corruption applied exclusively to a separate training subset. This drastic method, which deterministically removes all voxels on one side of a plane, is designed to rigorously test the model's large-scale, visually-conditioned generative capabilities. These samples are not mixed with those receiving VFM or other synthetic noises.

10.3. Additional Preprocessing in Real-World Scenarios

Compared to synthetic rendering datasets, real-world applications often lack ground-truth information, resulting in incomplete inputs that require additional preprocessing. To address this challenge, we leverage some large powerful model to estimate missing information. Specifically, we employ Pi3 [40] to predict camera parameters from real captured images, SAM2 [27] to generate precise object segmentation masks, OrientAnything [42] to estimate object orientation for canonical pose normalization, and StableNormal [46] to infer surface normal maps from RGB inputs.

10.4. K-Means View Selection Algorithm

Algo. 1 details the K-Means view selection strategy mentioned in Sec. 5.1. The objective is to select a diverse set of S views by clustering their angular positions. This method assumes all cameras point towards the origin, enabling the direct use of their 3D positions to derive angular features (longitude and latitude) for clustering. The algorithm partitions candidate views into S clusters and selects the most central view from each to ensure maximal angular coverage. For our experiments, total view count $N = 150$, the pitch angle for candidate views is constrained between $\theta_{\text{min}} = -10^\circ$ and $\theta_{\text{max}} = 30^\circ$.

Table 3. **Hyperparameters for Synthetic Corruption Modules.** This table details the parameters for the noise modules used in our training pipeline. For each corruption pass, a specific value is randomly sampled from the corresponding parameter list. The base resolution is 64^3.

Noise Type	Description	SDF Range	Probability List (p)	Cluster Size List (k)
Positive Noise (Floaters)				
Surface Noise	Single voxel addition	$(0, 0.04]$	[a]	—
Surface Noise	Single voxel addition	$(0.04, 0.08]$	[b]	—
Far-field Noise	Single voxel addition	$(0.15, 2.0]$	[c]	—
Clustered Floaters	Multi-voxel addition	$(0.15, 2.0]$	[d]	[f]
Negative Noise (Holes)				
Surface Erosion	Single voxel removal	$(0, 0.04]$	[a]	—
Surface Erosion	Single voxel removal	$(0.04, 0.08]$	[b]	—
Clustered Erosion	Multi-voxel removal	$(0, 0.04]$	[e]	[f]
Masking Noise				
Coarse Masking	Block removal at various resolutions	—	Foreground prob. in $[0.5, 1.0]$	Mask Res. $\{16, 8, 8, 4\}$

Parameter Lists:

[a] $\{0.01, 0.02, 0.03, 0.04, 0.05, 0.07, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40\}$
[b] $\{0.005, 0.01, 0.02, 0.03, 0.04, 0.05, 0.08, 0.12, 0.15, 0.20\}$
[c] $\{10^{-4}, 2.5 \times 10^{-4}, 5 \times 10^{-4}, 10^{-3}, 2 \times 10^{-3}\}$
[d] $\{10^{-5}, 2 \times 10^{-5}, 5 \times 10^{-5}, 10^{-4}\}$
[e] $\{8 \times 10^{-4}, 2 \times 10^{-3}, 5 \times 10^{-3}, 8 \times 10^{-3}\}$
[f] $\{8, 10, 12, 14, 16, 18, 20, 25, 30\}$

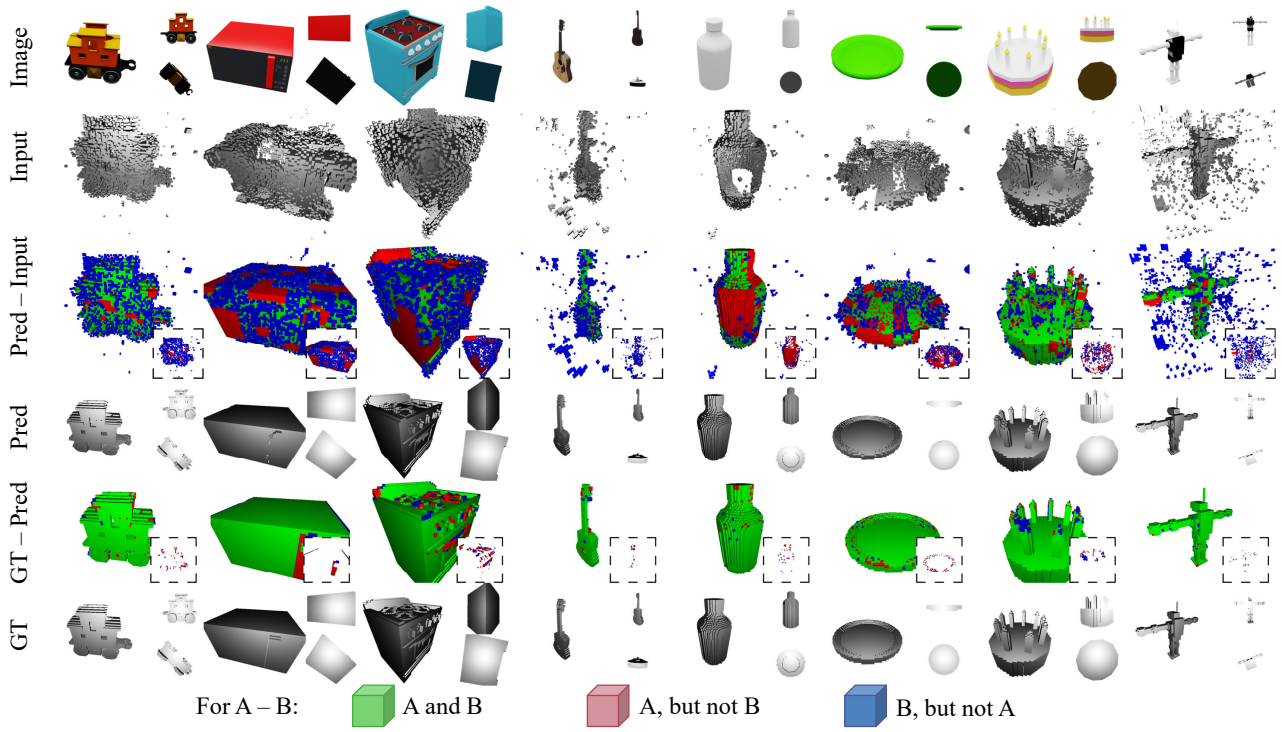


Figure 12. **More refinement results on synthetic noise.** We present additional results demonstrating the VIAFormer's robustness across a diverse set of objects.

11. Evaluation Metrics

1) IoU (Intersection over Union): IoU measures the overlap between the predicted voxel set $\hat{v}$ and the ground-truth

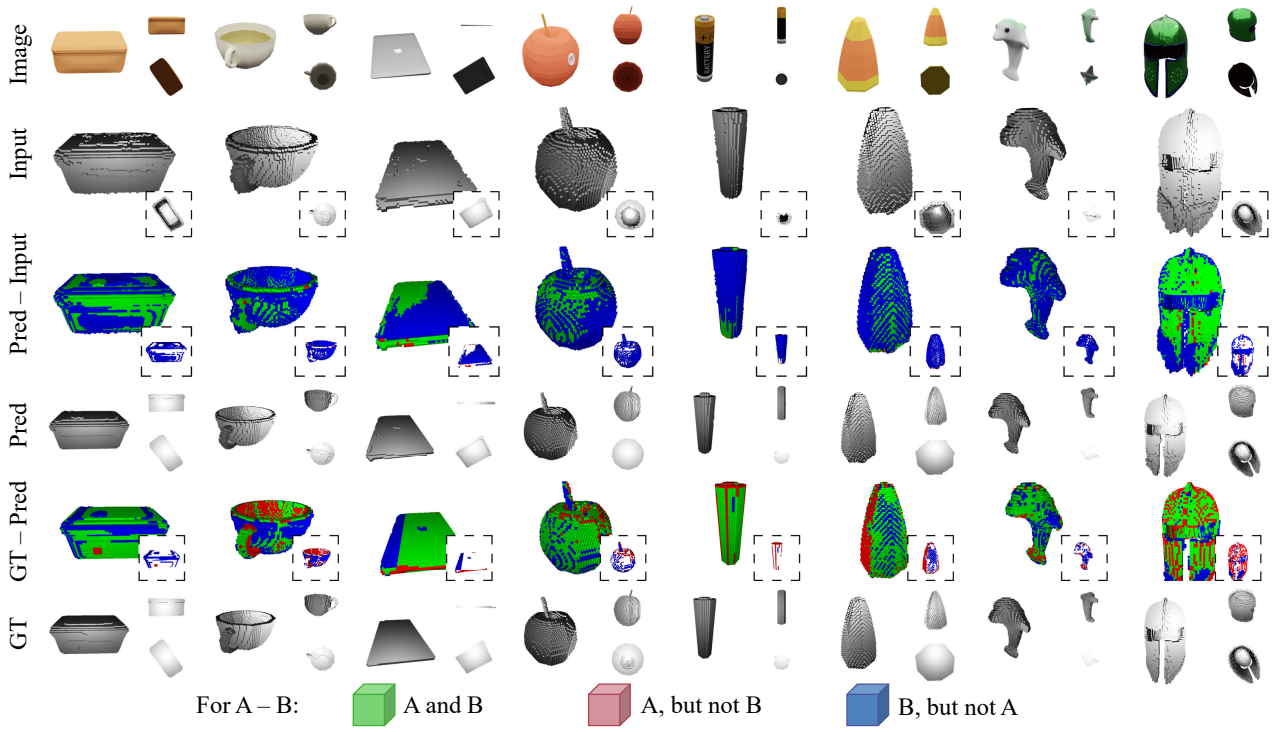


Figure 13. **More refinement results on VFM-derived noise.** We present additional results demonstrating the VIAFormer's robustness across a diverse set of objects.

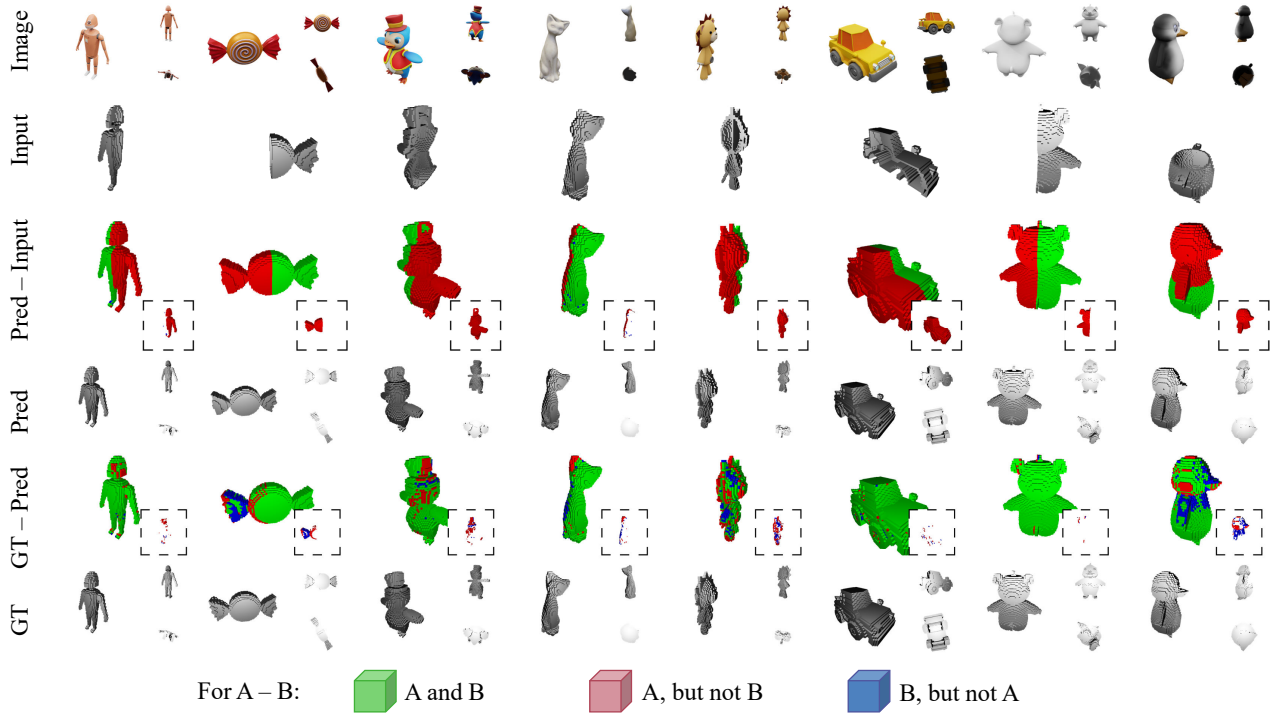


Figure 14. **Qualitative results for the half-space completion task.** The model is tasked with completing a shape given half of the geometry (**Input**), 4 input views and VFM-estimated Image Index. The **Pred - Input** row visualizes this process, showing the provided half (green) and the generated half (red). The final predictions (**Pred**) and low residual errors (**GT - Pred**) demonstrate the model's ability to perform large-scale conditional generation by fusing geometric and visual information. The rendered **Image** is for reference only.

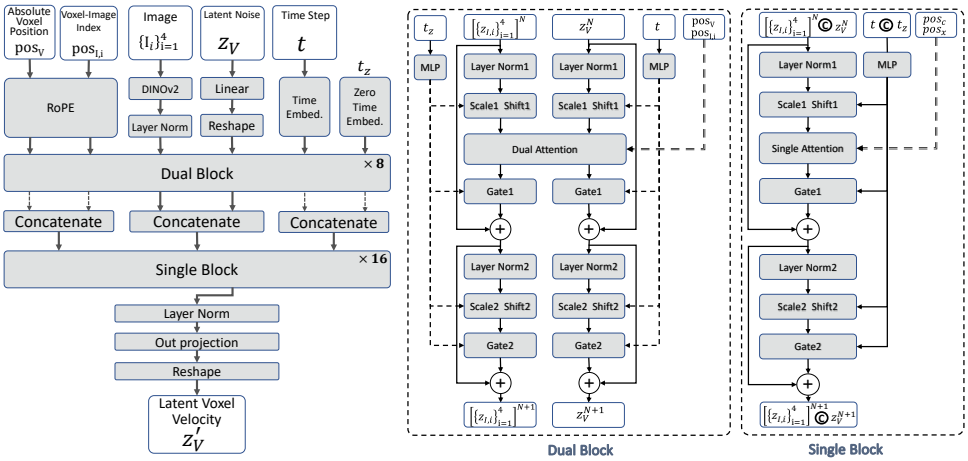
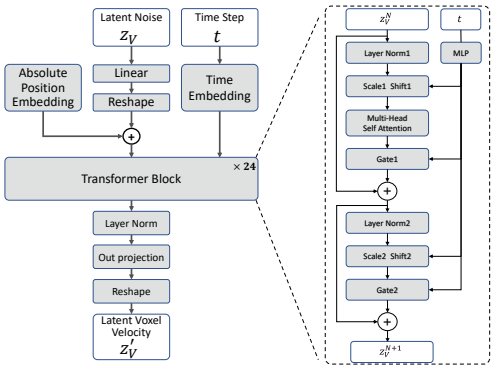
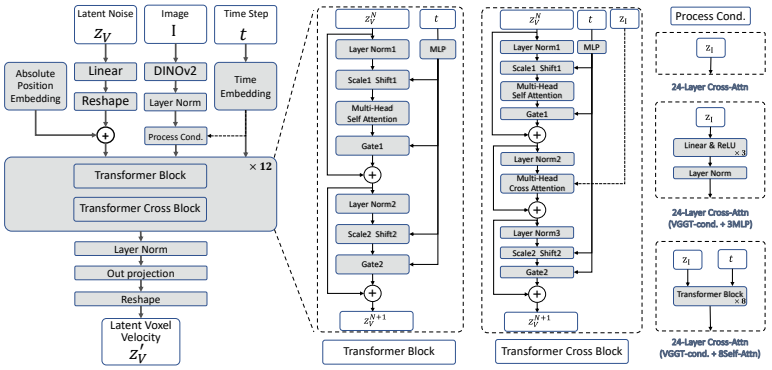


Figure 15. **The VIAFormer Architecture.** Our model is a two-stage transformer. The first stage uses 8 Dual Blocks (middle) to process and align parallel voxel (z_V) and image ($\{z_{I,i}\}_{i=1}^{S}$) streams. The second stage concatenates these streams and uses 16 Single Blocks (right) for global refinement. The model outputs the final latent voxel velocity.



(a) **The 24-Layer Self-Attention Baseline.** A geometry-only architecture that refines the noisy latent voxel grid (z_V) using a stack of 24 self-attention Transformer blocks, conditioned on time (t).



(b) **The 24-Layer Cross-Attention Baseline.** An architecture composed of alternating self-attention and cross-attention blocks to fuse image conditioning (z_I) into the voxel stream (z_V). We test three conditioning variants (right panel): direct injection, a 3-layer MLP adapter, and an 8-layer self-attention adapter.

Figure 16. **Baseline architectures for ablation studies.**

set v_{gt}:

$$\text{IoU}(\hat{v}, v_{\text{gt}}) = \frac{|\hat{v} \cap v_{\text{gt}}|}{|\hat{v} \cup v_{\text{gt}}|}. \tag{10}$$

2) Chamfer Distance (CD): To evaluate surface-level fidelity, we compute the Chamfer Distance. First, the predicted and ground-truth voxel grids are normalized to fit within a unit cube of $[-0.5, 0.5]^3$. We then extract their point clouds of grid $\text{P}(\hat{v})$ and $\text{P}(v_{\text{gt}})$. The CD is defined as the mean squared closest point distance between these two sets:

$$\begin{aligned}\text{CD}(\hat{v}, v_{\text{gt}}) = &\frac{1}{|\hat{v}|} \sum_{x \in \text{P}(\hat{v})} \min_{y \in \text{P}(v_{\text{gt}})} \|x - y\|_2^2 \\ &+ \frac{1}{|v_{\text{gt}}|} \sum_{y \in \text{P}(v_{\text{gt}})} \min_{x \in \text{P}(\hat{v})} \|y - x\|_2^2.\end{aligned} \tag{11}$$

12. Attention Map

12.1. Attention Map Theory

This section provides a theoretical overview of the Transformer attention mechanism, focusing on the generation and interpretation of attention maps. This foundation is essential for understanding the attention map visualizations presented in Fig. 3 and our diagnosis of "Attention Collapse" in Sec. 4.2.

In a Transformer attention layer, input tokens are first projected into three distinct representations: Query (Q), Key (K), and Value (V). For self-attention, these projections originate from the same input sequence. For cross-attention, Q is projected from the target sequence (e.g., voxels), while K and V are projected from the source sequence (e.g., image patches). Let the target sequence have length L_q and the source sequence have length L_{kv}. The input

Algorithm 1 K-Means View Selection

Require: A set of N camera positions $\mathcal{P} = \{p_i\}_{i=1}^{N}$, where $p_i = (x_i, y_i, z_i) \in \mathbb{R}^3$.
Require: The desired number of views S.
Require: Minimum and maximum pitch angles, θ_{min} and θ_{max}.
Ensure: A set of S selected view indices $\mathcal{I}_{\text{selected}}$.
Assumption: All cameras are oriented to look at the origin $(0, 0, 0)$.
Initialize an empty list for angular features $\mathcal{F} \leftarrow []$.
for each camera position $p_i = (x_i, y_i, z_i)$ in $\mathcal{P}$ **do**
  $\text{lng}_i \leftarrow \text{atan2}(y_i, x_i)$ ▷ Calculate azimuth
  $\text{lat}_i \leftarrow \text{atan2}(z_i, \sqrt{x_i^2 + y_i^2})$ ▷ Calculate pitch
  Append $(\text{lng}_i, \text{lat}_i)$ to $\mathcal{F}$.
end for
$\mathcal{I}_{\text{cand}} \leftarrow \{i \mid \theta_{\text{min}} \leq \mathcal{F}[i]_{\text{lat}} \leq \theta_{\text{max}}\}$ ▷ Filter views
if $|\mathcal{I}_{\text{cand}}| < S$ **then**
  $\mathcal{I}_{\text{cand}} \leftarrow \{1, 2, \ldots, N\}$ ▷ Fallback if too few views
end if
$\mathcal{F} \leftarrow \{\mathcal{F}[i] \mid i \in \mathcal{I}_{\text{cand}}\}$
Generate a random angle $\phi_{\text{rand}} \sim U(0, 2\pi)$.
for each index $i \in \{1, \ldots, N\}$ **do**
  $\mathcal{F}[i]_{\text{lng}} \leftarrow (\mathcal{F}[i]_{\text{lng}} + \phi_{\text{rand}}) \pmod{2\pi}$
end for
$(\mathcal{C}_1, \ldots, \mathcal{C}_S), (\mu_1, \ldots, \mu_S) \leftarrow \text{KMeans}(\mathcal{F}, S)$
  ▷ $\mathcal{C}_k$ are clusters of feature vectors, μ_k are centroids
$\mathcal{I}_{\text{selected}} \leftarrow \emptyset$.
for $k = 1$ to S **do**
  Find original index j^* of the feature in $\mathcal{F}$ that is closest to centroid μ_k:
  $j^* \leftarrow \underset{j \in \mathcal{I} \text{ s.t. the view with index } j \text{ is in cluster } \mathcal{C}_k}{\text{argmin}} \|\mathcal{F}[j] - \mu_k\|_2$
  Add j^* to $\mathcal{I}_{\text{selected}}$.
end for
return $\mathcal{I}_{\text{selected}}$

tensors are typically of shapes $\mathbb{R}^{B\times L_q\times C}$ and $\mathbb{R}^{B\times L_{kv}\times C}$ respectively.

For multi-head attention, the feature dimension C is divided among H parallel heads, with each head having a dimension of $d_k = C/H$. The Q, K, V tensors are then reshaped and transposed to isolate the heads for computation, resulting in the shapes:

$$Q \in \mathbb{R}^{B\times H\times L_q\times d_k}, \quad K, V \in \mathbb{R}^{B\times H\times L_{kv}\times d_k}. \tag{12}$$

The core of the mechanism is the scaled dot-product attention formula, which computes the output as a weighted sum of V:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{13}$$

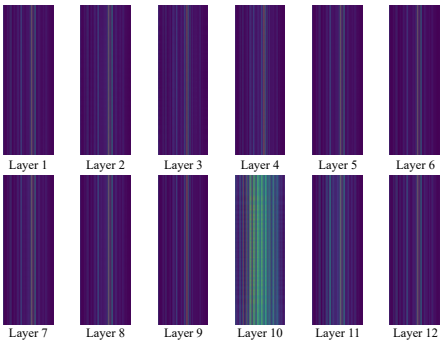


Figure 17. **Full visualization of Attention Collapse in the cross-attention baseline.** This figure displays the mean attention map (averaged across 16 heads) for each of the 12 cross-attention layers.

The component used for visualization is the attention map, which is the tensor of weights that pre-multiplies V:

$$\text{AttentionMap} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \in \mathbb{R}^{B\times H\times L_q\times L_{kv}}. \tag{14}$$

Its role is to weight V by multiplication of the batch matrix to produce the final attention output:

$$\underbrace{\text{AttentionMap}}_{[B,H,L_q,\mathbf{L_{kv}}]} @ \underbrace{V}_{[B,H,\mathbf{L_{kv}},d_k]} \Rightarrow \underbrace{\text{AttentionOutput}}_{[B,H,L_q,d_k]}. \tag{15}$$

This structure clarifies the role of the attention map: its element at (i, j) represents the weight assigned by the i-th query token to the j-th key token, thereby encoding a learned, directional relationship across modalities.

In visually intuitive domains such as image processing, it is common practice to visualize attention from a specific query token. The corresponding row of the attention map is extracted and reshaped back into a heatmap to illustrate that the model's attention is focused on semantically meaningful patches. However, VIAFormer operates within an abstract latent space—the velocity field defined by the Correctional Flow. This space lacks a direct, visually interpretable mapping. Consequently, we present the full, abstract $L_q \times L_{kv}$ attention maps directly. The goal of our visualizations is not to pinpoint focus on specific spatial locations, but rather to diagnose the global patterns of the attention mechanism, which is critical for identifying **Attention Collapse** case discussed in our analysis.

12.2. Attention Collapse Evidence

This section provides the full visual evidence for the "Attention Collapse" claim (Sec. 4.2) in Fig. 17, which shows the consistent uniform vertical stripe pattern in the mean attention maps for all 12 cross-attention layers of our baseline, a pattern indicating that every voxel token attends non-selectively to all image tokens, confirming the systemic failure to learn spatial correspondence.