R3DS: Reality-linked 3D Scenes for Panoramic Scene Understanding Supplemental Materials

In this supplement, we provide related works (Appendix A), additional examples and statistics for our R3DS dataset (Appendix B), dataset construction (Appendix C), details on our annotation interface (Appendix D), PanoSun task (Appendix E) and additional results (Appendix F).

A. Related Work

3D scene datasets. A spectrum of scene datasets have been used for scene understanding tasks. One type provides annotated 3D reconstructions of real scenes based on RGB-D videos [2, 5, 9, 14, 18, 24]. These datasets are usually subject to the limitations of RGB-D reconstruction, typically containing noise, artifacts such as holes, and poor reconstructions of thin structures, shiny objects, or light sources. Another type of 3D datasets is authored by manually designing 3D object assets [4, 8] and inserting them into synthetic 3D scenes [7]. However, such datasets lack the realism of real-world reconstructions and demand expert knowledge, making them expensive to create. A third, hybrid approach which is closest to our work creates 3D scene datasets by aligning existing object CAD models to real world data.

Datasets that align CAD models to real world. There have been a number of recent efforts in aligning CAD models with real-world data. Prior work [12, 20, 23] has annotated object images with 3D models, typically using keypoint correspondences to perspective images. These perspective images usually do not depict a complete scene; they typically focus on one or two objects and are limited in field of view, resulting in a sparse proxy of the real scene.

Another line of work aligns 3D CAD models to RGB-D scans either through annotation as in Scan2CAD [1], or automated heuristics as in iGibson [17]. OpenRooms [11] extends Scan2CAD [1] with photorealistic material annotations and focuses on inverse rendering tasks. Conceptually, these allow for more complete synthetic scene proxies. However, statistics from these datasets show that they are still relatively sparse (see **??**). In addition, the poor quality of reconstruction makes aligning CAD models challenging without referring to the original RGB images. A prominent exception is Replica [18] which has fairly high-quality reconstructions and the artist-created Replica-CAD [21]. However, creating such high quality "replicas" is labor intensive and costly. Szot et al. [21] report 900+ work hours required to model approximately 90 objects, resulting in a dataset of limited scale with 105 different layouts of what is effectively a single room.

More recently, Maninis et al. [13] introduced CAD-Estate, which aligns CAD models to RGB videos for over 19K spaces. Because the data is based on monocular video, the coverage of the spaces is incomplete. In addition, the annotation is relatively sparse, with an average of only 6 objects per scene.

Datasets for panoramic scene understanding. There have been relatively few datasets introduced for Panoramic Scene Understanding [6, 25, 26]. In the initial PanoContext dataset [26], the data did not have aligned CAD models and only included object cuboids. The ground truth data was collected on 2D panorama images by annotating visible cuboid vertices; 3D cuboids were obtained by minimizing the re-projection error from the annotated 2D vertices. Moreover, these 3D cuboids and the room layout are obtained with the assumption that the room layout is a cuboid and that the objects are vertically aligned. Thus, the resulting object layout may deviate from the real arrangement of objects. More recently, datasets for Panoramic Scene Understanding have been built by taking 3D scans, aligning CAD objects to them, and then generating panoramas [6, 25]. Compared to these datasets, our R3DS is manually curated for a larger number of distinct regions and provides support hierarchy and matching object set annotations.

B. R3DS dataset examples and statistics

We show a histogram of the region types covered by our dataset (Figure 1), and histograms of the object categories (Figures 2 and 4 to 7). We first show a histogram of the 20 most commonly occurring coarse object categories in Figure 2 and then fine-grained category distributions for some broader object categories such as 'Chair', 'Sofa', 'Table' and 'Lighting' in Figures 4 to 7. We also present a box plot of the physical size distribution (measured by volume in m^3) per category in Figure 3.

We show additional qualitative examples of scenes in our

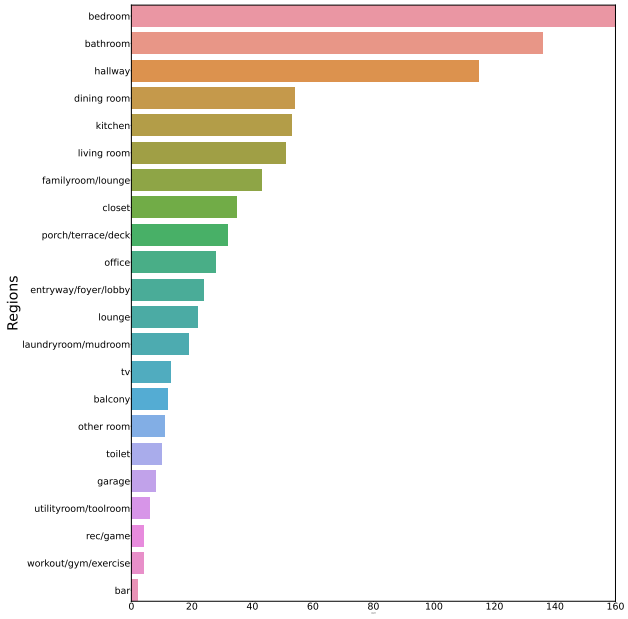


Figure 1. Histogram of all region (i.e. room) types in our R3DS dataset. The three most common region types are bedroom, bathroom and hallway, but there is a long-tail distribution with many other region types.

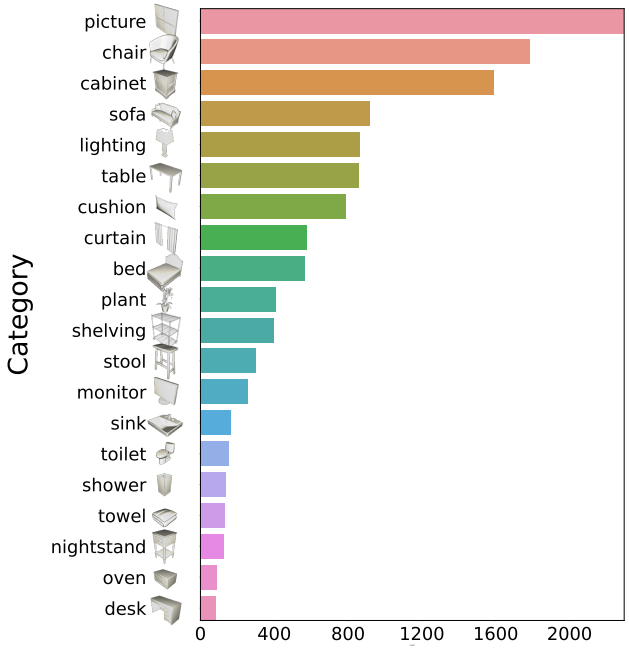


Figure 2. **Coarse Object Categories.** Histogram of the 20 most common object categories in R3DS. We see that the scenes in our dataset exhibit a long-tail distribution over common object categories occurring in real-world scenes.

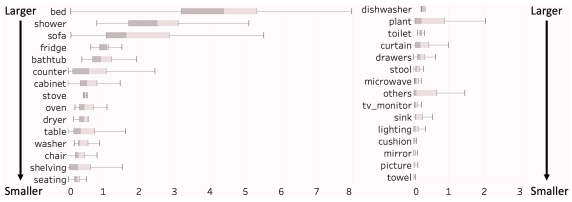


Figure 3. Box plot of the physical size distribution (measured by volume in m^3) per category, showing a broad spectrum of sizes with several small categories (right side).

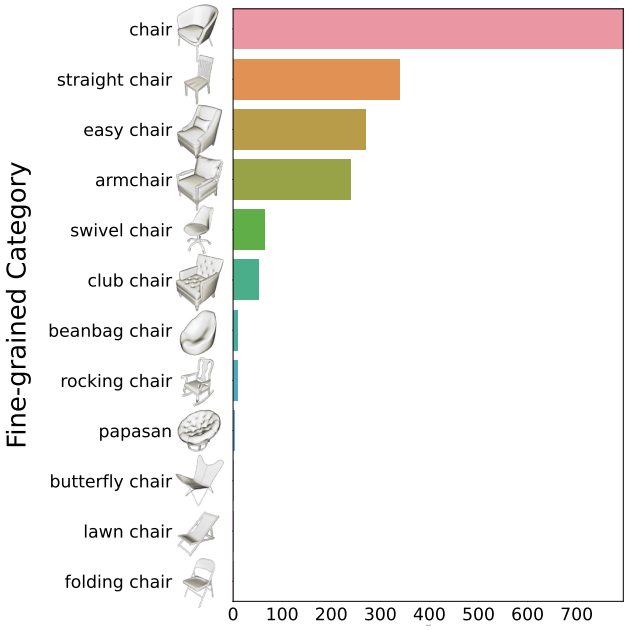


Figure 4. Histogram of the fine-grained categories for 'Chair'. We see a variety of types of chair instances present in scenes from our dataset.

R3DS dataset in Figures 8 and 9.

We also provide statistics of object support relations to architecture elements (Figure 10) and other objects (Figure 11). As expected, we see that chairs typically go on floors, while curtains are supported by walls. From Figure 11, we see that cushions are typically found on beds, chairs, and couches while towels are typically found on shelves. Similarly, in Figure 12 we show the object-to-region statistics. We see that some object categories tend to appear more frequently in a particular region (i.e. room) type. For example, couches are more frequently found in living rooms than in bedrooms.

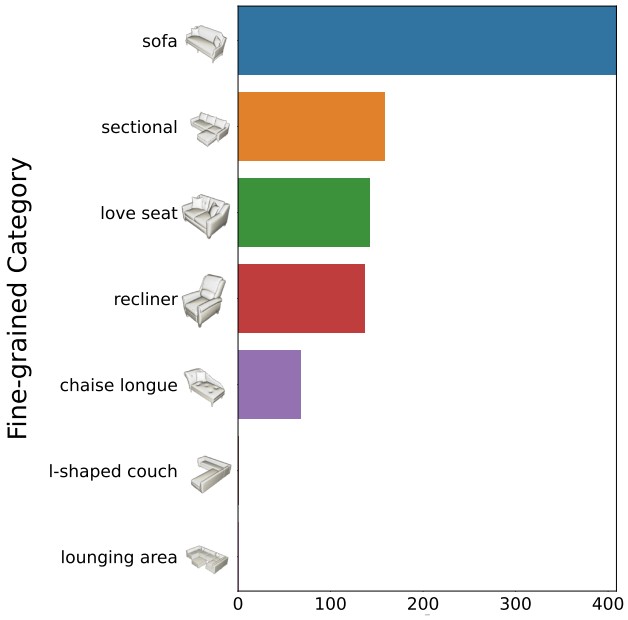


Figure 5. Histogram of the fine-grained categories for 'Sofa'. There are a variety of sofas in our scenes.

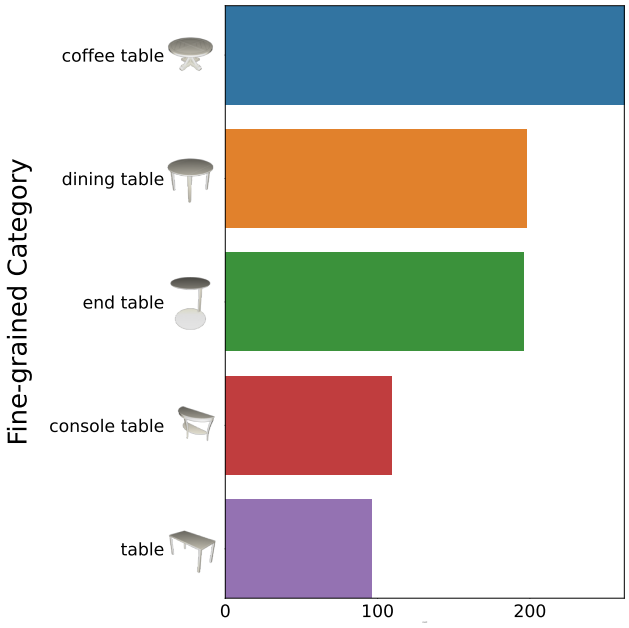


Figure 6. Histogram of the fine-grained categories for 'Table'. A variety of tables are present in our dataset scenes.

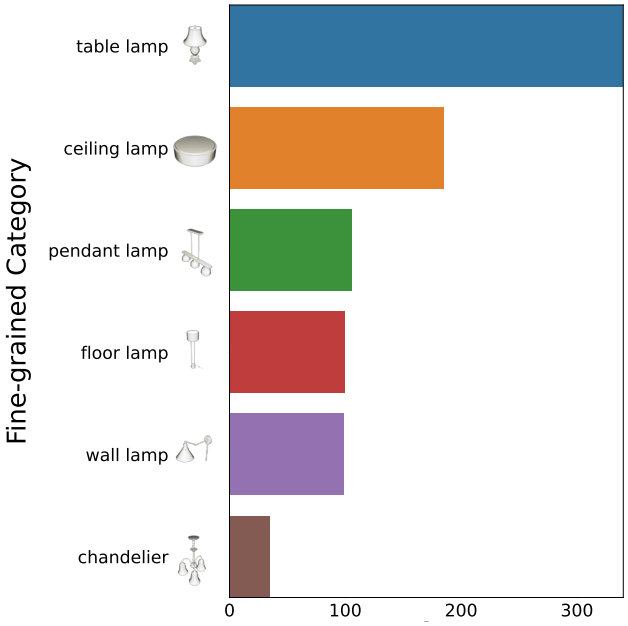


Figure 7. Histogram of the fine-grained categories for 'Lighting'. Like chairs, there are a variety of types of lighting fixtures in our scenes. Note that these types of lights are also found in dramatically different support relations with the architecture and other objects (e.g., table lamp vs ceiling lamp vs floor lamp).

Figure 8. Top-down overviews of various annotated scenes with objects colored by instances. Our dataset covers several region types including kitchen, bedroom, bathroom, office, lounge room and more. The object arrangements are dense, with objects supported by other objects (e.g., pillows on beds and couches) and by architectural elements (e.g., paintings on walls and curtains on windows).

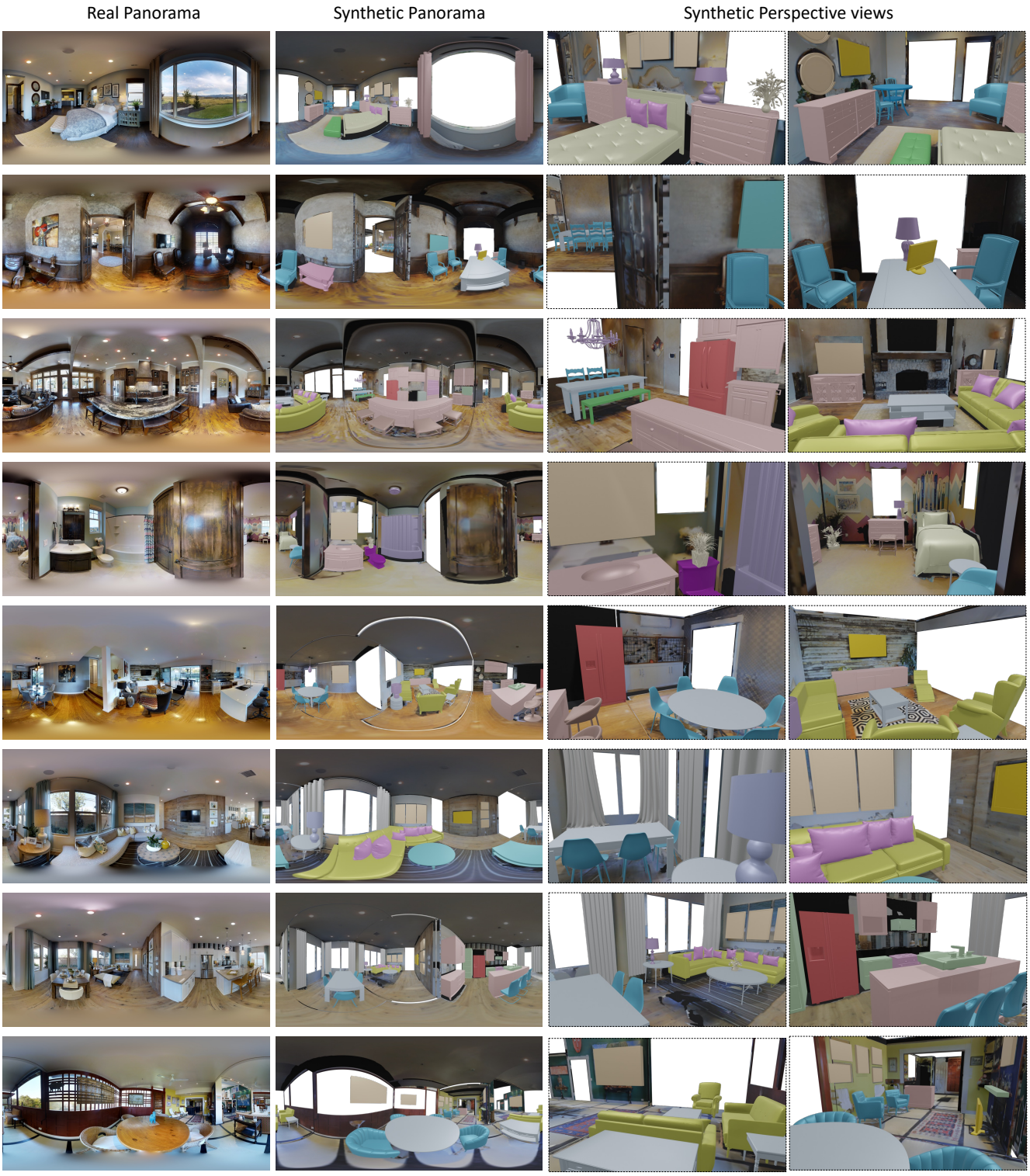


Figure 9. Examples of real-world panoramas and corresponded synthetic panoramas. The synthetic scenes are rendered using the annotated textured 3D architecture and all placed synthetic objects, from the same camera pose as the original panorama. The rightmost two columns show additional sampled perspective views within each scene. Objects are colored according to semantic category label. The scenes are densely populated with objects plausibly arranging on other objects and with respect to the architecture.

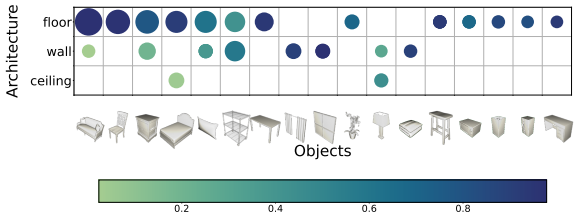


Figure 10. **Object-to-architecture support.** The plot shows support between objects and architecture elements (floor, wall, ceiling). The circle radius indicates the average number of the object type (ranges from 5 to 24) supported by a given arch type per panorama, whereas the color indicates the number of times the object type is supported by that arch type out of the total number of times the object type appears. For example, we see that on average, same number of shelves can be found on the floor and the wall in a panorama (circle radius), but overall shelves appear more on the wall than on the floor (color).

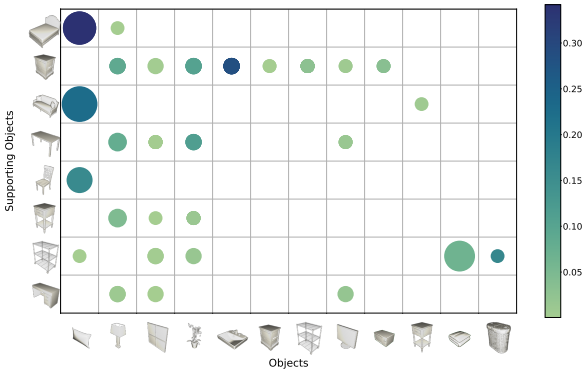


Figure 11. **Object-to-object support.** The plot shows support between various objects in R3DS. The circle radius indicates the average number of the object type (ranges from 1 to 10) supported by the parent object type per panorama, whereas the color indicates the number of times the object type is supported by the parent object type out of the total number of times the object type appears in the dataset. For example, we see that on average, more pillows are found on a couch than on a bed in a panorama (circle radius), but overall pillows appear more frequently on the bed in the dataset (color).

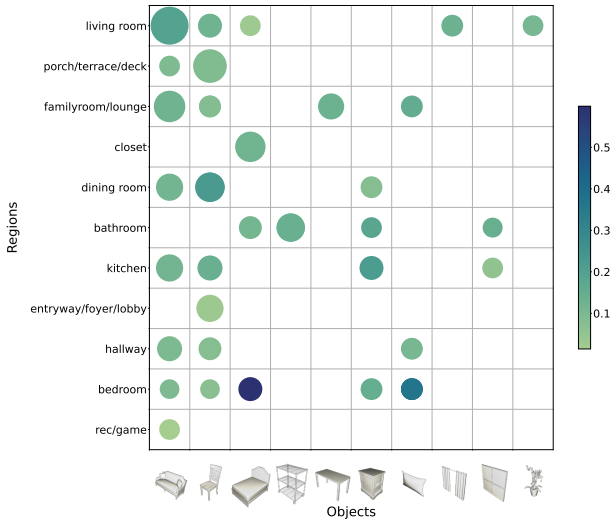


Figure 12. **Object-to-region.** The plot shows objects found in various regions in R3DS. The circle radius indicates the average number of the object type (ranges from 7 to 24) found in a region per panorama, whereas the color indicates the number of times the object type occurs in that region out of the total number of times the object type appears in the dataset. For example, we see that on average, similar number of chairs are found in dining room and lobby (circle radius), but overall chairs appear more frequently in the dining room in the dataset (color).

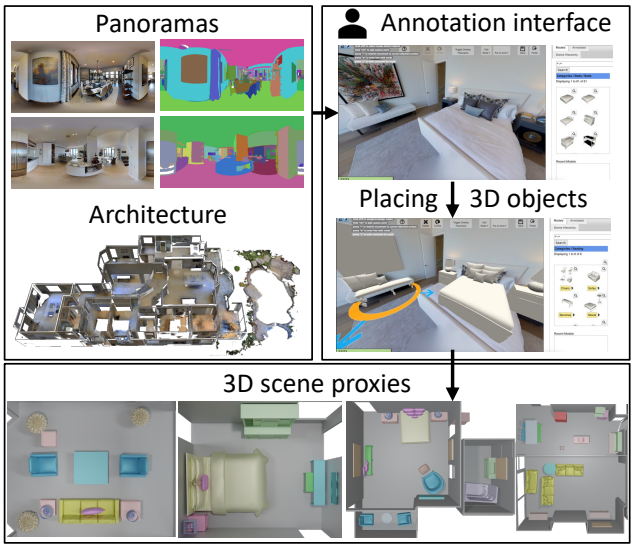


Figure 13. **R3DS annotation pipeline**. Annotators see an empty scene (architecture only). They then insert and manipulate 3D object models from a panorama viewpoint to create a populated 3D scene proxy corresponding to the panorama.

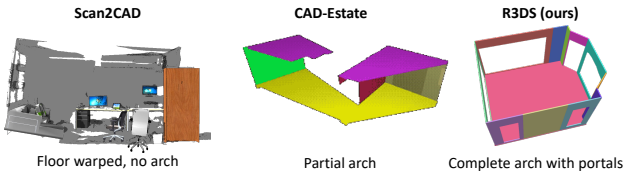


Figure 14. **Architecture comparison.** Compared to Scan2CAD (no architecture) and CAD-Estate (partial architecture), R3DS provides complete architecture with door/window portals.

C. Dataset construction process

We developed a 3D annotation interface (Fig. 13) showing a panorama of a room from Matterport3D and allowing users to insert 3D CAD objects into a 3D scene which is *visually overlaid* on the panorama. The 3D scene is initially empty, consisting only of 3D architectural geometry which specifies the walls, floor, ceiling as well as the placement of openings (e.g. doors, windows, and other openings) on the walls. We create this 3D architecture by taking 20 houses from Matterport3D, constructing an initial architecture based on the region and object annotations for the windows and doors, and manually refining the placement of walls and openings. By combining panoramas and 3D architectures, users can see through openings and annotate objects located in other rooms.

We ask annotators to select and place 3D object models to best match the panoramic image. We use CAD models from Wayfair [15] and ShapeNet [3] models collected from 3D Trimble warehouse. Wayfair provides a large collection of furniture CAD models that match real-world products and are sized based on real-world dimensions. However, it does not include bathroom fittings, electronic equipment and kitchen appliances, for which we manually scale and align CAD models from ShapeNet. Compared with ShapeNetCore, the CAD models we use are already sized to real-world sizes (instead of normalized to a unit cube).

To assist the annotators, we provide segmented masks of objects visible in the panorama. Since Matterport3D has annotated 3D object masks on the scans we use those annotations, but it is also possible to run an instance segmentation on the panorama. When the user clicks one of these masks, a search panel automatically opens and shows objects matching the clicked mask category label. For each mask, the annotator selects a matching object and positions and aligns it to match the mask. Annotators are instructed to choose objects which match the *shape* of the corresponding object in the panorama (rather than its color or texture). To help annotators focus on shape, we render all 3D objects in a neutral gray color. Annotators are also explicitly asked to select the same 3D asset for objects that should be the same; our interface provides a list of recently selected assets to make this process easier. In addition, annotators are instructed to add annotations for any objects that are not segmented (due to errors in Matterport3D) through simple clicks. These additional objects provide a more complete annotation that covers poorly reconstructed objects such as glass tables, lamps, and other small objects. The interface enforces that each object is placed on a support surface (either an architecture element or another object). The annotator can review their work by toggling off the panorama overlay or by switching to a perspective view of the 3D scene.

D. Annotation interface details

Our annotation interface consists of a web interface developed using `three.js` that allows users to insert 3D assets into the scene while *visually overlaid* on the panorama. To achieve this, our interface assumes that there is a set of panoramas with corresponding camera poses, and an 3D architecture on which the objects can be placed. We implement two viewing modes, panorama mode and architecture mode, to let users switch between overlaid panorama and underlying 3D scene.

Data Assets. We construct a parametric 3D architecture for 20 Matterport3D scenes. We take the region annotations that specify wall segments to create the initial 3D architecture. We then project annotations for the labels relating to windows and doors to get an initial estimate for the placement of doors and windows on the architecture. Next, we create a textured architecture by rendering the reconstructed scene onto the estimated surfaces of each architecture element plane (wall, floor, ceiling). Using a 3D interface that shows the architecture, we manually refining the wall

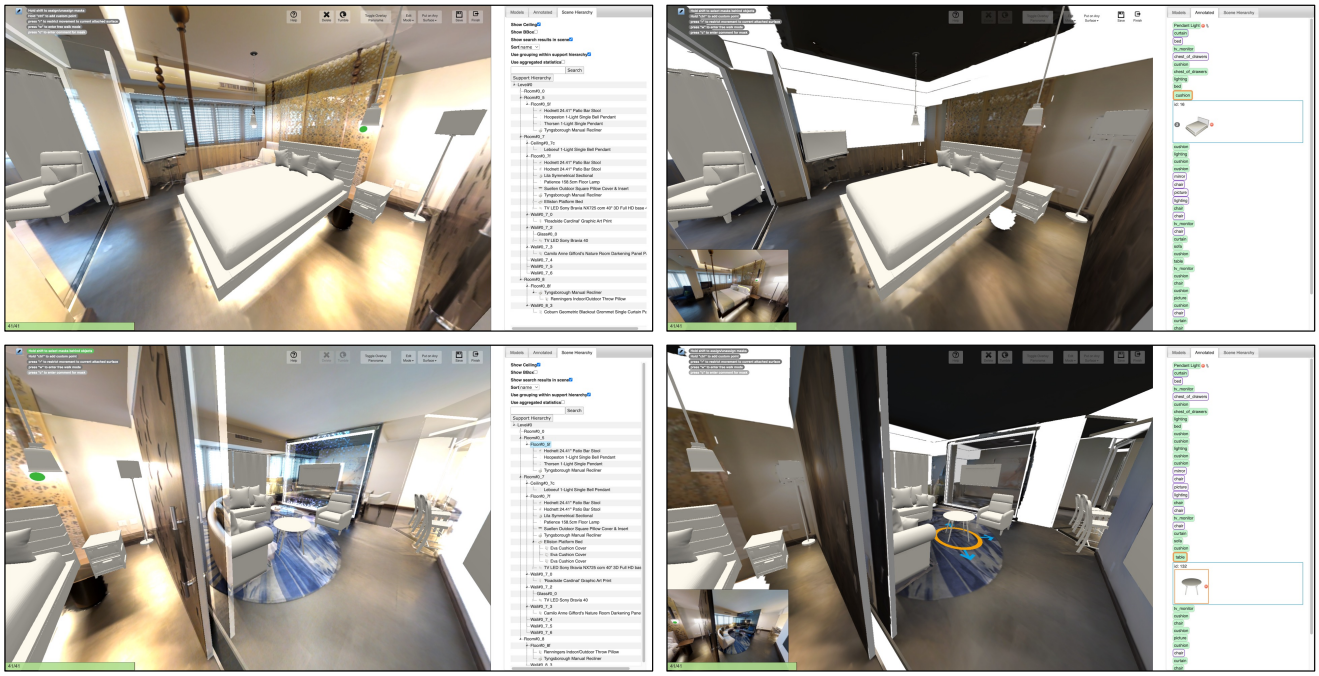


Figure 15. The R3DS interface allows users to select 3D CAD models and place them in the scene. Users can see the alignment of the object against the panorama. Camera controls allows the user to rotate the camera and to see different perspective views of the scene. The user can also toggle the overlaid panorama on (left) or off (right), to get a better view of the underlying 3D architecture and all objects placed into the scene thus far using the interface.

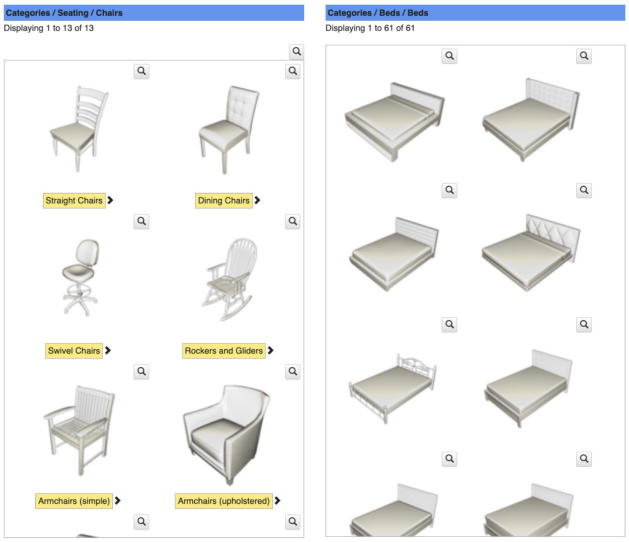


Figure 16. The user sees a list of candidate CAD models that is filtered depending on the semantic object category of the mask that was clicked in the panorama view. The list is hierarchical, allowing the user to refine the category into finer-grained categories such as the examples of chair types on the left side, and then select an appropriate instance of a chair within the finer category.

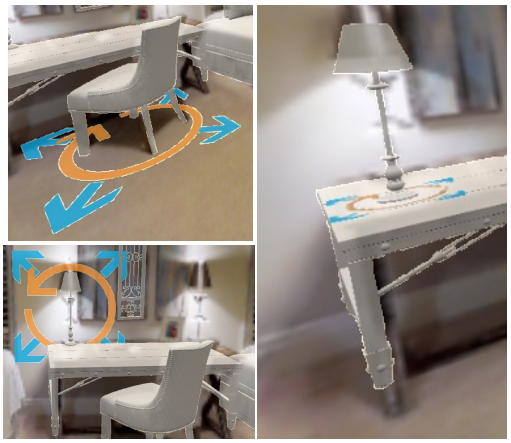


Figure 17. The interface allows annotators to attach objects to either architecture or other objects. The manipulator is oriented on the attached surface and allows for rotating the object whereas the arrows allow for scaling.

boundaries and the placement of doors and windows on the walls to correct any prominent errors. The projection of door and window annotations onto the walls is often noisy due to open doors, inaccurate windows, and noises in the annotation. We obtain each RGB panorama by stitching 6 skybox images from the same camera viewpoint. During

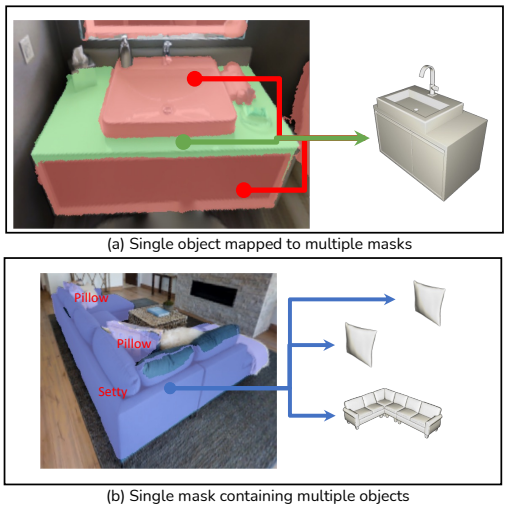
the data preprocessing stage, we also parse panoramas into semantic object instance masks to provide reference objects during annotation. We get these instance masks by rendering segmentations from Matterport3D's annotated object instance house meshes.

Annotation process. We describe a typical annotation workflow starting with an empty scene (see Figure 15). A user freely pans the camera to explore the whole scene while the overlay is kept in sync. After clicking an object to be annotated in the panorama, a list of candidate 3D shapes of the same category is shown in a side panel (see Figure 16). The user is instructed to identify the best matching 3D shape (see Figure 19). The inserted 3D shape is automatically placed at the location in the scene where the user initially clicked. The user can further manipulate the position, scale, and orientation of objects so that the object is aligned to the image (see Figure 17). The placement is attached to a specific surface already in the scene, thus creating a scene support hierarchy by construction.

We recruited annotators and instructed them to follow these guidelines: 1) *Completeness*: each mask should be annotated with a 3D model of an object. Some masks may be divided into parts for different objects and some masks may be merged into one (see detailed discussion of "mask-to-object assignment"). If an important object does not have a mask, it can still be added (see discussion of "custom masks"). 2) *Object match*: the categories, shapes and sizes of the placed objects match those observed (see "object selection" criteria) 3) *Spatial accuracy*: object placements and orientations should be as close to those observed in the panorama (see "object selection" criteria). There should be no collisions or floating objects.

Mask-to-Object Assignment. In some cases, it is overly restrictive to assume that there is a one-to-one correspondence between masks and objects. For example, an object may need to be assigned to multiple masks because the two masks correspond to parts of the same object, separated by occlusion. In other cases, we have masks that include multiple objects (see Figure 18). Our system supports these cases such that a user can place multiple models for the same mask by re-selecting a mask that already has a model assigned and inserting an additional model. For cases where a model is shared among multiple masks, the user first inserts the model having selected one of the masks. Then, the user can assign other relevant masks to the already inserted model. Handling of these cases enables us to correctly annotate densely cluttered arrangements such as kitchen cabinetry, sink units, and pillows on couches.

Object Selection. We decompose the requirement on semantically-matching objects into 4 sub-aspects (see Figure 19): category, shape, structural, and functional similarity. For example, a category mismatch constitutes a 'chair' being annotated with a 'table', a shape mismatch constitutes 'high-back armchair' being annotated with a 'dining-chair' model, a structural mismatch constitutes a 'single-seater chair' being annotated with a 'double-seater chair' and a functional mismatch constitutes a 'an armchair with no wheels' being annotated with a 'swivel chair with wheels and no arms' (Figure 19). We exclude door and window objects for annotations since they are represented as holes on the walls of the architecture and their placement can be largely automated.



(a) Single object mapped to multiple masks

(b) Single mask containing multiple objects

Figure 18. **Mask-to-object assignment.** Our annotation strategy allows for objects that need to be assigned to multiple instance masks (e.g., sink at top), and multiple masks needing to be assigned to the same object (e.g., couch and pillows at bottom).

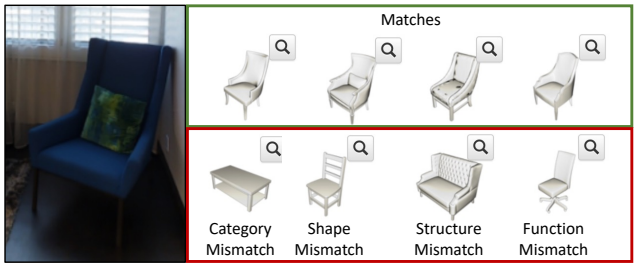


Figure 19. **Object selection in R3DS.** Annotators are instructed to select objects to insert in to a scene based on how well they match with the object observed in the panoramic image. Top: shows good object matches that an annotator would select following our instructions. Bottom: shows different types of mismatching objects that annotators are instructed to avoid.

Object alignment and support. Additionally, the objects can have two types of support structure: i) object-to-object support; and ii) object-to-architecture support. Object-to-object support ensures that two objects are supported by each other properly. For example, a microwave placed on a counter is by construction constrained to be on the counter

Figure 20. **Object alignment in R3DS.** (Left) Objects are closely aligned to the image. (Middle) Objects are properly supported by other objects. (Right) Objects are supported by appropriate architecture elements. In each column, the top is a cropped view from the panorama, the middle highlighted in green is a correctly placed object, while the bottom is an example of an error that annotators avoid using our annotation interface.

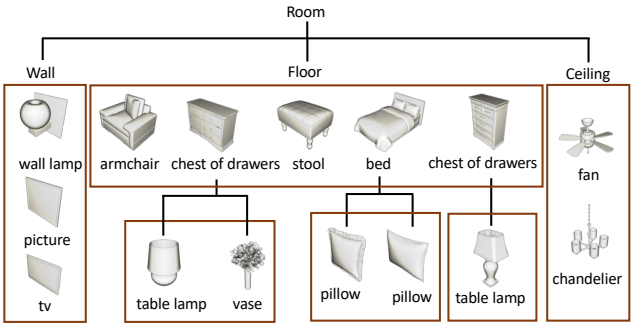


Figure 21. **Scene object support hierarchy.** Objects in a scene are supported by either architecture elements or other objects.

top, and not to float in midair. Similarly, in the object-to-architecture support case, an object placed on an architectural element (floor, wall or ceiling) is ensured to be supported by the planar surface of that element. This type of annotation also helps to disambiguate some otherwise physically implausible scenarios. For example, a chest of drawers is typically supported by the floor, and not by the adjacent wall (Figure 20). In Figure 10, we show a concrete example of how different objects are attached to architectural elements and supported by other objects.

Custom Masks. We further allow annotators to insert objects for which there are no existing instance masks to ensure the scenes are densely populated and objects are properly supported. In some cases, the user may decide to leave a mask unannotated. This could be because the mask is invalid or there are no viable models for the object. In this case, the user can mark the object as 'unannotated' and leave comments explaining the reason.

E. PanoSun task

Method. DeepPanoContext (DPC) [25] predicts the room layout, detects objects in 3D and recovers object meshes from a panorama image using a relation-based graph convolutional network and a differentiable relation optimization procedure. Since DPC has a publicly-available implementation, we use it to benchmark the R3DS data on the PanoSun task. We keep all hyperparameters unchanged except lowering the relation optimization loss weight of 3D bounding box back-projection from 10 to 1, since the ground truth 2D masks are noisy.

Datasets. We train and evaluate DPC on the iGibson-DPC (IG) [6, 16, 25], Structured3D (S3D) [27], and R3DS datasets. Zhang et al. [25] render 1,500 panoramas from 15 iGibson houses composed of 500+ objects spanning 57 object categories. We use the same data and splits for IG. Structured3D consists of 3500 houses and around 18K photo-realistic rendered panoramas in total. We use 14K for training and the remaining 4K for testing. Note that Structured3D does not provide ground truth object meshes.

To prepare R3DS for this task, we generate the ground truth room layout from the 3D architecture based on the camera viewpoint and obtain 3D oriented bounding boxes (OBBs) from all objects. We use 2D object masks from the Matterport3D mesh instance segmentation. We consider three variants of R3DS based on the input panorama: *R3DS-real* where we use the Matterport3D panoramas, *R3DS-syn* where we use rendered panoramas (at the same camera poses) from the annotated synthetic scenes, and *R3DS-mix* where we combine the two types of panoramas and double the available data. We follow the MP3D house split and merge the train and val sets to obtain a disjoint split of 15 train and 5 test houses. Based on the split, we have 696 annotated panoramas for train and 146 for test. To fairly evaluate methods trained on different datasets, we curate a list of 25 object classes common to all datasets.

Metrics. Following Zhang et al. [25] we use separate metrics for room layout estimation, 3D object detection, and scene relation prediction. For room layout estimation, we use 2D IoU for predicted 2D floorplan, 3D IoU for lifted 3D room geometry, and dRMSE for predicted depth with respect to the camera location. For 3D object detection, we report bounding box-based class-agnostic 3D IoU as well as mean average precision (mAP) across the 25 object classes, where an IoU greater than 0.15 counts as a "true" result. For scene relation prediction, we report F1 scores for relation classification. We also report the average number of objects colliding with each other or with architectural structures (wall, floor, ceiling). Specifically, we follow Zhang et al. [25] and measure collisions using the Separating Axis Theorem (SAT) to test whether object bounding boxes over-

Train	2D IoU ↑	3D IoU ↑	dRMSE ↓
DPC [25]	53.4	50.3	0.682
R3DS-real	55.1	53.1	0.610
R3DS-syn	59.0	56.1	0.629
R3DS-mix	**59.6**	**57.0**	**0.572**

Table 1. **Room layout estimation on *R3DS-real* test set.** DPC [25] was pretrained on IG and S3D. For the last three rows, we fine-tune the pretrained weights on variants of R3DS.

lap. Since bounding box-based collision is a poor proxy for real-world physical collision, we also compute mesh-based collision by checking if the meshes for object pairs have any interpenetrating triangles [10, 22].

F. Additional Results

Room layout estimation. For room layout estimation, DPC uses HorizonNet [19] pretrained on iGibson (IG) and Structured3D (S3D) panoramas. This model achieves good performance on IG data (91.0 3D IoU). When directly testing the official pretrained model on *R3DS-real* panoramas, we notice a significant performance drop compared to results on the rendered panoramas from iGibson (Tab. 1 shows that the 3D IoU drops to 50.3). By finetuning the pretrained model with *R3DS-real*, we can predict more precise room layouts for real cluttered scenes. Even only trained on *R3DS-syn*, we outperform the original DPC model by 5.6% and 5.8% on 2D and 3D IoU, respectively. This is likely due to renderings from *R3DS-syn* reflecting more realistic object arrangements in a room instead of pushing all objects against walls. Best performance on 2D IoU (59.6), 3D IoU (57.0) and depth RMSE (0.572) is achieved by fine tuning on *R3DS-mix*.

Error analysis. We conduct error analysis on 120 randomly sampled panoramas using the model pretrained on *S3D* to identify typical errors (see Fig. 23). Errors are categorized into 4 groups: (a) 60% panoramas have 2D perception errors due to the synthetic-to-real appearance gap; (b) 76.7% panoramas show detection failures due to occlusions; (c) 65% panoramas exhibit correct 2D detections but fail to correctly perform 3D predictions; and (d) 15.6% out of 45 panoramas with mirrors mistakenly predict virtual objects in mirrors.

Is relation optimization (RO) effective on R3DS? Test-time relation optimization (RO) was introduced by Zhang et al. [25] to reduce physical violations, floating objects, and misalignment between objects and architecture. The original cost function based on bounding box collisions succeeds in optimizing object poses, since there are few such collisions in the IG data originally used for evaluation (see in Tab. 2). However, the same data assumption does not hold for R3DS, which has more bounding-box-based collisions but nearly zero mesh-based collisions. We ablate the design of RO on R3DS-real in Table 3. By removing two optimization terms (bounding-box-based object-wise collision and touching step-by-step), the model outperforms the original one in 3D mAP (+1.4) but degrades in mesh-based and box-based collisions. We show that using mesh-based collision optimization leads to the best performance. The increase in collisions is unsurprising as the R3DS data reflects more cluttered real interiors.

Datasets	Mesh Collisions	Box Collisions			
		obj-obj	obj-wall	obj-floor	obj-ceil
IG	-	1.185	0.075	0.000	0.790
R3DS	0.0006	2.823	0.388	0.035	0.064

Table 2. **Comparison of the average number of bounding box and mesh-based object collisions per scene in IG and R3DS.** R3DS exhibits more bounding box-based collisions, but almost none of these are actual physical collisions between object meshes. Measuring collisions between bounding boxes is a poor collision measure for fully-populated, real-world scenes.

Rel. Opt.	3D mAP ↑	Mesh Collisions ↓	Box Collisions ↓	
			obj-obj	obj-arch
DPC	18.2	0.158	0.062	1.308
w/o obj col	18.9	1.342	1.130	1.301
w/o obj col+tch	19.6	1.219	1.062	1.295
w/ mesh col	19.7	1.027	0.856	1.394

Table 3. **Ablation of relation optimization (RO) on R3DS-real.** The 2nd and 3rd row remove optimization terms in RO. The last row replaces bounding-box collisions with mesh collisions.

Limitations. Our dataset construction relied on 3D architectures for each Matterport3D scan which are simplifications of the geometry of the real environment. One issue is imperfect wall positions, resulting in objects attached to these virtual walls being offset from the true surface. In addition, objects in our 3D scenes were placed without regard to the materials, meaning that the detailed surface appearance does not match that of the observed object. Future work can investigate transfer of surface appearance to the synthetic objects by projecting textures from the RGB-D data and 3D reconstructed meshes.

References

[1] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X. Chang, and Matthias Nießner. Scan2CAD: Learning CAD model alignment in RGB-D scans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[2] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. In *Proceedings of the Inter-*

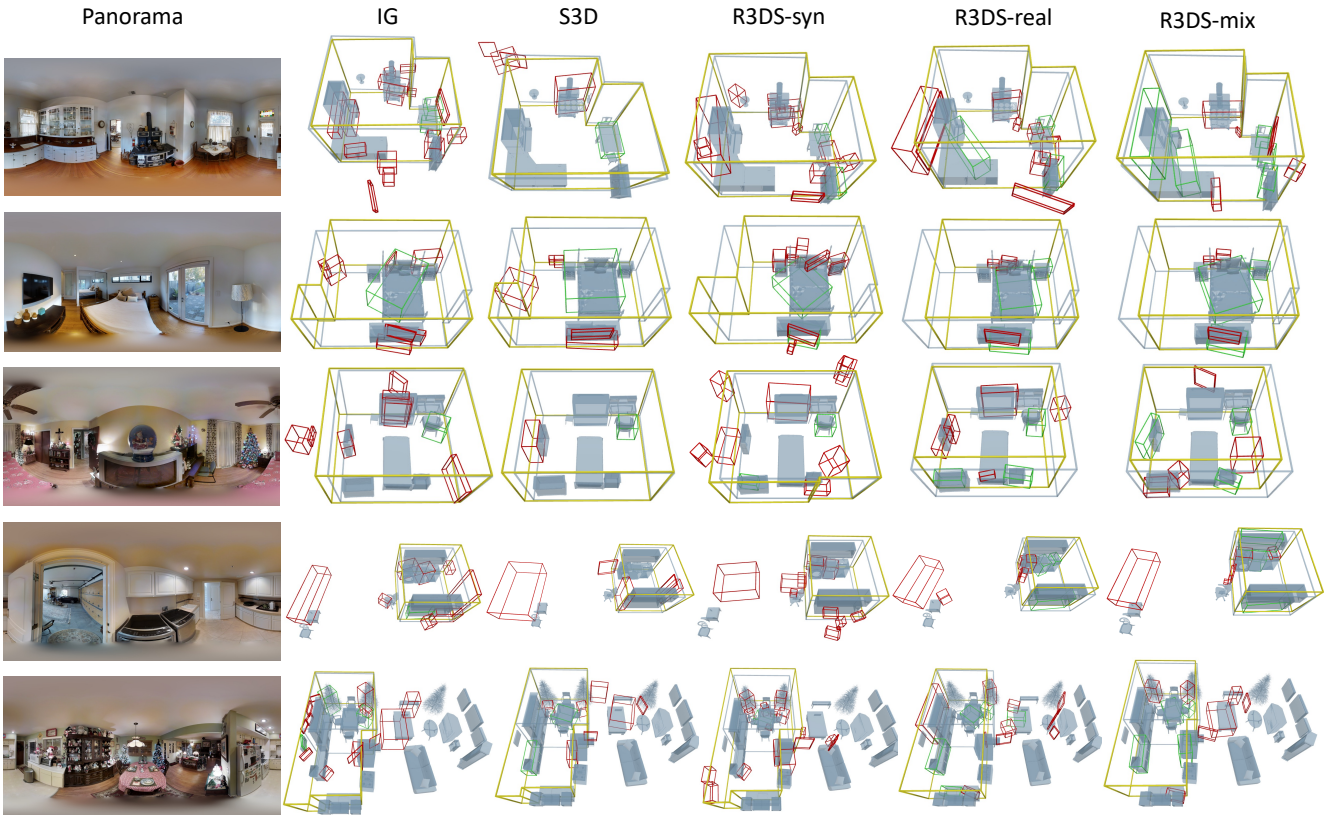


Figure 22. **Qualitative results for cross-dataset Panoramic Scene Understanding task**. Correct and incorrect object detections shown in green and red boxes. Ground truth room layout and meshes shown in gray color, while layout prediction is in yellow. Training on R3DS leads to fewer errors compared to training on other datasets, especially when mixed with real data.

national Conference on 3D Vision (3DV), pages 667–676. IEEE, 2017. 1

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 7

[4] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21126–21136, 2022. 1

[5] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5828–5839, 2017. 1

[6] Yuan Dong, Chuan Fang, Zilong Dong, Liefeng Bo, and Ping Tan. PanoContext-Former: Panoramic total scene understanding with a transformer. *arXiv preprint arXiv:2305.12497*, 2023. 1, 10

[7] Huan Fu, Bowen Cai, Lin Gao, Lingxiao Zhang, Cao Li, Zengqi Xun, Chengyue Sun, Yiyun Fei, Yu Zheng, Ying Li, et al. 3D-FRONT: 3D Furnished Rooms with layOuts and semaNTics. *arXiv preprint arXiv:2011.09127*, 2020. 1

[8] Huan Fu, Rongfei Jia, Lin Gao, Mingming Gong, Binqiang Zhao, Steve Maybank, and Dacheng Tao. 3D-Future: 3D Furniture shape with TextURE. *arXiv preprint arXiv:2009.09633*, 2020. 1

[9] Binh-Son Hua, Quang-Hieu Pham, Duc Thanh Nguyen, Minh-Khoi Tran, Lap-Fai Yu, and Sai-Kit Yeung. SceneNN: A scene meshes dataset with annotations. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 92–101. IEEE, 2016. 1

[10] Tero Karras. Maximizing parallelism in the construction of bvhs, octrees, and k-d trees. In *Proceedings of the Fourth ACM SIGGRAPH / Eurographics Conference on High-Performance Graphics*, pages 33–37. Eurographics Association, 2012. 11

[11] Zhengqin Li, Ting-Wei Yu, Shen Sang, Sarah Wang, Meng Song, Yuhan Liu, Yu-Ying Yeh, Rui Zhu, Nitesh Gundavarapu, Jia Shi, Sai Bi, Zexiang Xu, Hong-Xing Yu, Kalyan Sunkavalli, Milos Hasan, Ravi Ramamoorthi, and Manmohan Chandraker. OpenRooms: An end-to-end open framework for photorealistic indoor scene datasets. In *Pro-*

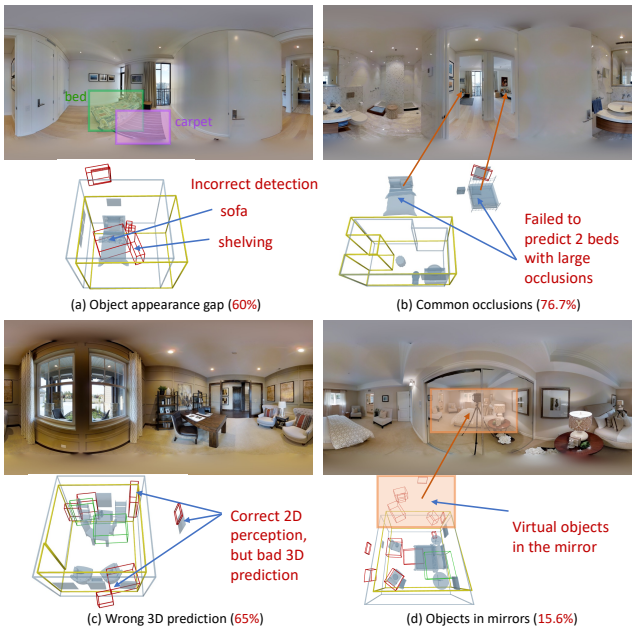


Figure 23. **Error analysis on R3DS panoramas using DPC pre-trained on S3D.** Typical prediction errors are categorized into 4 groups: (a) Appearance gap between real and synthetic images. (b) Detection failure due to common occlusions. (c) Bad performance of monocular 3D prediction. (d) Perception error for objects in mirrors. Numbers in parentheses mean the error rates.

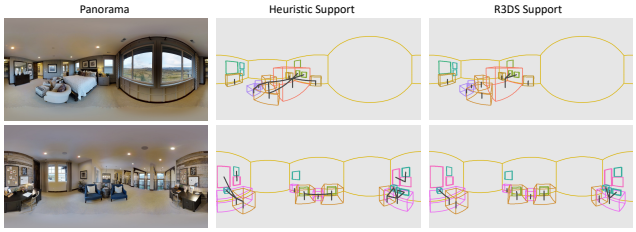


Figure 24. **Comparison between heuristic and R3DS support relations.** Dark lines between objects show support relations. Heuristic support often mistakenly assigns relations to spatially close objects. In contrast, R3DS support relations are annotated.

ceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2021. 1

[12] Joseph J Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing ikea objects: Fine pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2992–2999, 2013. 1

[13] Kevis-Kokitsi Maninis, Stefan Popov, Matthias Nießner, and Vittorio Ferrari. CAD-estate: Large-scale CAD model annotation in RGB videos. *arXiv preprint arXiv:2306.09011*, 2023. 1

[14] Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-Matterport 3D dataset (hm3d): 1000 large-scale 3D environments for embodied AI. *arXiv preprint arXiv:2109.08238*, 2021. 1

[15] Shrenik Sadalgi. Wayfair's 3D Model API. https://www.aboutwayfair.com/tech-innovation/wayfairs-3d-model-api, 2016. [Online; accessed 15-Nov-2023]. 7

[16] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martın-Martın, Linxi Fan, Guanzhi Wang, Shyamal Buch, Claudia D'Arpino, Sanjana Srivastava, Lyne P Tchapmi, Kent Vainio, Li Fei-Fei, and Silvio Savarese. iGibson, a simulation environment for interactive tasks in large realistic scenes. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2021. 10

[17] Bokui Shen, Fei Xia, Chengshu Li, Roberto Martín-Martín, Linxi Fan, Guanzhi Wang, Claudia Pérez-D'Arpino, Shyamal Buch, Sanjana Srivastava, Lyne Tchapmi, et al. iGibson 1.0: A simulation environment for interactive tasks in large realistic scenes. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pages 7520–7527. IEEE, 2021. 1

[18] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 1

[19] Cheng Sun, Chi-Wei Hsiao, Min Sun, and Hwann-Tzong Chen. Horizonnet: Learning room layout with 1d representation and pano stretch data augmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1047–1056, 2019. 11

[20] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3D: Dataset and methods for single-image 3D shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2974–2983, 2018. 1

[21] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in Neural Information Processing Systems*, 34:251–266, 2021. 1

[22] Dimitrios Tzionas, Luca Ballan, Abhilash Srikantha, Pablo Aponte, Marc Pollefeys, and Juergen Gall. Capturing hands in action using discriminative salient points and physics simulation. *International Journal of Computer Vision (IJCV)*, 118(2):172–193, 2016. 11

[23] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. Objectnet3D: A large scale database for 3D object recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 160–176. Springer, 2016. 1

[24] Karmesh Yadav, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Theo Gervet, John Turner, Aaron Gokaslan, Noah Maestre, Angel Xuan Chang, Dhruv Batra, Manolis Savva, et al. Habitat-Matterport 3D semantics dataset. *arXiv preprint arXiv:2210.05633*, 2022. 1

[25] Cheng Zhang, Zhaopeng Cui, Cai Chen, Shuaicheng Liu, Bing Zeng, Hujun Bao, and Yinda Zhang. DeepPanoContext: Panoramic 3D scene understanding with holistic scene context graph and relation-based optimization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 12632–12641, 2021. 1, 10, 11

[26] Yinda Zhang, Shuran Song, Ping Tan, and Jianxiong Xiao. PanoContext: A whole-room 3D context model for panoramic scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 668–686. Springer, 2014. 1

[27] Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3D: A large photo-realistic dataset for structured 3D modeling. *arXiv preprint arXiv:1908.00222*, 2019. 10