PartDiffuser: Part-wise 3D Mesh Generation via Discrete Diffusion

Supplementary Material

A. Dataset Construction

As a supplement to the dataset introduction in the main text, we provide a detailed description of the dataset construction process. We utilize Objaverse [8] and 3D-Front [10] as our primary data sources. The data preprocessing pipeline consists of four main steps: data filtering, segmentation configuration, serialization augmentation, and block filtering.

Considering computational costs and training stability, we first filter the raw mesh data, retaining only models with a face count fewer than 4000. The filtered data is randomly divided into training, validation, and test sets in a 9:1:1 ratio.

In the segmentation phase, we employ PartField [23] to process the meshes. PartField requires explicit specifications for the minimum and maximum number of clusters. To adapt to geometric structures of varying complexity, we dynamically determine these bounds based on the mesh resolution. Specifically, for a mesh with F faces, the minimum cluster count K_{min} and the maximum cluster count K_{max} are calculated as:

$$K_{min} = \min\left(\left\lfloor \frac{F \times 0.5}{500} \right\rfloor, 4\right),$$
$$K_{max} = \min\left(\left\lfloor \frac{F \times 2.0}{500} \right\rfloor, 4\right)$$

where $\lfloor \cdot \rfloor$ denotes the floor operation. These bounds are passed to PartField to ensure that the segmentation granularity aligns with the mesh complexity while remaining within a manageable range for the generative model.

Serialization and Augmentation. After obtaining the semantic segmentation results, we construct a part adjacency graph, and then randomly select two different parts as starting nodes to perform Breadth-First Search traversals to perform data augmentation on each mesh sample. Subsequently, we employ BPT [36] to encode the mesh following these orders.

Block Filtering To optimize the training efficiency of the discrete diffusion model and prevent an excessive proportion of Pad Tokens in the sequence from causing ineffective computation, we perform a secondary filtering on the encoded sequences. We mandate that the encoded length of each semantic part must fall within the range of 128 to 1024 tokens. For blocks satisfying this condition, we uniformly pad them to a length of 1024 using Pad Tokens. Samples containing blocks that fall outside this range are discarded. The distribution of mesh face counts in the the training set is shown in Figure 6.

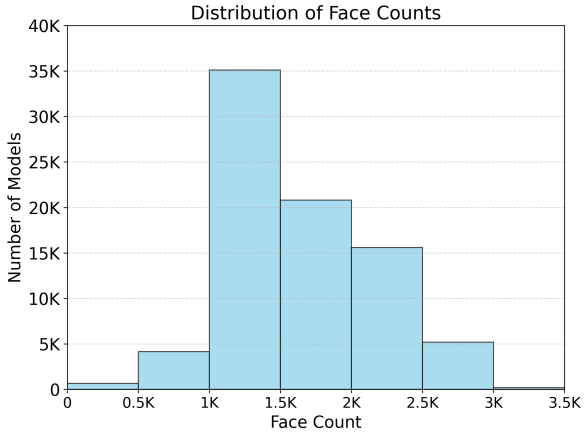

Figure 6. Mesh with different faces in the dataset.

B. Training and Sampling Details

B.1. Training Strategy

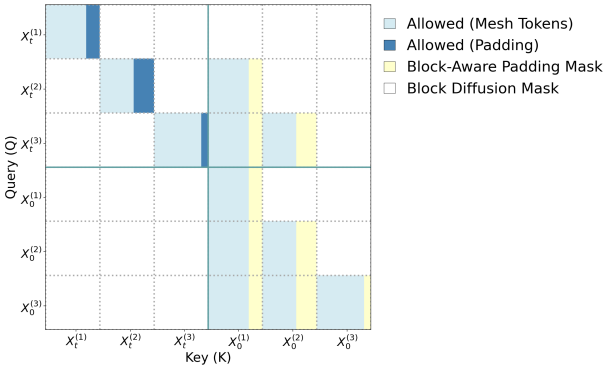

Figure 7. Visualization of the composite attention mask during the parallel training phase, using $N = 3$ parts as an example. This mask governs the attention mask used in Section 3.3.4, managing interactions across the N noisy blocks, denoted X_t, and the N clean blocks, denoted X_0. The legend details the four distinct attention behaviors: standard allowance for mesh tokens, specific allowance for padding tokens, and blockage enforced by either the Block Diffusion Mask or the Block-Aware Padding Mask.

As introduced in the main text, the self-attention module employs a specialized composite mask. This mask combines two distinct components via logical intersection. The

Block Diffusion Mask stemming from BD-LM [1] enables fine-grained control over the diffusion process by regulating the flow of information between different blocks. The second component, the Block-Aware Padding Mask, works in tight coordination with this hybrid structure. It supports the bidirectional intra-block attention by permitting full visibility within the current denoising block, including padding tokens. Conversely, for the autoregressive inter-block attention, it strictly prevents attention to any padding tokens in other blocks. An example of the composite mask used during the training phase is illustrated in Figure 7.

During the training phase, the model operates in a parallel mode, receiving the complete token sequence at once. This sequence is logically divided into $2N$ blocks (N blocks for the noisy version X_t and N blocks for the clean version X_0). The composite attention mask is applied to this parallel block structure. This parallelization strategy allows the model to efficiently learn the precise correspondence between token sequences and specific part geometries.

To balance training stability with generative coverage, we adopt a two-stage training curriculum. In the pre-training stage, we adopt the *clipped noise schedule* $\mathcal{U}[\beta, \omega]$ [1] to ensure efficient acquisition of core geometric features. In the fine-tuning stage, the model is fine-tuned under a full linear schedule $\mathcal{U}[0, 1]$, enabling it to adapt to the complete denoising trajectory required during inference and improving overall robustness across all noise levels.

B.2. Sampling Strategy

Algorithm 1 Sampling Strategy

Require: Hierarchical geometric condition $C_{\text{pc}} = [C_{\text{global}}, C_{\text{part}_1}, \ldots, C_{\text{part}_N}]$, Diffusion steps T, block length L_{block}

Initialize accumulated sequence $X_{\text{accum}} \leftarrow \emptyset$
for $i \leftarrow 1$ to N **do**
  $Z \leftarrow \text{SamplePrior}(B, L_{\text{block}})$
  $X_{\text{context}} \leftarrow \text{concat}(X_{\text{accum}}, Z)$
  $C_{\text{dyn}} \leftarrow \text{concat}(C_{\text{global}}, C_{\text{part}_i})$
  for $t \leftarrow T$ down to 1 **do**
    // Pass full context, focus on last block
    $P(X_0|X_t) \leftarrow \text{Model}(X_{\text{context}}, t, C_{\text{pc}}, \text{block\_idx} = i)$
    // CrossAttn uses C_{dyn} due to block_idx
    $Z_{\text{denoised}} \leftarrow \text{SampleStep}(P(X_0), X_{\text{context}}, t, t-1)$
    $X_{\text{context}} \leftarrow \text{update\_context}(X_{\text{accum}}, Z_{\text{denoised}})$
  end for
  $X_{\text{accum}} \leftarrow \text{concat}(X_{\text{accum}}, Z_{\text{denoised}})$
end for
return X_{accum}

Algorithm 1 provides the pseudo-code for our semi-autoregressive sampling process. The generation reconstructs one semantic part at a time, consisting of N strides corresponding to the total number of parts determined by the input condition C_{pc}.

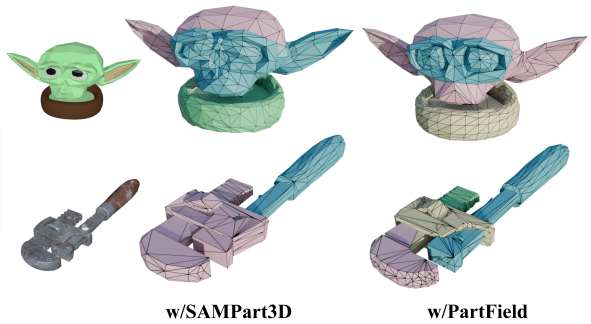

Figure 8. Experiment on robustness to different segmentation strategies.

C. Sampling Analysis

C.1. Robustness Analysis

To analyze robustness, we conducted a brief experiment using SAMPart3D [41] as an alternative segmentation upstream.

Quantitatively, using SAMPart3D on Objaverse yields a CD of 19.11×10^{-3}, closely matching the original 17.81 using PartField. The analysis demonstrates that our model is reasonably robust to different segmentation strategies. The denoising process prioritizes the completeness and smoothness of local point cloud geometry rather than strict semantic boundaries. However, as shown in the comparison, different segmentation strategies can affect local detail representation. Coarser segments may reach this capacity sooner, leading to relative simplification, while more balanced partitions utilize the token budget effectively for higher fidelity.

C.2. Sampling Process

To intuitively illustrate the core mechanism of PartDiffuser, we visualize the part-by-part sampling process in Figure 9. Crucially, the visualization highlights a distinct behavior compared to conventional fully AR approaches. In AR methods, mesh faces are typically generated sequentially along a fixed topological trajectory or backbone. In contrast, the intra-part diffusion process exhibits a parallel generation pattern: faces do not propagate linearly from a seed point. Instead, valid geometry materializes simultaneously across disparate spatial locations within the part's volume. As observed in the intermediate steps, the model does not merely extend a continuous partial surface; rather, it often constructs a sparse structural "framework" or "skeleton" of the part first. This aligns with the confidence-based nature of discrete diffusion, where the model prioritizes high-confidence structural tokens. The model's ability to establish the overall geometric scaffolding may help mitigate the risk of local over-complexity.

Figure 9. Visualization of the part-wise sampling process. The process evolves from left to right: (1) The first part being denoised, (2) The first part completed, (3) The second part being denoised, (4) The second part completed, and so on. This highlights how PartDiffuser combines part-level autoregression with discrete diffusion.

Besides, the visualization also verifies semi-autoregressive design, where subsequent parts are explicitly conditioned on the completed geometry of their predecessors. It can be observed that in the noisy stages, the newly emerging faces tend to align with the boundaries of the previously generated fixed parts. This may demonstrate that the model utilizes the context from previous parts to ensure topological connectivity.