# OpenScene: 3D Scene Understanding with Open Vocabularies —Supplementary Material—

Songyou Peng[1,2,3] Kyle Genova[1] Chiyu "Max" Jiang[4] Andrea Tagliasacchi[1,5]
Marc Pollefeys[2] Thomas Funkhouser[1]

[1] Google Research [2] ETH Zurich [3] MPI for Intelligent Systems, Tübingen [4] Waymo LLC [5] Simon Fraser University

pengsongyou.github.io/openscene

In this **supplementary document**, we first provide implementation details in Sec. 1. Next, we supply additional investigation of our methods in Sec. 2. Full results of open-vocabulary object retrieval experiments are shown in Sec. 3. More results of open-vocabulary scene exploration results can be found in Sec. 4.

## 1. Implementation Details

**3D Distillation.** We implement our pipeline in PyTorch [10]. To distill $\mathcal{E}^{3D}$, we use Adam [6] as the optimizer with an initial learning rate of $1e{-}4$ and train for 100 epochs. For MinkowskiNet we use a voxel size of 2cm for ScanNet and Matterport3D experiments, and 5cm for nuScenes. For indoor datasets, we input all points of a scene to the 3D backbone to have the full contexts, but for the distillation loss (Eq. 2) in the paper we only supervise with 20K uniformly sampled point features at every iteration due to the memory constraints. For nuScenes, we input all Lidar points within the half-second segments, and only train with point features at the last time stamp. We use a batch size of 8 for ScanNet and Matterport3D with a single NVIDIA A100 (40G). For nuScenes, we use a batch size of 16 with 4 A100 GPUs. It takes around 24 hours to train, and 0.1 seconds for inference. Moreover, for all dataset we only take in the 3D point position as input to the MinkowskiNet during distillation.

**More Details of Feature Fusion.** For Matterport3D and nuScenes, we use all images of each scene for fusion, while for ScanNet, we sample 1 out of every 20 video frames.

As for the occlusion test, for dataset like ScanNet and Matterport3D where the depth map is provided for each RGB image, we do occlusion test to guarantee that a pixel is only paired with a *visible* surface point. For every surface point, we first find its corresponding pixel in an image, and we can obtain the distance between that pixel and 3D point. The 3D points and pixel are only paired when the difference between the distance and the depth value of that pixel is smaller than a threshold $\sigma$. The threshold $\sigma$ is proportional to the depth value $D$. We use $\sigma = 0.2D$ for ScanNet due to the highly noisy depths and $\sigma = 0.02D$ for Matterport. For pixels with "invalid" regions of the depth map, we do not project their features to 3D points.

For nuScenes Lidar points, since no depth images are provided, no occlusion test is conducted, and we only use the synchronized images and the corresponding Lidar points on the last timestamp of a 0.5 second segment.

**Our nuScenes Evaluation.** Unlike ScanNet and Matterport where each 3D surface point usually having multiple corresponding images, in nuScenes most Lidar points only have one corresponding view, maximum two views at the same time stamp. Therefore, we directly project the single pixel feature to most Lidar points, and use average pooling when there are 2 views. There are 16 classes in the nuScenes lidarseg benchmark, and some of these class names are ambiguous. Since our method can take in arbitrary text prompts, we can pre-define some non-ambiguous classes names for each class, and then map the predictions from these non-ambiguous classes back to the 16 classes. The pre-defined classes names are listed in Table 1.

**MSeg [7] Voting.** MSeg supports a unified taxonomy of 194 classes. We use their official image semantic segmentation code[1] and their pretrained MSeg-3m-1080p model. MSeg already provided the mapping from some of 194 classes to 20 ScanNet classes, so we directly use the mapping. For Matterport3D, we simply add the mapping from "ceiling" in the MSeg labelset. As for nuScenes, we manually define the mapping from MSeg to nuScenes 16 labelsets. However, for the "construction vehicles", "traffic cone", and 'other flat', there is no mapping at all, so we set them to unknown.

As for the majority voting for MSeg multi-view predictions, what we do is the following. Given a surface point and its corresponding multi-view MSeg semantic segmen-

[1] https://github.com/mseg-dataset/mseg-semantic

| nuScenes 16 labels | Our pre-defined labels |
|---|---|
| barrier | barrier, barricade |
| bicycle | bicycle |
| bus | bus |
| car | car |
| construction vehicle | bulldozer, excavator, concrete mixer, crane, dump truck |
| motorcycle | motorcycle |
| pedestrian | pedestrian, person |
| traffic cone | traffic cone |
| trailer | trailer, semi trailer, cargo container, shipping container, freight container |
| truck | truck |
| driveable surface | road |
| other flat | curb, traffic island, traffic median |
| sidewalk | sidewalk |
| terrain | grass, grassland, lawn, meadow, turf, sod |
| manmade | building, wall, pole, awning |
| vegetation | tree, trunk, tree trunk, bush, shrub, plant, flower, woods |

Table 1. **Label Mappings for nuScenes 16 Classes.** Here we list the total 43 pre-defined non-ambiguous class names corresponding to the 16 nuScenes classes.

| | 3D-only model | | 2D-3D ensemble model | |
|---|---|---|---|---|
| Evaluation Datasets | ScanNet20 | Matterport40 | ScanNet20 | Matterport40 |
| | mIoU (%) | mAcc (%) | mIoU (%) | mAcc (%) |
| Distill w/ ScanNet images | 46.1 | 37.6 | 47.7 | 46.4 |
| Distill w/ Matterport images | 38.0 | 47.1 | 47.1 | 50.9 |

Table 2. **Domain Transfer with Open Vocabularies.**. These results show that it is possible to apply our models trained on ScanNet [2] to a novel 3D semantic segmentation task with a different labelset in Matterport3D [1], and vice versa. Since our trained models are task-agnostic (they predict only CLIP features), they can be applied to arbitrary label sets without retraining.

tation, we take the class in the majority of views as the voting results for this point. If there are two classes and only two views, we directly flip the coin to decide point labels.

**Simple Prompt Engineering.** Given a set of text prompts, we use a simple prompt engineering trick before extract CLIP text features. For each object class "XX" (except for "other") we modify the text prompts to "a XX in a scene", for instance "a chair in a scene". With such a simple modification, we observe +2.3 mIoU performance boost with our LSeg ensemble model for ScanNet evaluation. We apply the trick for all our benchmark comparison experiments.

## 2. Additional Analysis

**Can we transfer to another dataset with different labelsets?** Here we investigate the ability of our trained models to handle domain transfer between 3D segmentation benchmarks with different labelsets. We train on one dataset (e.g., ScanNet20) and then test on another (e.g., Matterport40) without any retraining (Table 2). Since our trained model is task agnostic (it predicts only CLIP features), it does not over-fit to the classes of the training set, and thus can transfer to other datasets with different classes directly. Doing the same using a fully-supervised approach would require a sophisticated domain-transfer algorithm (e.g., [3]).

| | Random | Median | Mean |
|---|---|---|---|
| mIoU | 38.2 | 40.1 | **41.4** |
| mAcc | 60.1 | 62.2 | **63.6** |

Table 3. **Ablation on Multi-view Fusion Strategy.** We report mIoU and mAcc on ScanNet [2] with our OpenSeg feature fusion.

**Ablation on multi-view fusion strategy.** We ablate different multi-view feature fusion strategies in Table 3. Random means that having multiple features corresponding to one surface point, we randomly assign one feature to the 3D point. For Median, we take the feature that has the smallest Euclidean distance in the feature space to all other features. As can be seen, the simple average pooling yields the best results, and we use it for all our experiments.

**Visualization of our 2D-3D ensemble model.** In Fig. 1 we study our ensemble model on how to select 2D and 3D features for prediction on a Matterport3D house based on different labelsets. First, we can notice that our ensemble model uses 3D features for those large areas like floors and walls, while 2D features are preferrable for smaller objects. Second, when comparing the feature selections using 21 and 160 classes, we can see that when the number of classes increases, our ensemble model selects more 2D features for the segmentation. The possible reason is that 2D image features can better understand those fine-grained concept than purely from 3D point clouds. For example, on the bottom-right there is a pool table there. When using 21 class labels, it is segmented as a table, so 3D features are preferrable. When using 160 class labels for 2D-3D ensemble, it is much easier to understand the concept of "pool table" using 2D images than 3D point clouds.

**Definition of Zero-Shot Learning (ZSL).** The terminology for ZSL is ambiguous in the literature. In a theoretical ZSL system, there should be no training data of any kind from seen classes. However, almost all real-world ZSL systems utilize general-purpose feature extractors pretrained for proxy tasks on large datasets. For example, 3DGenZ [9] proposes a ZSL variant utilizing image features pretrained on ImageNet (see section 4.5 in their paper), while OpenSeg [4], LSeg [8], CLIP [11], and ALIGN [5] propose ZSL methods trained on alt-text, as we also do. The authors of those papers all describe their methods as ZSL. We followed the same terminology as the latter one.

## 3. Full Results of Open-vocabulary Object Retrieval

The main paper demonstrates that open-vocabulary search can be used to retrieval 3D points matching specific queries from a database of 3D scenes (Figure 4 and Table 6 of the main paper). This section provides the full set of details and results for that experiment.

To produce these results, our 2D-3D ensemble method was used to produce features for every 3D mesh vertex in the Matterport3D test set, using NY160 as the labelset for ensembling. Then, for each given search query, the vertices were sorted based on their cosine similarities to the CLIP embedding of the text query to produce a ranked retrieval list. To avoid flooding the top matches with nearby vertices, we allow the retrieval list to have at most one match per region (i.e., room, as defined by the Matterport3D dataset).

When selecting the queries to use in the experiment, we limited our selections to raw category names provided with the ground truth of the dataset. This allows us to reason about how many examples exist in the test set so that we can know how many matches to expect. Among those candidate queries, we used two strategies to select ones to test: 1) we chose several of the most specific categories (e.g., "yellow egg-shaped vase") in order to test the method on the most difficult cases, and 2) we chose all raw categories with 15 ground truth examples in the entire Matterport3D dataset that had clear definitions (which excepted "lounge chair," "side table," and "office table") in order to avoid bias in the selection of queries (i.e., no cherry-picking).

Figures 2-5 show the ranked retrieval lists for these queries (with the best match first). The number of expected matches according to the dataset ground truth is listed in parentheses under the query text on the left. The best matching point is highlighted with a wireframe red sphere in each image. Images with green/red borders indicate which matches are correct/incorrect, where we say a match is incorrect if it is lower in the ranked list than the last instance in the ground truth. The images with gray borders are included only to show the behavior of the algorithm for near misses – they are not expected to be matches because their rank is the list is beyond the number of examples labeled in the ground truth.

From these results, we see that the algorithm is able to retrieve very specific objects from the database with great precision. For example, when queried with "yellow egg-shaped vase," its top match is indeed a match (which was not labeled in the ground truth), and the following retrieval results are tan vase, a pumpkin, and a white egg-shaped vase with gold decorations. Similarly, when queried with "teddy bear," it retrieves two teddy bears (neither labeled in the ground truth), a stuffed monkey, and a stuffed lion among the top four matches. Among all the queries in all of the experiments, the only false positive is where a bowl of stones on a toilet is ranked 15th in the retrieval list for "telephone" behind two of the ground truth instances, which are ranked 25th and 29th. In this example, all other 29 of the top 30 matches are correct (20 are shown in Figure 4).

These results suggest that the open-vocabulary features computed with our 2D-3D ensemble algorithm are very effective at retrieving object types with specific names. Further experiments are required to understand the limitations.

## 4. More Results of Open-vocabulary 3D Scene Exploration

The main paper demonstrates that open-vocabulary queries can be used to explore the content of 3D scenes via text queries (Figures 1 and 6 of the main paper). This section provides more examples demonstrating the power of open-vocabulary exploration. Please see the supplemental video for a live demo.

For each query, the user types a text string (e.g., "glass"), it gets encoded with the pre-trained CLIP text encoder, we compute the cosine similarity with features we've computed for every 3D vertex, and then color the vertices by similarity (yellow is high, green is middle, blue is low).

Figures 6-11 show results for a broad range of queries, including ones that describe object categories in Fig. 6, room types in Fig. 7, activities in Fig. 8, colors in Fig. 9, materials in Fig. 10, and abstract concepts in Fig. 11.

Please note the power of using language models learned via CLIP to reason about scene attributes and abstract concepts that would be difficult to label in a supervised setting. For example, searching for "store" highlights 3D points mainly on closets and cabinets (middle-right of Fig. 8), and searching for "cluttered" yields points in a particularly busy closet (top-right of Fig. 11). These examples demonstrate the power of the proposed approach for scene *understanding*, which goes far beyond semantic segmentation.

**Quantitative results on 3DSSG dataset [13].** In order to quantitatively assess our performance in 3D scene exploration, we conduct an experiment on the 3DSSG dataset. 3DSSG is a dataset extended from 3RScan [12] that has annotations in object-level material estimation. In Table 4, we show results of predicting material classes for every 3D point in the 3DSSG test set using variants of our approach trained on ScanNet, and compare them to a fully-supervised MinkowskiNet trained for this task on 3DSSG. The conclusions are similar to the ones made in the paper: 1) 2D-3D ensembling is our best variant, 2) it is worse than a fully-supervised approach for classes with many training examples, and 3) it is better for classes with fewer examples.

| | mIoU | mAcc |
|---|---|---|
| MinkowskiNet (Fully supervised) | **23.5** | 30.6 |
| Ours - 2D Fusion | 18.6 | 31.9 |
| Ours - 3D Model distilled on ScanNet | 15.3 | 26.4 |
| Ours - 2D-3D Ensemble | 20.1 | **35.6** |

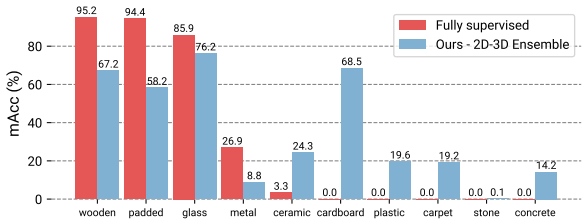



Table 4. **Comparison on 3DSSG [13] in Material Estimation.** We report the average of 10 material classes in test set. Classes are sorted left-to-right in the bar chart by the number of training examples.

## References


[1] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. In *3DV*, 2017. 2

[2] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 2

[3] Hehe Fan, Xiaojun Chang, Wanyue Zhang, Yi Cheng, Ying Sun, and Mohan Kankanhalli. Self-supervised global-local structure modeling for point cloud domain adaptation with reliable voted pseudo labels. In *CVPR*, 2022. 2

[4] Golnaz Ghiasi, Xiuye Gu, Yin Cui, and Tsung-Yi Lin. Open-vocabulary image segmentation. In *ECCV*, 2022. 2

[5] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, 2021. 2

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 1

[7] John Lambert, Zhuang Liu, Ozan Sener, James Hays, and Vladlen Koltun. Mseg: A composite dataset for multi-domain semantic segmentation. In *CVPR*, 2020. 1

[8] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and René Ranftl. Language-driven semantic segmentation. In *ICLR*, 2022. 2

[9] Björn Michele, Alexandre Boulch, Gilles Puy, Maxime Bucher, and Renaud Marlet. Generative zero-shot learning for semantic segmentation of 3d point clouds. In *3DV*, 2021. 2

[10] Adam Paszke et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 1

[11] 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 *ICML*, 2021. 2

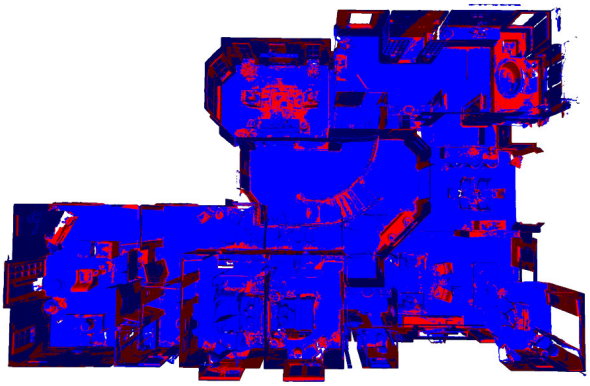
[12] Johanna Wald, Armen Avetisyan, Nassir Navab, Federico Tombari, and Matthias Nießner. Rio: 3d object instance re-localization in changing indoor environments. In *ICCV*, 2019. 3

[13] Johanna Wald, Helisa Dhamo, Nassir Navab, and Federico Tombari. Learning 3d semantic scene graphs from 3d indoor reconstructions. In *CVPR*, 2020. 3, 4

Input 3D Geometry

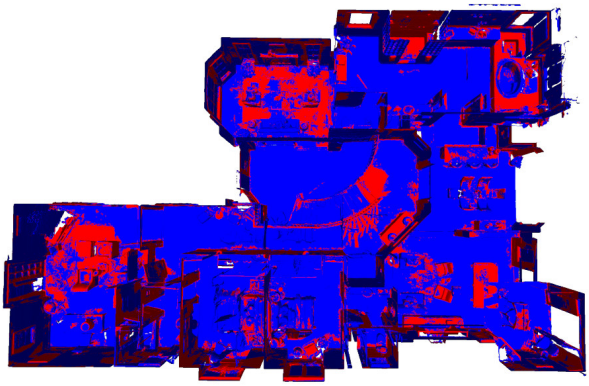

160-class label sets

Our top 21-class semantic segmentation results



2D-3D features selected for 21-class segmentation

Our top 160-class semantic segmentation results



2D-3D features selected for 160-class segmentation

Figure 1. **Study of Our 2D-3D Ensemble Model.** We show semantic segmentation results and the feature selection of our ensemble model on a Matterport3D house. We show the comparison between the 21-class and 160-class prediction. As can be seen, when the number of classes increases, our ensemble model selects more 2D features for the segmentation. The reason can be that, when involving more fine-grained or long-tailed classes, 2D image features can better understand those fine-grained concept than purely from 3D point clouds. Points using 2D features for final segmentation are marked as red, while points with 3D features are marked as blue.

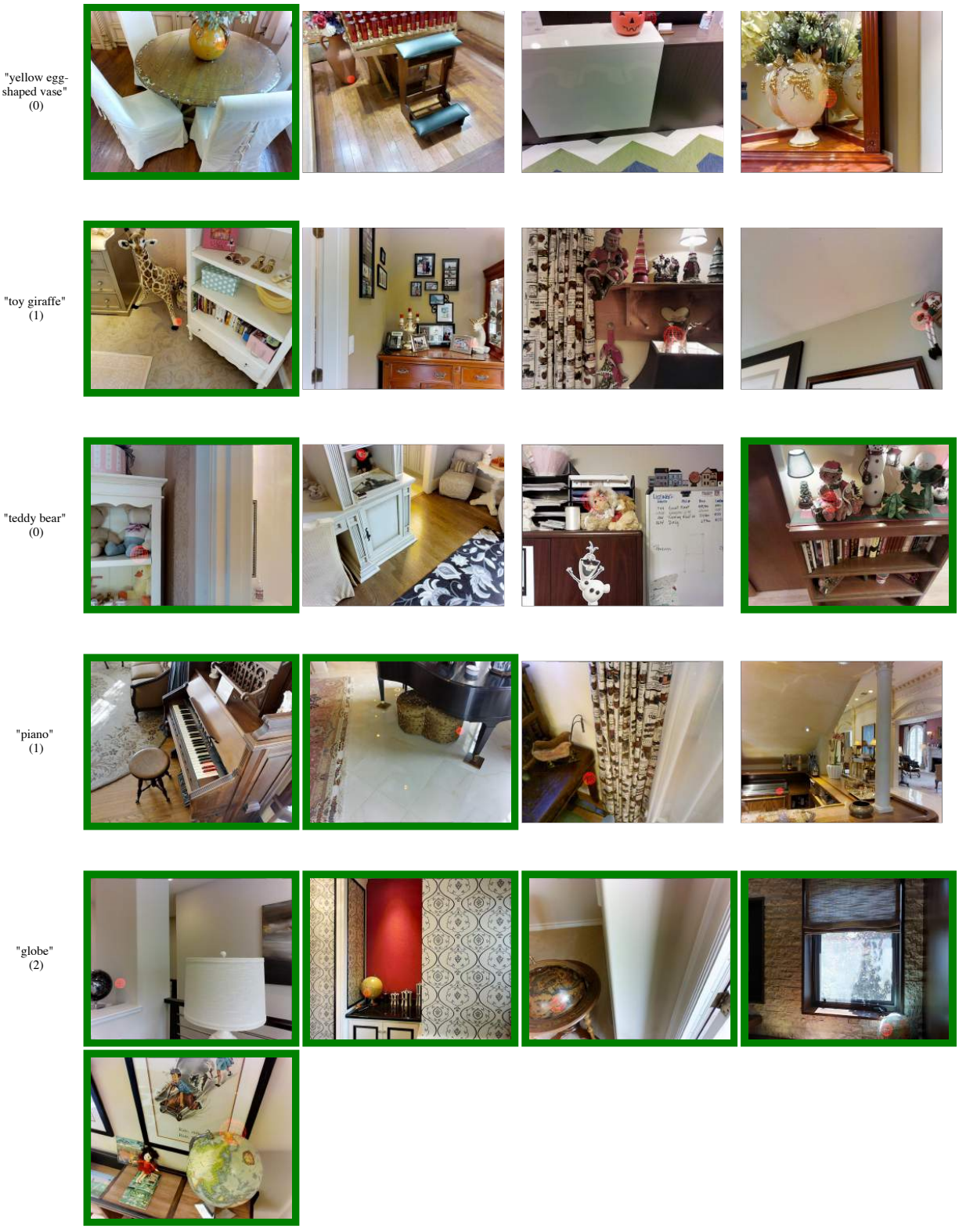



Figure 2. **Example object retrieval results (page 1 of 4)**. The query text is in the left column, with the number of ground truth instances in the Matterport test set listed in parentheses below. The images show top matching 3D points in the Matterport test set ranked from left to right (note the red wireframe sphere around the matching point in each image). Correct matches are marked with green borders. The one incorrect match is marked with a red border (in page 3 of 4). Others marked with gray borders are not wrong (since there are no further objects matching the query according to the ground truth), but are shown as examples of near matches.

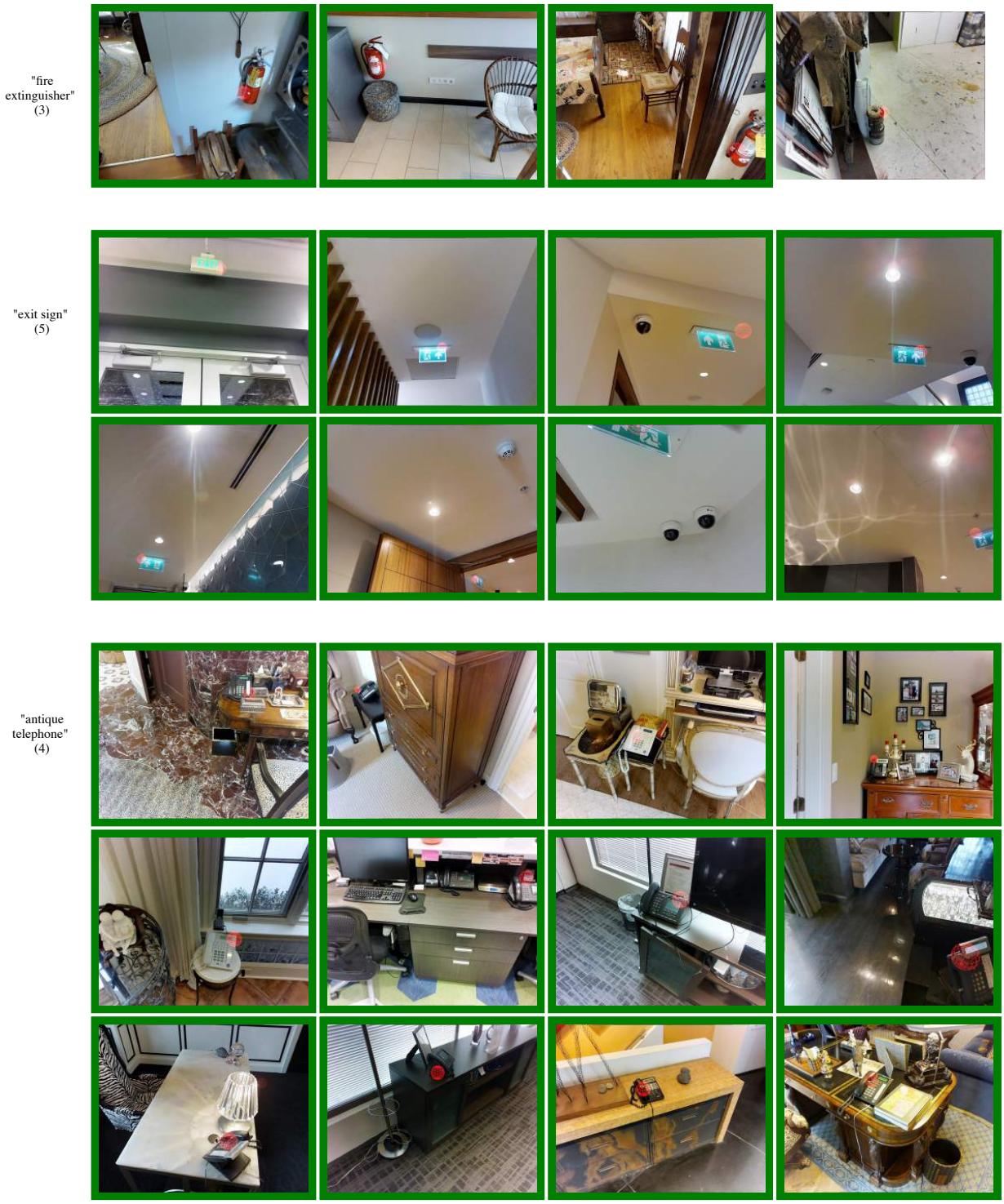



Figure 3. **Example object retrieval results (page 2 of 4)**. See caption of Figure 2 for details.

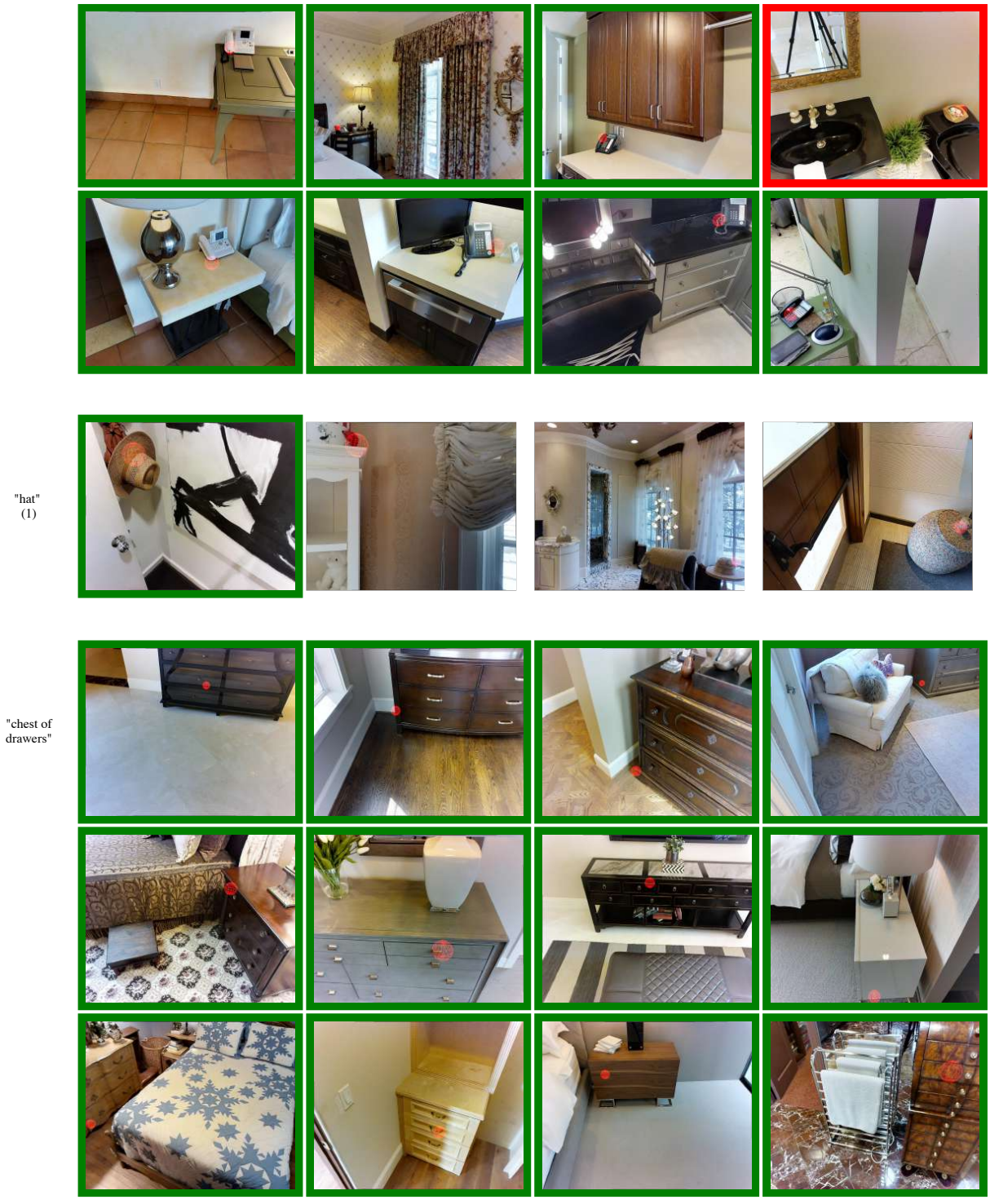



Figure 4. **Example object retrieval results (page 3 of 4)**. See caption of Figure 2 for details.

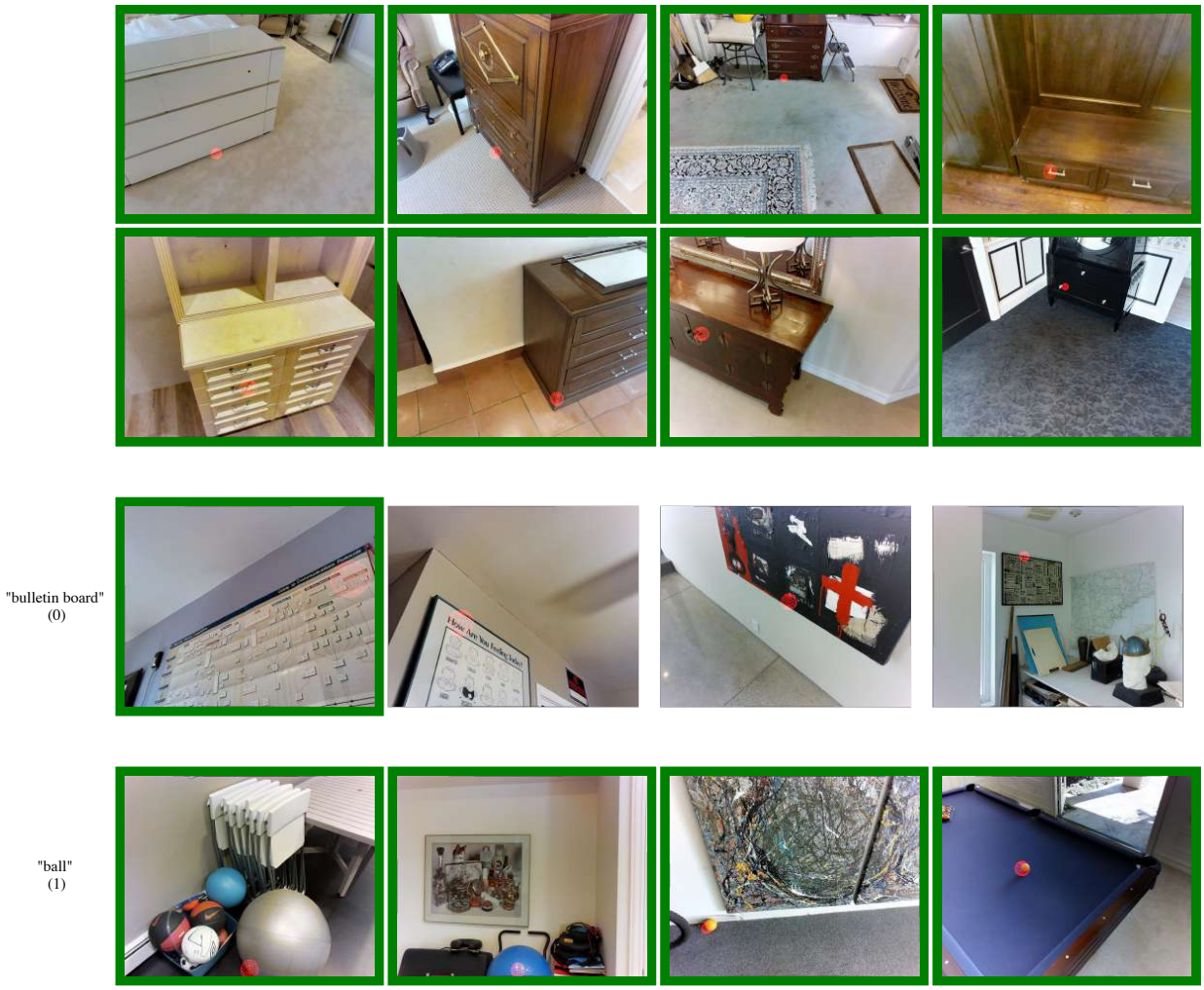



Figure 5. **Example object retrieval results (page 4 of 4)**. See caption of Figure 2 for details.

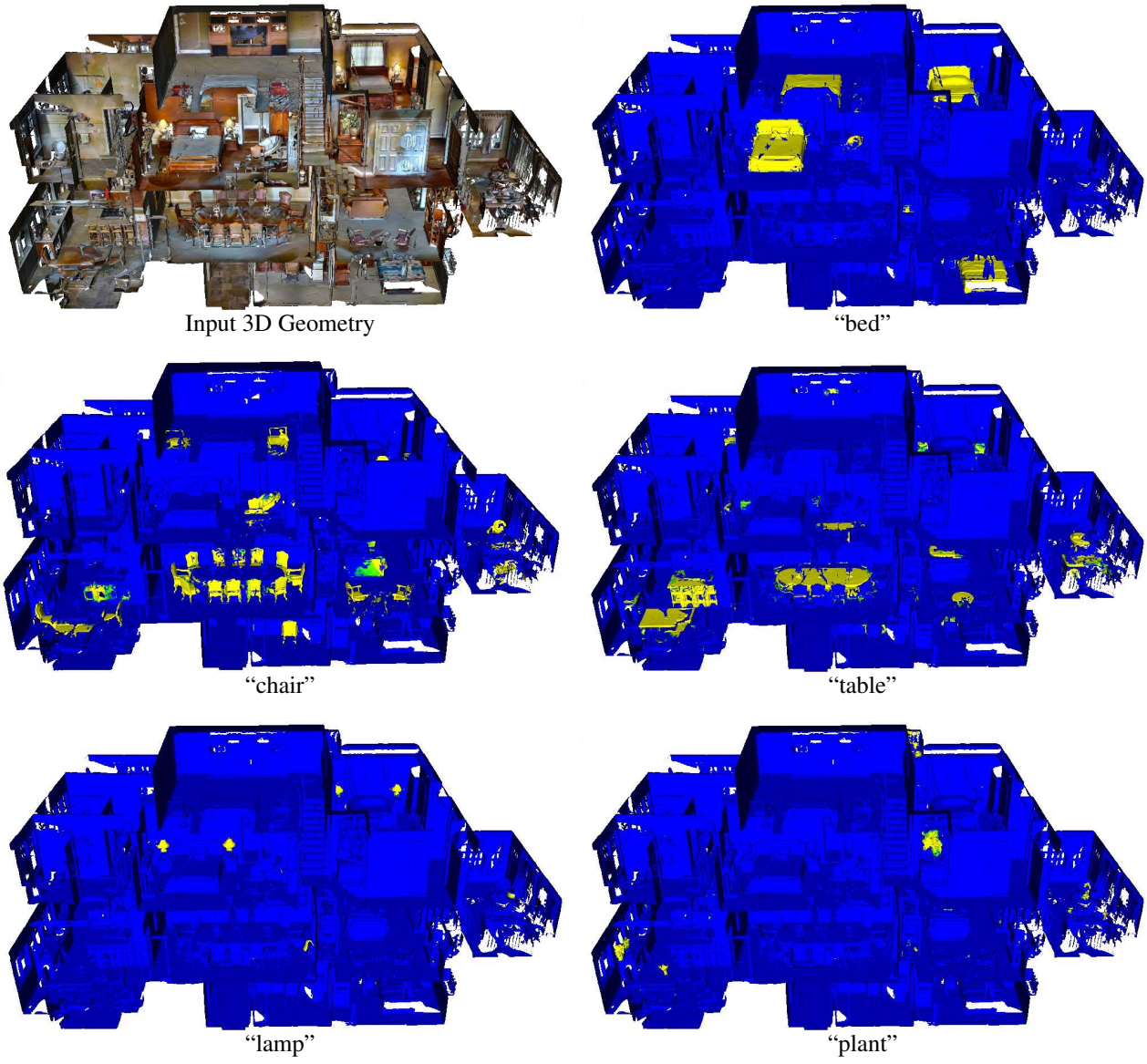
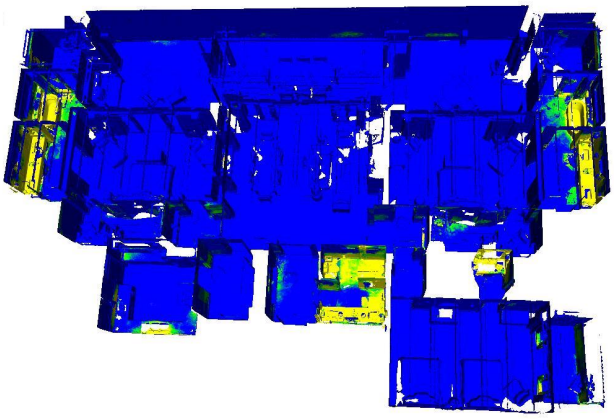
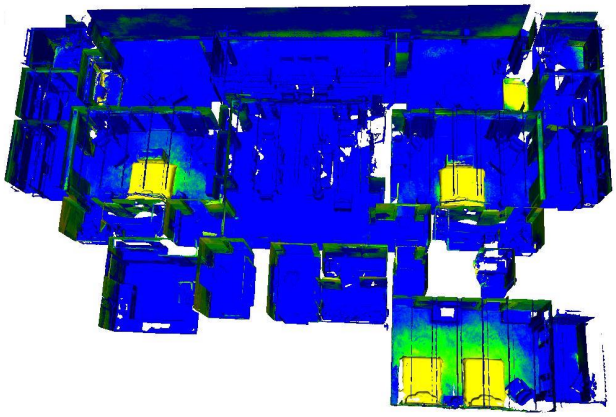



Figure 6. **Open-Vocabulary Queries for Common Object Types.**

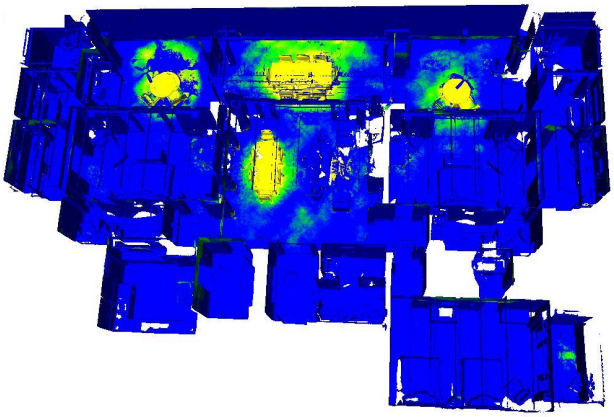
Input 3D Geometry



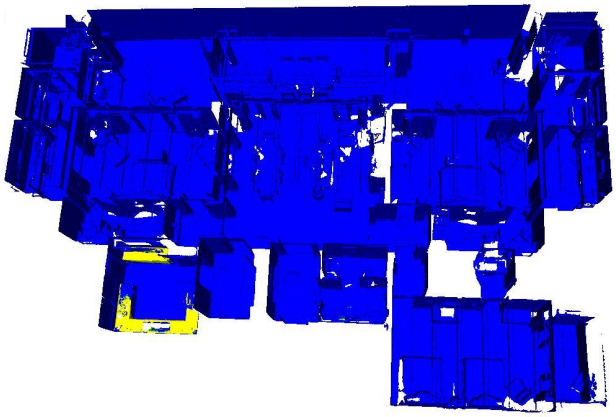
“bathroom”



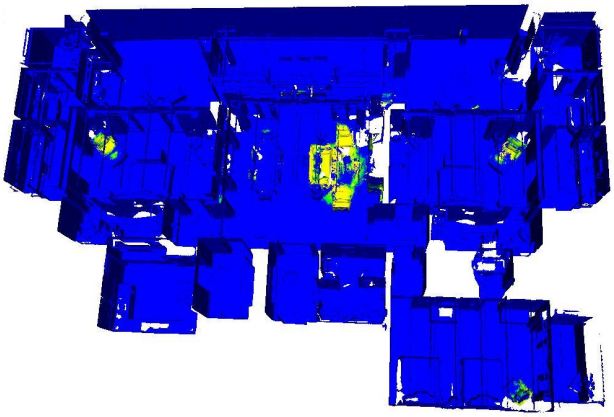
“bedroom”



“dining room”

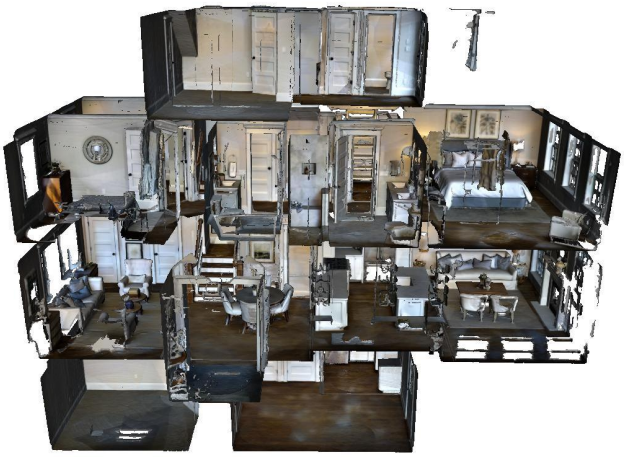


“kitchen”

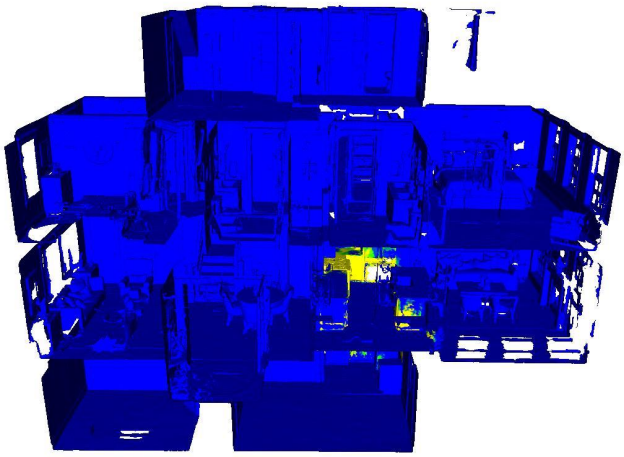


“living room”

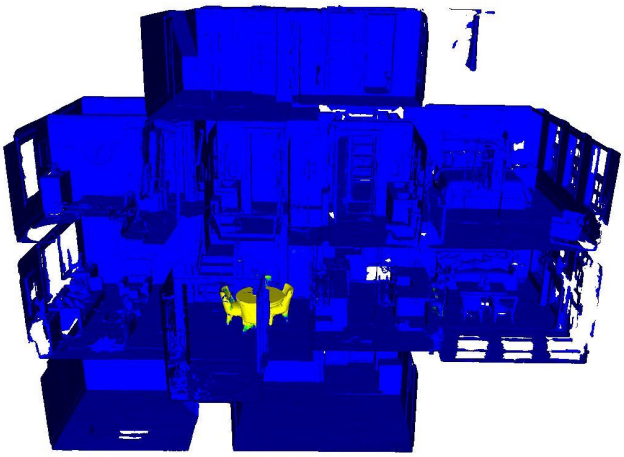
Figure 7. **Open-Vocabulary Queries for Room Types.**



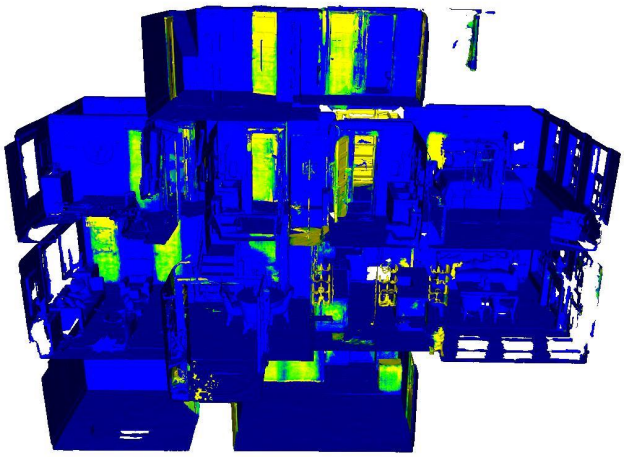
Input 3D Geometry



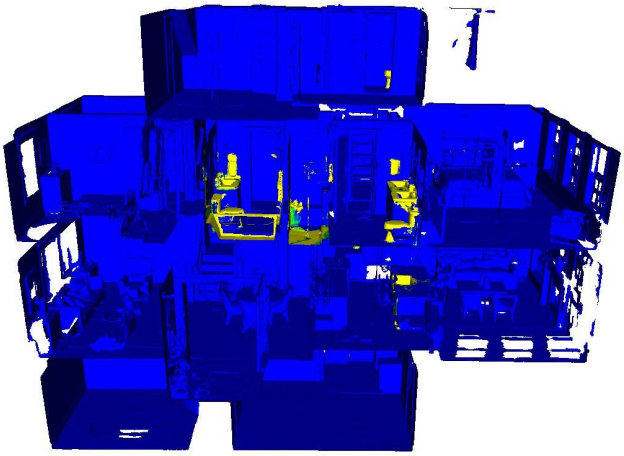
“cook”



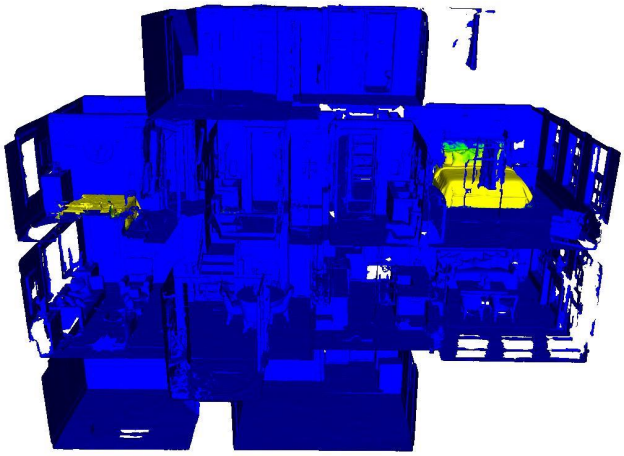
“dine”



“store”



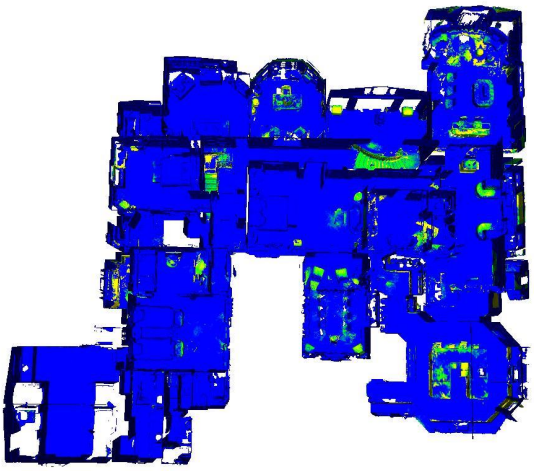
“wash”



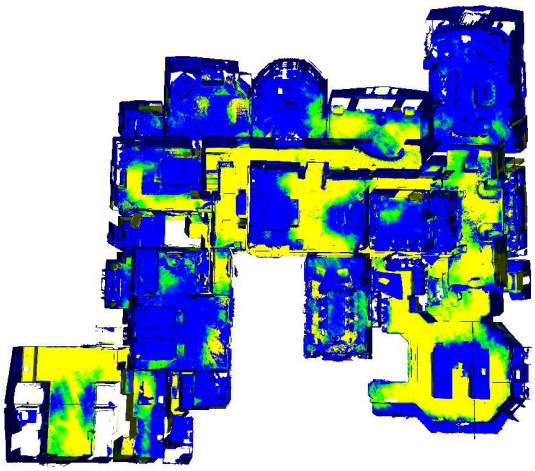
“sleep”

Figure 8. **Open-Vocabulary Queries for Activity Sites.**

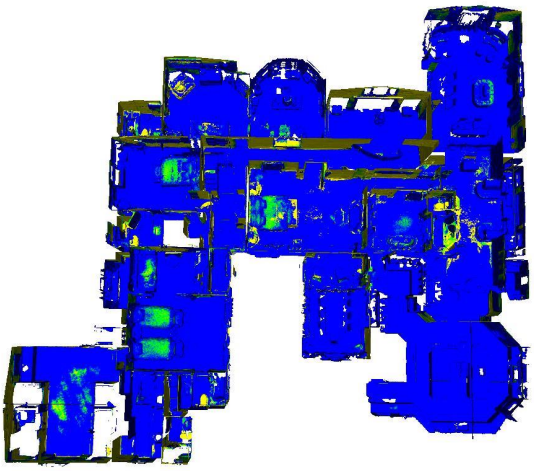
Input 3D Geometry



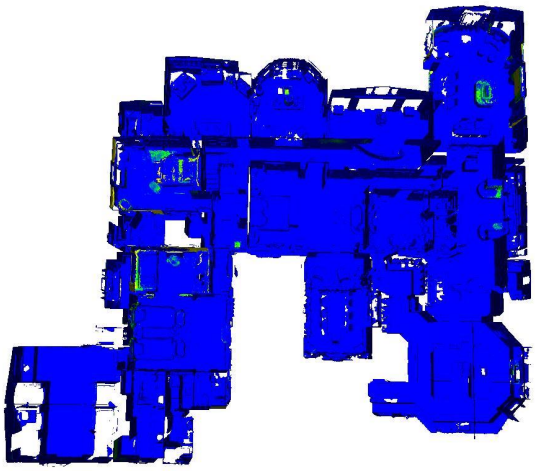
“black”



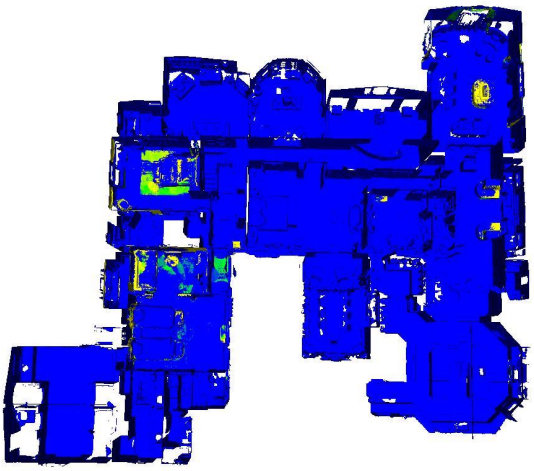
“brown”



“white”



“orange”



“red”

Figure 9. **Open-Vocabulary Queries for Colors.**

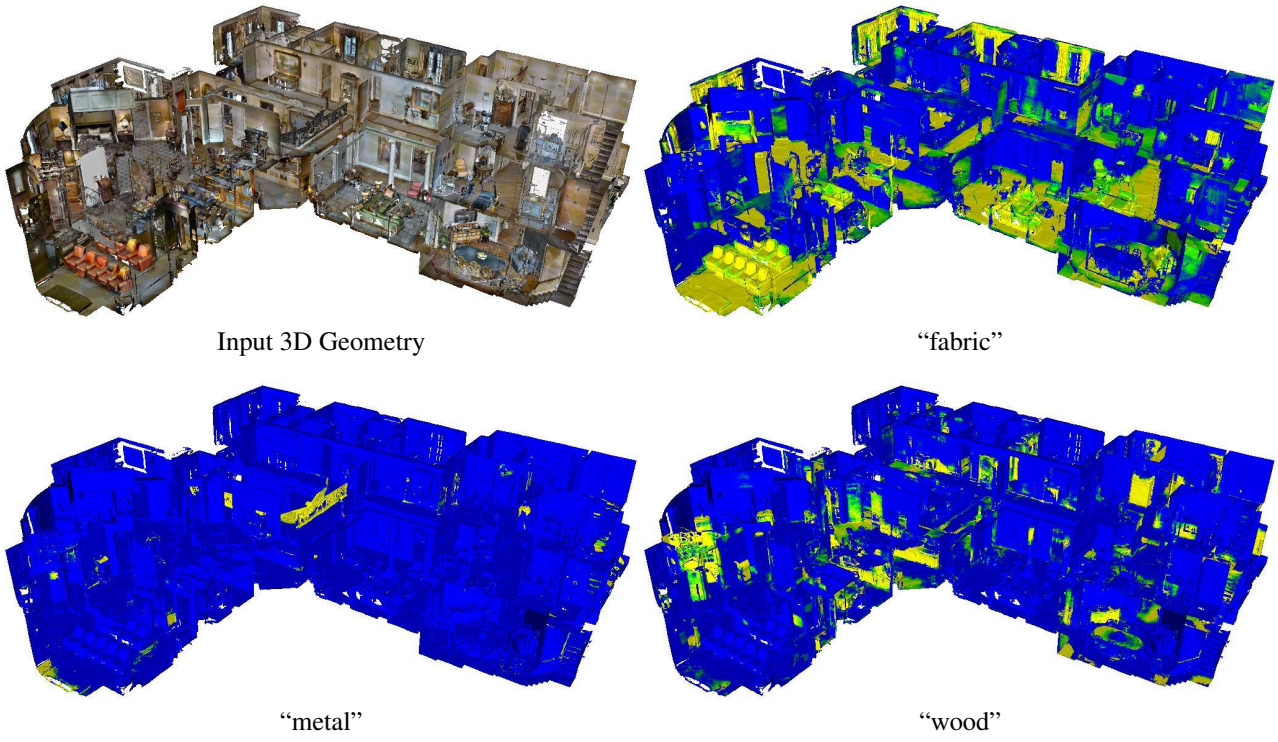



Figure 10. **Open-Vocabulary Queries for Materials.**

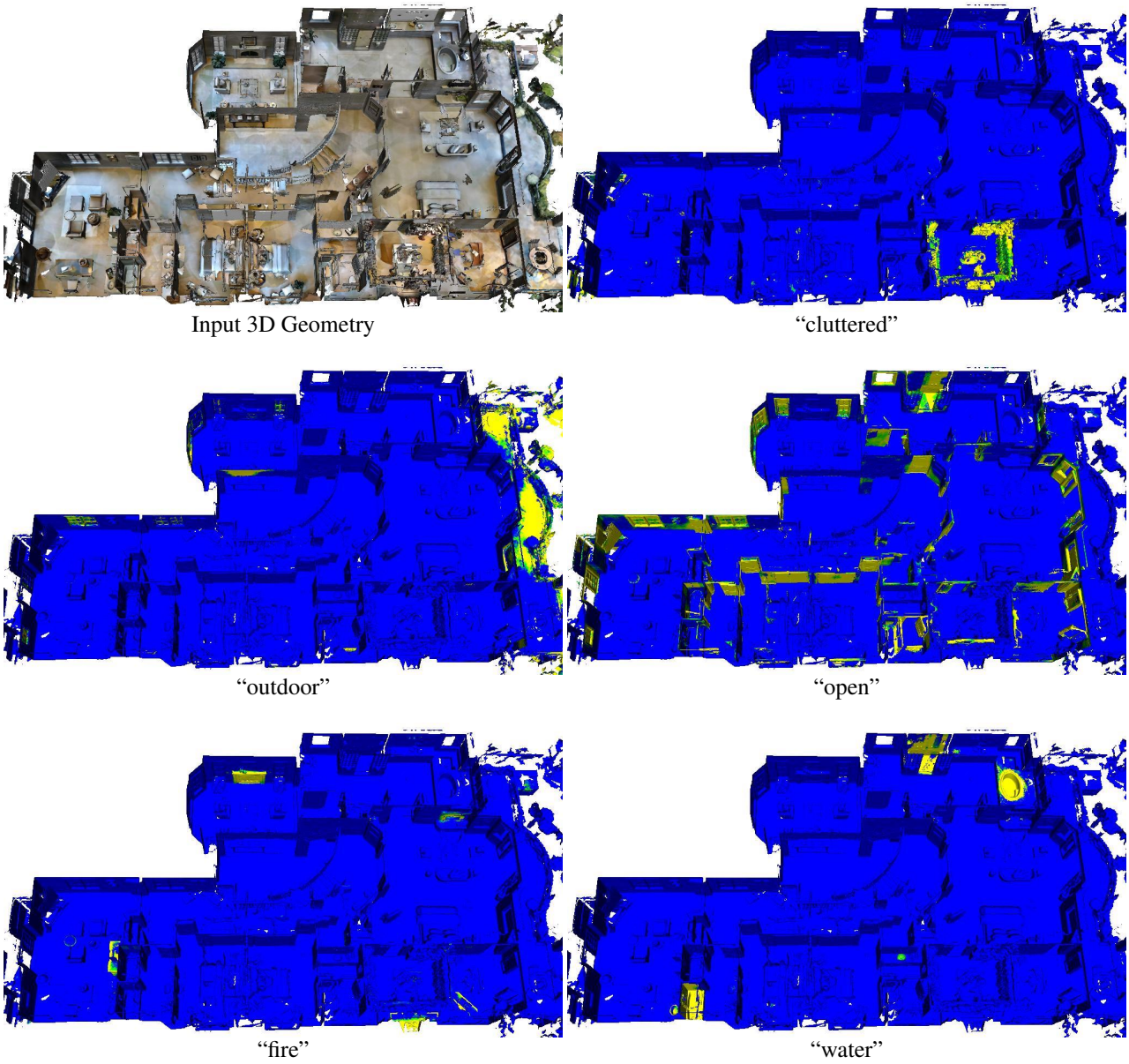



Figure 11. **Open-Vocabulary Queries for Abstract Concepts.**