Diffusion Based 3D Assets Generation Foundation Models

Yangguang Li
liyangguang256@gmail.com
The Chinese University of Hong Kong
2025-10

Having witnessed the success of diffusion-based generative paradigms in the image and video domains, the question arises: how should 3D asset generation evolve?

— Diffusion Based 3D Assets Generation Foundation Model

Tutorial Outline

3D Data Preparation:
- Open Source Datasets
- Data Filtering
- Data Preprocessing

3D VAE and Representation:
- 3DShape2VecSet VAE
- 3D Sparse Structure / Structured Latent VAE

3D Shape Generation Models:
- 3DShape2VecSet Generation
- 3D Sparse Structure / Structured Latent Generation

3D Texture Generation Models:
- Texture Dataset and Representation
- Texture Generation Models

3D Data Preparation:

Various Original Datasets

SHAPENET

Watertight Direction Align Many Low-Quality Data

Input

Data Processing

Training Data

Watertight Direction Align Filtered High-Quality Data

Output

3D Data Preparation:

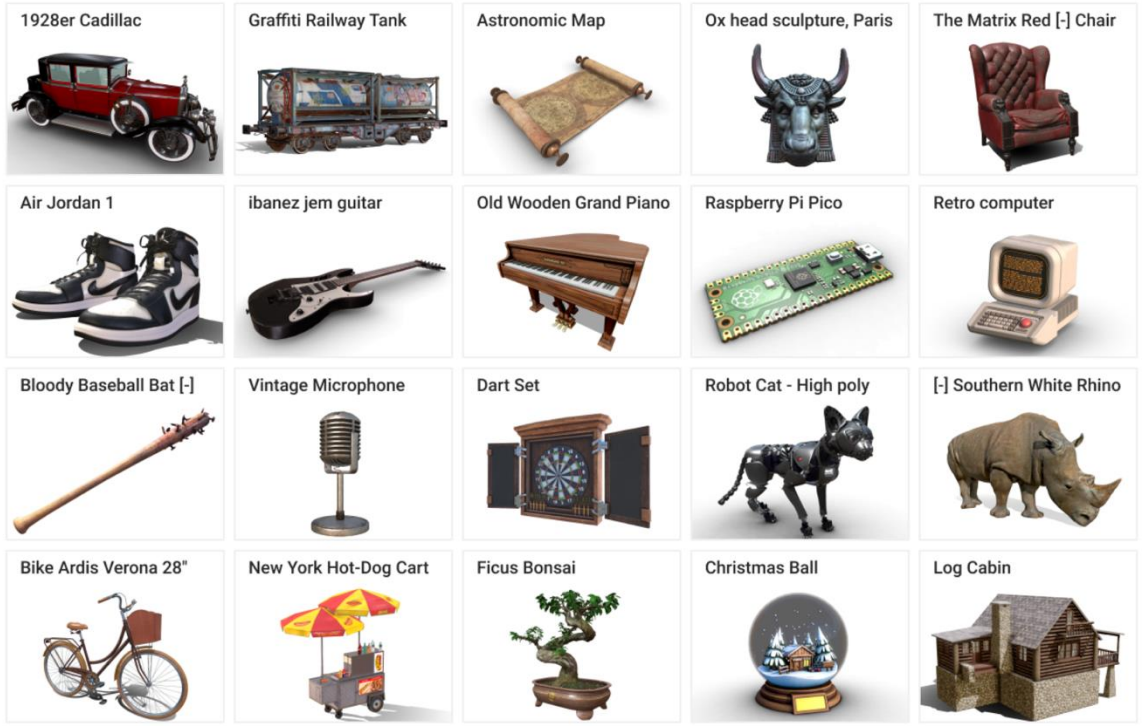
- Open Source Datasets

- Data Filtering and Fixing
  - Data Scoring Procedure
  - Data Filtering Procedure
  - Data Fixing and Augmentation Procedure
  - Field Data Producing

- Data Preprocessing
  - Watertight Converting without Deformation
  - Watertight Converting with Deformation

3D Data Preparation:

- Open Source Datasets



Objaverse [1]

Dataset	Size	Highlight
Objaverse[1]	800K	The first open-source large-scale 3D object dataset, the quality of objects varies greatly and requires targeted filtering.
Objaverse-XL[2]	10M	The largest open-source 3D dataset. It is quite noisy and contains a significant number of low-quality objects.
ABO[3]	8K	Complex geometries and high-resolution materials, containing 63 categories.
3D-FUTURE[4]	16.5K	Rich geometric details and informative textures.
HSSD[5]	14K	High-quality, human-authored synthetic 3D indoor dataset.
Toys4K[6]	4K	Contains 105 object categories, featuring a diverse set of object instances within each category, and is often used as a test set.
ShapeNet[7]	50K	Contains about 55 categories, with relatively low overall quality but covering common object classes.

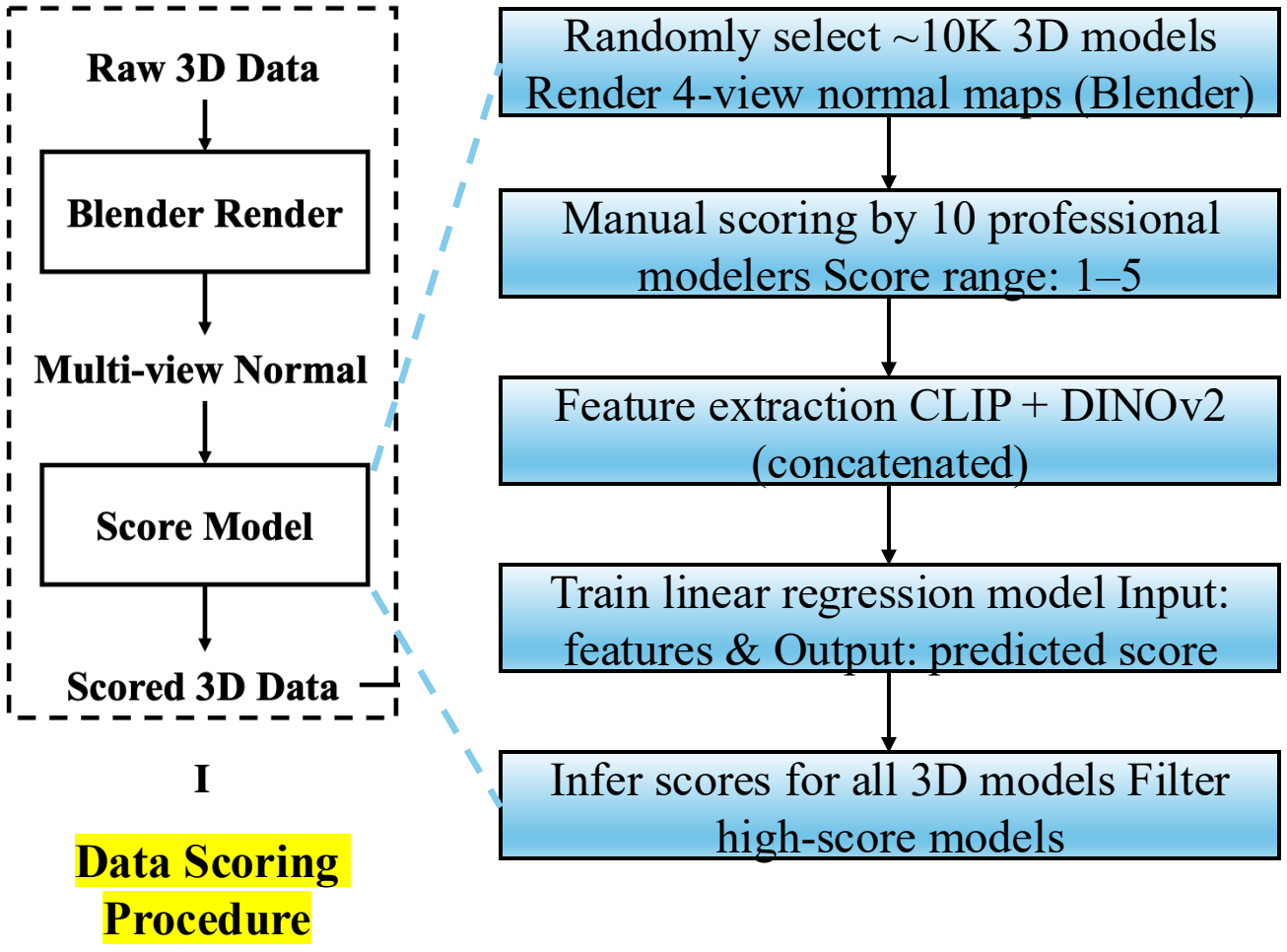
[1]Objaverse: A Universe of Annotated 3D Objects; [2]Objaverse-XL: A Universe of 10M+ 3D Objects; [7] ShapeNet: An Information-Rich 3D Model Repository;

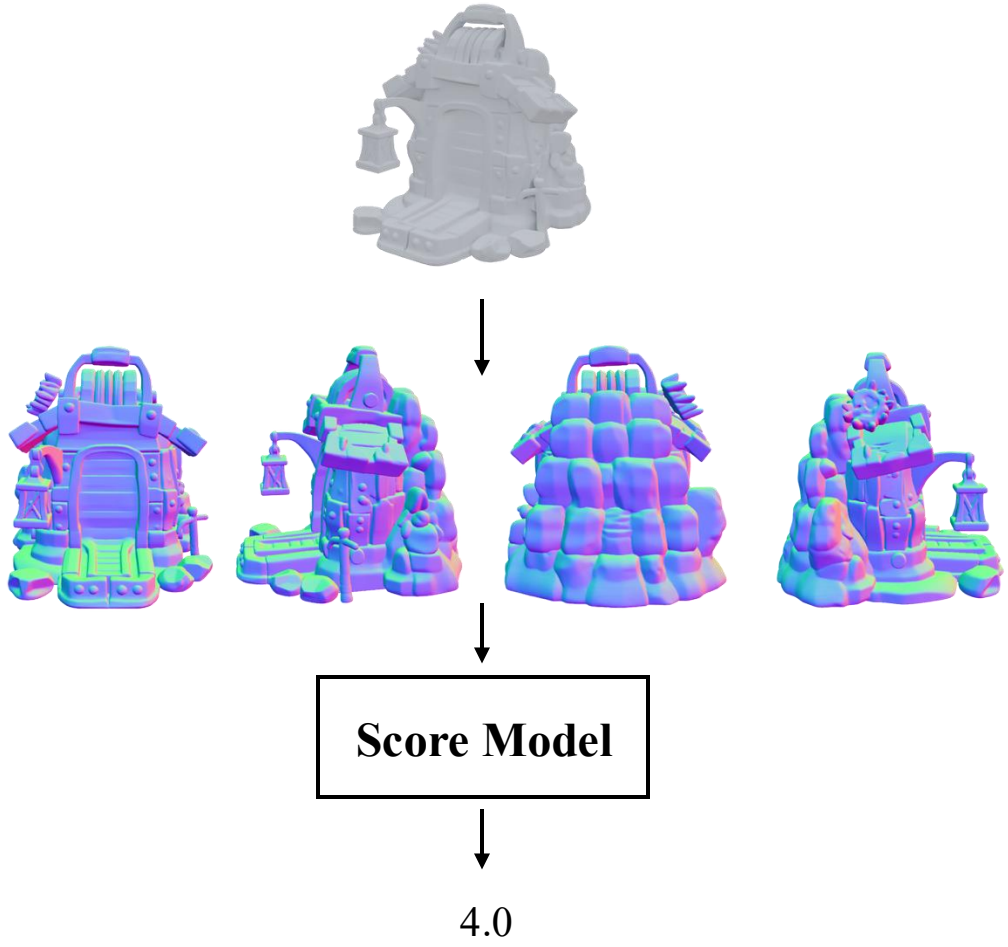
[3] ABO: Dataset and benchmarks for real-world 3d object understanding; [4] 3D-FUTURE: 3d furniture shape with texture.

[5]Habitat Synthetic Scenes Dataset (HSSD-200); [6] Using shape to categorize: Low-shot learning with an explicit shape bias.

3D Data Preparation:

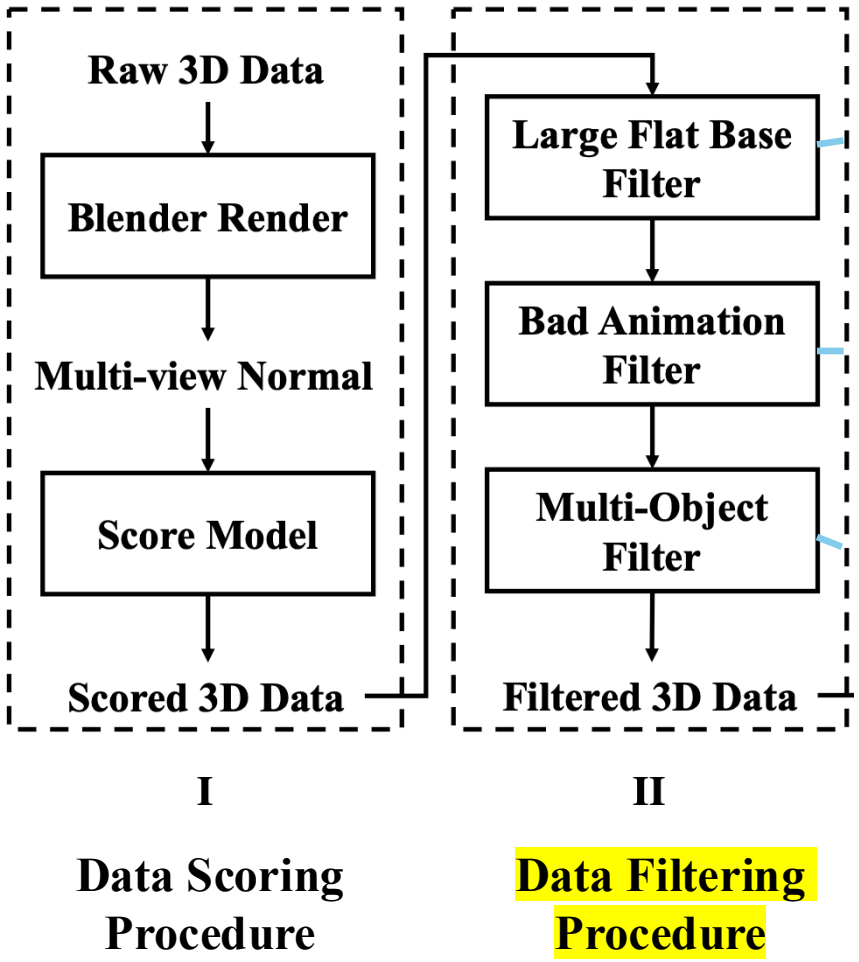
- Data Filtering and Fixing





3D Data Preparation:

- Data Filtering and Fixing



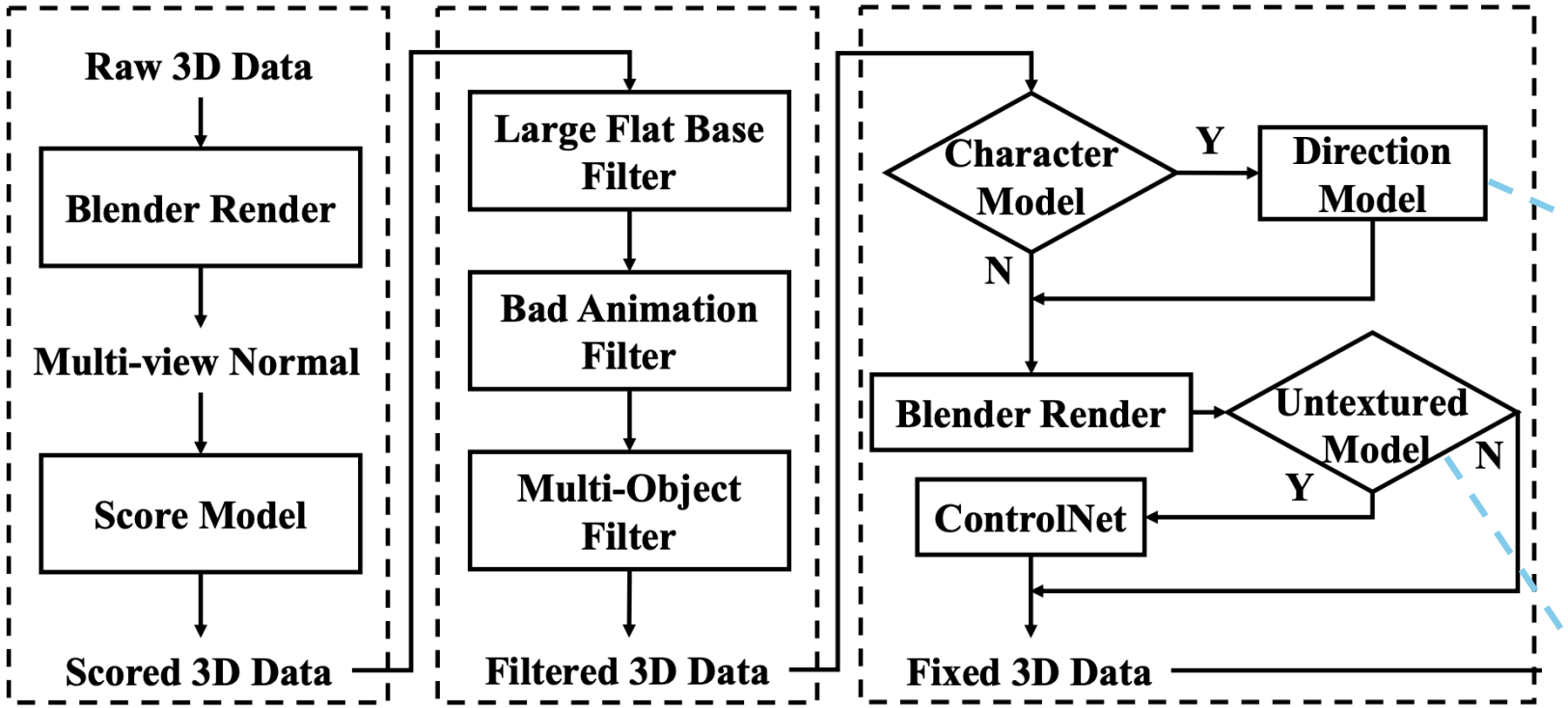
The surface is divided into patches and evaluated based on centroid positions, normal vectors, and plane area.

Blender detects animated models, resets them to the first frame, and filters out those still showing rendering errors.

Filtering is performed by computing the largest connected component ratio in the opaque mask and the solidity of both the component and the entire mask.

3D Data Preparation:

- Data Filtering and Fixing



- Sample 24 orientations around the x, y, and z axes;
- Render six orthogonal views for each model;
- Concatenate their DINOv2 features to train an direction estimation model.

- Render multi-view normal maps for untextured models;
- Converted multi-view normal maps into corresponding RGB images using ControlNet++.

3D Data Preparation:

- Data Filtering and Fixing

Method	Release Date	Data Souce	Data Size after Filtering	Filtering Methods
CraftsMan[1]	2024.05.23	Objaverse	170k	Filtering out meshes of inferior quality, such as those with point clouds, thin structures, holes, and texture-less surfaces.
CLAY[2]	2024.05.30	ShapeNet Objaverse	527K	Filtering out unsuitable data, such as complex scenes and fragmented scans.
Trellis[3]	2024.12.02	Objaverse Objaverse-XL ABO 3DFUTURE HSSD	500K	Employing a pretrained aesthetic assessment model to evaluate the quality of each 3D asset.
TripoSG[4]	2025.02.10	ShapeNet Objaverse Objaverse-XL	2M	Data scoring procedure to filter out high quality data; Data filter procedure to filter large flat base data, bad animation data and multi-object data; Data fixing and augmentation procedure to fix the direction of character object and re-generate texture for untextured object.

[1] CraftsMan3D: High-fidelity Mesh Generation with 3D Native Generation and Interactive Geometry Refiner

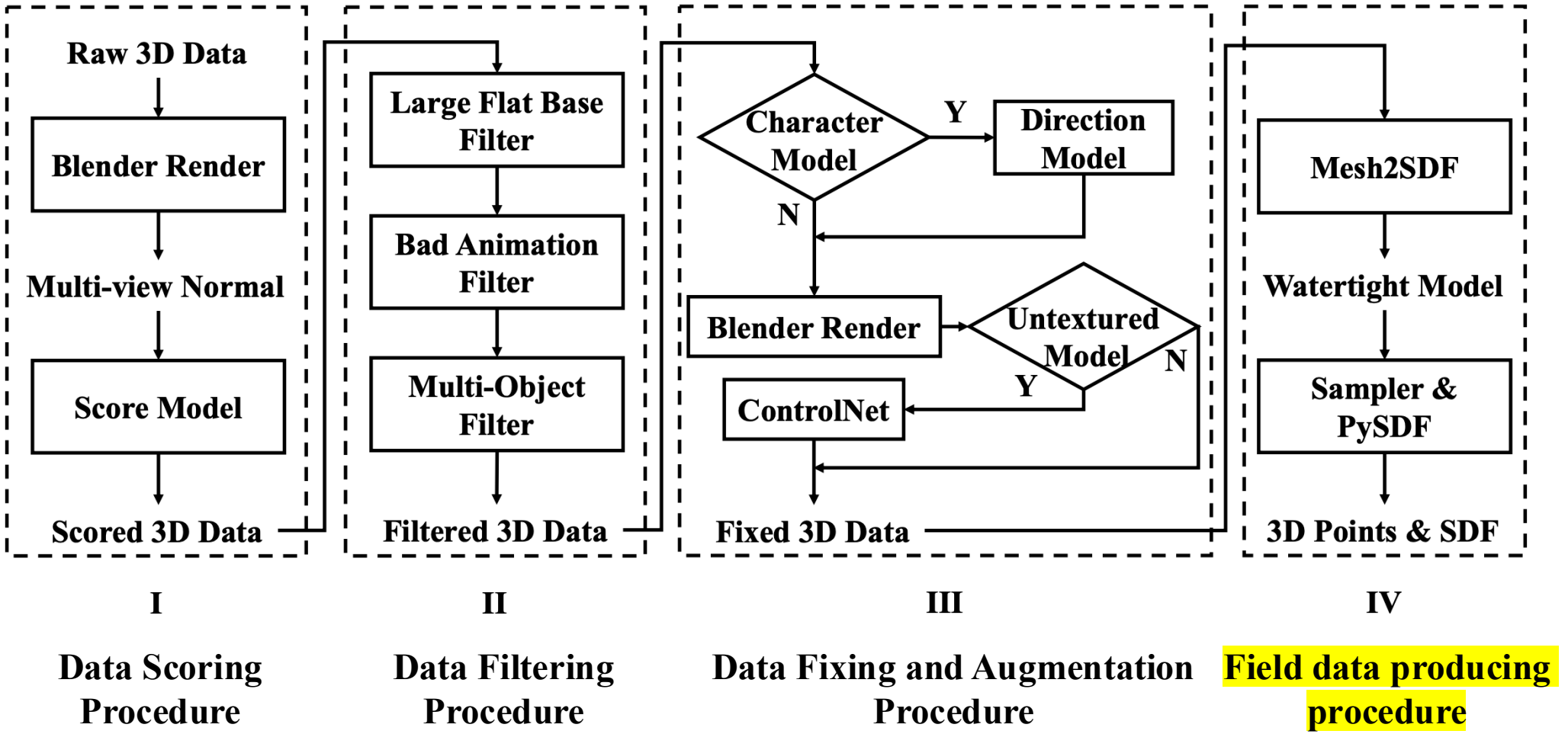
[2] CLAY: A Controllable Large-scale Generative Model for Creating High-quality 3D Assets

[3] Structured 3D Latents for Scalable and Versatile 3D Generation

[4]TripoSG: High-Fidelity 3D Shape Synthesis using Large-Scale Rectified Flow Models

3D Data Preparation:

- Field Data Producing



3D Data Preparation:

- Data Preprocessing: *Watertight Converting without Deformation - SDF2Watertight*

0. Input

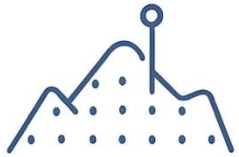
Input Raw 3D Data: watertight or non-watertight

1. UDF Construction

Construct Unsigned Distance Field (UDF)

2. UDF Resetting

Reset UDF value of invisible grid area

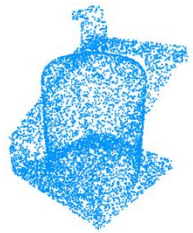


3. Mesh Extraction

Generate watertight mesh with matching cube

4. Post-Processing

Filter components by area, ambient occlusion

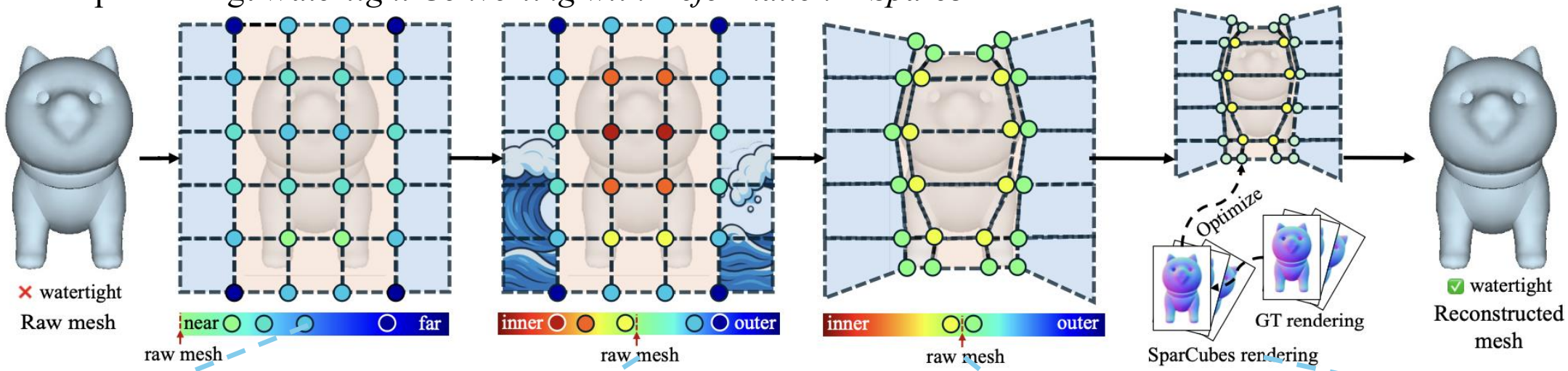


5. Samping

Sample 3D points and normals on mesh

3D Data Preparation:

- Data Preprocessing: *Watertight Converting with Deformation - Sparc3D*



Step 1: Active voxel extraction and UDF computation

•Select sparse **active voxels** within a narrow band around the surface.

•Compute the unsigned distance field (UDF) for each corner vertex to form a sparse voxel grid.

Step 2: Flood fill for coarse sign labeling

•Apply a volumetric flood fill algorithm to distinguish inside and outside.

•Construct the coarse SDF to provide consistent sign assignment.

Step 3: Gradient-based deformation optimization

•Instead of refining the global SDF, optimize sparse cube vertices to better align the zero level set.

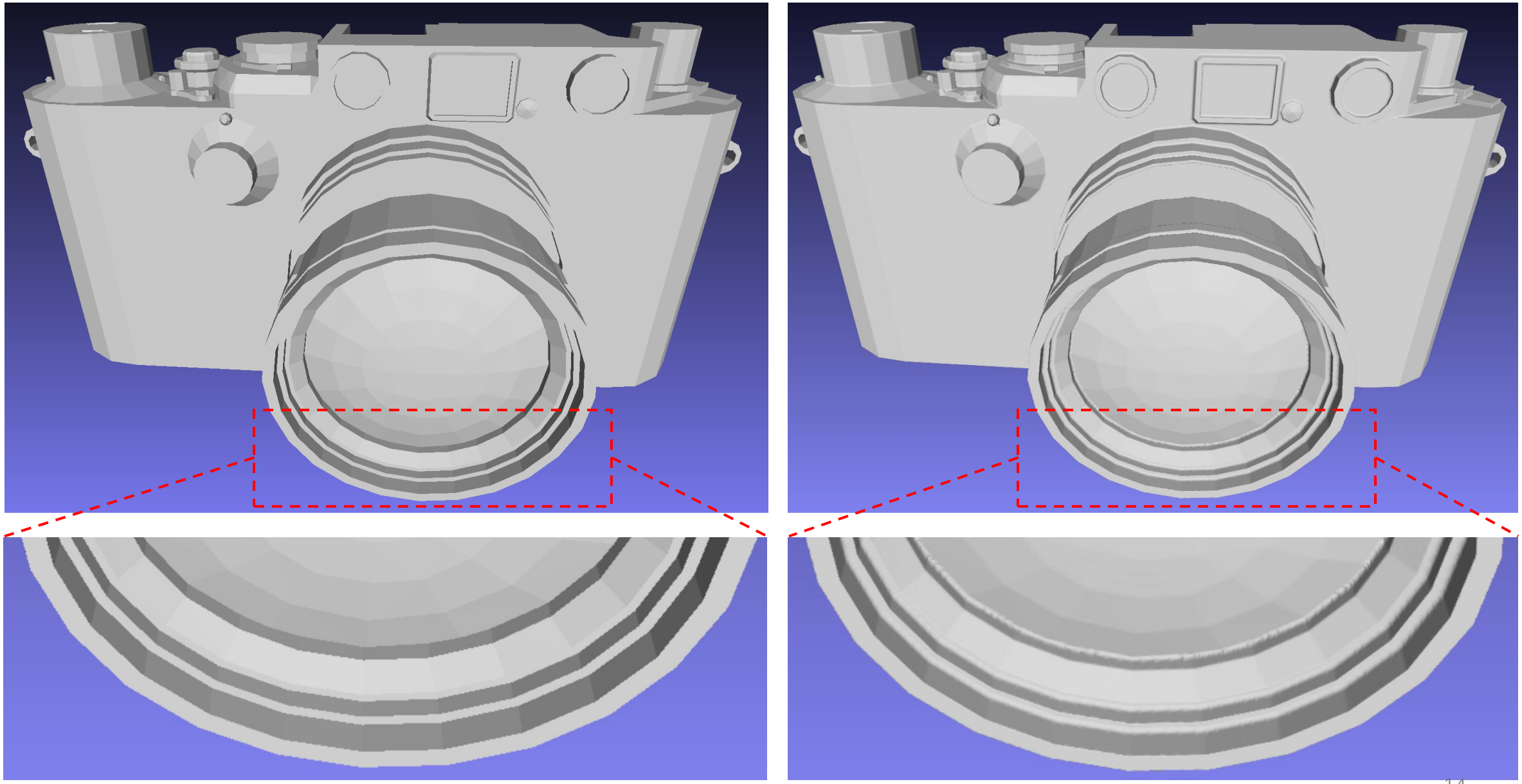
•Displace vertices slightly along the UDF gradient to improve sign estimation and geometric accuracy.

Step 4: Rendering-based refinement

•Render depth image and normal map in visible voxels.

•Compute losses between rendered and reference depth and normal maps for optimization.

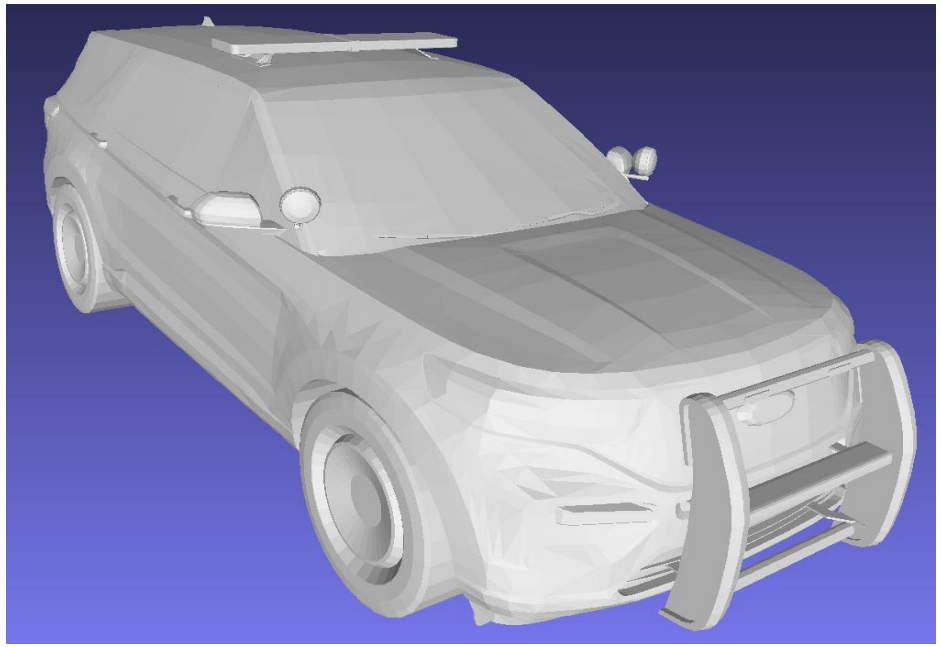
3D Data Preparation:

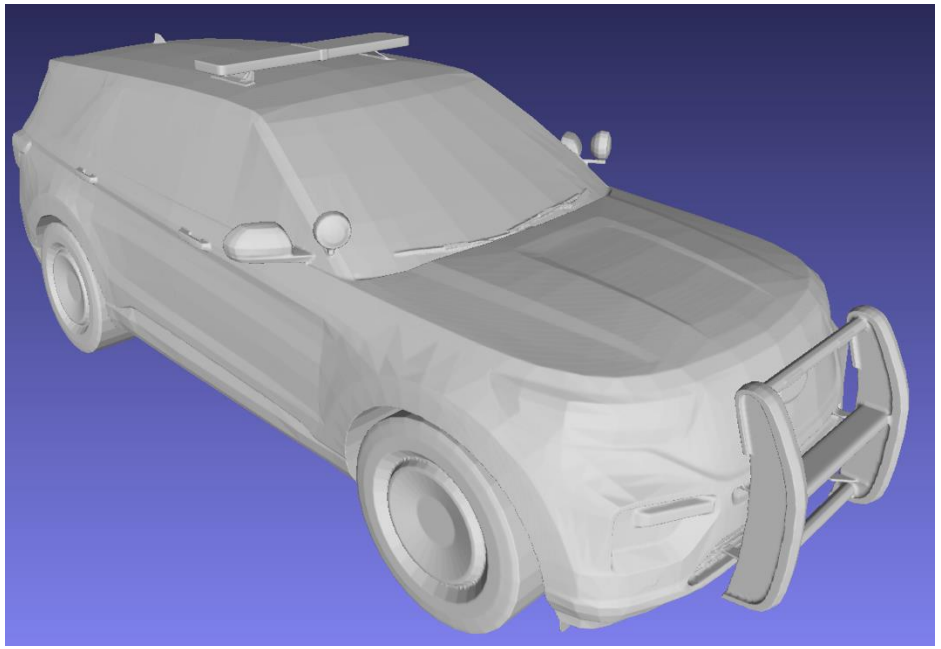


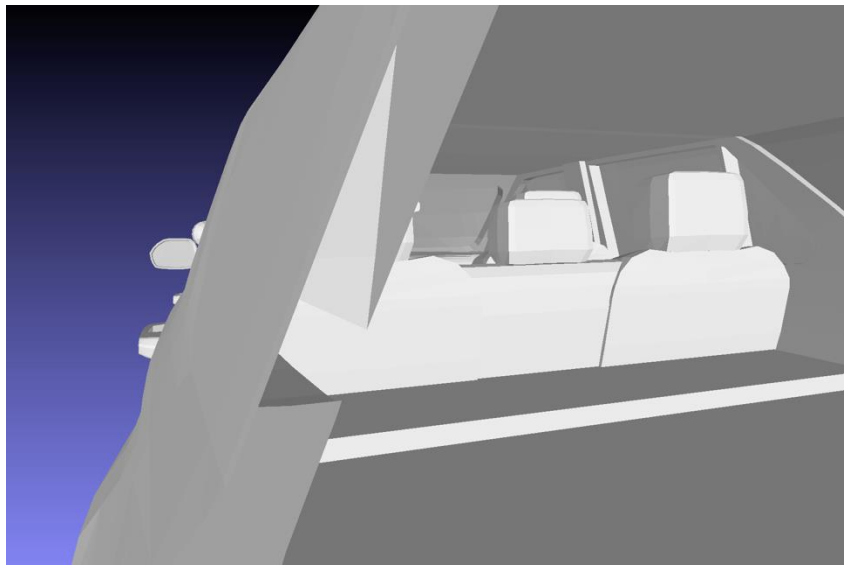
Raw Mesh

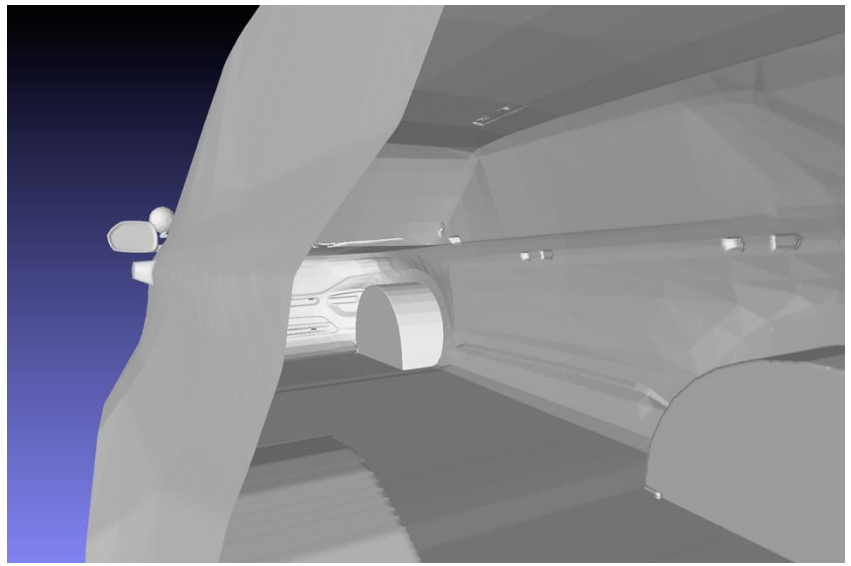
Watertight Mesh

3D Data Preparation:









Raw Mesh

Watertight Mesh

3D VAE and Representation:

Preprocessing Mesh

Input

Render

Point Cloud Sample

Watertight & Occupancy & SDF & Deformation

VAE Encoder

Latents

VAE Decoder

Reconstruction Mesh

Output

3D VAE and Representation:

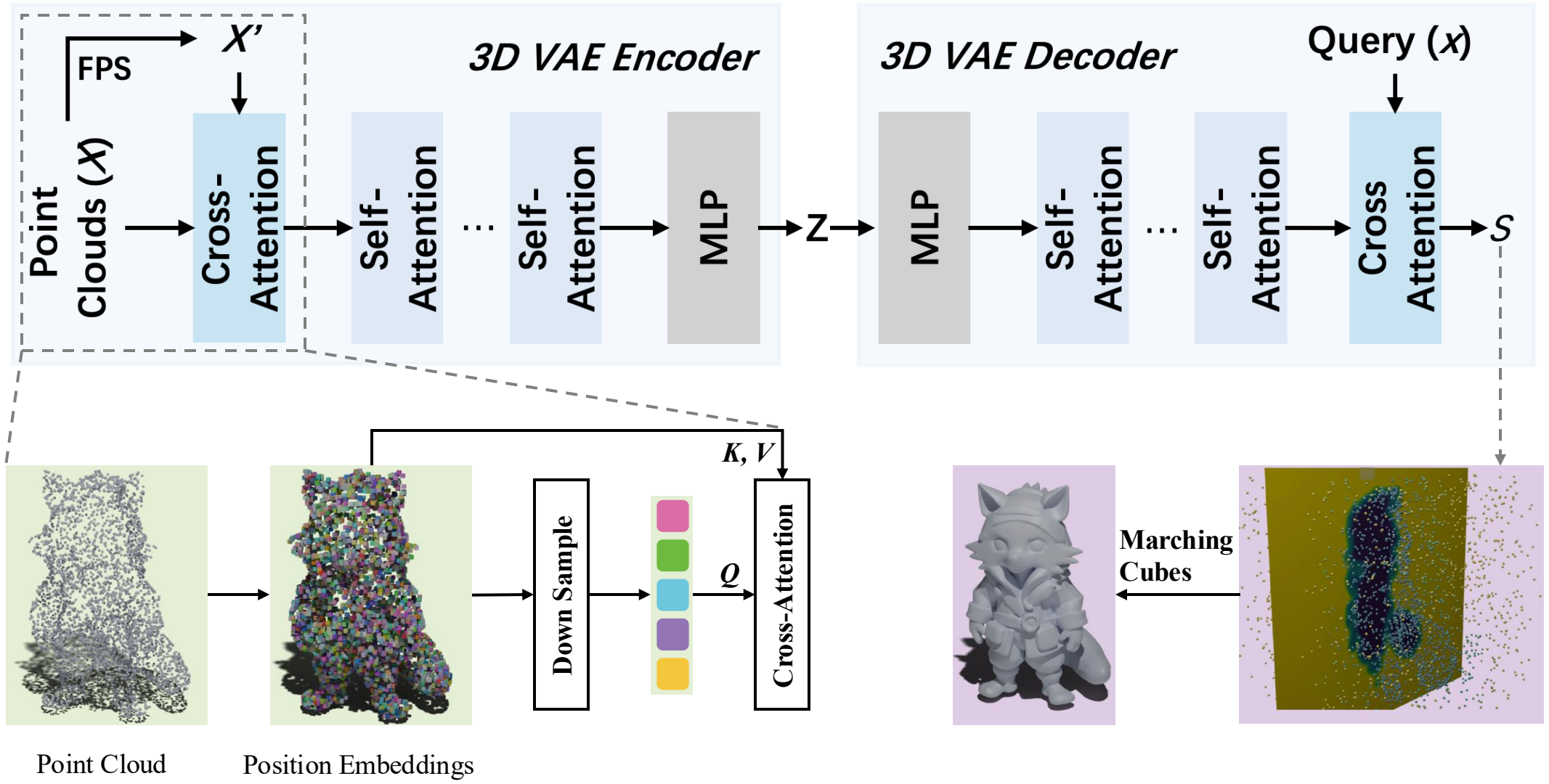
- **3DShape2VecSet VAE**
  - Architecture
  - Occupancy with BCE - 3DShape2VecSet
  - TSDF with MSE - TripoSG
  - TSDF with BCE - ShapeGen
  - Sharp Edge Input - Dora

- **3D Sparse Structure / Structured Latent VAE**
  - Basic Architecture - Trellis
  - Hybrid Representation - Trellis
  - SparseFlex VAE - SparseFlex
  - Sparse SDF VAE – Direct3D-S2
  - Sparconv VAE – Sparc3D

- **Hybrid Architecture VAE – Ultra3D**

3D VAE and Representation:

- 3DShape2VecSet VAE: Architecture



3D VAE and Representation:

- 3DShape2VecSet VAE: Occupancy with BCE

- **SDF to Occupancy:**

$$Sign(SDF(x)) = \begin{cases} +, & outside \\ -, & inside \end{cases}$$

$$\downarrow$$

$$f(x) = \begin{cases} 1, & x \text{ inside } (occupied) \\ 0, & x\ outside\ (free) \end{cases}$$

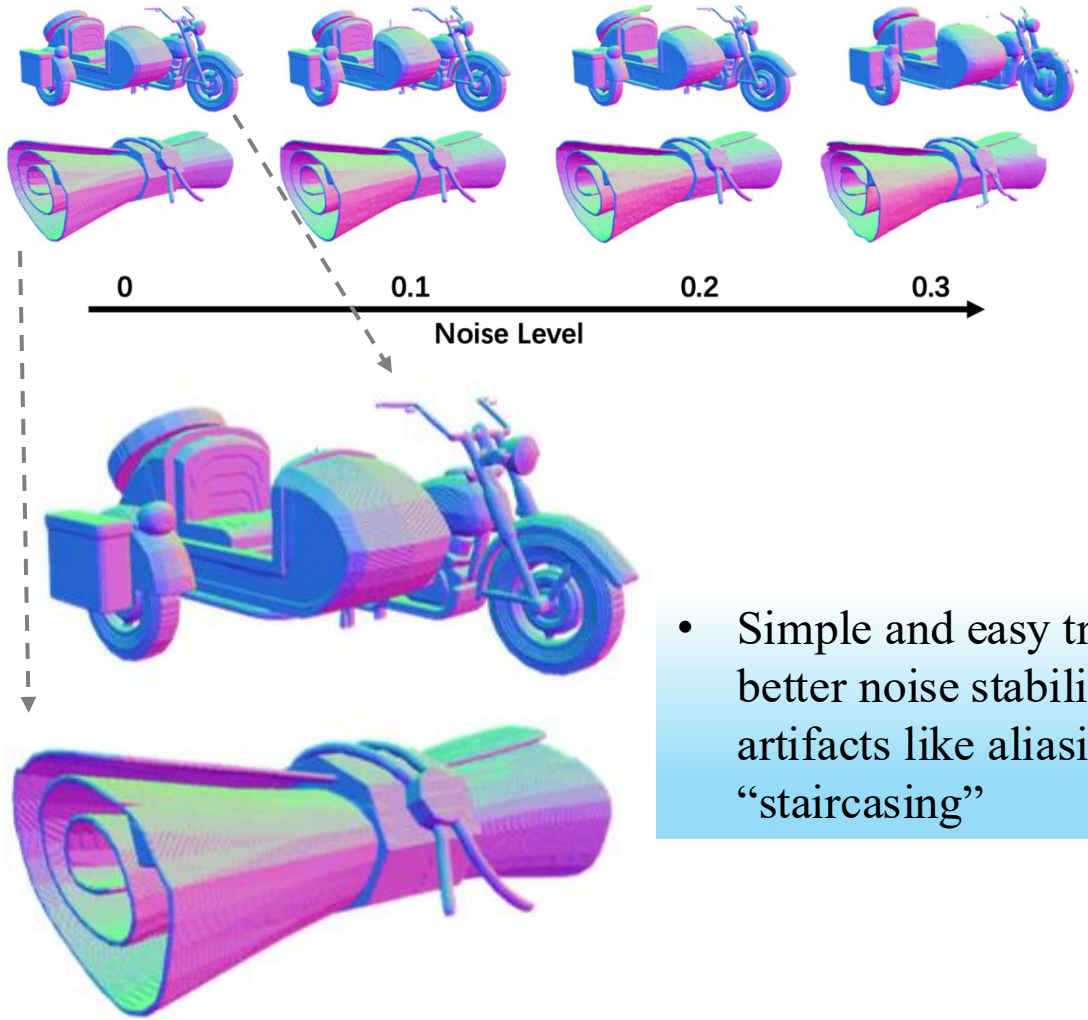
- **Occupancy with BCE supervision:**

$$L_{BCE} = (y\log(\hat{y}) + (1-y)\log(1-\hat{y})$$

$y \in \{0,1\}$, Occupancy ground truth

$\hat{y} \in [0,1]$, Predicted occupancy probability

- **Features of occupancy:**



- Simple and easy training, better noise stability, **but** with artifacts like aliasing or "staircasing"

3D VAE and Representation:

- 3DShape2VecSet VAE: TSDF with MSE

- **SDF to TSDF:**

$$y = TSDF(x) = \frac{clip(SDF(x),\ -\sigma,\ \sigma)}{\sigma}$$

Discarding distant irrelevant values

- Improves numerical stability
- Reduces storage
- Highlights essential geometry for reconstruction

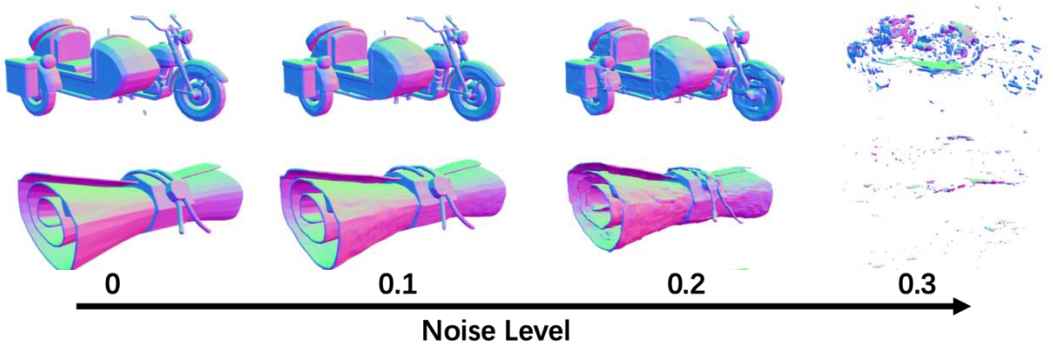
- **TSDF with MSE supervision:**

$$L_{SDF} = |y - \hat{y}| + \|y - \hat{y}\|_2^2$$

$y \in [-1, 1]$, TSDF ground truth

$\hat{y} \in [-1, 1]$, Predicted TSDF

- **Features of TSDF:**



- Better reconstruction but poor noise stability;
- Make the generation model trained with the VAE easy to generate fragmented thin shell structure.

- L1 is more robust to outliers and preserves the overall trend;
- L2 (MSE) is more sensitive to small errors and helps with fine-grained fitting;

3D VAE and Representation:

- 3DShape2VecSet VAE: TSDF with BCE

- **TSDF with BCE supervision:**

$$y = TSDF(x) = \frac{clip(SDF(x),\ -\sigma,\ \sigma) + \sigma}{2\sigma}$$

Discarding distant irrelevant values

- Improves numerical stability
- Reduces storage
- Highlights essential geometry for reconstruction

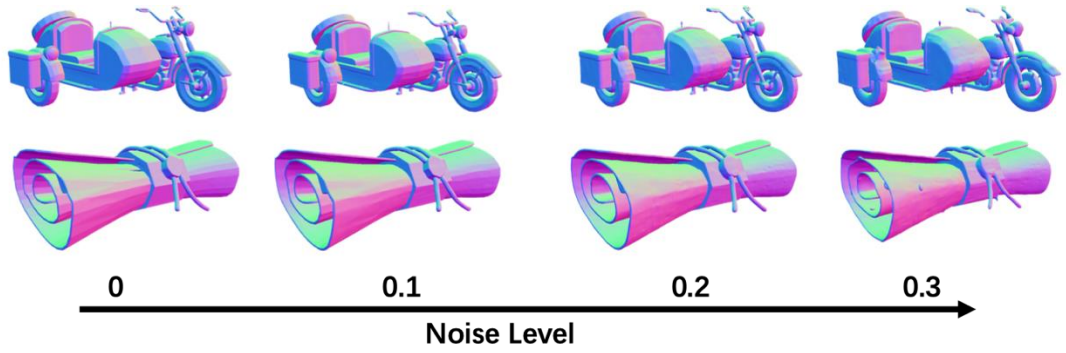
- **TSDF with BCE supervision:**

$$L_{BCE-SDF} = (y\log(\hat{y}) + (1-y)\log(1-\hat{y})$$

$y \in [0, 1]$, TSDF ground truth

$\hat{y} \in [0, 1]$, Predicted probability

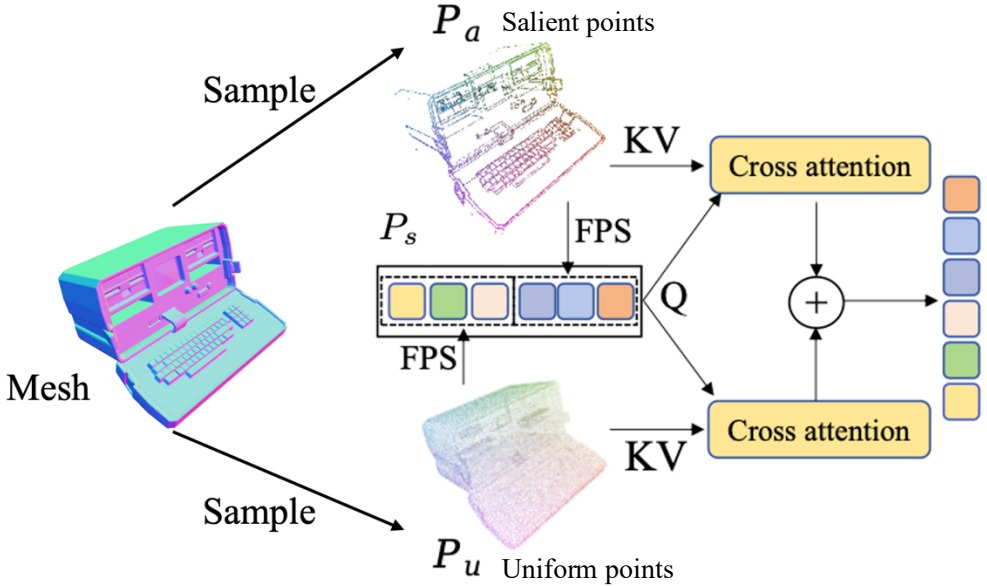
- **Features of TSDF with BCE:**

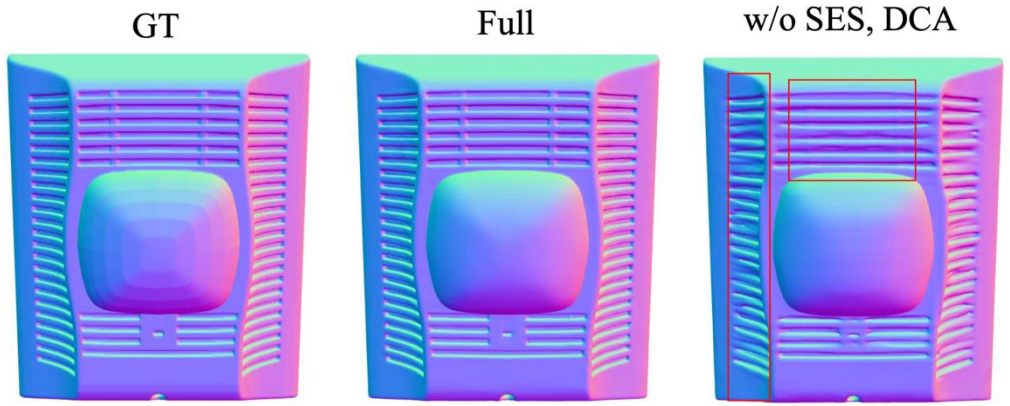


- Combines the advantages of TSDF smoothness and Occupancy noise stability

3D VAE and Representation:

- 3DShape2VecSet VAE: Sharp Edge Input

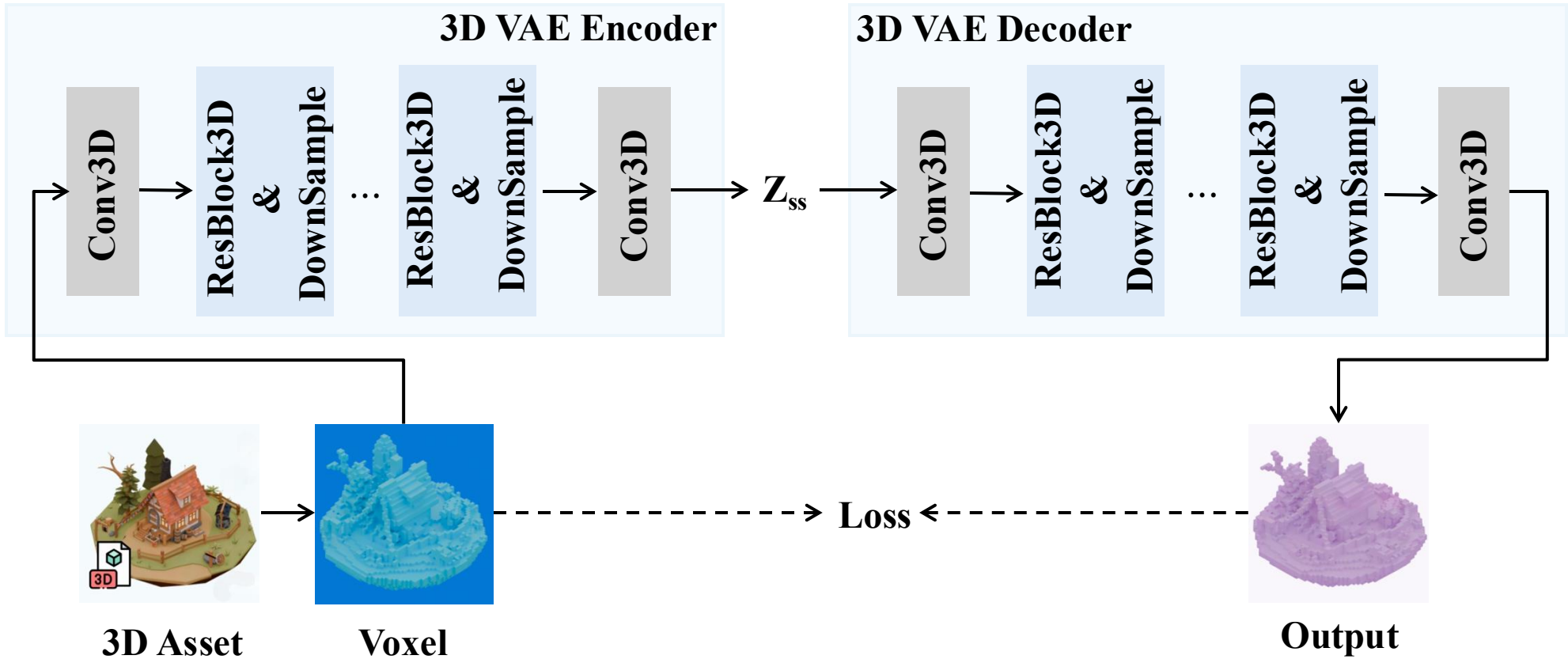




- **Sharp Edge Sampling —> Salient points**
  - **Detect sharp edges** by checking if the dihedral angle between adjacent faces is larger than a threshold.
  - **Extract salient vertices** from these edges and remove duplicates.
  - **Sample salient points**:
    - If enough vertices → use FPS to select them.
    - If too few → add extra points by interpolation.
    - If none → no salient points.
- **Dual Cross-Attention(DCA):**
  - **DownSampling**: Apply FPS to uniform points *Pu* and salient points *Pa* to get the joint query *Ps*.
  - **Dual Cross-Attention**: Compute cross-attention features separately:
    - For uniform points: *Cu*=CrossAttn(*Ps,Pu,Pu*)
    - For salient points: *Ca*=CrossAttn(*Ps,Pa,Pa*)
  - **Feature Fusion**:
    - Sum the two attention results: *C*=*Cu*+*Ca*
- **Advantage: Sharp edge and clearer details**

3D VAE and Representation:

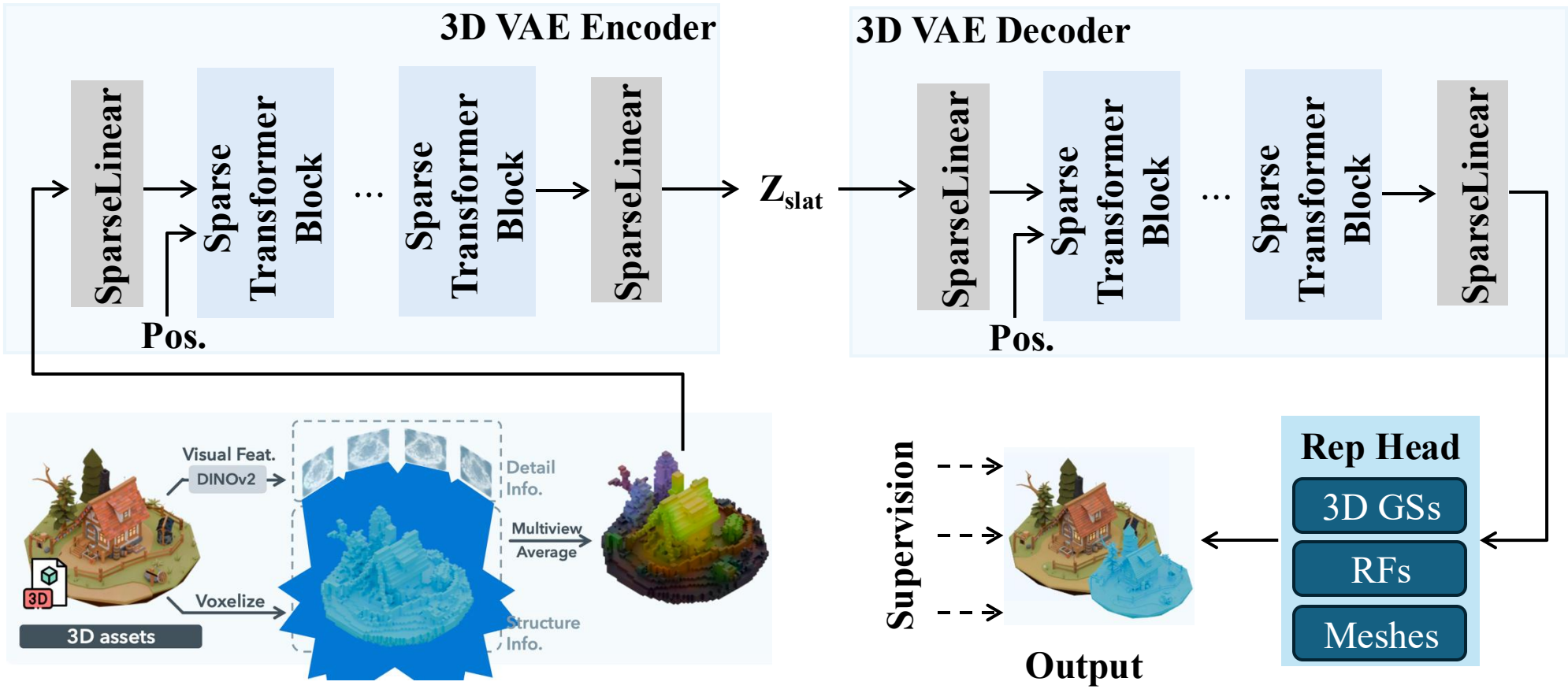
- 3D Sparse Structure VAE: Basic Architecture - Trellis



Voxel input -> Voxel output and supervision

3D VAE and Representation:

- 3D Sparse Structured Latent VAE: Basic Architecture - Trellis



Voxel with RGB feature input -> Voxel output with RGB/Depth/Normal supervision

3D VAE and Representation:

- 3D Sparse Structured Latent VAE: Basic Architecture - Trellis

These decoders share the same architecture except for their output layers:

1. 3D Gaussians:

$$\mathcal{D}_{GS} : \{(z_i, p_i)\}_{i=1}^{L} \rightarrow \{\{(o_i^k, c_i^k, s_i^k, \alpha_i^k, r_i^k)\}_{k=1}^{K}\}_{i=1}^{L}$$

$$x_i^k = p_i + \tanh(o_i^k)$$

- Latents are decoded into multiple Gaussian primitives.
- Gaussian parameters: position, color, scale, opacity, rotation.
- Position offsets constrained by voxel anchors via *tanh*.
- Losses: L1, D-SSIM, LPIPS.

2. Radiance Fields:

$$\mathcal{D}_{RF} : \{(z_i, p_i)\}_{i=1}^{L} \rightarrow \{(v_i^x, v_i^y, v_i^z, v_i^c)\}_{i=1}^{L}$$

- Represented by local radiance fields via CP-decomposition.
- Factors along x, y, z directions and color channels.
- Losses: similar to Gaussian case.

3. Meshes:

$$\mathcal{D}_{M} : \{(z_i, p_i)\}_{i=1}^{L} \rightarrow \{\{(w_i^j, d_i^j)\}_{j=1}^{64}\}_{i=1}^{L}$$

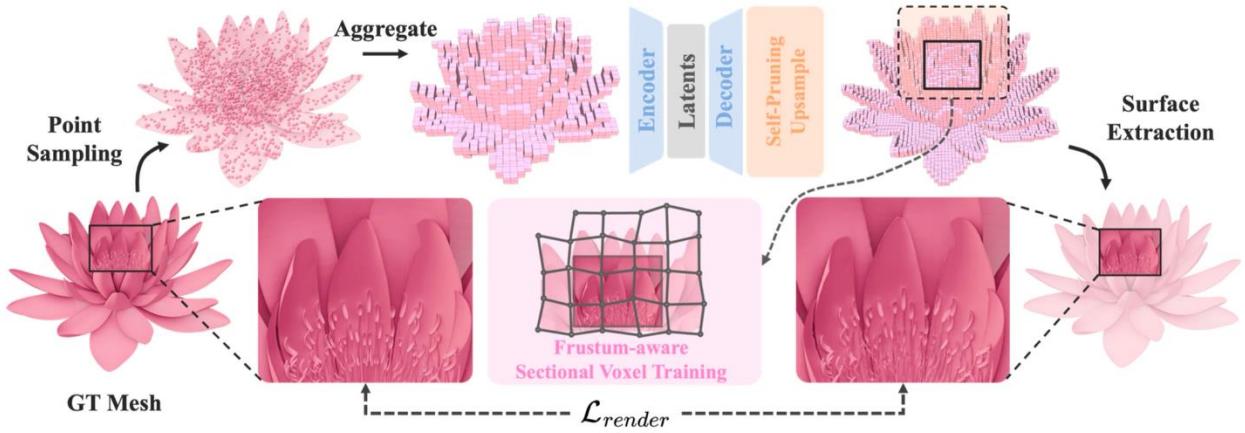
- Representation via FlexiCubes: 45 the flexible parameters w^j and 8 signed distance values d^j for each voxel.
- High-resolution output through convolutional upsampling.
- Extract meshes via marching from 0-level iso-surfaces.
- Loss: L1 on depth and normal maps.

4. Overall Comparison

- **3D Gaussians**: Point-based representation, flexible rendering, losses include perceptual metrics.
- **Radiance Fields**: Voxel-based representation using CP-decomposition for efficiency.
- **Meshes**: Explicit geometry with FlexiCubes, producing high-resolution mesh surfaces.

3D VAE and Representation:

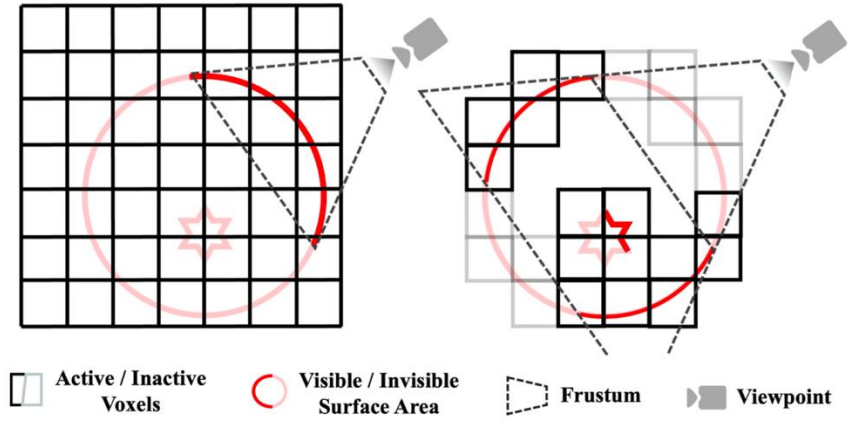
- 3D Sparse Structured Latent VAE: SparseFlex VAE - SparseFlex



- **Not RGB input –> RGB/Depth/Normal supervision; but Point cloud input –> RGB/Depth/Normal supervision**

- **Encoder-decoder:** PointNet Encoder + Basic Architecture

- **SparseFlex Representation:**
  - Sparse voxel-based structure, focusing only near the surface.
  - Components: Voxel centers, SDF + deformation at corners, interpolation weights.
  - Advantages: memory efficiency, supports open surfaces, differentiable, enables end-to-end training.

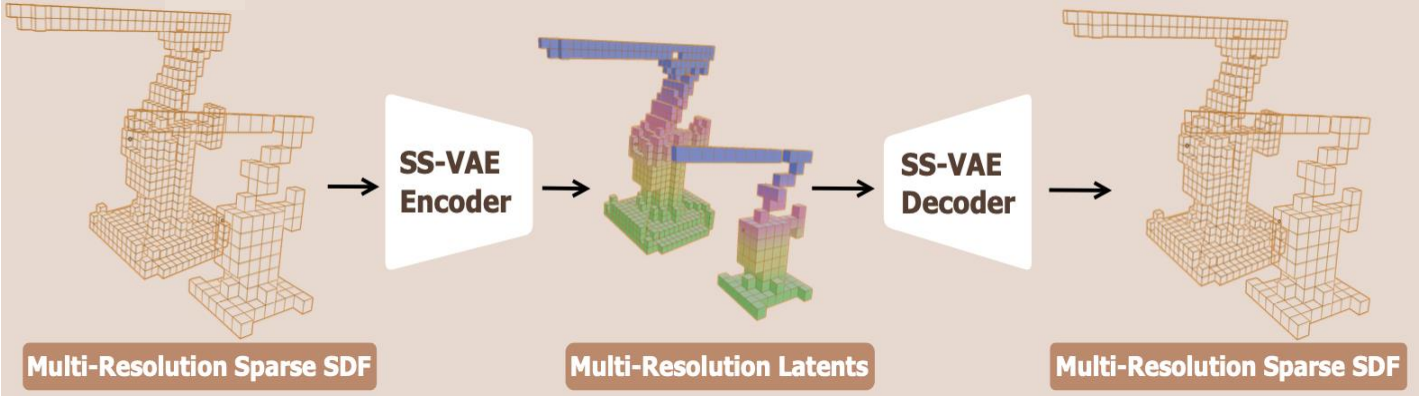


Training Strategy:

- **Frustum-aware sectional voxel training**: activates only visible voxels within the camera's frustum
- **Adaptive frustum**: dynamically adjusts clipping planes to maintain proportion of active voxels.
- **Interior reconstruction**: places camera inside the object for supervising internal structure

3D VAE and Representation:

- 3D Sparse Structured Latent VAE: Sparse SDF VAE – Direct3D-S2



- **Not RGB input –> RGB/Depth/Normal supervision; not Point cloud input –> RGB/Depth/Normal supervision; but SDF input –> SDF supervision**

- Proposes Sparse SDF VAE (SS-VAE), an end-to-end symmetric encoder-decoder framework that unifies input and supervision modality and directly operates on sparse SDF volumes.

- Focuses only on near-surface voxels ($|s(x)|<\tau$), reducing computation while maintaining precision.

Encoder: Combines sparse 3D CNN and Transformer networks.

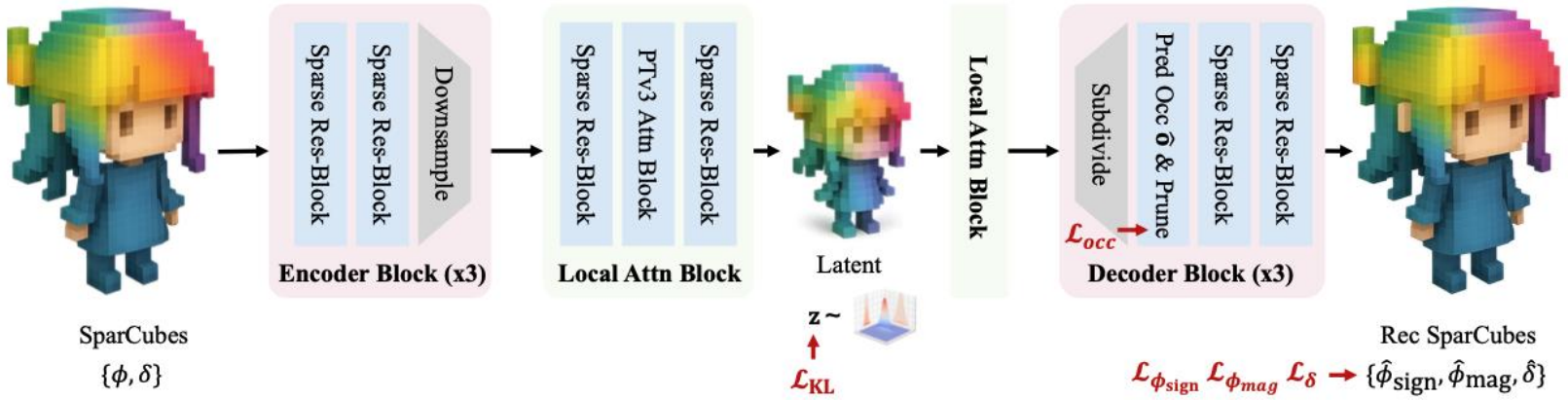
- Extracts local geometric features through residual sparse 3D CNN, average pooling, and progressively downsamples.

Decoder: Symmetric to the encoder.

- Leveraging attention layers and sparse 3D CNNs to progressively upsample the latent representation to reconstruct.

3D VAE and Representation:

- 3D Sparse Structured Latent VAE: Sparconv VAE – Sparc3D



- **Not RGB input –> RGB/Depth/Normal supervision; not Point cloud input –> RGB/Depth/Normal supervision; not SDF input –> SDF supervision; but SparCubes input –> SparCubes supervision**

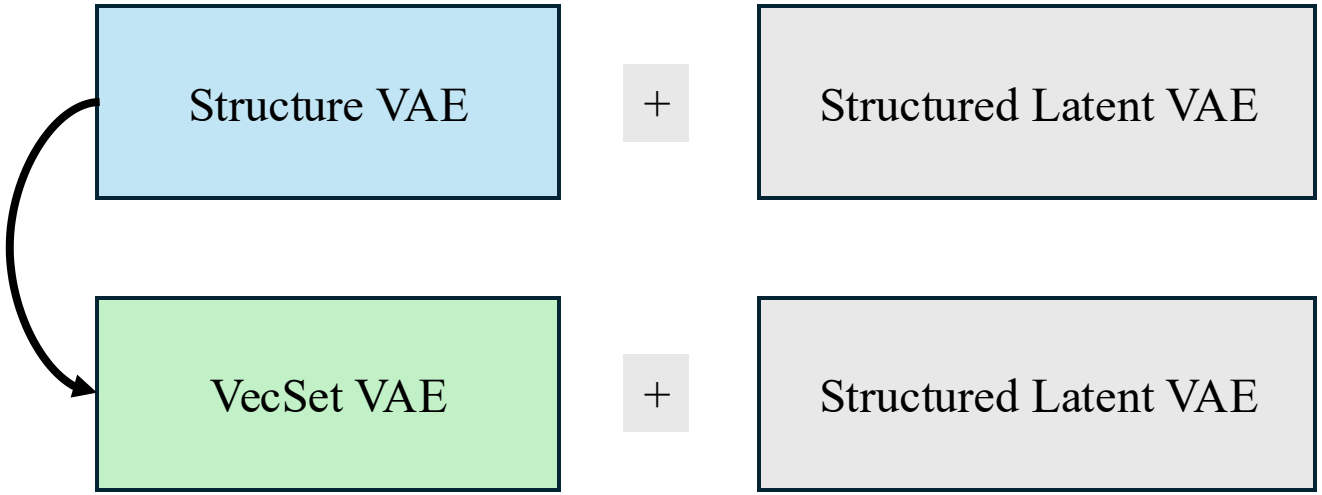
- **SparCube**: ϕ is the signed distance value and δ is the geometric deformation of sparse cube corner vertices;

- **Encoder:** 3 downsampling blocks, each with 2 sparse residual conv blocks + downsampling. Uses GroupNorm, Swish activation, voxel shuffling, linear channel compression, and a PTV3 local attention module after the final block.

- **Decoder:** 3 upsampling blocks with linear channel expansion, shuffle upsampling, 2 sparse residual blocks; outputs occupancy mask (pruned) and SparCubes parameters.

3D VAE and Representation:

- Hybrid Architecture VAE – Ultra3D



- **Analysis for High Resolution:**
  - Sparse Structure VAE struggles to balance quality and efficiency: high compression reduces tokens and harms performance, while low compression increases tokens and slows processing.
  - VecSet VAE offers compact representation and efficient generation, using only a few thousand tokens significantly fewer than structure-based sparse voxel methods.

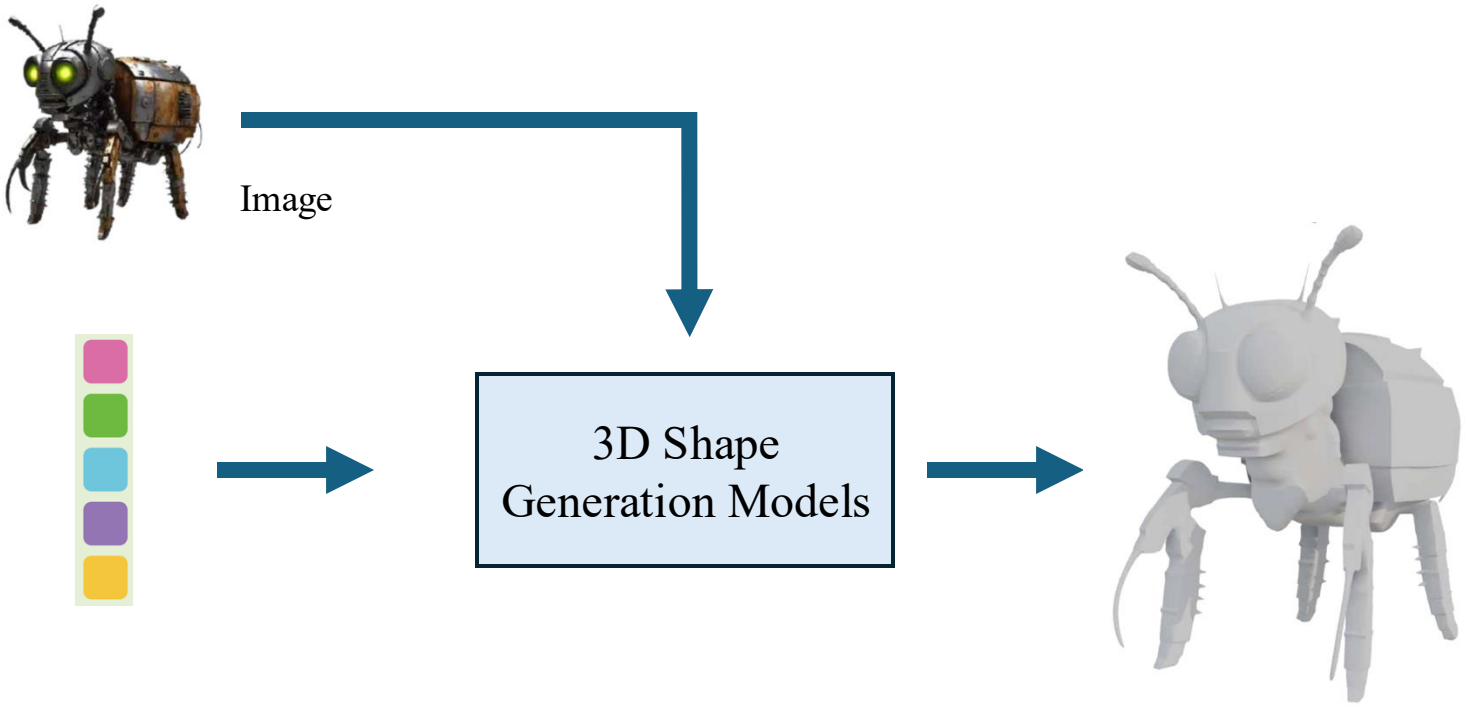
- **Method for High Resolution:**
  - VecSet VAE reconstructs mesh at 512/1024 resolution, voxelizes it into sparse voxels (64/128), combines with inputs feature, and refines geometry via a structured latent VAE.

3D VAE and Representation:

- Method Comparison

Method	VAE	Representative	Highlight
VecSet VAE	Occupancy with BCE	3DShape2Vecset /CLAY	Simple and easy training, better noise stability, but with artifacts like aliasing or “staircasing”
	TSDF with MSE	TripoSG	Better reconstruction but poor noise stability; Easy to generate fragmented thin shell structure
	TSDF with BCE	ShapeGen	Combines the advantages of TSDF smoothness and Occupancy noise stability
	Sharp Edge Input	Dora	Sharp edge and clearer details
3D Sparse Structure(SS) / Structured Latent(SL) VAE	Basic Architecture: SS+SL VAE; RGB feature input –> RGB/Depth/Normal supervision	Trellis	Basic architecture; No watertight preprocessing; Locality advantage; Better potential than VecSet VAE
	SS + SparseFlex VAE; Point cloud input –> RGB/Depth/Normal supervision	SparseFlex	High resolution; No watertight preprocessing; Better details, interior structure, open surface
	SS + Sparse SDF VAE; SDF input –> SDF supervision	Direct3D-S2	High resolution; Watertight preprocessing; Consistent input/output modality, more efficient architecture
	SS + Sparconv VAE; SparCubes input –> SparCubes supervision	Sparc3D	High resolution; Better and stability presentation; Watertight preprocessing Consistent input/output modality, more efficient architecture
Hybrid Architecture VAE	VecSet + Sparconv VAE; SparCubes input –> SparCubes supervision	Ultra3D	High resolution; Watertight preprocessing; VecSet VAE offers compact representation and efficient generation

3D Shape Generation Models:



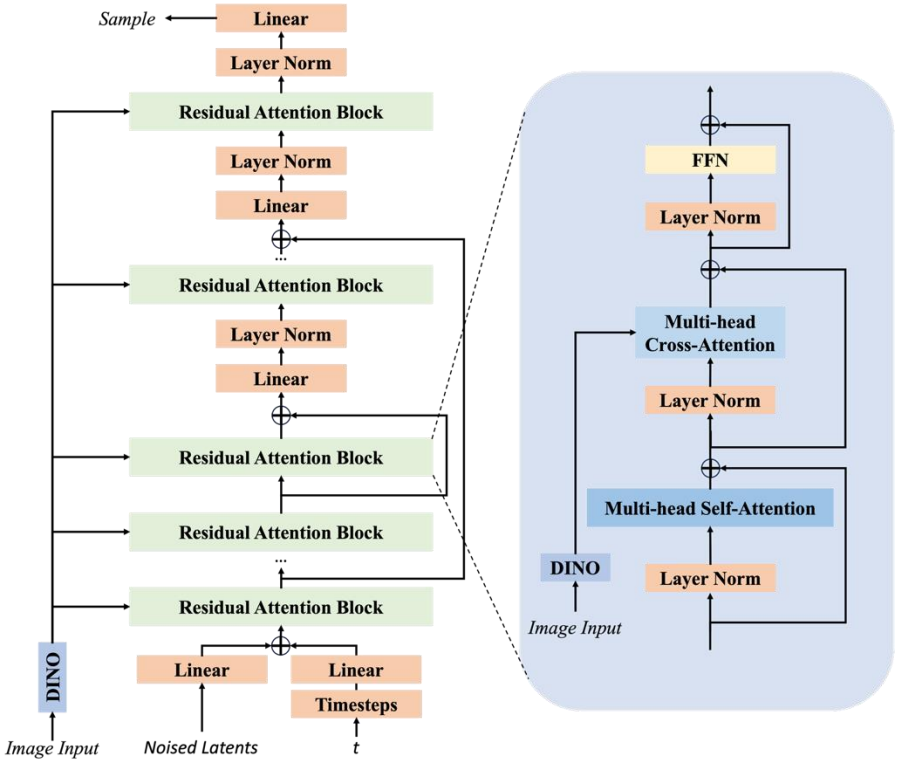
3D Shape Generation Models:

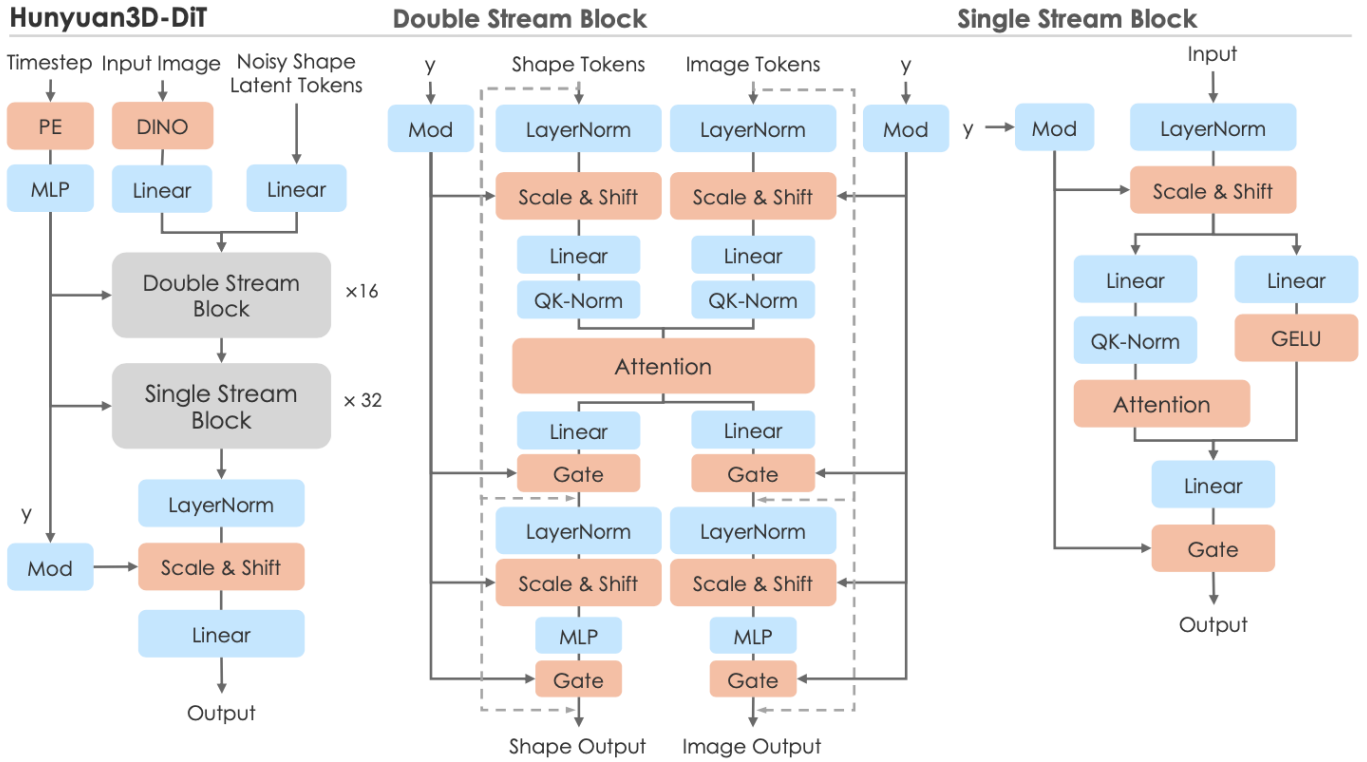
- **3DShape2VecSet Generation**
  - Architecture
  - Linear Attention
  - Noise sampling strategy

- **3D Sparse Structure / Structured Latent Generation**
  - Sparse Structure Generation Model - Trellis
  - Structured Latent Generation Model - Trellis
  - Spatial Sparse Attention DiT Model - Diect3D-S2
  - Part Attention DiT Model - Ultra3D

3D Shape Generation Models:

- 3DShape2VecSet Generation: Architecture





TripoSG Architecture:

- Using DiT backbone
- Timestep input: concatenate the timestep and latent
- Image conditioning: cross-attention input
- Using skip-connection

Hunytuan3D-2 Architecture:

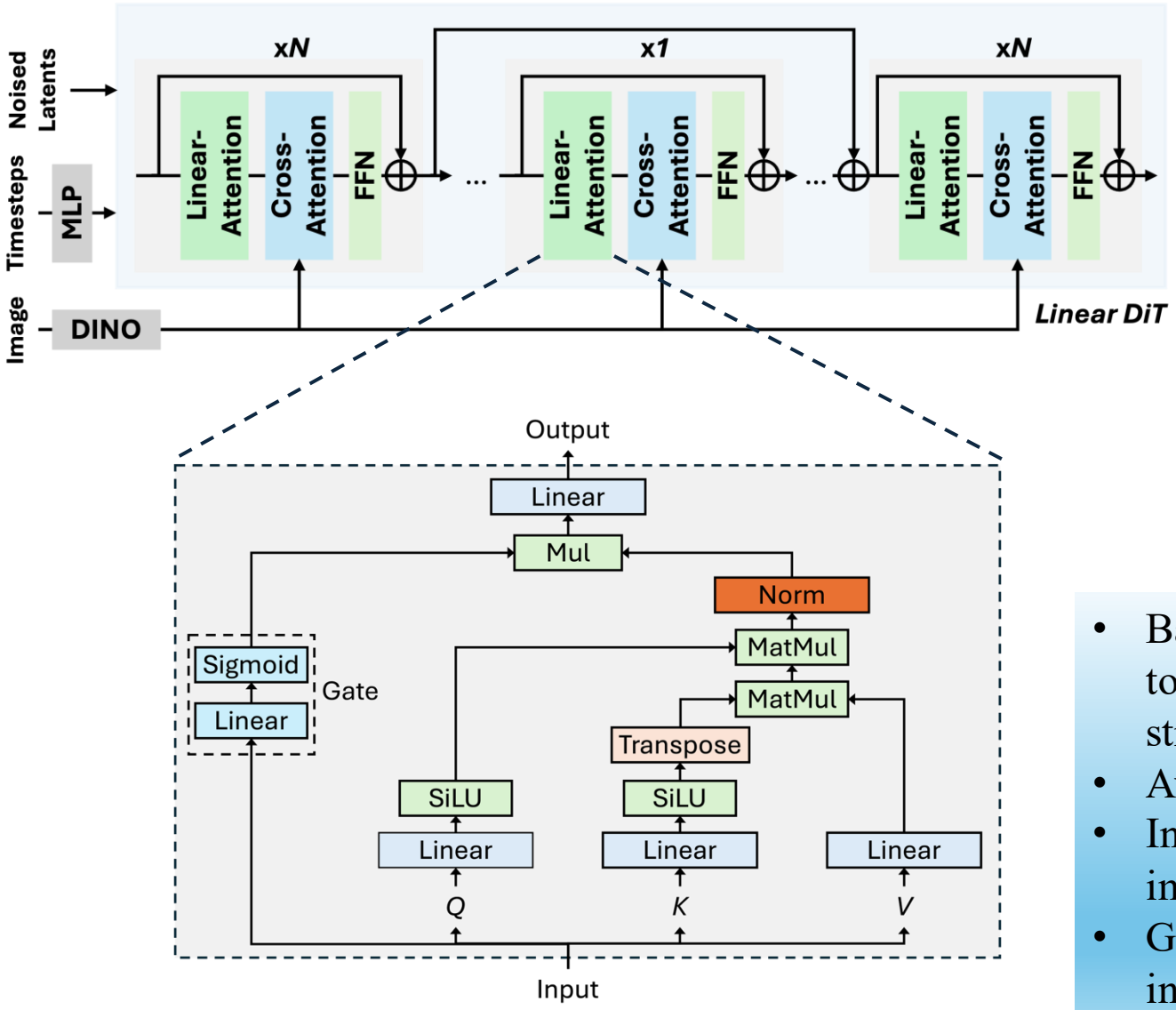
- Using MMDiT backbone
- Timestep input: AdaLN input
- Image conditioning: self-attention input

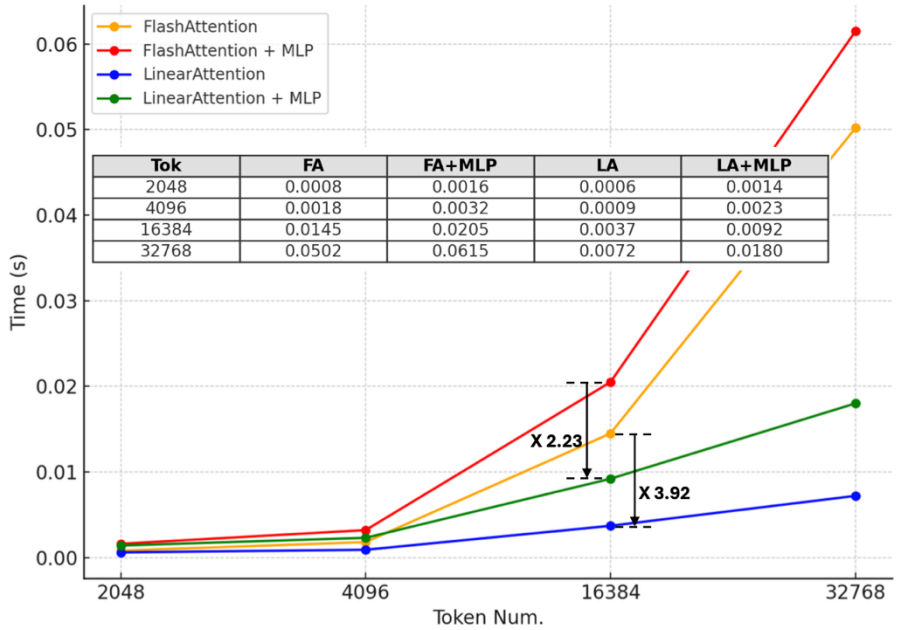
TripoSG: High-Fidelity 3D Shape Synthesis using Large-Scale Rectified Flow Models

Hunyuan3D 2.0: Scaling Diffusion Models for High Resolution Textured 3D Assets Generation

3D Shape Generation Models:

- 3DShape2VecSet Generation: Linear Attention





Tok	FA	FA+MLP	LA	LA+MLP
2048	0.0008	0.0016	0.0006	0.0014
4096	0.0018	0.0032	0.0009	0.0023
16384	0.0145	0.0205	0.0037	0.0092
32768	0.0502	0.0615	0.0072	0.0180

- Based on VecSet method, compress raw 3D data into 1D tokens, where each token inherently captures global structural information.
- Avoiding the locality degradation issues of 2D.
- Inspired by LightNet, incorporating Linear Attention in image-to-3D model to reduce time cost.
- Gating mechanism improves the stability of training and inference.

ShapeGen: Towards High-Quality 3D Shape Synthesis

3D Shape Generation Models:

- 3DShape2VecSet Generation: Noise sampling strategy

- **Formulation**

Method	Expression (x_t)	Trajectory
DDPM	$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon$	Complex nonlinear curve
EDM	$x_t = x_0 + \sigma(t)\epsilon$	Smooth nonlinear curve
Rectified Flow	$x_t = t x_0 + (1-t)\epsilon$	Straight line

- **Rectified Flow Improvements for 3D Generation**

a) Logit-Normal Sampling

- Intermediate timesteps (t≈0.5) are more difficult to predict than early or late timesteps.
- Introduce logit-normal sampling weights to emphasize these mid-range timesteps during training.

$$\pi_{\ln}(t; m, s) = \frac{1}{s\sqrt{2\pi t(1-t)}} \exp\left(-\frac{(\log(t/(1-t)) - m)^2}{2s^2}\right)$$

Where m, s : biasing location and distribution width parameter

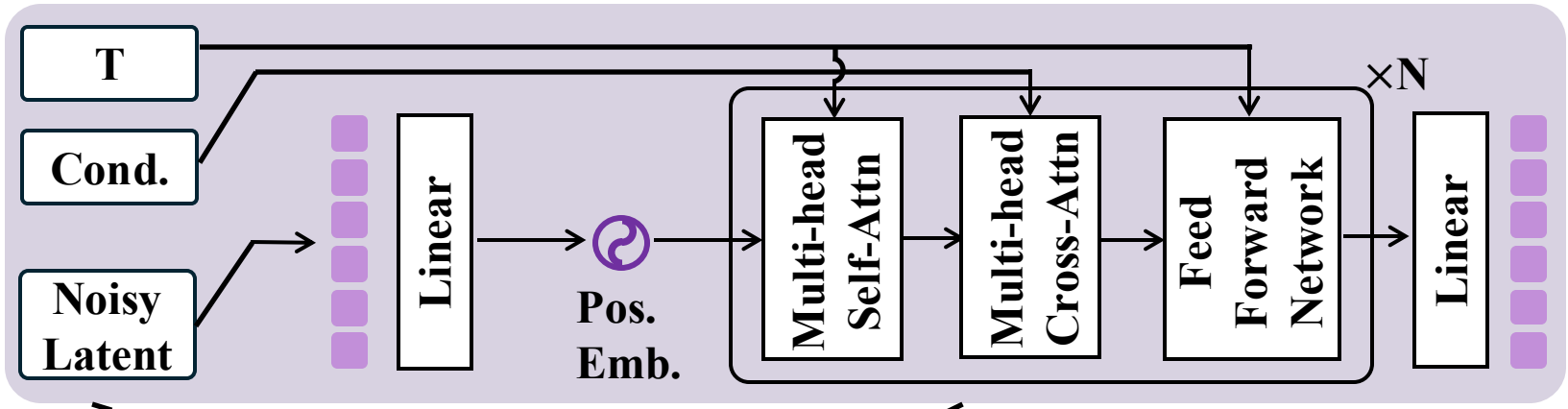
b) Resolution-Dependent Shifting of Timestep

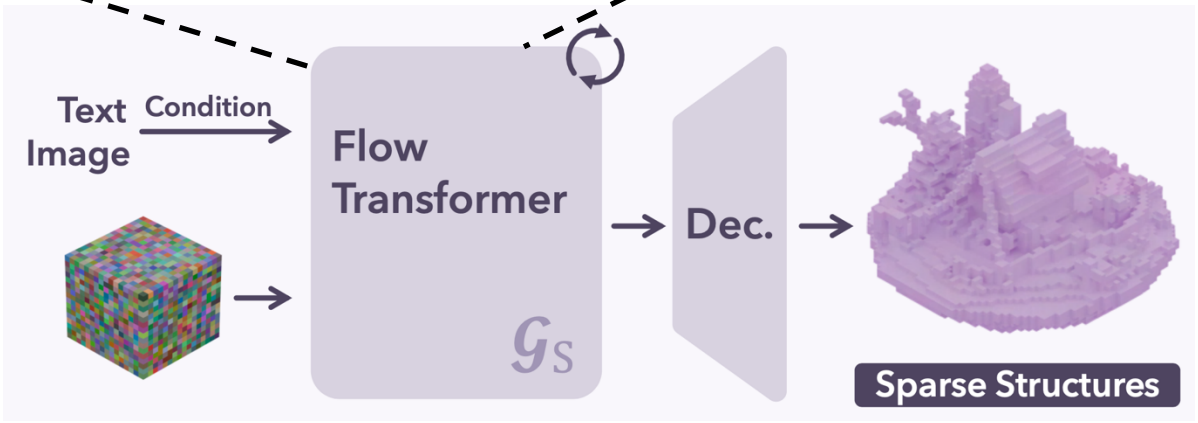
- Higher-resolution training requires more noise to sufficiently corrupt the signal. At the same timestep, the uncertainty in the noised latent decreases as resolution increases.
- Adjust timesteps when moving from base resolution n to fine-tuned resolution m, to keep uncertainty consistent.

$$\longrightarrow t_m = \frac{\sqrt{\frac{m}{n}}\, t_n}{1 + \left(\sqrt{\frac{m}{n}} - 1\right) t_n}$$

3D Shape Generation Models:

- 3D Sparse Structure Generation Model: Basic Architecture - Trellis

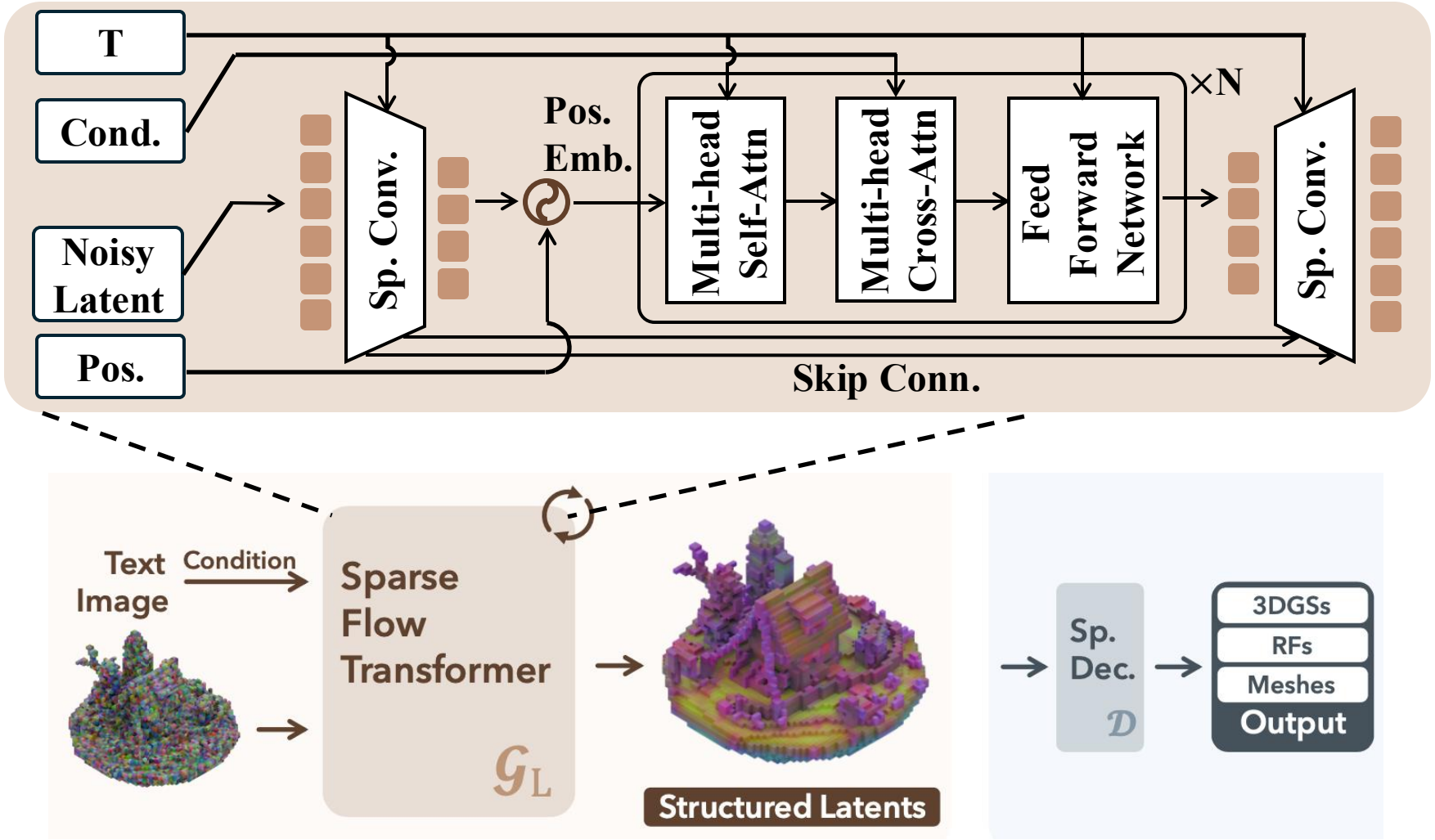




1) A standard transformer architecture
2) Dense noisy grid is serialized into a fixed-length sequence
3) Timestep injected using AdaLN along with a gating mechanism
4) Text or image condition injected through cross-attention
5) The Rectified Flow noise sampling strategy

3D Shape Generation Models:

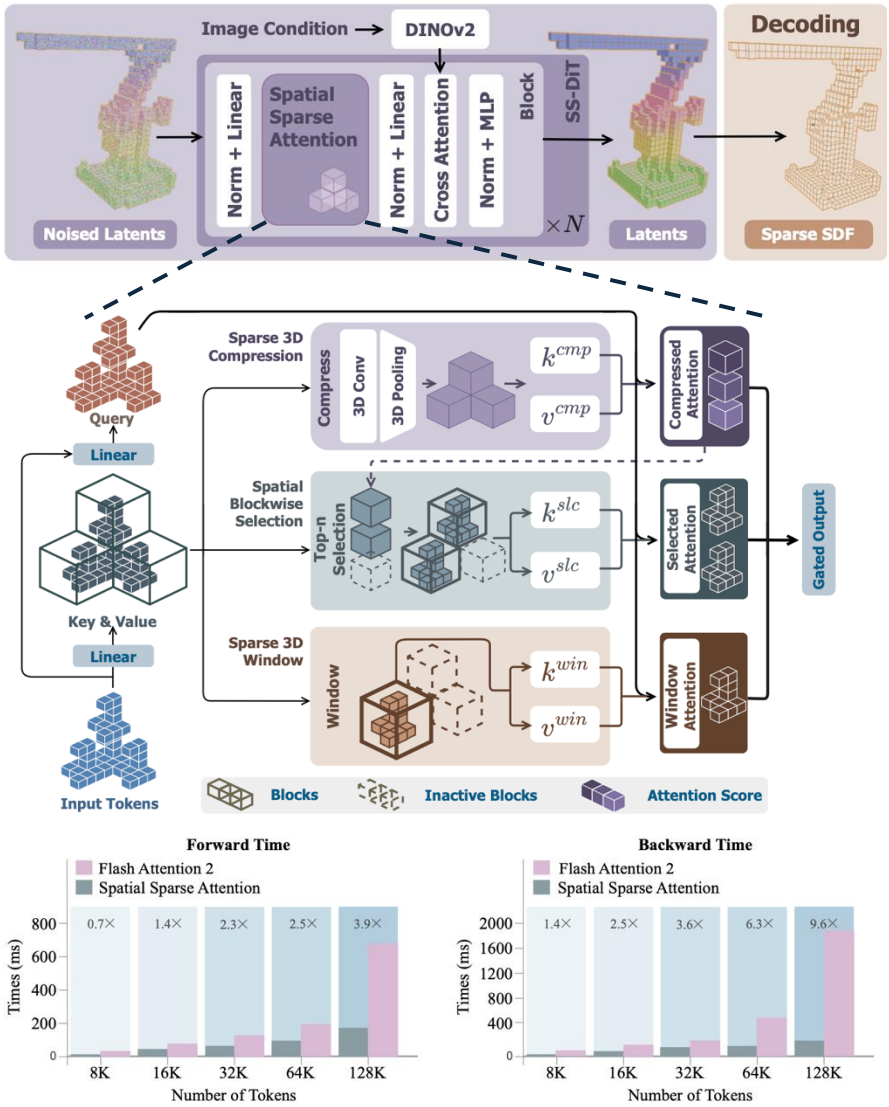
- 3D Structured Latent Generation Model: Basic Architecture - Trellis



1) Sparse convolutional downsampling block packs the input noised latent
2) Sparse noisy latent grid is serialized into a flexible-length sequence
3) Convolutional upsampling block skip connections to the downsampling block
4) Timestep injected using AdaLN along with a gating mechanism
5) Text or image condition injected through cross-attention
6) The Rectified Flow noise sampling strategy

3D Shape Generation Models:

- 3D Structured Generation: Spatial Sparse Attention DiT - Diect3D-S2



Spatial Sparse Attention:

a) **Sparse 3D Compression:**
Compress spatially coherent blocks with sparse 3D convolution and pooling, preserving global information while greatly reducing token count.

b) **Spatial Blockwise Selection:**
Further select the most relevant blocks compute the attention score between queries and compressed blocks select the top-k blocks, all tokens within the selected blocks are retained Grouped-Query Attention (GQA) is applied to improve efficiency.

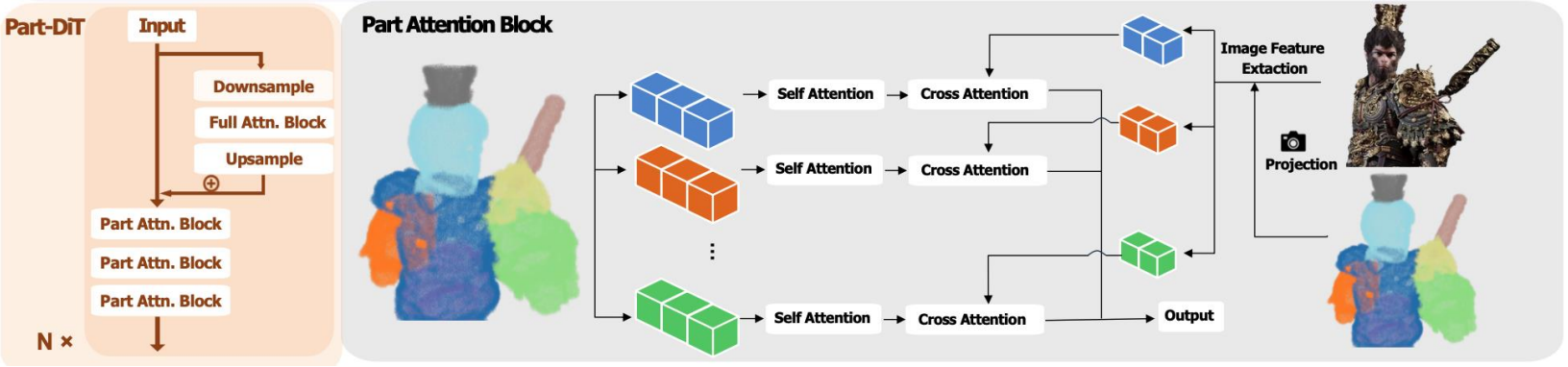
c) **Sparse 3D Window:**
Tokens are partitioned into non-overlapping windows in 3D space, within each window, active tokens are dynamically aggregated and local attention is applied.

Sparse Conditioning Mechanism: Extracts only foreground tokens from the input image for cross-modal interaction.

Direct3D-S2: Gigascale 3D Generation Made Easy with Spatial Sparse Attention

3D Shape Generation Models:

- 3D Structured Generation: Part Attention DiT Model - Ultra3D



	Part Self Attention	Part Cross Attention
Speedup Rate	6.7×	4.1×

	DiT Training	DiT Inference
Speedup Rate	3.1×	3.3×

Part Attention:

- Groups sparse voxels based on externally provided part information.
- Each voxel is assigned a part index, and attention is restricted within the same part.

Part Self Attention:

- Each token only attends to tokens within the same part, enforced by a masking strategy.
- This reduces the computation cost by nearly a factor of **K**, where K is the number of parts.

Part Cross Attention:

- Each 3D part group is projected onto the 2D image plane, and corresponding pixels inherit the same part index.
- A voxel token only attends to image tokens that share its part index.
  - Greatly reduces cross-attention cost.
  - Preserves semantic alignment between 3D and 2D tokens.

Ultra3D: Efficient and High-Fidelity 3D Generation with Part Attention

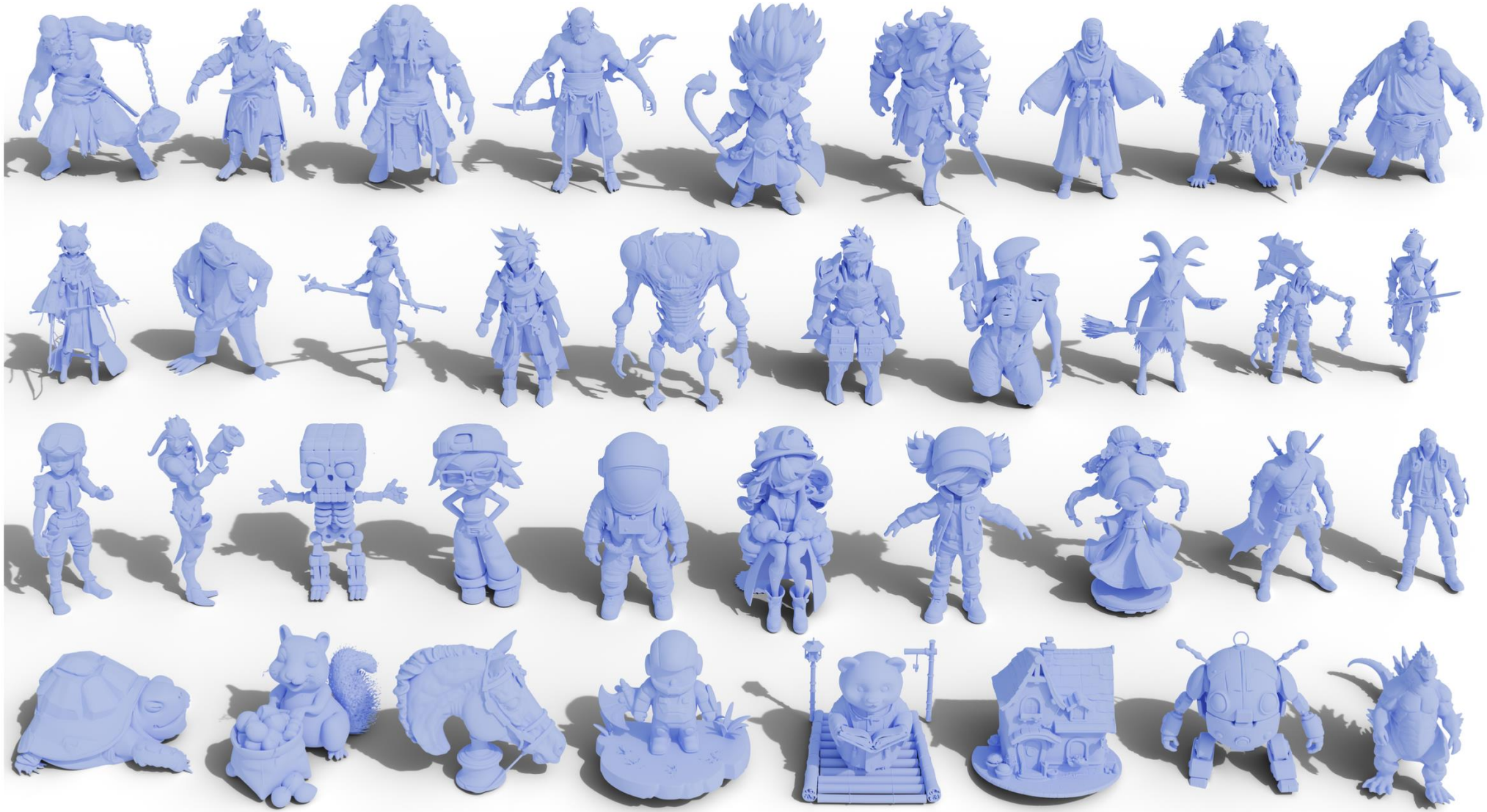
3D Shape Generation Models:

- Method Comparison

3DShape2VecSet Generation	3D Sparse Structure Generation	3D Structured Latent Generation Model
• TripoSG • Hunyuan3D-2 • ShapeGen	• Trellis	• Trellis • Direct3D-S2 • Ultra3D
• Using DiT or MMDiT backbone • Timestep input: concatenate or AdaLN input • Image conditioning: cross-attention or self-attention input • Rectified Flow • Easy training • Relatively low training cost	• Using DiT-like architecture • Timestep input: AdaLN input • Image conditioning: cross-attention input • Rectified Flow • Weakly generation training performance under VAE with high compression ratio	• Using DiT-like architecture • Timestep input: AdaLN input • Image conditioning: cross-attention input • Rectified Flow • Spatial Sparse Attention for efficient training • Part Attention for efficient training • Long sequences of tokens lead to high training costs

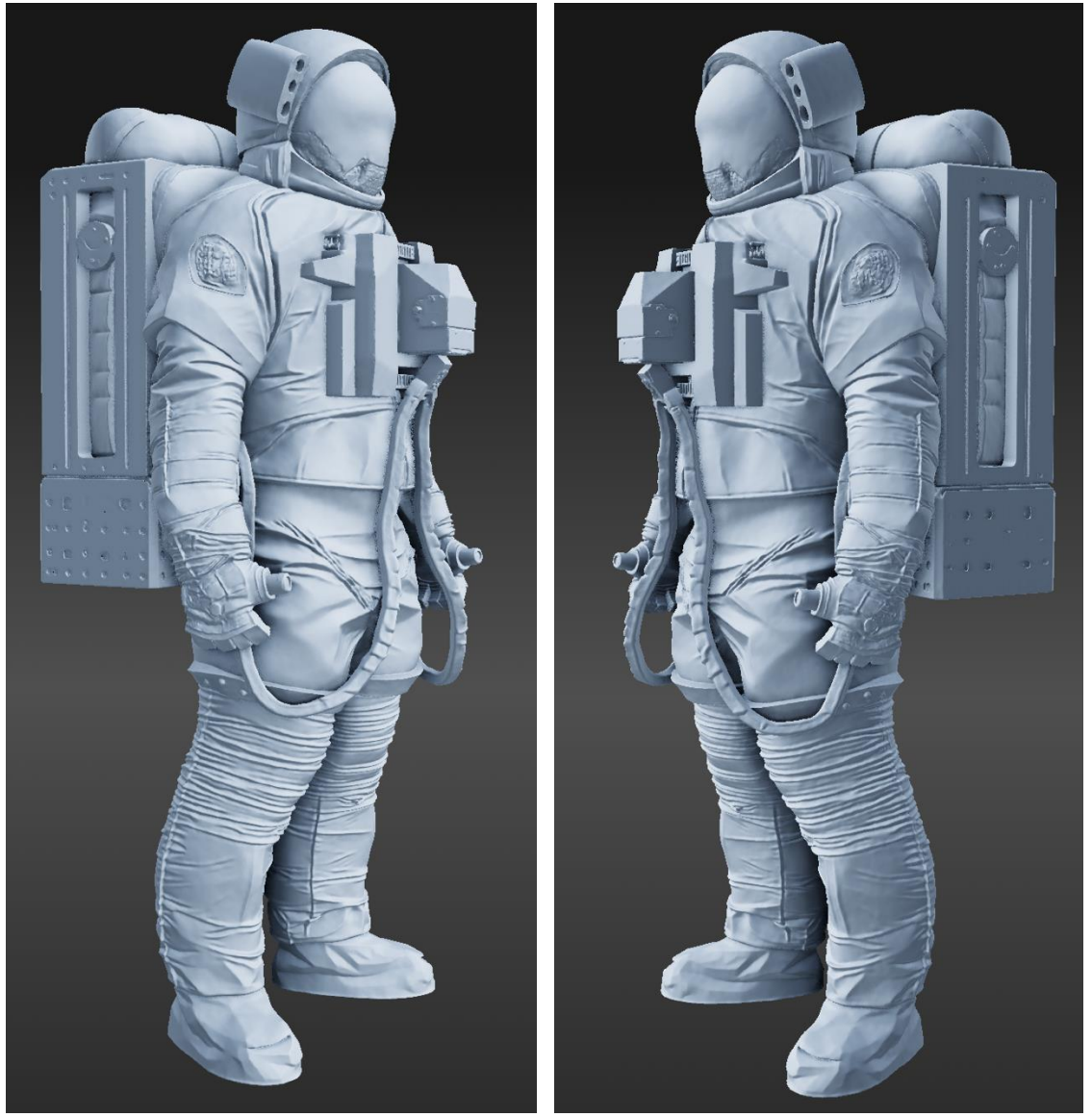
3D Shape Generation Models:

- Image-to-3D Results

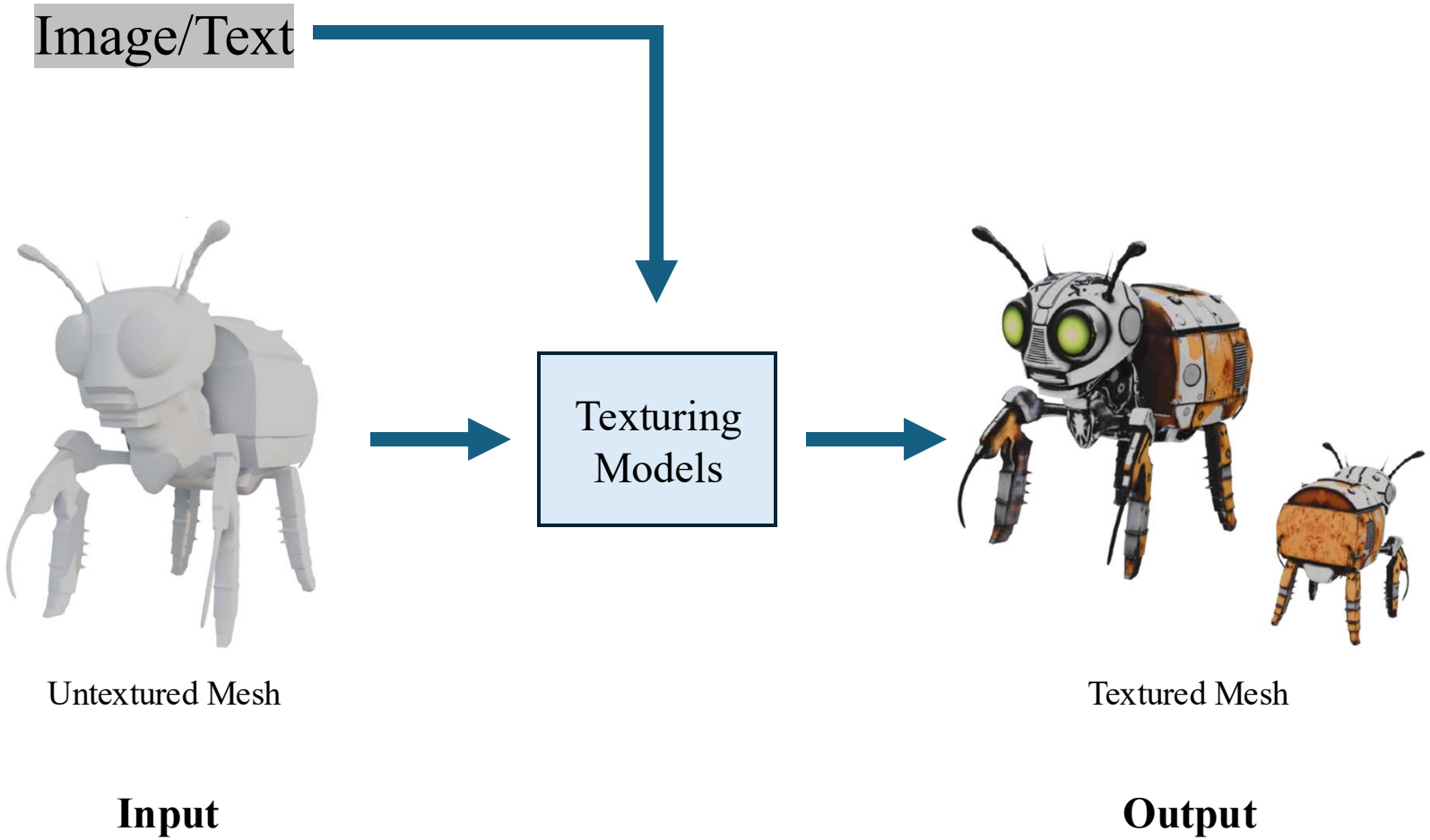


3D Shape Generation Models:

- Image-to-3D Results



3D Texture Generation Models:

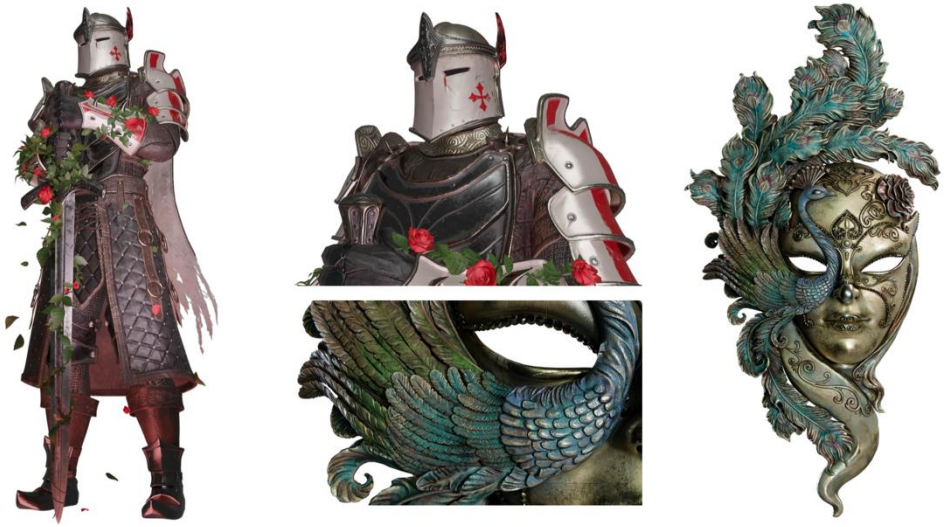


3D Texture Generation Models:

- Texture Dataset
- Texture Representation
- Multi-view Back-Projection + UV Inpainting
  - Meta 3D TextureGen
  - TripoSG
- Single-view Back-Projection + UV Diffusion
  - TEXGen
- Multi-view Back-Projection + Large Texturing Model
  - UniTEX
- Method Comparison
- Generation Results

3D Texture Generation Models:

- Texture Dataset



Dataset	# Objects	Type	PBR Material	High-Resolution Texture
ShapeNet [3]	51K	synthetic	✗	✗
3D-Future [11]	10K	synthetic	✗	✓
ABO [6]	8K	synthetic	✓	✓
OmniObject3D [18]	6K	real	✗	✗
GSO [10]	1K	real	✗	✓
DTC [9]	2K	real	✓	✓
Objaverse [8]	818K	both	(✓)*	(✓)*
Objaverse-XL [7]	10M	both	(✓)*	(✓)*
TexVerse (Ours)	858K	both	(✓)*	✓

TexVerse: 858K unique high-resolution textured 3D objects curated from Sketchfab

- Selected models with texture resolutions ≥ 1024 pixels.
- Excluded models tagged or described with "NoAI".
- Retained only models under distributable Creative Commons licenses via Sketchfab's API.

Self-Filtering: Filter texture data from Objaverse(-XL) in a similar way to geometry data filtering

- Filter out the untextured data.
- Filter out the large flat, bad animation, and multi-object texture data.
- Labeling annotation and training a score model for filtering to get final high-quality textured 3D data.

TexVerse: A Universe of 3D Objects with High-Resolution Textures. 2025.09. Arxiv

3D Texture Generation Models:

- Texture Representation

Essential nature of surface: Can be regarded as a 2D signal embedded in 3D space.

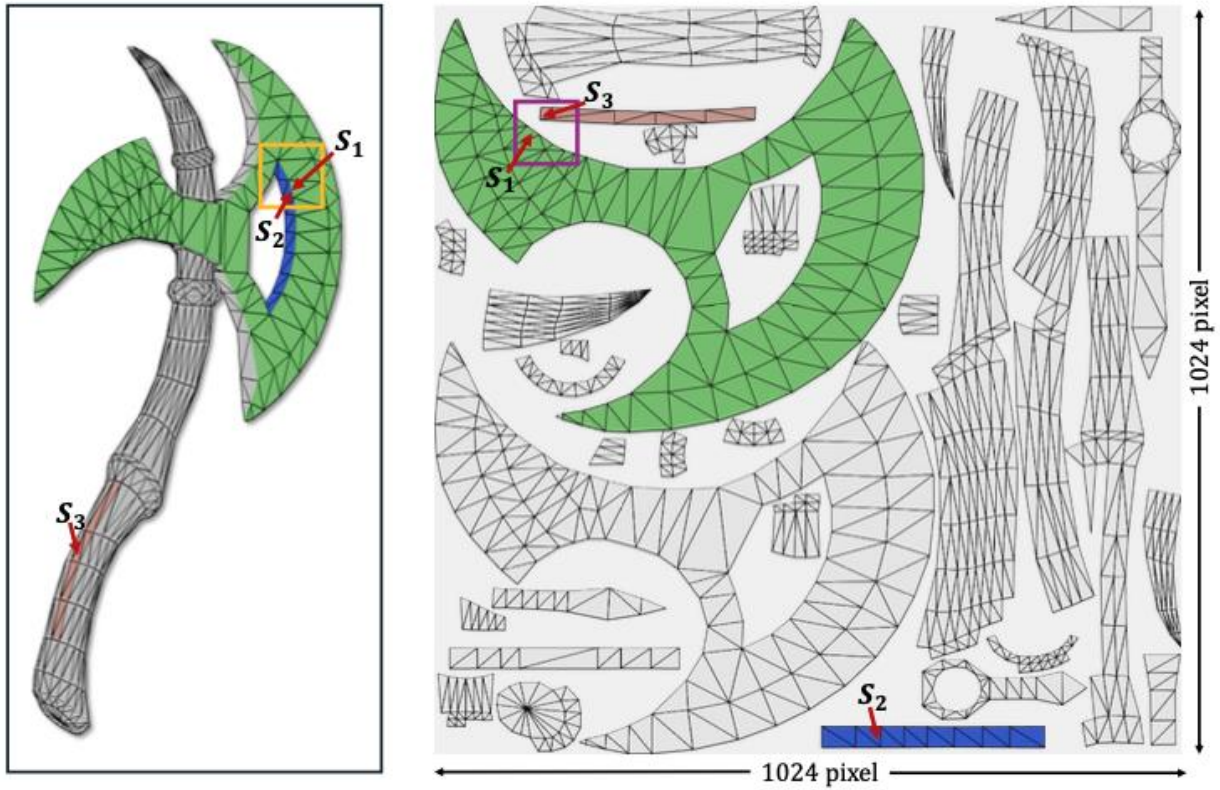
Traditional operation: UV mapping, which flattens the 3D structure into a compact 2D representation.

Main function: Allows 3D attributes (e.g., textures) to be organized and represented on a 2D plane.

UV Attributes:

- The 2D UV space effectively captures local neighborhood dependencies within each island, improving texture generation efficiency.

- However, it inevitably loses global consistency across different islands due to the fragmentation of UV mapping.

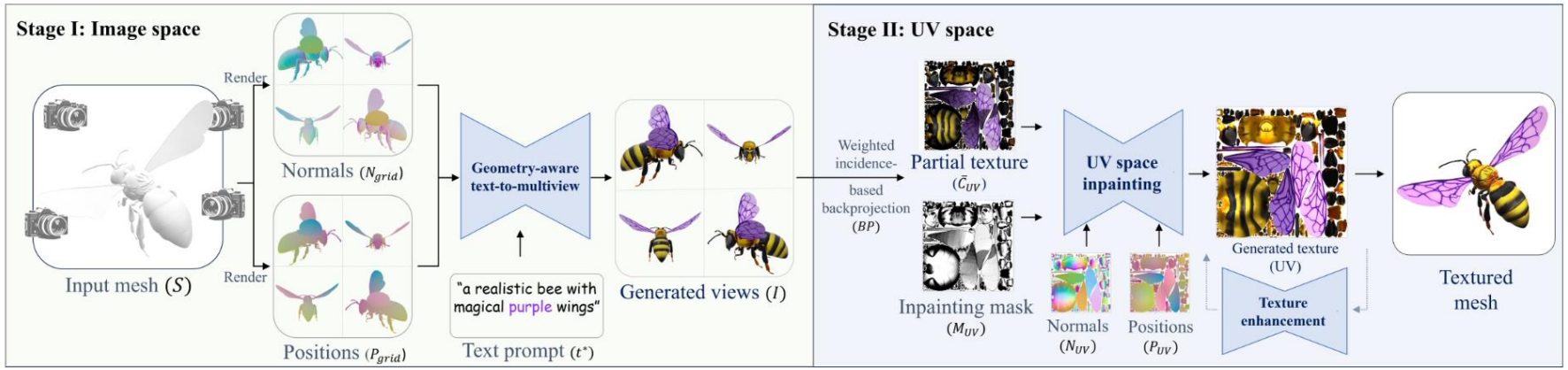


(a) A triangle mesh (b) UV map

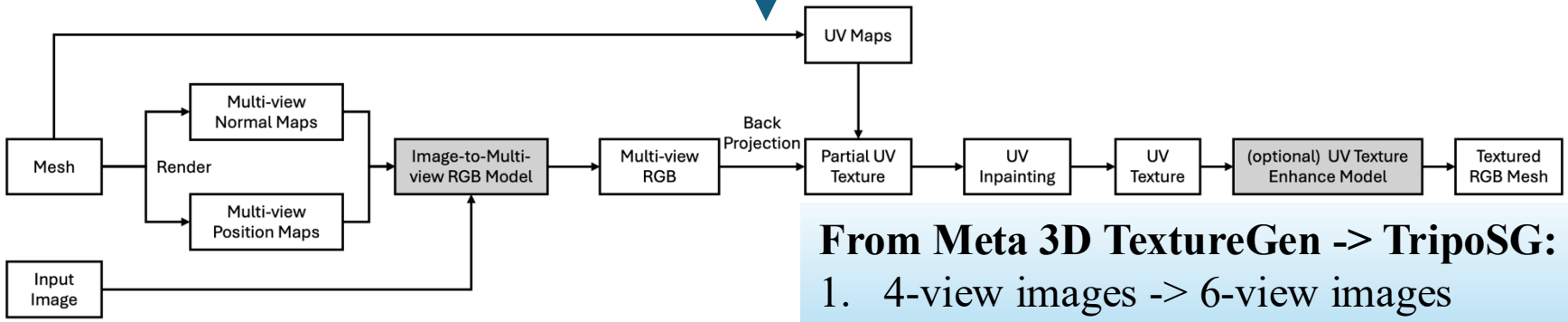
TEXGen: a Generative Diffusion Model for Mesh Textures. 2024.11. Siggraph Asia.

3D Texture Generation Models:

- Multi-view Back-Projection + UV Inpainting



[1] Meta 3D TextureGen



[2] TripoSG

Texturing Pipeline:

- Image-to-Multiview Generation
- UV-Space Back-Projection from Multiview
- UV-Space Inpainting
- Super-Resolution Enhancement (Optional)

From Meta 3D TextureGen -> TripoSG:

1. 4-view images -> 6-view images
2. Text condition -> Image+Text condition
3. EMU based prior model -> SDXL based prior model
4. Width and height stitching -> Channel concatenating
5. Low resolution multi-view -> High resolution multi-view
6. UNet based diffusion model for UV inpainting -> LaMa based UV inpainting

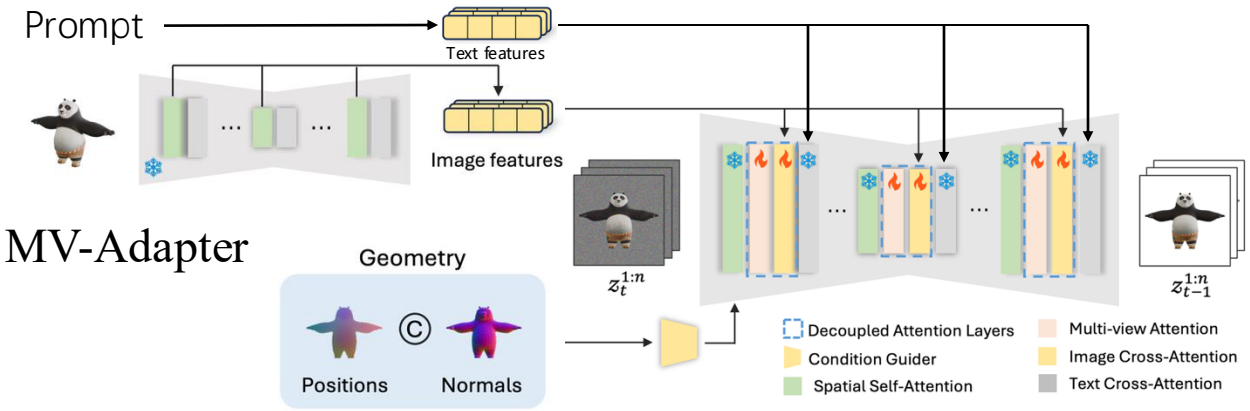
[1] Meta 3D TextureGen: Fast and Consistent Texture Generation for 3D Objects. 2024.07. Arxiv
[2] TripoSG: High-Fidelity 3D Shape Synthesis using Large-Scale Rectified Flow Models. 2025.02. Arxiv

3D Texture Generation Models:

- Multi-view Back-Projection + UV Inpainting

Image-to-Multiview Training:

- Fine-tune SDXL with high-quality textured 3D data.
- Input: a single-view RGB image and Prompt + multi-view (6-view) normal and position maps.
- Output: multi-view (6-view) RGB images.



1) Pre-trained U-Net to encode the reference image
2) Prompt encoder provides prompt information
3) Condition guider encodes geometry condition
4) Decoupled attention layers contain multi-view attention and image cross-attention

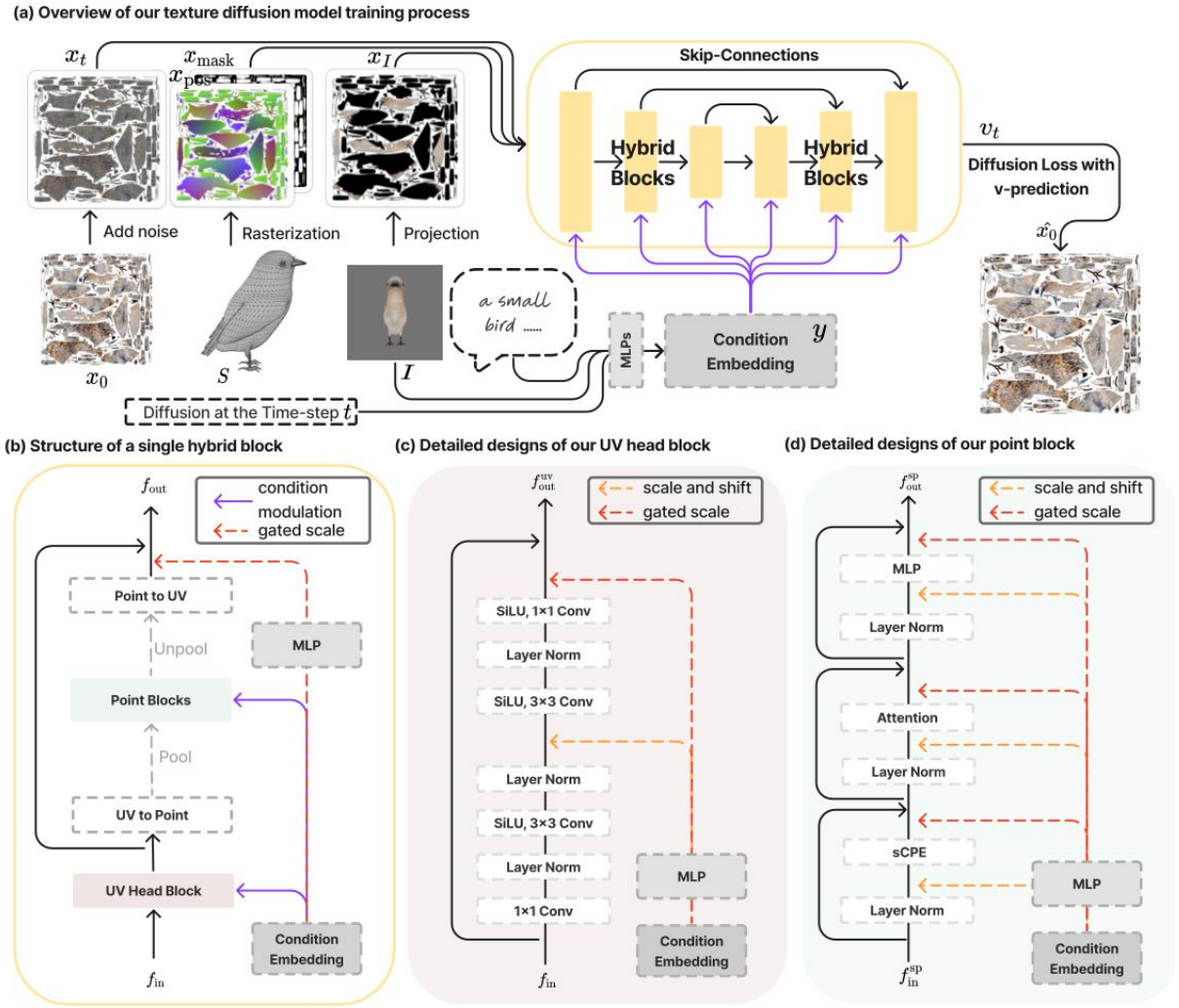
UV-Space Back-Projection & Inpainting: Convert the generated multi-view RGB into UV space

- Use the position maps and normal maps to transform into the UV space to obtain the correspondence with the UV normal map and UV position map.
- Perform **weighted incident-based back-projection** to back-project multi-view RGB into an initial UV texture.
- Apply **LaMa inpainting** in UV space to fill sparse regions caused by occlusions.

[1] MV-Adapter: Multi-view Consistent Image Generation Made Easy
[2] Meta 3D TextureGen: Fast and Consistent Texture Generation for 3D Objects
[3] LaMa: Resolution-robust Large Mask Inpainting with Fourier Convolutions

3D Texture Generation Models:

- Single-view Back-Projection + UV Diffusion: TEXGen



Can generation be performed directly based on UV space?

TEXGen: Propose a hybrid 2D-3D network that learns both fine 2D details and global 3D consistency.

Conditional Inputs -> Output:

- Single-view image + Text prompt + Partial texture + UV space position/mask maps → Guide denoising.

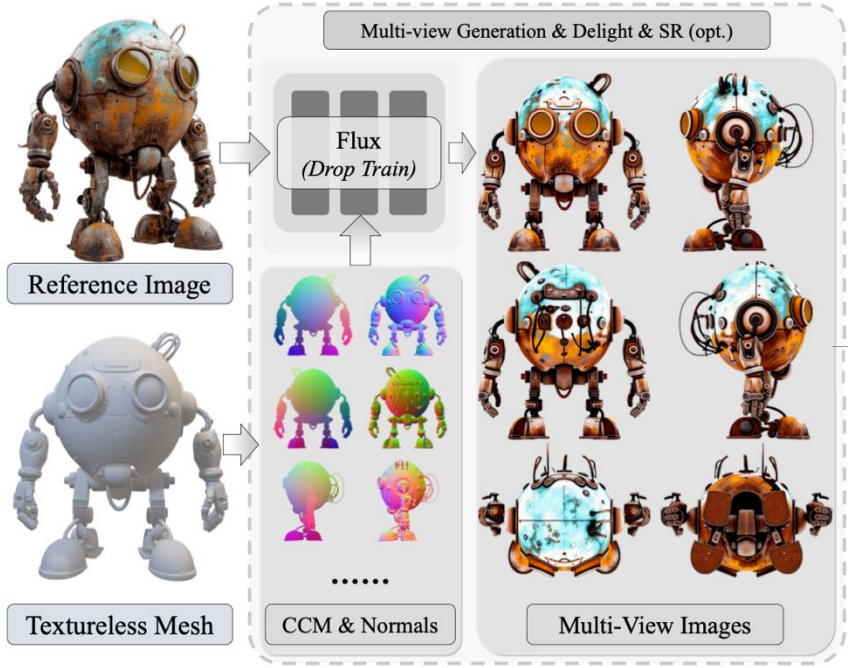
Hybrid Block Design:

- **2D branch**: Convolutions capture local high-resolution details.
- **3D branch**: Point-based modeling preserves cross-island consistency.
- **Fusion**: Serialized attention + Position encoding + Condition modulation balance detail and global coherence.

TEXGen: a Generative Diffusion Model for Mesh Textures. 2024.11. Siggraph Asia.

3D Texture Generation Models:

- Multi-view Back-Projection + Large Texturing Model: UniTEX



Efficient DiT Tuning for Multi-View Generation:

- **Flux-1** generates shaded 6-view images from reference images + normal maps + canonical coordinate maps (CCMs).
- **Flux-2** produces delighting and diffuse color for final texturing, optionally enhanced with super-resolution.

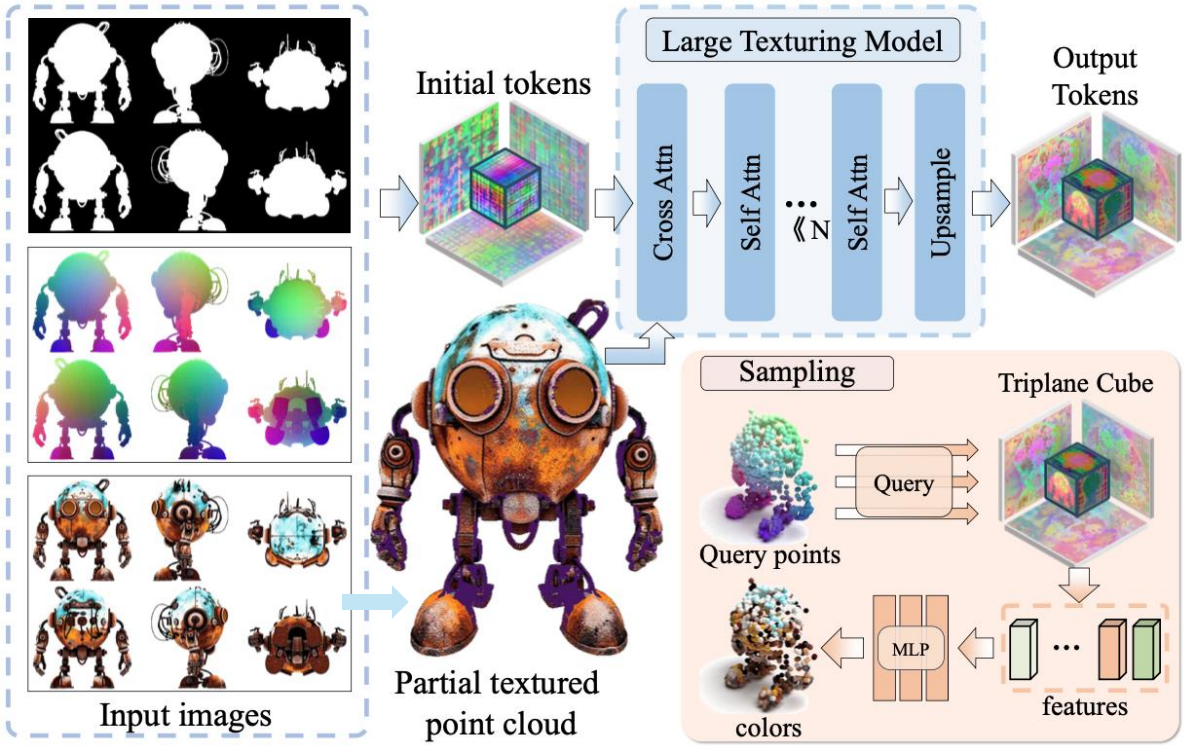
Challenge: Six-view generation requires heavy conditioning (images + geometry per view), leading to massive token input and slow convergence.

UniTEX: propose **Drop Training,** only a subset of tokens is retained per step, forcing the model to learn multi-view consistency from partial inputs.

UniTEX: Universal High Fidelity Generative Texturing for 3D Shapes. 2025.05. Arxiv.

3D Texture Generation Models:

- Multi-view Back-Projection + Large Texturing Model: UniTEX



Large Texturing Model (LTM): Replace UV-based inpainting with a **3D functional regression** approach.

- **Initialize** the Triplane-Cube representation (high-resolution triplane + low-resolution cube).
- **Encode** six orthogonal-view images, CCM, and alpha maps, and add them into the initial representation.
- **Incorporate** partial textured geometry point clouds into the triplane-cube via cross-attention.
- **Process** flattened features with transformer self-attention → reshape back to triplane & cube → upsample to obtain final features.
- **Query** triplane-cube features with an MLP to predict the color of any 3D point.

Training Objective — Texture Functions:

- **Motivation**: Extend texture from surfaces to continuous volumetric functions.
- **Benefit**: Allows dense volumetric supervision with richer training signals.
- **Definition**: Map 3D coordinate x to the nearest surface point Ω and assign its color.
- **Significance**: Aligns with volumetric geometry (e.g., SDF/UDF) for unified 3D learning.

UniTEX: Universal High Fidelity Generative Texturing for 3D Shapes. 2025.05. Arxiv.

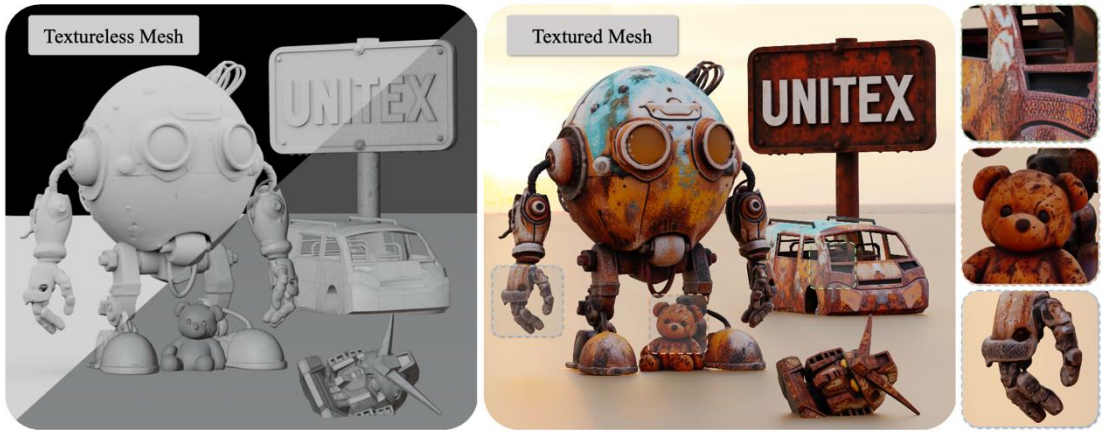
3D Texture Generation Models:

- Method Comparison

Method-1	Method-2	Method-3
Multi-view Back-Projection + UV Inpainting: • Meta 3D TextureGen • TripoSG	Single-view Back-Projection + UV Diffusion: • TEXGen	Multi-view Back-Projection + Large Texturing Model: • UniTEX
• Suffers from topological ambiguity • Suboptimal texture quality • Easy building	• Suffer from fragmentation, breaking global 3D consistency, but TEXGen fixed it • Hard to prepare data • Hard training high resolution UV maps	• Regress the texture in 3D functional space • Bypass the topology ambiguity • Better 3D consistency and performance

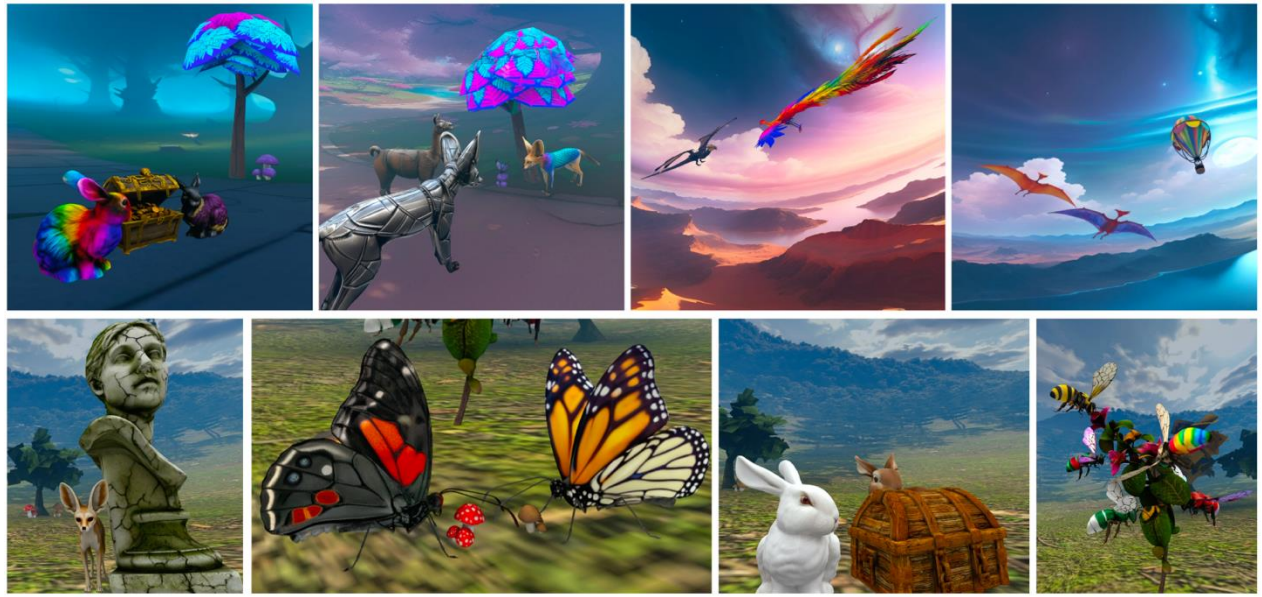
3D Texture Generation Models:

- Generation Results



[1] UniTEX

[2] TEXGen



[3] Meta 3D TextureGen

[1] UniTEX: Universal High Fidelity Generative Texturing for 3D Shapes. 2025.05. Arxiv.

[2] TEXGen: a Generative Diffusion Model for Mesh Textures. 2024.11. Siggraph Asia.

[3] Meta 3D TextureGen: Fast and Consistent Texture Generation for 3D Objects. 2024.07. Arxiv

香港中文大學
The Chinese University of Hong Kong

ICCV
OCT 19-23, 2025
HONOLULU
HAWAII

Thanks