PartNeXt: A Next-Generation Dataset for Fine-Grained and Hierarchical 3D Part Understanding

Penghao Wang Yiyang He Xin Lv Yukai Zhou
Lan Xu Jingyi Yu Jiayuan Gu[*]
ShanghaiTech University
`https://authoritywang.github.io/partnext`

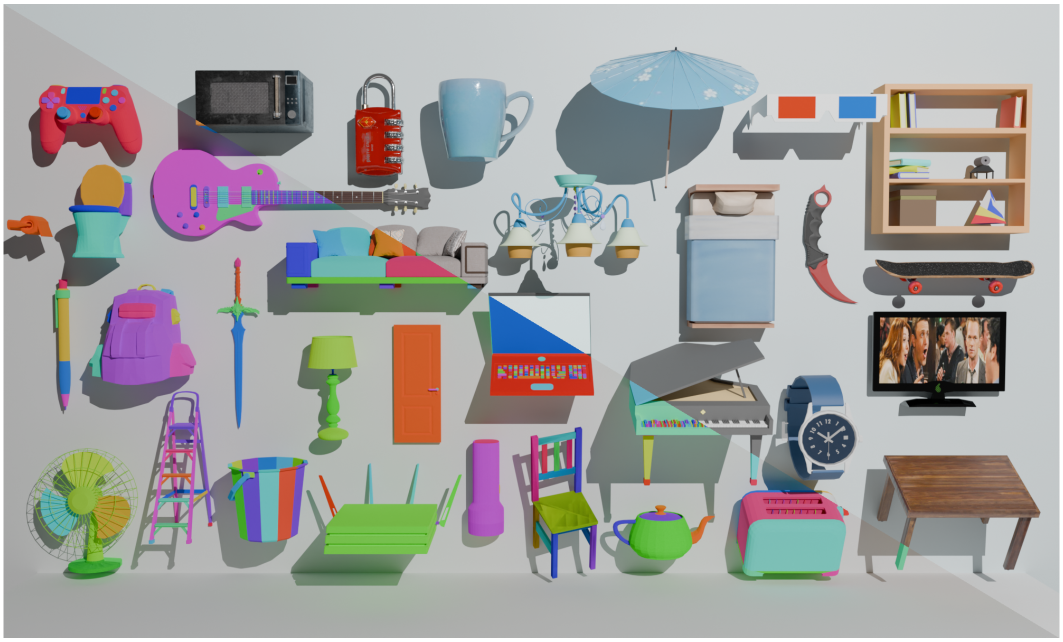


Figure 1: We present **PartNeXt**, a next-generation dataset tailored for fine-grained, hierarchically structured 3D part understanding.

Abstract

Understanding objects at the level of their constituent parts is fundamental to advancing computer vision, graphics, and robotics. While datasets like PartNet have driven progress in 3D part understanding, their reliance on untextured geometries and expert-dependent annotation limits scalability and usability. We introduce PartNeXt, a next-generation dataset addressing these gaps with over 23,000 high-quality, textured 3D models annotated with fine-grained, hierarchical part labels across 50 categories. We benchmark PartNeXt on two tasks: (1) class-agnostic part segmentation, where state-of-the-art methods (e.g., PartField, SAMPart3D) struggle with fine-grained and leaf-level parts, and (2) 3D part-centric question answering, a new benchmark for 3D-LLMs that reveals significant gaps in open-vocabulary part grounding. Additionally, training Point-SAM on PartNeXt yields substantial gains over PartNet, underscoring the dataset's superior quality and diversity. By

---

[*] Corresponding Author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Track on Datasets and Benchmarks.

combining scalable annotation, texture-aware labels, and multi-task evaluation, PartNeXt opens new avenues for research in structured 3D understanding.

1 Introduction

Understanding objects at the level of their constituent parts is fundamental to numerous tasks in computer vision, computer graphics, and robotics. From semantic and instance-level segmentation [26, 53, 48, 25] to generative modeling [18, 22] and robot manipulation [45, 42], fine-grained part reasoning enables models to infer structure, affordance, and function in ways that align closely with human perception and action. As David Marr famously argued in his theory of vision [30], the construction of intermediate representational primitives—such as parts—is a critical step in transforming raw sensory data into higher-level perceptual and behavioral intelligence.

The PartNet dataset [32] has driven progress in part-level 3D understanding. It provides over 573K part annotations across 26K 3D models spanning 24 object categories, organized in expert-defined hierarchies. It has catalyzed a wave of research in 3D part segmentation [27, 54], affordance analysis [6, 50], and retrieval-based generation [17, 21]. Despite its impact, several limitations in data quality and the annotation tool hinder its broader usability and scalability. **First, some annotations in PartNet require remeshing the mesh, which may lead to missing textures on some objects and deformation of the mesh geometry.** This restricts the availability of visual cues such as color and material, which are often essential for both human annotators and learning-based models to accurately recognize and segment object parts. As a result, many existing methods [39, 26] either rely solely on geometry or operate on small textured subsets like PartNet-Mobility [44]. **Second, its annotation interface demands significant expertise in 3D modeling, presenting a barrier to crowdsourcing and large-scale expansion.** For instance, annotators have to manually draw curves to cut meshes and carefully inspect sliced cross-sections to identify interior components—procedures that are time-consuming and unintuitive for non-experts.

Creating a new 3D part-level annotated dataset is both necessary and highly challenging. While there has been notable progress in annotating 2D multi-granular masks [15], part-level annotation in 3D introduces a unique set of difficulties. Unlike 2D images, 3D objects can contain complex interior structures, making segmentation significantly more demanding. First, designing an intuitive and easy-to-use annotation interface for non-expert users is non-trivial—especially when annotating interior or occluded parts. Second, incentivizing annotators to produce useful and fine-grained segmentations requires thoughtful task design and guidance, as well as mechanisms to ensure consistency and quality. Third, if hierarchical labels are desired, defining a coherent and extensible taxonomy that generalizes across categories remains an open problem.

In this work, we introduce **PartNeXt**, a next-generation dataset for fine-grained and hierarchical 3D part understanding. PartNeXt contains 23,519 high-quality, textured meshes annotated with detailed part masks spanning 50 object categories. The models are sourced from Objaverse [5], ABO [3], and 3D-FUTURE [8], ensuring a wide diversity of appearances and geometries while covering several widely used 3D object datasets. **To support efficient and scalable annotation, we develop a fully web-based interface tailored for crowdsourcing.** The interface features a dual-panel layout: one panel displays the unannotated regions of the mesh, while the other shows parts that have already been annotated. Annotators assign parts by selecting faces from the unannotated panel and moving them into the annotated panel. This intuitive and visually guided workflow minimizes the need for specialized 3D expertise and significantly boosts annotation throughput, especially for complex objects with interior structures. Importantly, PartNeXt annotations are performed directly on textured meshes. To enable this, we implement custom algorithms for visualizing partially segmented, high-resolution textured meshes and provide a multi-granular suite of face-based selection tools that enhances annotation flexibility, efficiency, and precision. **We further enhance scalability and consistency by integrating AI tools into the annotation workflow.** Specifically, we use CLIP-based[37] filtering to select high-quality, category-consistent assets and employ GPT-4o[13] to bootstrap part hierarchies across categories. The resulting dataset supports both high annotation quality and broad taxonomic coverage.

To showcase the utility of our dataset, we first evaluate performance on class-agnostic part segmentation, which assesses a model's ability to identify and segment semantically meaningful parts without relying on category-specific priors. We find that state-of-the-art methods such as PartField [25], SAM-

Part3D [48], and SAMesh [39] perform noticeably worse on our fine-grained dataset, particularly when segmenting leaf-level parts in the hierarchy. In addition, we introduce a new benchmark: 3D part-centric question answering. This task is tailored for 3D Large Language Models (3D LLMs), and evaluates their ability to perform open-vocabulary part grounding, detection, and question answering. We benchmark leading 3D LLMs, including ShapeLLM [35], 3D-LLM [11], and PointLLM [46], and observe that current models struggle with part-centric queries—underscoring the complexity and significance of this challenge. Lastly, we train the interactive segmentation model Point-SAM [53] on PartNeXt, which substantially outperforms its counterpart trained on PartNet. It demonstrates the enhanced quality, diversity, and utility of our dataset for fine-grained 3D understanding.

2 Related Work

3D Datasets Large-scale 3D repositories have played a pivotal role in advancing graphics, vision, and robotics. ShapeNet [1] is one of the earliest efforts, aggregating a large collection of textured meshes organized under WordNet synsets. Its core subset provides about 51K models with filtered mesh and texture quality, and has been widely used for 3D shape analysis [33, 34, 55] and synthetic data generation [31, 7, 9]. More recently, Objaverse [5] expanded the scale of 3D data to over 800K richly tagged objects sourced from Sketchfab. Objaverse-XL [4] further expands this effort.

In addition to broad-category collections, several datasets target specific domains. Amazon-Berkeley Objects (ABO) [3] includes 7,953 artist-designed textured meshes accompanied by product images and metadata from Amazon. 3D-FUTURE [8] offers approximately 16K industrial CAD furniture models with high-resolution textures created by professional designers, accompanied by photorealistic synthetic renderings. Thingi10K [52] curates 10K 3D-printable models from Thingiverse. Apart from mesh-based datasets, multi-view image collections have been used to reconstruct 3D shapes from real-world observations, like CO3D [38], MVImgNet [51], OmniObject3D [43].

3D Part Annotation Despite many existing 3D object datasets, part-level annotations remain limited due to high annotation costs, the inherent complexity of 3D labeling, and underexplored interface design. PartNet [32] pioneered large-scale fine-grained, hierarchical part annotations, providing detailed part masks for approximately 26K shapes across 24 categories. It enabled benchmarks in semantic, hierarchical, and instance-level part segmentation. Prior to PartNet, ShapeNet-Part[49] offered coarse part labels for 16 categories from ShapeNet, and remains a popular benchmark for semantic part segmentation. Other efforts include Fusion360 [41], which focuses on CAD models and B-Rep. Several extensions of PartNet have further enriched the landscape. PartNet-Mobility[44] adds kinematic joint annotations to over 2K articulated objects, supporting research on articulated object understanding [18, 22, 17, 21]. GAPartNet[10] redefines part semantics by grouping them according to functionality. 3D AffordanceNet [6] augments selected PartNet shapes with point-wise probabilistic affordance scores to support functional reasoning. In addition to synthetic datasets, part-level annotations have also been introduced in real-world scanned data, such as ScanObjectNN [40] and AKB-48 [24]. Recently, PartObjaverse-Tiny [48] was introduced to evaluate open-vocabulary part segmentation, comprising 200 complex 3D objects with semantic and instance annotations.

Our new dataset, PartNeXt, offers part annotations across a broader range of categories than PartNet (50 vs 24). We introduce a novel, web-based annotation interface designed for efficient labeling, particularly of interior structures. Unlike PartNet, which provides separate, untextured part meshes, PartNeXt directly annotates parts on textured meshes. This avoids common issues such as the need for extra alignment between ShapeNet and PartNet when textures are required.

3D Part Understanding Part annotations enable many downstream tasks relevant to 3D part understanding, such as part segmentation, part assembly [20, 12, 19], part-based generative models [18, 22], and articulated object reconstruction [14, 23, 29, 2]. PartNet has proposed a detection-by-segmentation method to address instance part segmentation and benchmarked a set of close-vocabulary segmentation methods on semantic part segmentation. Recently, there has been growing interest in open-vocabulary part segmentation. PartField [25] learns feature embeddings and applies clustering to generate segmentations. SAMesh [39] combines multi-view SAM guidance with a tailored connectivity detection algorithm to produce high-quality segmentations. Point-SAM [53], following SAM [15], extends interactive segmentation to 3D point clouds.

Besides, recent efforts have extended large-scale vision-language models to support spatial reasoning in 3D environments. Models such as PointLLM [46], ShapeLLM [35], 3D-LLM [11], and GPT4Point [36] aim to align 3D point clouds with language queries, enabling tasks such as object classification, 3D captioning, and functionality understanding.

3 Data Annotation

In this section, we present the data annotation process of **PartNeXt**, a next-generation 3D dataset with fine-grained and hierarchical part annotations. We detail our data collection and preprocessing pipeline (Sec. 3.1), the construction of consistent and functionality-aware part hierarchies (Sec. 3.2), and the design of our scalable annotation system (Sec. 3.3) optimized for both efficiency and accuracy. Sec. 3.4 describes comprehensive statistics that highlight the scale, diversity, and richness of PartNeXt.

3.1 Data Collection and Preprocessing

We collected high-quality 3D models from several large-scale public datasets, including Objaverse [5], ABO [3], and 3D-FUTURE [8]. Among these, ABO and 3D-FUTURE primarily focus on household furniture CAD models and provide reliable category annotations, allowing straightforward filtering to select relevant categories. In contrast, Objaverse spans a much broader range of categories and exhibits greater variation in quality, including a large number of 3D scans and multi-object scenes, which makes consistent category annotation significantly more challenging.

To curate a clean subset from Objaverse, we first applied a series of filters based on metadata. Specifically, we removed: (1) models containing animation, (2) models with more than 130k faces, and (3) models tagged as scans or architectural objects. Since Objaverse does not provide explicit category annotations, we adopted a text similarity-based classification approach to label and further filter the remaining high-quality models.

We began by defining a set of around 100 common object categories. Using CLIP's text encoder, we encoded both the category names and the descriptive captions provided by Cap3D [28] for each object. Each object was assigned the category whose embedding had the highest cosine similarity to its caption. To ensure label reliability, we discard any object whose highest similarity score was below 0.75. Finally, we selected the 50 categories with the largest number of objects.

3.2 Hierarchy Definition and Example Generation

Parts are typically organized in hierarchical structures, reflecting different levels of granularity based on functionality or semantics. A well-defined hierarchy not only facilitates the interpretation of part semantics and functionality, but also improves annotation accuracy by providing annotators with clear and consistent guidance. Following the strategy introduced in PartNet [32], we predefined a structured part hierarchy for each category, along with detailed definitions and illustrative examples for all part nodes within the hierarchy.

In PartNet, part hierarchies were defined by experts based on several criteria, including being *well-defined, consistent, compact, hierarchical, atomic, and complete*. However, these principles were implied implicitly within expert-designed templates. Notably, part hierarchies can also be informed by an object's manufacturing process and intended functionality. To better capture these aspects, we refine and formalize the hierarchy design criteria as follows:

- **Functionality-aware**: Top-level components should consist of the largest indivisible, functionally meaningful parts.
- **Hierarchical**: Deeper levels should be composed of sub-parts as defined during the manufacturing process.
- **Exhaustive variants**: When a part has multiple variants, all possible types should be enumerated under the same parent node to explicitly differentiate between them.
- **Atomicity**: Leaf nodes should represent all parts that cannot be further subdivided.
- **Consistency**: Parts with the same function and structure should be defined consistently across different object categories.

Manually defining detailed hierarchical structures—particularly when enumerating diverse part variants—is a challenging and labor-intensive task. To address this, we leverage GPT-4o [13] to assist in generating hierarchies for each category. Our prompts incorporate the aforementioned design principles to guide the model in producing coarse hierarchies. To further improve coverage of diverse part variants (e.g., a chair may have a foot base or a pedestal base), we collect rendered images for each category and provide GPT-4o with them to refine hierarchies. All AI-generated hierarchies are subsequently reviewed and refined by human experts to ensure accuracy and consistency.

Fine-grained part annotation often involves highly specialized terminology, which can lead to ambiguity during the labeling process. To address this, we provide visual examples for each part to aid annotators. Leveraging the image generation capabilities of GPT-4o, we produce labeled reference images for each part node. These generated samples are manually reviewed, and when necessary, inaccurate or missing examples are supplemented or replaced with high-quality images curated from online sources to ensure clear and faithful visual representations of each component.

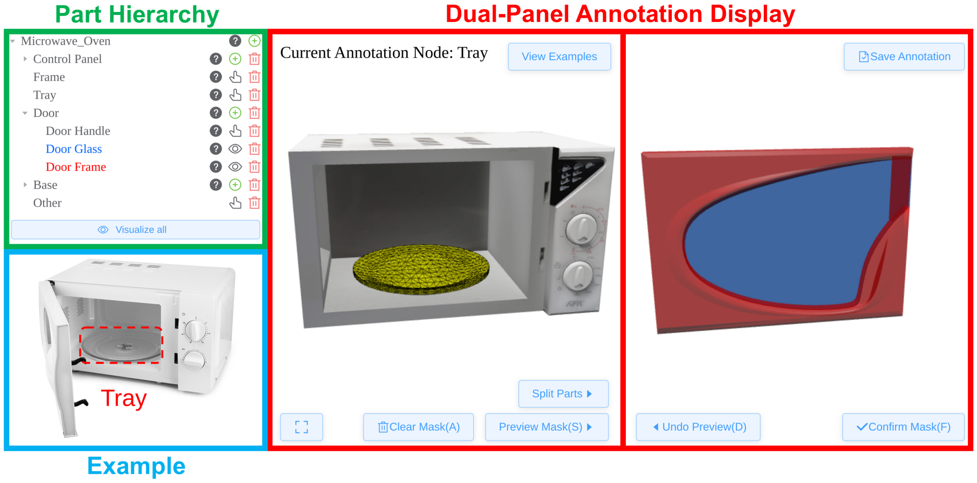


Figure 2: **Illustration of our annotation interface**. The example shows a microwave containing an internal tray. The dual-panel layout allows annotators to first label external parts such as the "door" (as shown in the right panel with already segmented meshes), and then proceed to annotate internal components like the "tray" (visible in the unsegmented mesh in the left panel). This design effectively mitigates occlusion issues during annotation.

3.3 Annotation System Design

Our annotation system is fully web-based to support scalable crowdsourcing (see Figure 2). It is designed with three key features to enhance annotation efficiency and quality: (1) a hierarchical annotation workflow, (2) a dual-panel interface, and (3) a suite of selection tools. Please refer to the supplementary video for a demonstration of our annotation interface.

Hierarchical annotation workflow We adopt a hierarchical annotation workflow to guide annotators in labeling complex 3D structures. The panel presents a collapsible tree structure representing the predefined part hierarchy. By default, only top-level components are visible; annotators progressively expand the tree and label leaf nodes. This progressive design helps disambiguate multiple part instances with the same semantics but different parent nodes—for example, distinguishing between a cushion on the backrest and one on the seat of a chair. For uncommon or unexpected components not captured by the predefined hierarchy, annotators can create an "Other" node at any level.

Dual-panel interface The main interface features a dual-panel layout: the left panel displays the unsegmented 3D mesh, while the right panel shows the segmented result from the same viewpoint. Annotators interactively select mesh faces corresponding to the currently active part node, and once confirmed, the selected region is transferred from the left panel to the right. Each annotated part is assigned a unique color, synchronized with its corresponding node in the hierarchy tree. The dual-panel design is particularly useful for visualizing and labeling interior parts that may be initially occluded in the unsegmented view.

Selection tools Previously, PartNet supported selecting mesh subgroups or provided mesh-cutting tools to split parts on a remeshed surface. However, manual mesh cutting is often time-consuming and require experience with mesh processing. In contrast, we ask annotators to work directly on the original textured mesh, and provide three face-selection tools tailored for different use cases:

- **Connected Component Selection:** We provide a connectivity-based selection tool that enables annotators to select mesh regions by simply clicking on a face. The tool automatically selects the entire connected component containing the clicked face. To accommodate meshes with duplicated vertices, we offer two modes for computing connectivity—one that considers exact mesh topology and another that merges nearby vertices—ensuring robust performance across diverse mesh qualities.
- **Bounding-Box selection:** We include a bounding-box selection tool that selects all mesh faces within a user-defined 2D bounding box projected from the current viewpoint. This tool is particularly effective for quickly selecting coherent regions on over-segmented or highly detailed meshes.
- **Per-Face Selection:** For precise control, manual face-by-face selection is supported, allowing annotators to directly add or remove individual faces, offering fine-grained control over the selected regions and complementing the coarser selection tools.

Compared to the selection tools provided in PartNet, our system supports flexible combinations of different selection modes. For example, annotators can first select a connected component and then refine the selection by removing a subset of faces. By operating directly at the face level on textured meshes, our approach eliminates the need for remeshing and preserves the original texture information.

3.4 Statistic

We hired 35 professional annotators, and 5 top-performing annotators responsible for data verification and quality control. The annotation backend was deployed on a central server equipped with dual Intel Xeon 6326 CPUs, while annotators operated on consumer-grade PCs powered by Intel Core i5-12400F processors. Prior to labeling, all annotators completed a two-day training session using a curated set of test models to ensure consistency and accuracy. On average, each 3D model required approximately 5 to 6 minutes to annotate.

Our constructed dataset, PartNeXt, provides 350187 annotated instances for 23,519 objects across 50 categories. Specifically, 14,811 instances were sourced from Objaverse, 2,633 from ABO, and 6,075 from 3DFuture. Detailed statistics of the dataset are presented in Table 1. For more statistics, please refer to the appendix. Each annotation underwent at least one review, with a total of 5,211 corrections made. The maximum number of corrections for a single annotation reached 8. The defined part hierarchy spans from a minimum depth of 4 to a maximum depth of 10.

	All	Axe	Bag	Bed	Bookcase	Bottle	Buck	Cam	Chair	Chandeller	Mouse	Control	Cup	Door	Fan	Flashlight	F-Lamp
#S	23519	1628	69	1454	574	1204	168	8	4277	1355	129	78	361	9	93	118	78
#P	350187	7142	589	33661	15018	4196	1974	81	60819	51678	520	1640	1056	48	950	715	718

	Fork	Glass	Guitar	Ham.	H-phone	Keyboard	Knife	Lamp	Lap.	Micro.	Monitor	Mug	P-lock	Pen	Piano	Pickaxe
#S	76	291	279	583	136	228	998	950	94	67	332	498	18	146	106	124
#P	487	2833	9574	2843	1225	22045	3815	11073	9385	1042	1941	1133	83	864	6945	560

	Plier	Sciss.	Screw	Shovel	Skate.	Sofa	Spoon	S-Ladder	Sword	Table	Teapot	Toast.	Toilet	Umb.	Wash.	Watch	WiFi	Wrench
#S	65	40	71	73	141	3139	101	22	151	2326	348	28	137	65	19	115	8	141
#P	523	229	218	525	2461	48135	285	433	647	31836	3180	549	1133	1111	280	1256	123	610

Table 1: **PartNeXt Dataset Statistic**. #S represent number of annotated objects, while #P as the number of total annotated parts.

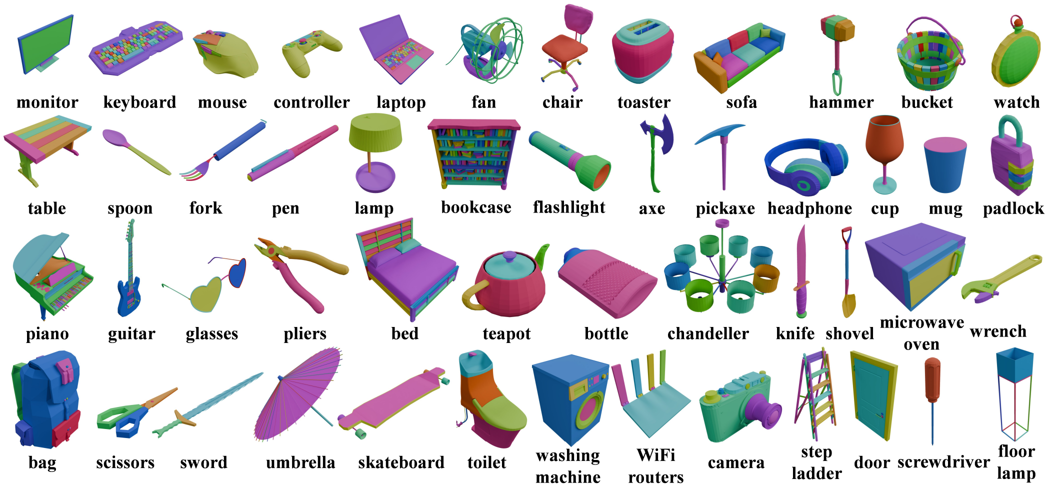


Figure 3: **PartNeXt dataset**. We visualize example shapes with fine-grained part annotations for the 50 object categories in PartNeXt.

4 BenchMark and Experiments

Based on our PartNeXt dataset, we introduce two benchmarks: **class-agnostic 3D part instance segmentation** and **part-centric 3D question answering**. Additionally, we examine the quality of our dataset by training a transformer-based interactive 3D segmentation method, and show superior results compared to PartNet.

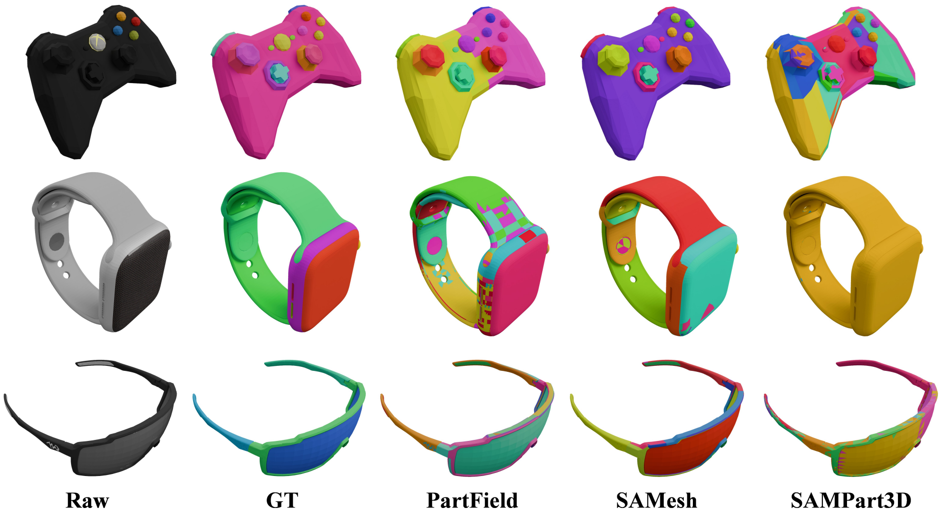


Figure 4: **Part Segmentation Results on PartNeXt**. PartField struggles to separate connected regions, SAMesh excels at fine-grained segmentation but over-segments, while SAMPart3D lacks continuity in weak textures and granularity control.

4.1 Class-Agnostic 3D Part Instance Segmentation

3D part segmentation plays a crucial role in understanding both the compositional structure and functionality of 3D objects. To systematically evaluate existing part segmentation methods, especially class-agnostic ones, we establish a segmentation benchmark based on our PartNeXt dataset. Specifically, we select 5 objects per category, resulting in a total of 250 objects in the evaluation set. All leaf nodes in the annotation hierarchy are treated as ground truth labels, thereby enabling an

Method	Category mIoU (%)										mIoU
	Bed	Bottle	Chair	Knife	Table	Controller	Fan	Glasses	Monitor	Wrench	
SAMPart3D	17.51	47.71	28.49	61.08	25.86	24.00	31.12	28.34	25.70	40.53	36.78
SAMesh	**82.59**	35.63	**72.57**	51.19	**64.81**	**47.71**	**56.72**	33.38	45.16	52.17	**51.57**
PartField	24.77	**67.91**	43.78	**68.22**	53.26	41.57	46.66	**55.57**	**45.97**	**60.53**	50.22

Table 2: Per-category IoU and mean IoU (mIoU) comparison across different methods. We present results for 10 representative categories: the first five are PartNet categories, while the latter five are novel categories beyond PartNet. Please refer to the appendix for segmentation results on more categories.

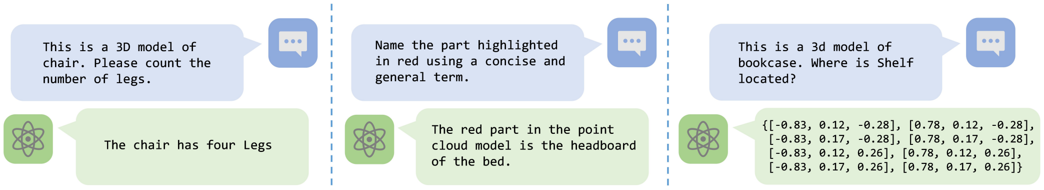
assessment of each method's ability to perform fine-grained segmentation. Following the evaluation protocol of PartField [25], we adopt mean Intersection-over-Union (mIoU) as our evaluation metric. For each ground truth part, we calculate the IoU with all predicted parts and record the maximum IoU achieved as the model's score for that part. These maximum IoU values are then averaged across all parts to obtain the final mIoU.

We evaluate recent SOTA part segmentation methods, including PartField[25], SAMPart3D[48], and SAMesh[39]. As shown in Figure 4 and Table 2, our analysis reveals distinct characteristics of each method: SAMesh demonstrates relatively strong performance in fine-grained segmentation but tends to produce over-segmented results. PartField occasionally fails to separate adjacent connected regions, while SAMPart3D struggles to maintain segmentation continuity in weakly-textured areas and exhibits inconsistent granularity control. Overall, current 3D part segmentation methods still face significant challenges in handling fine-grained segmentation tasks, particularly in balancing segmentation granularity with semantic consistency.

4.2 Part-Centric 3D Question Answering

To advance research on part-level reasoning in 3D vision, we propose a benchmark dedicated to evaluating model understanding of 3D object part structures. This benchmark is designed to assess a model's capability to understand, localize, and reason about fine-grained part structures in complex 3D objects. Specifically, we include three representative tasks: part counting, part classification, and part grounding. These tasks collectively probe a model's semantic granularity, structural awareness, and spatial correspondence in the context of 3D objects.

- **Part Counting:** Given a 3D object represented as a point cloud and a target part name, the model should predict the total number of target part instances in the object, testing object decomposition, with a downstream LLM converting text to numbers.
- **Part Classification:** Given ground-truth labels, points of a specific part are rendered in red, and the model should name the highlighted region, with a downstream LLM verifying correctness.
- **Part Grounding:** Given the full point cloud and a part query, the model should locate the part by predicting its bounding box corners.



This is a 3D model of chair. Please count the number of legs.

The chair has four Legs

Name the part highlighted in red using a concise and general term.

The red part in the point cloud model is the headboard of the bed.

This is a 3d model of bookcase. Where is Shelf located?

```
{[-0.83, 0.12, -0.28], [0.78, 0.12, -0.28],
 [-0.83, 0.17, -0.28], [0.78, 0.17, -0.28],
 [-0.83, 0.12, 0.26], [0.78, 0.12, 0.26],
 [-0.83, 0.17, 0.26], [0.78, 0.17, 0.26]}
```

(a) Part Counting (b) Part Classification (c) Part Grounding

Figure 5: **Representative prompt–response pairs used to evaluate 3D part-level understanding.** (a) Part Counting: the model is requested to enumerate the number of legs in a chair. (b) Part Classification: the model must name the part highlighted in red within the point-cloud bed. (c) Part Grounding: the model is asked to localize the "Shelf" of a bookcase by outputting the eight corner coordinates of its bounding box.

We evaluate three existing multimodal language models adapted for 3D tasks: PointLLM [46], ShapeLLM [35], and 3DLLM [11]. The prompts used in our benchmark can be broadly categorized into two types: (1) category-aware prompts, where the category of the 3D object is provided to the model prior to querying; and (2) category-agnostic prompts, where the model is required to infer part-level semantics and structure without any prior knowledge of the object class.

Table 3 reports the performance of all three models across the above tasks. We observe that their performance on part counting and grounding remains limited. These results suggest that current 3D vision-language models lack fine-grained structural understanding and call for further research in this direction.

	Part count(MAE)			Classification(Acc)		Grounding(IoU)		
	3DLLM	PointLLM	ShapeLLM	PointLLM	ShapeLLM	3DLLM	PointLLM	ShapeLLM
w category	2.16	1.87	1.72	0.22	0.25	\	\	0.33
wo category	2.46	1.79	1.85	0.18	0.08	\	\	0.30

Table 3: Comparison of different 3D object part-level understanding models across three tasks. Note that '\' means fails to response reasonable boundingbox.

4.3 Analysis on 3D Promptable Segmentation

Evaluation Dataset	Training Dataset	IoU@1	IoU@3	IoU@5	IoU@7	IoU@10
PartNet-Mobility	PartNet	39.0	53.7	58.6	60.9	62.9
	PartNeXt	40.2	57.5	63.2	65.0	67.4
	Mixture	**40.4**	**58.3**	**64.1**	**66.9**	**68.7**
PartNeXt	PartNet	39.9	53.9	58.4	60.4	60.3
	PartNeXt	44.3	60.1	63.2	64.8	65.9
	Mixture	**45.3**	**61.7**	**65.3**	**66.6**	**67.6**

Table 4: Comparison of Point-SAM [53] models trained on different datasets. The metric IoU@k is reported for 3D promptable segmentation, where k denotes the number of prompt points.

To further demonstrate the quality of our dataset, we conduct experiments by training a high-capacity 3D promptable segmentation model, Point-SAM [53], on PartNeXt. For a comprehensive comparison, we train the model on three different dataset configurations: (1) PartNet, (2) PartNeXt, and (3) a combination of both. The setup is designed to evaluate the effect of the training dataset on the model's generalization performance. Following the evaluation protocol established in Point-SAM, we use the PartNet-Mobility dataset [44] to evaluate models on 3 categories (scissors, refrigerators, and doors). Additionally, to further assess zero-shot transfer, we also hold out 3 categories (scissors, microwave oven, and floor lamp) from the PartNeXt training data as the test set. All models are trained on 4 NVIDIA A6000 GPUs, with a batch size of 16 for a total of 50,000 optimization steps.

The results in Table 4 indicate a clear performance improvement when the model is trained on the mixture dataset. Notably, training on the PartNeXt dataset alone also yields significant gains. These findings validate the high quality of our part annotations and highlight the benefit of expanding category diversity to enhance model generalization. Such a richly annotated and diverse dataset provides a crucial foundation for improving the generalizability of future 3D foundation models.

5 Limitations and Conclusion

Limitations Currently, PartNeXt faces three main limitations. Firstly, to ensure high-quality annotations, PartNeXt currently includes only 23,519 models, but we are actively working on expanding the dataset by incorporating more data from ObjaverseXL [4]. Secondly, each category in PartNeXt requires a carefully predefined fine-grained part hierarchy, which limits our ability to annotate open-vocabulary datasets. We are exploring open-vocabulary part annotations through deeper integration with VLMs. Thirdly, PartNeXt currently provides only plain part name annotations for each node. Introducing caption or physical attribute annotations for both category and part could greatly enrich the information of the PartNeXt dataset.

Conclusion In this work, we present PartNeXt, a next-generation dataset tailored for fine-grained, hierarchically structured 3D part understanding. Leveraging a web-based annotation system guided by connectivity-aware part prompts, we efficiently annotate a large-scale collection of 23519 3D models spanning 50 diverse categories, with 350187 parts in total. By directly annotating on textured mesh, our dataset can also provide native texture for each part, which is an essential modality for comprehensive 3D understanding. The proposed class-agnostic segmentation and 3D part-centric QA benchmark enables a comprehensive evaluation of a model's understanding at the part level. Through our experiments, we observe that current 3D understanding models still exhibit significant limitations in understanding fine-grained part-level semantics. Therefore, we envision our proposed dataset, PartNeXt, as a high-quality foundational dataset to support the development of the next generation of part-level 3D understanding models.

6 Acknowledgement

This work was supported by Shanghai Pujiang Program (24PJA080), the National Natural Science Foundation of China under Grant W2431046, the MoE Key Lab of Intelligent Perception and Human-Machine Collaboration (ShanghaiTech University), the Shanghai Frontiers Science Center of Human-centered Artificial Intelligence, and the HPC Platform of ShanghaiTech University. We also appreciate Benyuan AI Data for providing support in data annotation, as well as the 35 annotators for their valuable contributions.

References

[1] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository, 2015.

[2] Qiuyu Chen, Aaron Walsman, Marius Memmel, Kaichun Mo, Alex Fang, Dieter Fox, and Abhishek Gupta. Urdformer: A pipeline for constructing articulated simulation environments from real-world images. In Dana Kulic, Gentiane Venture, Kostas E. Bekris, and Enrique Coronado, editors, *Robotics: Science and Systems XX, Delft, The Netherlands, July 15-19, 2024*, 2024.

[3] Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, et al. Abo: Dataset and benchmarks for real-world 3d object understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21126–21136, 2022.

[4] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *Advances in Neural Information Processing Systems*, 36:35799–35813, 2023.

[5] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13142–13153, 2023.

[6] Shengheng Deng, Xun Xu, Chaozheng Wu, Ke Chen, and Kui Jia. 3d affordancenet: A benchmark for visual object affordance understanding. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1778–1787, 2021.

[7] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017.

[8] Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Steve Maybank, and Dacheng Tao. 3d-future: 3d furniture shape with texture. *International Journal of Computer Vision*, 129:3313–3337, 2021.

[9] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *Advances In Neural Information Processing Systems*, 35:31841–31854, 2022.

[10] Haoran Geng, Helin Xu, Chengyang Zhao, Chao Xu, Li Yi, Siyuan Huang, and He Wang. Gapartnet: Cross-category domain-generalizable object perception and manipulation via generalizable and actionable parts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7081–7091, 2023.

[11] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. *Advances in Neural Information Processing Systems*, 36:20482–20494, 2023.

[12] Jialei Huang, Guanqi Zhan, Qingnan Fan, Kaichun Mo, Lin Shao, Baoquan Chen, Leonidas Guibas, and Hao Dong. Generative 3d part assembly via dynamic graph learning. In *The IEEE Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[13] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[14] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5616–5626, 2022.

[15] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.

[16] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

[17] Long Le, Jason Xie, William Liang, Hung-Ju Wang, Yue Yang, Yecheng Jason Ma, Kyle Vedder, Arjun Krishna, Dinesh Jayaraman, and Eric Eaton. Articulate-anything: Automatic modeling of articulated objects via a vision-language foundation model. *arXiv preprint arXiv:2410.13882*, 2024.

[18] Jiahui Lei, Congyue Deng, William B Shen, Leonidas J Guibas, and Kostas Daniilidis. Nap: Neural 3d articulated object prior. *Advances in Neural Information Processing Systems*, 36:31878–31894, 2023.

[19] Yichen Li, Kaichun Mo, Yueqi Duan, He Wang, Jiequan Zhang, Lin Shao, Wojciech Matusik, and Leonidas Guibas. Category-level multi-part multi-joint 3d shape assembly. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024.

[20] Yichen Li, Kaichun Mo, Lin Shao, Minghyuk Sung, and Leonidas Guibas. Learning 3d part assembly from a single image. *European conference on computer vision (ECCV 2020)*, 2020.

[21] Xinyu Lian, Zichao Yu, Ruiming Liang, Yitong Wang, Li Ray Luo, Kaixu Chen, Yuanzhen Zhou, Qihong Tang, Xudong Xu, Zhaoyang Lyu, et al. Infinite mobility: Scalable high-fidelity synthesis of articulated objects via procedural generation. *arXiv preprint arXiv:2503.13424*, 2025.

[22] Jiayi Liu, Denys Iliash, Angel X Chang, Manolis Savva, and Ali Mahdavi-Amiri. Singapo: Single image controlled generation of articulated parts in objects. *arXiv preprint arXiv:2410.16499*, 2024.

[23] Jiayi Liu, Ali Mahdavi-Amiri, and Manolis Savva. Paris: Part-level reconstruction and motion analysis for articulated objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 352–363, 2023.

[24] Liu Liu, Wenqiang Xu, Haoyuan Fu, Sucheng Qian, Qiaojun Yu, Yang Han, and Cewu Lu. Akb-48: A real-world articulated object knowledge base. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14809–14818, 2022.

[25] Minghua Liu, Mikaela Angelina Uy, Donglai Xiang, Hao Su, Sanja Fidler, Nicholas Sharp, and Jun Gao. Partfield: Learning 3d feature fields for part segmentation and beyond. *arXiv preprint arXiv:2504.11451*, 2025.

[26] Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su. Partslip: Low-shot part segmentation for 3d point clouds via pretrained image-language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21736–21746, 2023.

[27] Tiange Luo, Kaichun Mo, Zhiao Huang, Jiarui Xu, Siyu Hu, Liwei Wang, and Hao Su. Learning to group: A bottom-up framework for 3d part discovery in unseen categories. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[28] Tiange Luo, Chris Rockwell, Honglak Lee, and Justin Johnson. Scalable 3d captioning with pretrained models. *Advances in Neural Information Processing Systems*, 36:75307–75337, 2023.

[29] Zhao Mandi, Yijia Weng, Dominik Bauer, and Shuran Song. Real2code: Reconstruct articulated objects via code generation. *arXiv preprint arXiv:2406.08474*, 2024.

[30] David Marr. *Vision: A computational investigation into the human representation and processing of visual information*. W. H. Freeman and Company, San Francisco, 1982.

[31] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016.

[32] Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 909–918, 2019.

[33] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[34] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[35] Zekun Qi, Runpei Dong, Shaochen Zhang, Haoran Geng, Chunrui Han, Zheng Ge, Li Yi, and Kaisheng Ma. Shapellm: Universal 3d object understanding for embodied interaction. In *European Conference on Computer Vision*, pages 214–238. Springer, 2024.

[36] Zhangyang Qi, Ye Fang, Zeyi Sun, Xiaoyang Wu, Tong Wu, Jiaqi Wang, Dahua Lin, and Hengshuang Zhao. Gpt4point: A unified framework for point-language understanding and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26417–26427, 2024.

[37] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[38] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10901–10911, 2021.

[39] George Tang, William Zhao, Logan Ford, David Benhaim, and Paul Zhang. Segment any mesh: Zero-shot mesh part segmentation via lifting segment anything 2 to 3d. *arXiv preprint arXiv:2408.13679*, 2024.

[40] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1588–1597, 2019.

[41] Karl DD Willis, Yewen Pu, Jieliang Luo, Hang Chu, Tao Du, Joseph G Lambourne, Armando Solar-Lezama, and Wojciech Matusik. Fusion 360 gallery: A dataset and environment for programmatic cad construction from human design sequences. *ACM Transactions on Graphics (TOG)*, 40(4):1–24, 2021.

[42] Shijie Wu, Yihang Zhu, Yunao Huang, Kaizhen Zhu, Jiayuan Gu, Jingyi Yu, Ye Shi, and Jingya Wang. Afforddp: Generalizable diffusion policy with transferable affordance. *arXiv preprint arXiv:2412.03142*, 2024.

[43] Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, et al. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 803–814, 2023.

[44] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11097–11107, 2020.

[45] Pengwei Xie, Rui Chen, Siang Chen, Yuzhe Qin, Fanbo Xiang, Tianyu Sun, Jing Xu, Guijin Wang, and Hao Su. Part-guided 3d rl for sim2real articulated object manipulation. *IEEE Robotics and Automation Letters*, 8(11):7178–7185, 2023.

[46] Runsen Xu, Xiaolong Wang, Tai Wang, Yilun Chen, Jiangmiao Pang, and Dahua Lin. Pointllm: Empowering large language models to understand point clouds. In *European Conference on Computer Vision*, pages 131–147. Springer, 2024.

[47] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[48] Yunhan Yang, Yukun Huang, Yuan-Chen Guo, Liangjun Lu, Xiaoyang Wu, Lam Edmund Y., Yan-Pei Cao, and Xihui Liu. Sampart3d: Segment any part in 3d objects. *arXiv preprint arXiv:2411.07184*, 2024.

[49] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016.

[50] Chunlin Yu, Hanqing Wang, Ye Shi, Haoyang Luo, Sibei Yang, Jingyi Yu, and Jingya Wang. Seqafford: Sequential 3d affordance reasoning via multimodal large language model. *arXiv preprint arXiv:2412.01550*, 2024.

[51] Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Chenming Zhu, Zhangyang Xiong, Tianyou Liang, et al. Mvimgnet: A large-scale dataset of multi-view images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9150–9161, 2023.

[52] Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797*, 2016.

[53] Yuchen Zhou, Jiayuan Gu, Tung Yen Chiang, Fanbo Xiang, and Hao Su. Point-sam: Promptable 3d segmentation model for point clouds. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

[54] Yuchen Zhou, Jiayuan Gu, Xuanlin Li, Minghua Liu, Yunhao Fang, and Hao Su. Partslip++: Enhancing low-shot 3d part segmentation via multi-view instance segmentation and maximum likelihood estimation. *arXiv preprint arXiv:2312.03015*, 2023.

[55] Chenyang Zhu, Kai Xu, Siddhartha Chaudhuri, Li Yi, Leonidas J Guibas, and Hao Zhang. Adacoseg: Adaptive shape co-segmentation with group consistency loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8543–8552, 2020.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the paper's main contributions, including the new PartNeXt dataset and its advantages over PartNet. These claims are supported by statistics3.4 and experiments4, and are consistent with the paper's scope and results.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [NA]

   Justification: As we present a new dataset, we do not discuss limitations.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include theoretical results, theorems, or formal proofs. Its primary contributions lie in dataset construction, annotation methodology, and empirical benchmarking. Therefore, the question regarding theory assumptions and proofs is not applicable in this context.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: This paper introduces a new dataset, PartNeXt, and evaluates it through two benchmark tasks (part segmentation and part-centric reasoning) as well as one training task. The experimental settings, evaluation metrics, baseline methods, and implementation details (e.g., mesh rendering settings, evaluation scripts) are clearly described in both the main paper4 and the appendix. The dataset and benchmark code will be publicly released.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will publicly release the PartNeXt dataset, including annotation files and benchmark tasks. The codebase provides data loaders, evaluation scripts, and training pipelines for all benchmark tasks. Detailed instructions for environment setup, command-line usage, and data preparation steps are included in the supplementary material to ensure faithful reproduction of experimental results.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental settings and details can be found in the appendix and supplemental materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

Justification: We do not provide this.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See 3.4 and 4.3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The paper's research strictly follows the NeurIPS Code of Ethics. There are no ethical concerns related to data usage, privacy, or potential societal harms. The authors have ensured compliance with all guidelines and have preserved anonymity in the review process.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: The work focuses on creating a 3D models dataset and conducting relevant benchmark tests. This foundational research primarily advances computer vision and graphics techniques without direct societal applications or foreseeable societal harms. Therefore, no significant positive or negative societal impacts are discussed or expected.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The dataset do not pose high risks of misuse or dual-use. The data is curated and sourced responsibly, without involving scraped content from the internet or sensitive information. Therefore, no specific safeguards were deemed necessary for the release.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All external datasets and code assets used in the paper are properly cited with their original sources. For each dataset (e.g., PartNet), we mention the version, provide URLs where available, and explicitly state the license (e.g., CC BY 4.0). No web-scraped or unlicensed content was used. All licenses and terms of use have been carefully respected.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We provide the datasets publicly with format documentation.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [Yes]

Justification: We employed external annotators to label our dataset. All annotators were provided with detailed written instructions and visual examples, which are included in the supplementary material. Annotators were compensated fairly, with payment rates exceeding the minimum wage in their respective regions, in line with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [Yes]

    Justification: External annotators were involved in the study to label 3D models. The tasks posed no foreseeable risks, and participants gave informed consent. According to our institution's policies, this type of annotation task is exempt from full IRB review but was assessed and cleared through the standard ethics process.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: LLMs were used only in auxiliary components of the data preprocessing pipeline, which are not central to the core methods proposed in the paper.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Detail of Annotation Platform

We first provide additional details about the annotation platform we designed and used for large-scale data annotation. A video demonstrating our annotation workflow is included in the supplementary materials.

During the data annotation process, we found that some categories are difficult to distinguish using text similarity-based methods. Additionally, some models contain multiple objects. To address these issues, we merged some categories. Specifically, due to the limitations of Cap3D's multi-view captioning approach, it's difficult to ensure accurate captions for every viewpoint, which can result in inaccurate label. One of the most prominent cases we observed was in tool-related categories such as pliers, scissors, and hammers. Therefore, we merged some categories to allow annotators to select the correct category. For models containing multiple objects, we also grouped these into a single category to allow annotators to label all objects in the scene. Our merged categories include:

- **Tool**: axe, fork, hammer, pickaxe, pliers, scissors, screwdriver, shovel, spoon, wrench, and knife.
- **Liquid Container**: mug, bottle, cup.
- **Light**: chandelier, floor lamp, desk lamp.
- **Computer Device**: keyboard, mouse, monitor, laptop, controller, desk, chair, mug, desk lamp, headphones.

After completing the annotations, we decomposed these merged categories back into their original, individual categories. For models with multiple objects, we also split the original model to extract individual objects, thereby obtaining separate models.

B Dataset Statistic

Here we provide detail statistic of PartNeXt dataset, as shown in Table 5

	All	Axe	Bag	Bed	Bookcase	Bottle	Buck	Cam	Chair	Chandeller	Mouse	Control	Cup	Door	Fan	Flashlight	F-Lamp
#S	23519	1628	69	1454	574	1204	168	8	4277	1355	129	78	361	9	93	118	78
#P	350187	7142	589	33661	15018	4196	1974	81	60819	51678	520	1640	1056	48	950	715	718
P_{Med}	15	7	8	27	24	6	12	15	19	24	8	24	6	9	15	10	14
D_{Med}	4	3	3	8	4	3	4	4	4	4	5	4	3	5	5	3	4
D_{Max}	9	3	3	9	5	3	5	5	6	5	5	4	3	5	5	3	4

	Fork	Glass	Guitar	Ham.	H-phone	Keyboard	Knife	Lamp	Lap.	Micro.	Monitor	Mug	P-lock	Pen	Piano	Pickaxe
#S	76	291	279	583	136	228	998	950	94	67	332	498	18	146	106	124
#P	487	2833	9574	2843	1225	22045	3815	11073	9385	1042	1941	1133	83	864	6945	560
P_{Med}	11	17	46	7	10	88	6	13	82	15	7	4	9	10	68	7
D_{Med}	4	5	5	3	3	4	3	4	4	4	3	3	5	5	4	3
D_{Max}	7	5	5	3	3	4	3	4	4	4	3	3	5	5	5	3

	Plier	Sciss.	Screw	Shovel	Skate.	Sofa	Spoon	S-Ladder	Sword	Table	Teapot	Toast.	Toilet	Umb.	Wash.	Watch	WiFi	Wrench
#S	65	40	71	73	141	3139	101	22	151	2326	348	28	137	65	19	115	8	141
#P	523	229	218	525	2461	48135	285	433	647	31836	3180	549	1133	1111	280	1256	123	610
P_{Med}	14	9	5	13	25	18	5	17	6	14	12	20	12	19	21	19	23	7
D_{Med}	4	4	3	6	6	3	3	3	3	5	3	4	4	5	4	6	4	4
D_{Max}	4	4	3	6	6	4	3	4	3	8	3	6	4	6	4	7	4	5

Table 5: **PartNeXt Dataset Statistic**. #S represent number of annotated objects, #P as the number of total annotated parts, P_{Med} is median number of parts, D_{Med} is median number of hierarchy depth, D_{Max} is maximum number of hierarchy depth.

Here we provide the visualization of our predefined hierarchy and more annotations. As shown in Figure 10, Figure 11, Figure 12, Figure 13.

Although we provide annotators with fine-grained hierarchies, we cannot guarantee that the hierarchy covers all possible parts. Therefore, during annotation, annotators are allowed to label parts that are not included in the hierarchy as “Other”. We also provide statistics on the “Other” parts, as shown in Table 6.

Number of Other	**0**	**1**	**2**	**3**	**4**	**5**	**>5**
Percentage %	82.22	13.82	1.77	0.30	0.38	0.20	1.30

Table 6: **Distribution of the Number of Other Instances.** The table shows the percentage of occurrences for each count category.

C Comparison with Recent 3D Part Dataset

We present a comparison between PartNeXt and recent 3D part-level datasets, as shown in Table 7. This includes detailed information on several 3D object part-level datasets such as PartNet[32] and PartObjaverseTiny from SAMPart3D[48].

Dataset	**Texture**	**Raw Geometry**	**Semantic**	**Hierarchy**	**# Shape**	**# Category**
PartNet	×	×	✓	✓	26,671	24
PartObjaverseTiny	✓	✓	✓	×	200	8
PartNeXt	✓	✓	✓	✓	23,519	50

Table 7: **Comparison between PartNeXt and recent 3D Part Datasets.** Raw Geometry indicates if the part remains the raw geometry (PartNet lacks texture and raw geometry due to remesh operation).

The PartNet[32] dataset is annotated directly on textureless models and requires remeshing during the annotation process. As a result, the final annotations lack both texture and original geometry, omitting important color information and potential geometric details (such as mesh topology and face distribution), which are crucial for comprehensive 3D understanding. Besides, PartNet requires manually drawn cutting lines after remeshing to achieve segmentation. As a result, the boundaries of the parts are often not smooth. In contrast, our method leverages per-face annotations, effectively avoiding these issues, the comparison is shown in Figure 6.

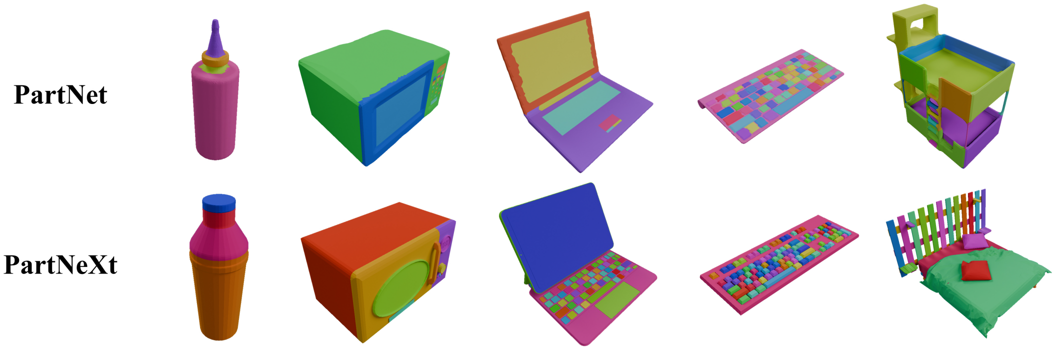


Figure 6: **Visualization of PartNet and PartNeXt results**. Since PartNet uses remeshing to obtain finer-grained parts, the mesh undergoes deformation, lacks texture, and requires manually drawn cutting lines after remeshing to achieve segmentation. As a result, the boundaries of the parts are often not smooth.

PartObjaverseTiny[48] retains the original geometry during annotation, but it lacks a hierarchical structure, making it unsuitable for models that require multi-granularity segmentation (e.g., PointSAM[53]).

In terms of scale, our dataset is comparable to PartNet[32] in quantity, whereas PartObjaverseTiny[48] contains only 200 models—insufficient for large-scale 3D applications. As for category diversity, our dataset includes more than twice the number of categories compared to PartNet[32]. Although PartObjaverseTiny[48] claims to have subcategories under its eight main categories, the limited quantity constrains its scalability for broader 3D model tasks.

In summary, PartNeXt offers clear advantages in terms of dataset scale and category diversity. It also preserves complete texture and original geometric structure, along with hierarchical annotations, enabling more nuanced and semantically rich 3D understanding tasks. We believe that PartNeXt will provide a stronger foundation and broader potential for future research in 3D segmentation and understanding.

D Detail of Segmentation BenchMark

We provide the detailed evaluation result and settings for our proposed benchmark on class-agnostic 3D part instance segmentation. All evaluations were conducted on a server equipped with 2 Intel(R) Xeon(R) Platinum 8581C processor and 4 NVIDIA L40 GPUs across experiments for fairness.

As shown in Table 8, we report the per-category metrics for each method included in the benchmark.

We evaluate each method using its publicly available codebase, and follow their original settings and hypermeters. SAMesh [39] utilizes textureless meshes, PartField [25] uses colorless point clouds, sampled from mesh surfaces, SAMPart3D [48] requires point clouds with color and normal attributes, so we sample colorized point clouds from textured meshes as input. Following the evaluation code of PartField, we convert the predicted masks from all methods into a unified format before computing the mIoU.

Method	Category mIoU (%)									
	Axe	Bed	Bookcase	Bottle	Bucket	Camera	Chair	Chandelier	Com_Keyboard	Com_Mouse
SAMPart3D	61.53	17.51	16.78	47.71	28.08	27.34	28.49	17.26	32.52	28.28
SAMesh	71.50	82.59	39.30	35.63	60.45	39.57	72.57	37.05	89.98	45.77
PartField	65.60	24.77	26.68	67.91	31.76	50.05	43.78	26.48	27.14	52.05
	Controller	Cup	Door	Fan	Flashlight	Floor_lamp	Fork	Glasses	Guitar	Hammer
	24.00	55.06	42.76	31.12	37.02	36.84	59.96	28.34	6.75	60.74
	47.71	37.78	64.66	56.72	32.79	59.87	36.26	33.38	33.68	43.33
	41.57	72.90	63.48	46.66	45.11	70.59	66.03	55.57	23.13	64.58
	Bag	Headphone	Knife	Lamp	Laptop	Microwave	Monitor	Mug	Padlock	Pen
	42.60	28.33	61.08	36.60	6.92	18.29	25.70	48.70	60.35	40.60
	68.69	45.41	51.19	58.49	24.08	42.16	45.16	82.25	78.12	33.48
	52.88	44.08	68.22	53.32	12.34	38.48	45.97	75.37	68.85	56.83
	Pickaxe	Piano	Pliers	Screwdriver	Scissors	Shovel	Skateboard	Sofa	Spoon	Step_ladder
	57.63	23.10	36.27	68.72	47.11	35.35	23.51	23.03	68.26	50.26
	66.94	39.10	36.63	35.47	55.96	38.03	43.45	54.48	44.13	81.22
	68.48	19.51	50.90	61.78	63.90	50.91	42.44	55.01	62.18	52.29
	Sword	Table	Teapot	Toaster	Toilet	Umbrella	Watch	Wrench	Washing_machine	Wifi_routers
	59.87	25.86	43.36	25.49	29.70	32.58	27.65	40.53	17.87	25.86
	63.72	64.81	73.24	64.26	45.41	47.01	50.46	52.17	41.82	32.75
	51.85	53.26	73.83	46.98	38.46	38.45	31.29	52.94	39.88	60.53

Table 8: Full quantitative comparison on our segmentation benchmark.

E Detail of LLM BenchMark

We provide detailed evaluation settings for our proposed Part-Centric 3D Question Answering BenchMark.

This benchmark experiments are conducted on the same hardware setup: a server equipped with 2 Intel(R) Xeon(R) Platinum 8581C processor and 4 NVIDIA L40 GPUs.

Since the models we evaluate require point cloud inputs, we uniformly sample 8192 points from the mesh using area-based sampling via the Trimesh library to ensure fairness in evaluation.

In all experiments, we use publicly available code. However, we observe that current LLM for 3D understanding shows limited capabilities in instruction following and response formatting. Even with carefully designed prompts, these models struggle to generate well-formatted outputs. To address this issue, we utilize a language model for output postprocess. We deploy the Qwen3-14B[47] AWQ quantized model using the vLLM framework[16] on 4 NVIDIA GeForce 2080Ti GPUs for postprocess.

For the Part Count task, the outputs are converted into a single numerical value, the prompt is shown in Listing 1. For the Part Classification task, we directly use the LLM to verify the correctness of the prediction with predicted label and ground truth label as input, output is either "True" or "False". The prompt is shown in Listing 2

Listing 1: System prompt for part count output conversion.

```
system_prompt = """
    You are a precise information extraction system. Analyze
       the given sentence describing the quantity of
       components and output a JSON object containing the
       numerical value.
    Follow these rules:

    Extract only the exact numerical quantity mentioned (either
        as digits or words)
    Ignore non-quantitative descriptors like size/color/
       material
    Return 0 if no quantity is specified
    Use this format: {"number": <extracted_value>}

    Examples:
    Input: "The chair contains three cushions"
    Output: {"number": 3}

    Input: "There are 8 buttons on the device"
    Output: {"number": 8}

    Input: "The lamp comes with a bulb"
    Output: {"number": 1}

    Input: "No additional parts included"
    Output: {"number": 0}
"""
```

Listing 2: System prompt for part classification correctness verification.

```
system_prompt = """
    You are a precise semantic verification system.
    Given a sentence describing a part of an object and a
       ground truth label, determine whether the described
       part corresponds to the given label.

    Follow these rules:

    - Focus on the identity or function of the described part.
    - Determine whether the described part refers to the same
       concept as the ground truth label.
    - Ignore attributes like color, position, size, material
       unless they are essential to identify the part.
    - Do not require exact word match - semantic equivalence is
        acceptable (e.g., "display" matches "Screen").
    - Output only a JSON object in this format: {"output": "
       True"} or {"output": "False"}.
    - Do not include any other text.

    Examples:

    Input:
    Sentence: "The red-highlighted section is the soft part
       people rest their heads on while sleeping."
    Ground truth: "Pillow"
    Output:
    {"output": "True"}

    Input:
    Sentence: "The red-highlighted component is the handle used
        to carry the toolbox."
    Ground truth: "Handle"
    Output:
    {"output": "True"}

    Input:
    Sentence: "The highlighted area is the internal CPU used
       for processing data."
    Ground truth: "Pillow"
    Output:
    {"output": "False"}

    Input:
    Sentence: "The indicated region is a curved support that
       connects both lenses at the nose."
    Ground truth: "Temple"
    Output:
    {"output": "False"}

"""
```

E.1 Detail of Part Count Task

For the Part Count task, we detail how prompts are provided to the model. We first select a set of parts from the hierarchies of each object category that are most meaningful for this task, for example, keys on a keyboard, fan blades of a fan, legs of a table or chair, and arms of a chandelier. The selected parts are typically repeated within a single object, and their counts vary across different object instances.

After processing with a 3D understanding LLM and a separate language model for output post-processing, we obtain a predicted part count for each specified part of every object. We evaluated the performance of different 3D understanding LLMs using **Mean Absolute Error** (MAE) as the evaluation metric. The calculation is as follows:

Given an object i with n_i parts that have predicted and ground-truth counts, the object-level MAE is defined as:

$$\mathrm{MAE}_i = \frac{1}{n_i} \sum_{j=1}^{n_i} |\hat{c}_{i,j} - c_{i,j}| \tag{1}$$

where $\hat{c}_{i,j}$ and $c_{i,j}$ denote the predicted and ground-truth count of the j-th part in object i, respectively.

The overall MAE for the 3D understanding LLM is then computed by averaging over all M objects:

$$\mathrm{MAE}_{\mathrm{avg}} = \frac{1}{M} \sum_{i=1}^{M} \mathrm{MAE}_i \tag{2}$$

E.2 Detail of Part Classification Task

For the Part Classification task, we detail how prompts are provided to the model. Since our dataset includes segmentation masks, when querying the LLM about a specific part, we highlight the corresponding point cloud segment in red to serve as an effective 3D prompt. Note that, as some objects may inherently contain red regions, we intentionally avoid selecting objects with large red areas to prevent ambiguity. Examples of our processed point cloud prompts are shown in Figure 7.

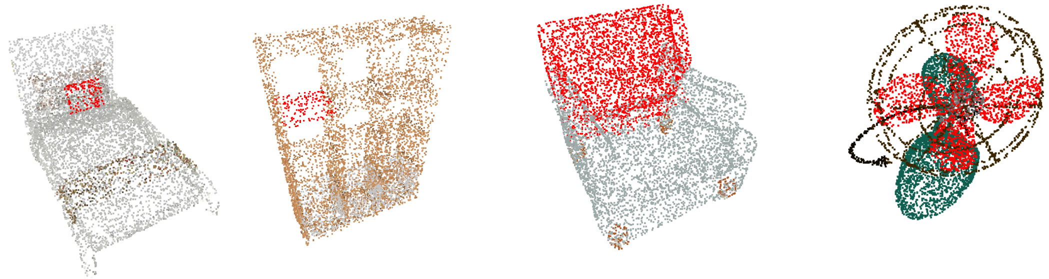


Figure 7: Visualization of Prompted Input for Part Classification Task, the target part is highlighted in red as 3d prompt.

E.3 Detail of Part Grounding Task

For ShapeLLM[35], we follow the original evaluation code, requiring the model to output the coordinates of the eight vertices of the bounding box. We then compute the minimum and maximum values to calculate the IoU. An example of the output from ShapeLLM[35], along with the ground truth, is shown in Figure 8. These results indicate that current 3D language models are still in the early stages when it comes to the Part Grounding task.

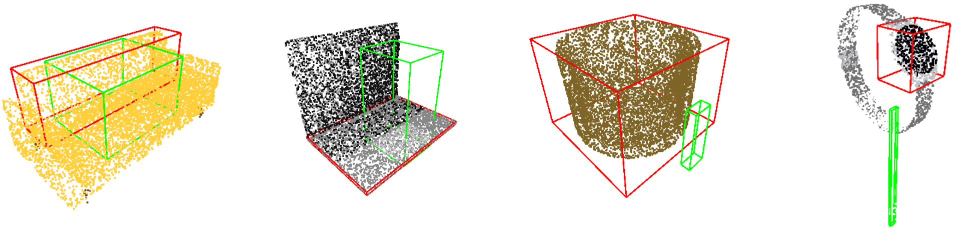


Figure 8: Visualization of ShapeLLM grounding result, the red box is the ground truth, while the green box is predict result.

In Figure 9, we present several output examples from 3DLLM[11] and PointLLM[46]. As shown, both types of models struggle to generate precise bounding box coordinates, indicates the weak ability of part grounding.

Prompt 1

```
Where is the screen?
```

Output 1

```
The screen is located in the middle of the computer setup.
```

Prompt 2

```
This is a monitor. Where is the screen?
```

Output 2

```
The screen is turned off, hence black.
```

Prompt 3

```
Where is the screen? Please answer with the coordinates of the
    eight vertices of its bounding box.
```

Output 3

```
The screen is at the center of the bounding box. The vertices
    are (0,0), (0,1), (1,1), and (1,0).
```

Prompt 4

```
This is a monitor. Where is the screen? Please answer with the
    coordinates of the eight vertices of its bounding box.
```

Output 4

```
The screen is located at the center of the monitor.
```

Prompt 5

```
Where is the screen? Please answer with the coordinates of the
    eight vertices of its bounding box, in the format: [[x1, y1
    , z1], [x2, y2, z2], ..., [x8, y8, z8]].
```

Output 5

```
The screen is located at the origin of the bounding box, which
    is also its center. The eight vertices of the box are: [[0,
     0, 0], [0, 0, 1], [0, 1, 0], [0, 1, 1], [1, 0, 0], [1, 0,
    1], [1, 1, 0], [1, 1, 1].
```

Prompt 6

```
This is a monitor. Where is the screen? Please answer with the
    coordinates of the eight vertices of its bounding box, in
    the format: [[x1, y1, z1], [x2, y2, z2], ..., [x8, y8, z8
    ]].
```

Output 6

```
The screen is at the center of the monitor.
```

Figure 9: Example response of 3DLLM and PointLLM under different prompts.

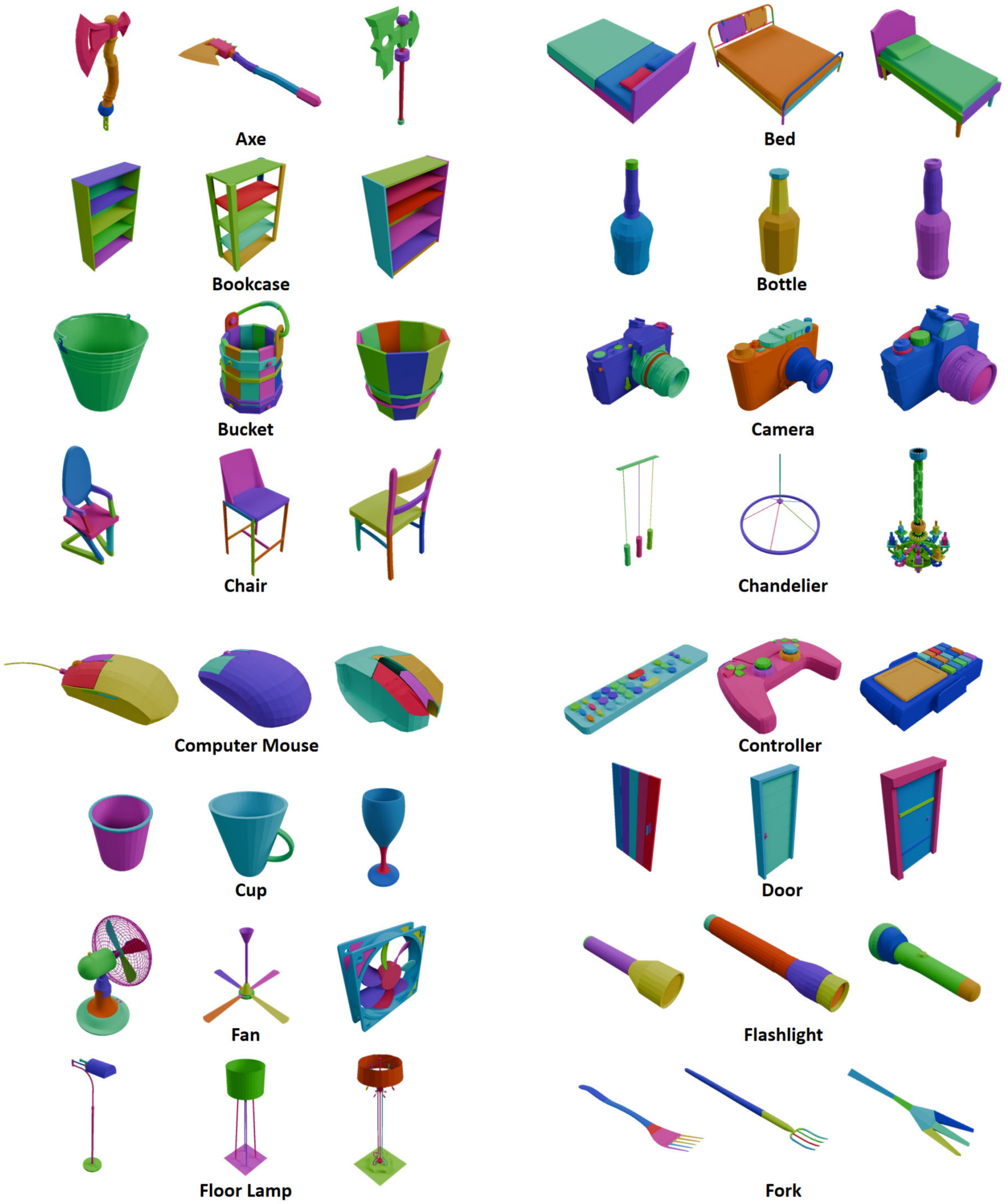


Figure 10: Visualization of our Annotation Results.

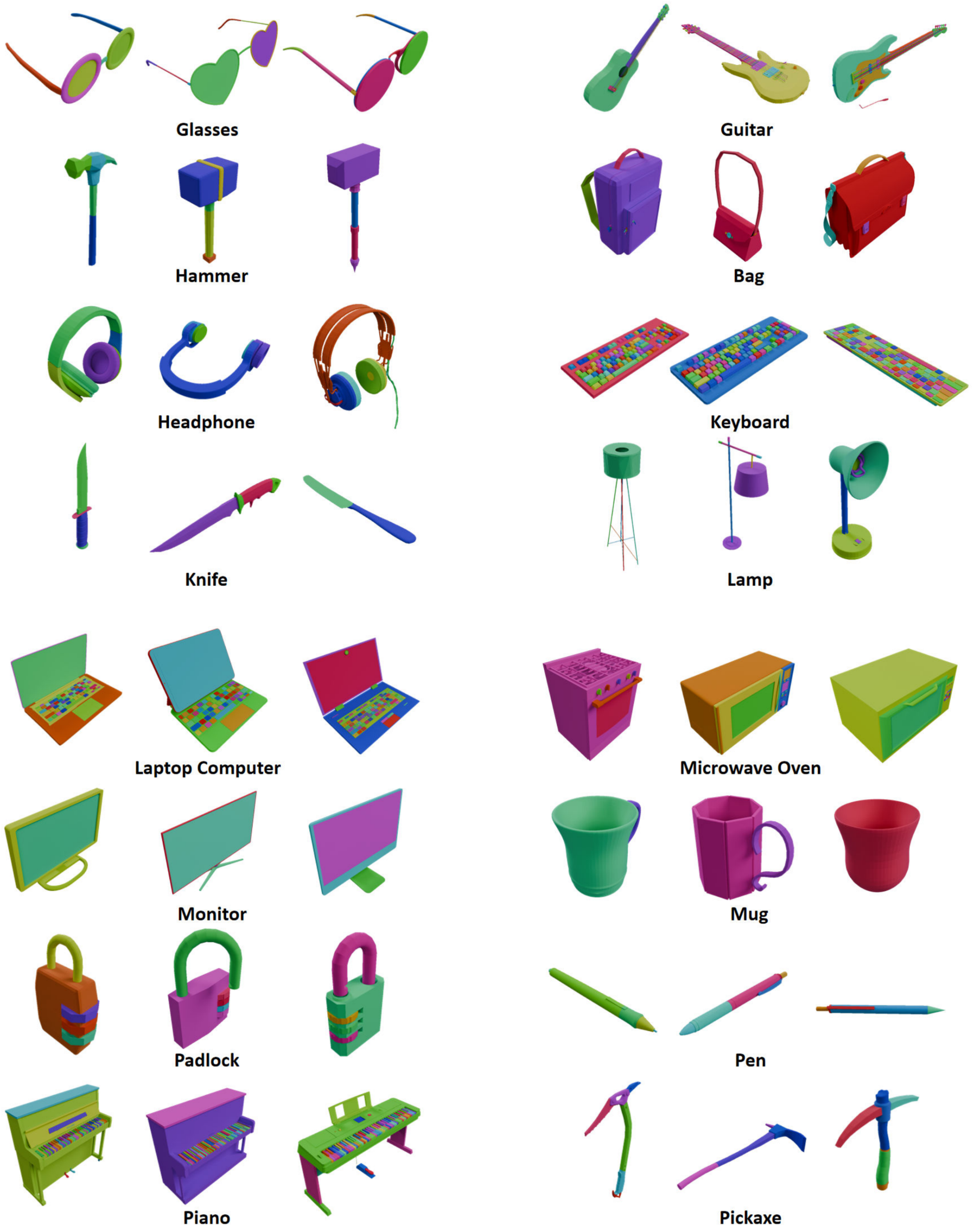


Figure 11: Visualization of our Annotation Results.

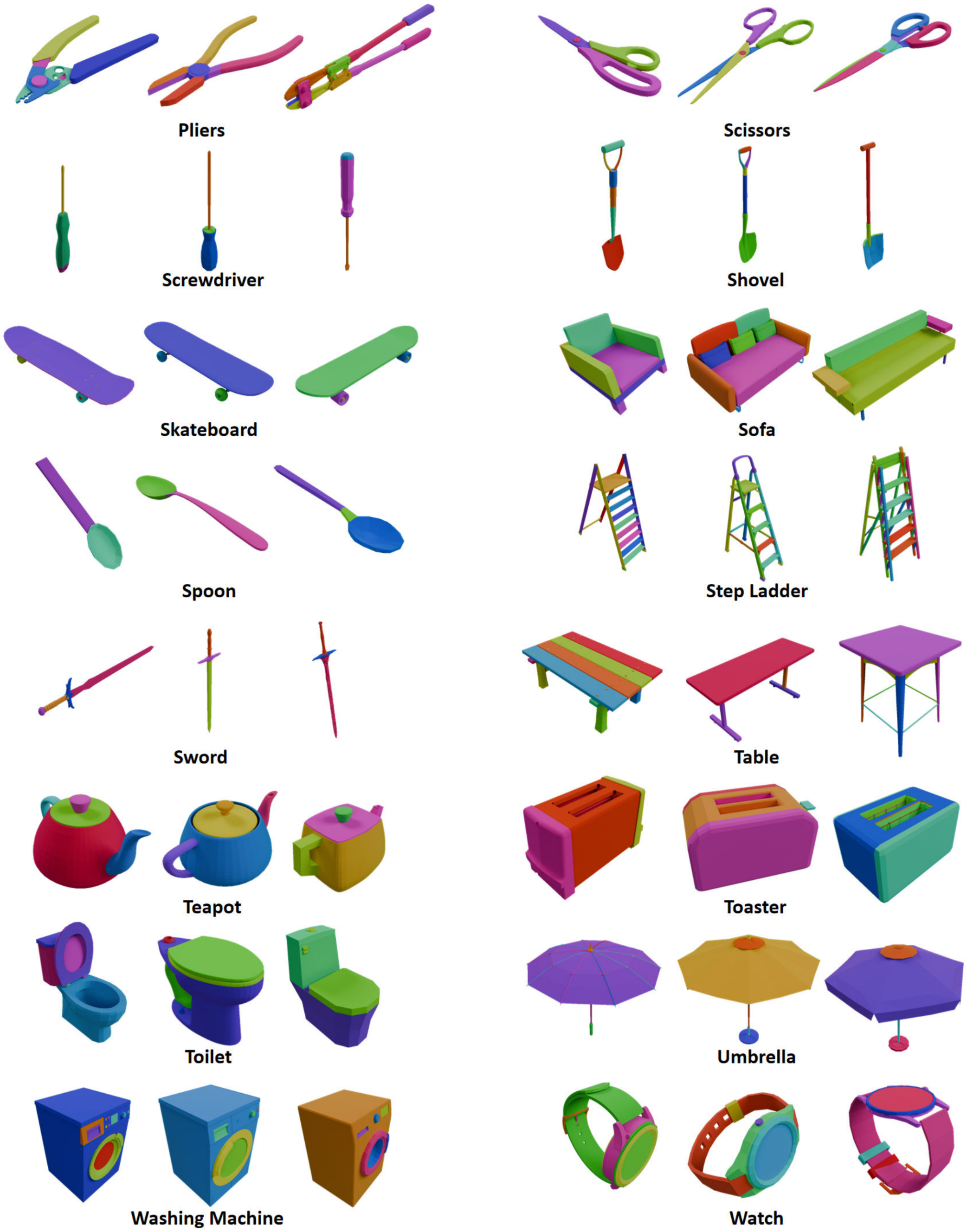


Figure 12: Visualization of our Annotation Results.

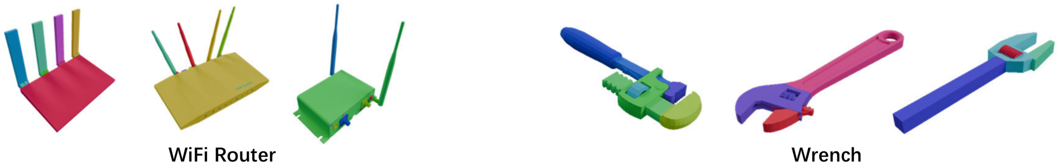


Figure 13: Visualization of our Annotation Results.

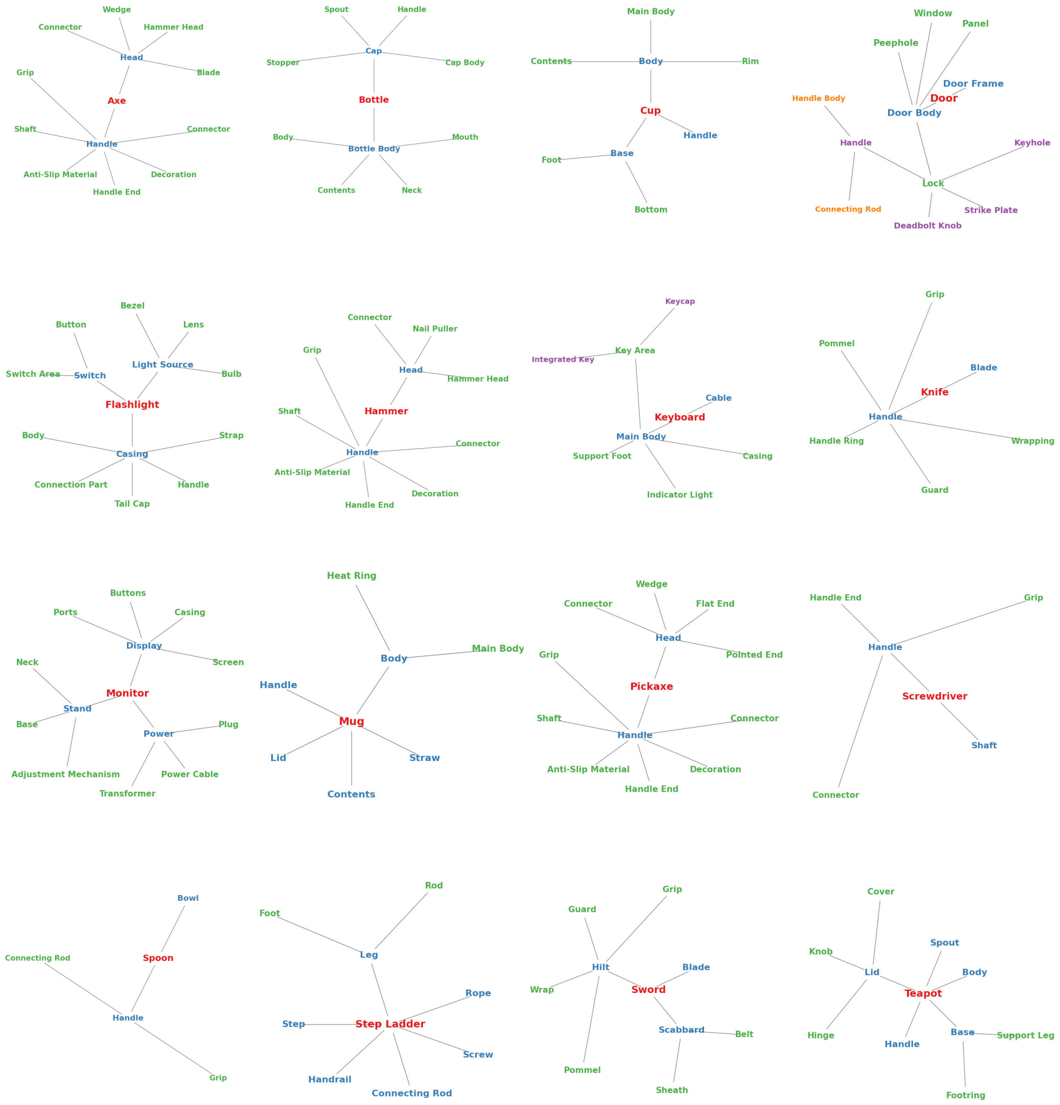


Figure 14: Visualization of our Predefined Hierarchy

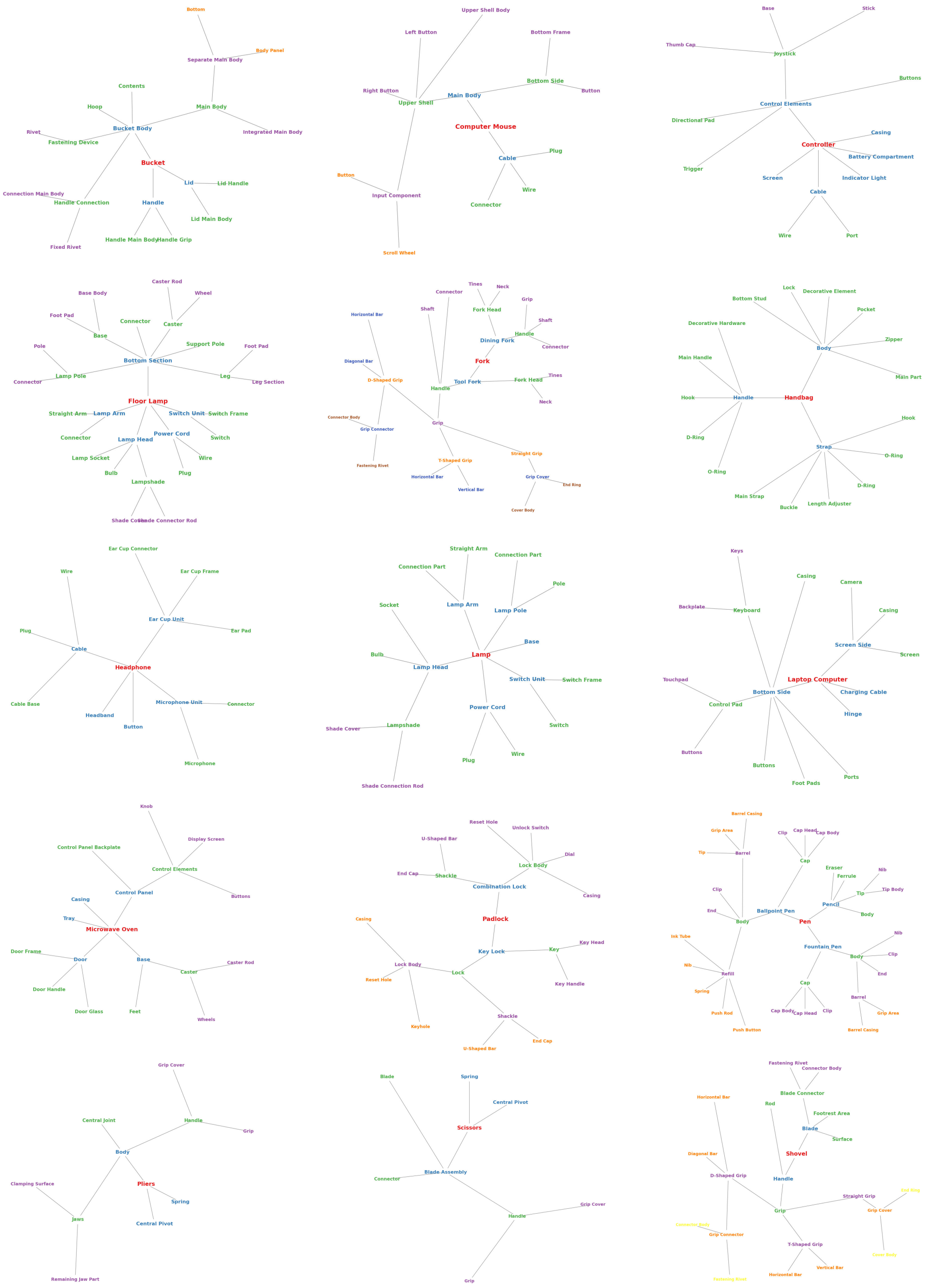


Figure 15: Visualization of our Predefined Hierarchy

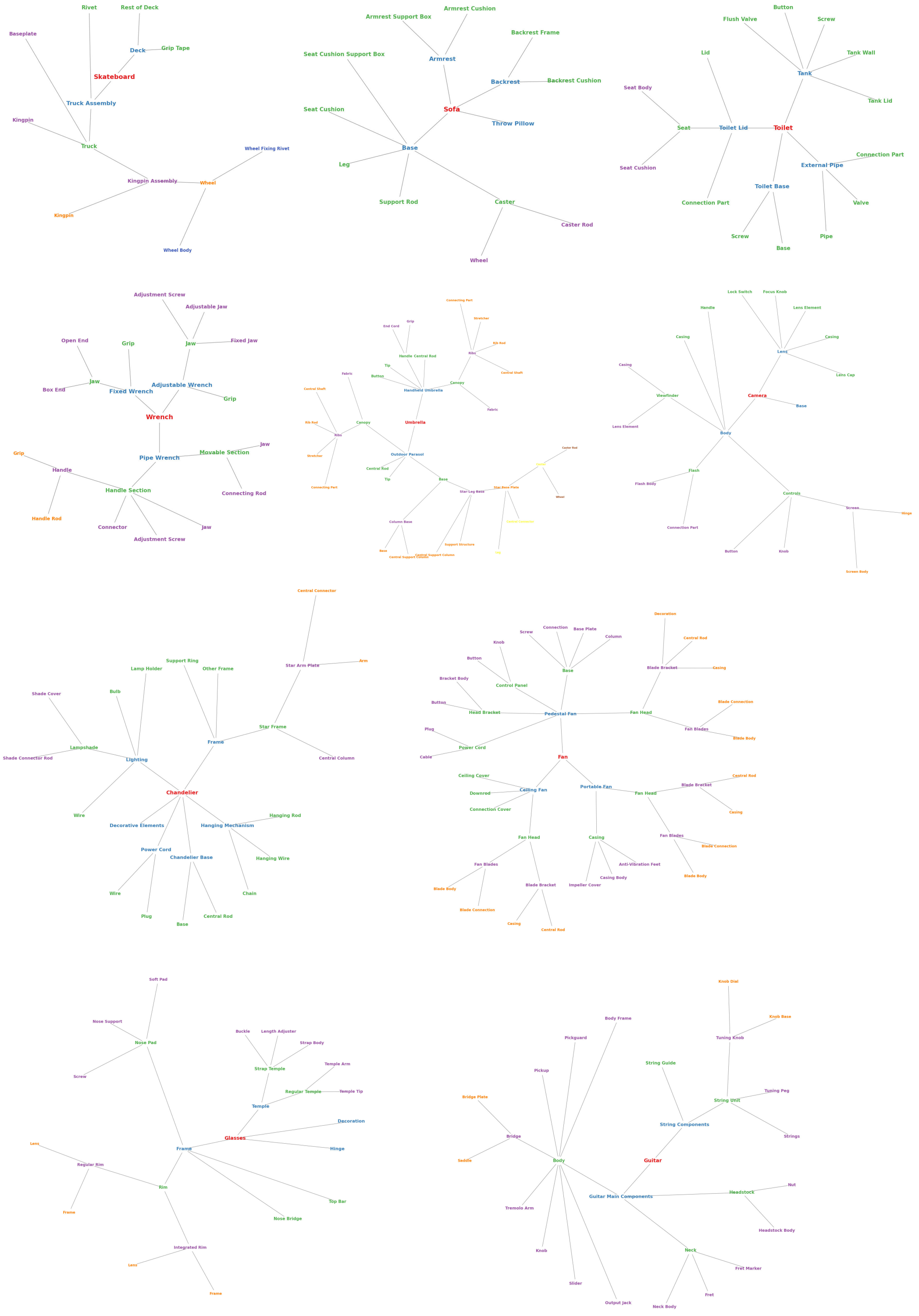


Figure 16: Visualization of our Predefined Hierarchy

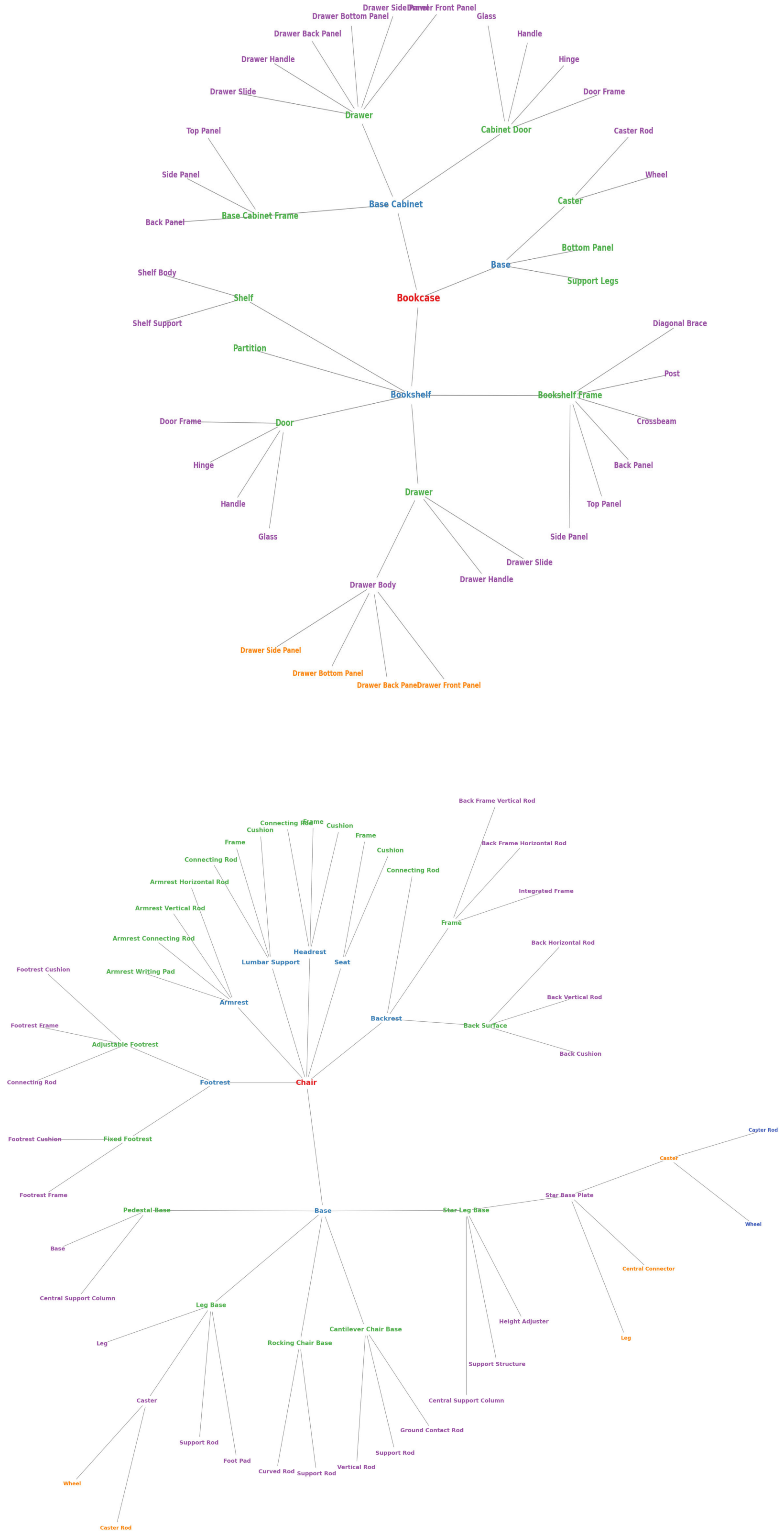


Figure 17: Visualization of our Predefined Hierarchy

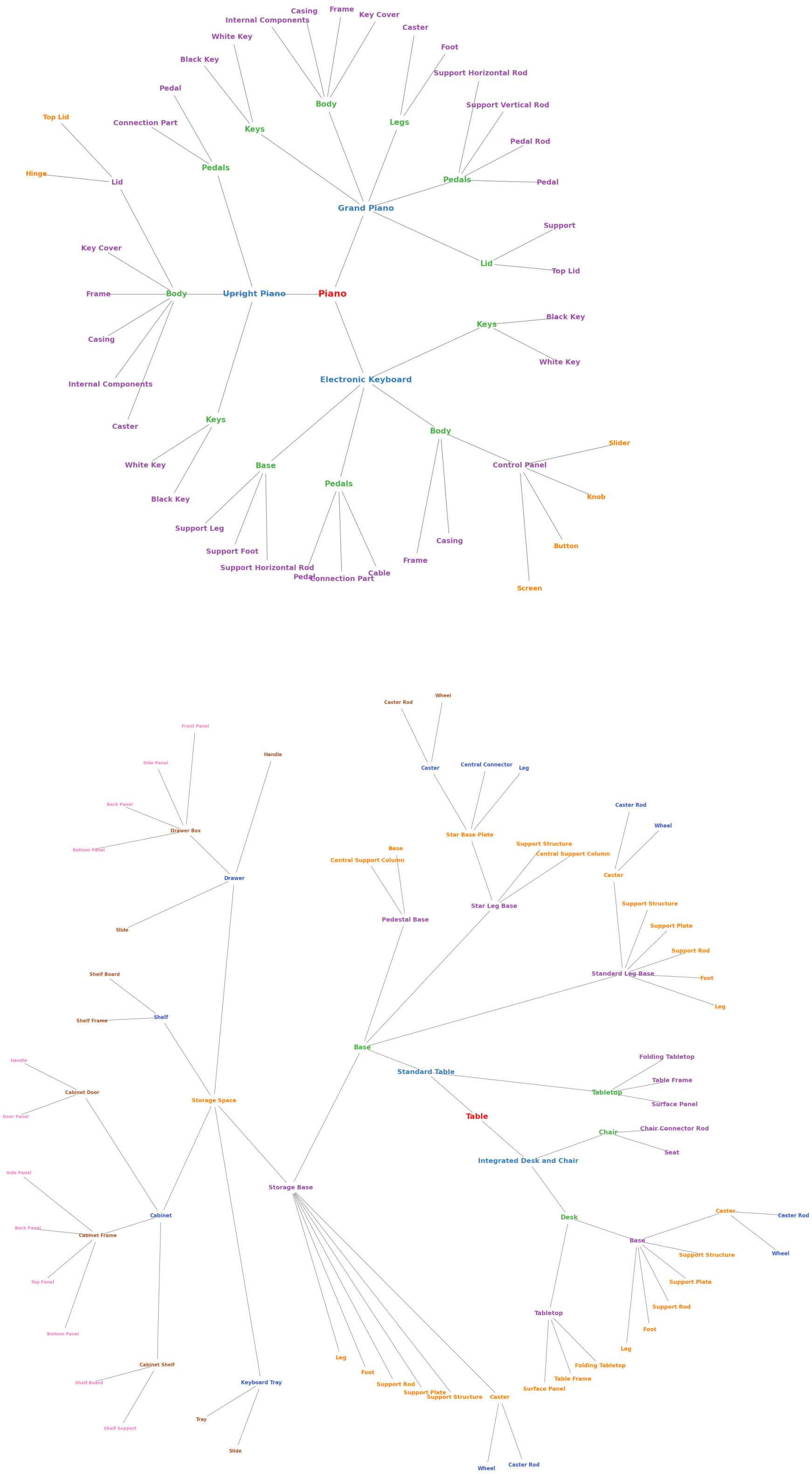


Figure 18: Visualization of our Predefined Hierarchy

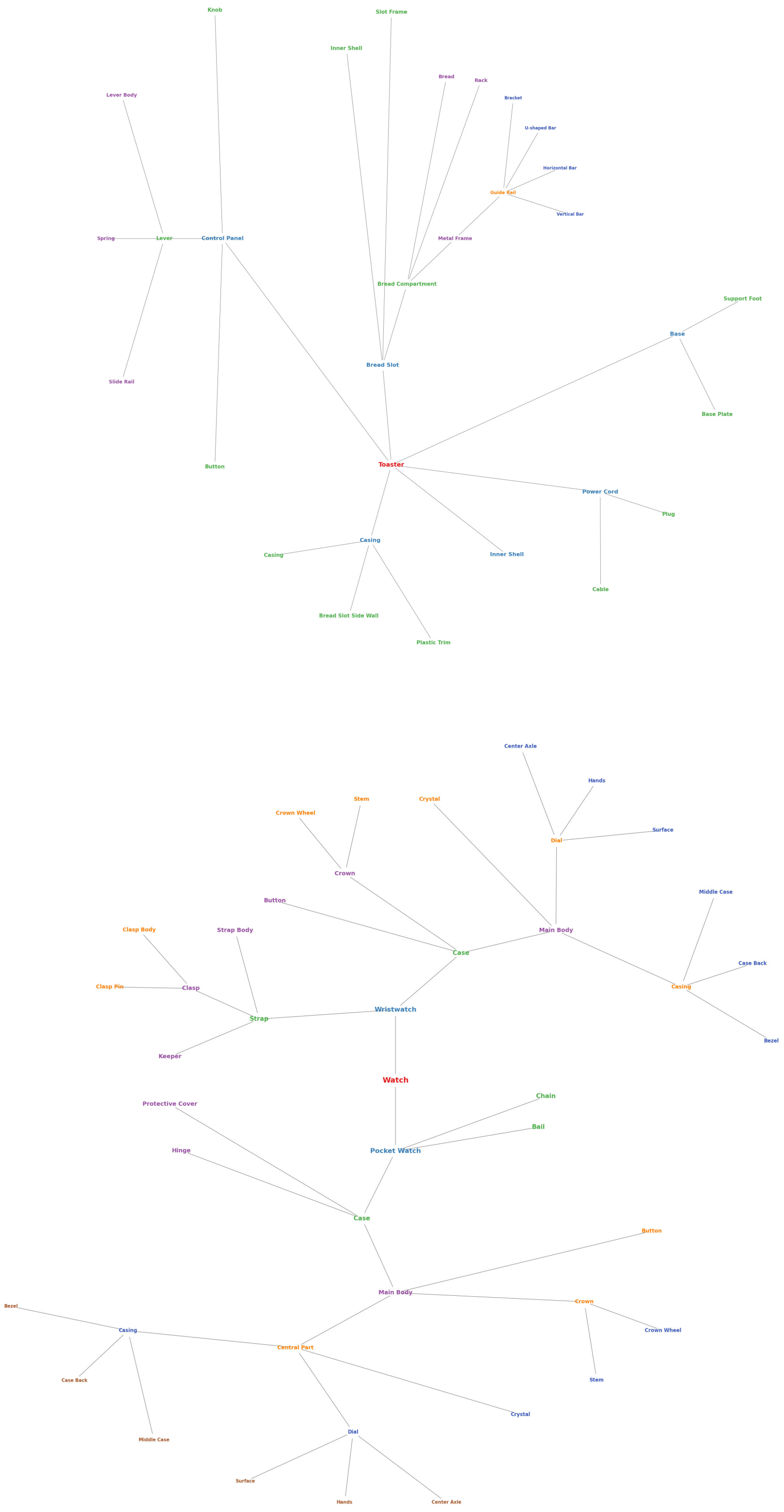


Figure 19: Visualization of our Predefined Hierarchy

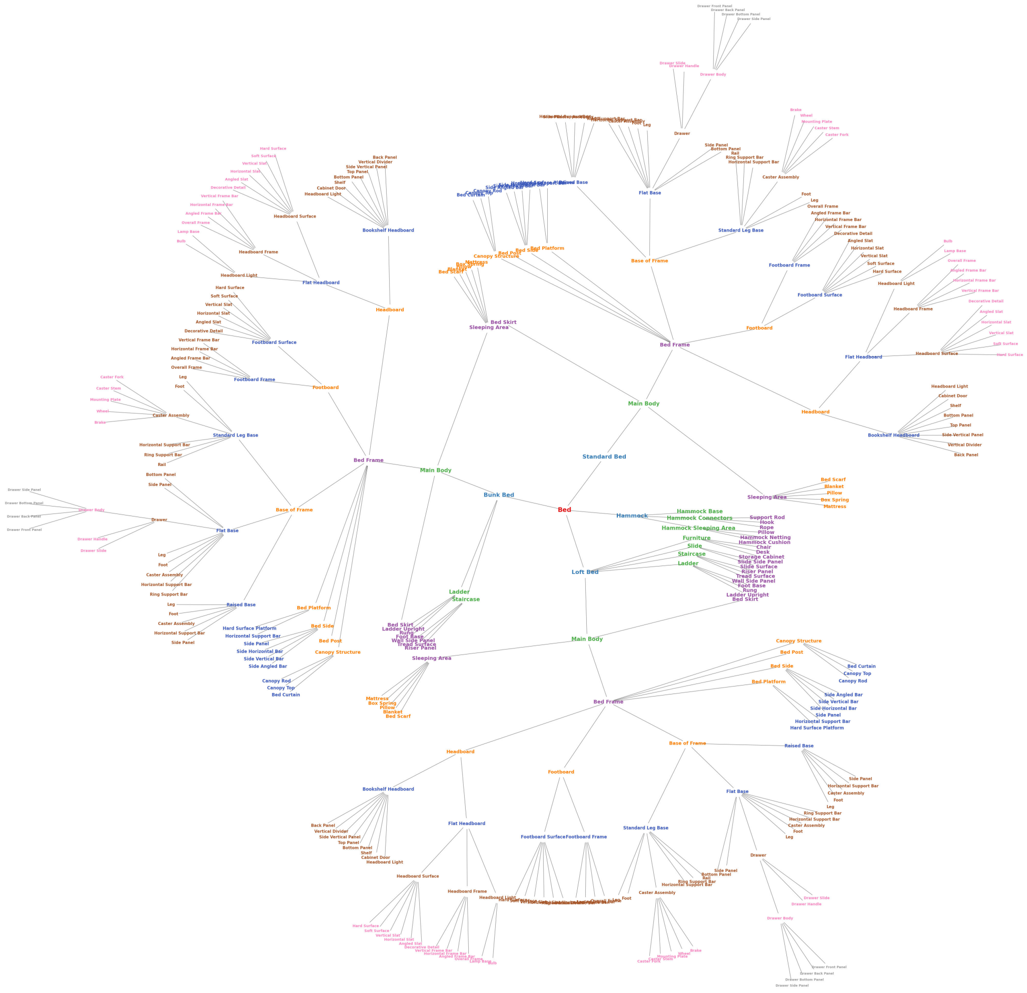


Figure 20: Visualization of our Predefined Hierarchy