SUMO: The Scene Understanding and Modeling Challenge from 360° to 3D

http://sumochallenge.org

Lyne P. Tchapmi[1], Daniel Huber[2], Richard Skarbez[3], Ilke Demir[4]
Jimmy Wu[5], Xingyuan Sun[5], Osama Sakhi[6], Zhile Ren[6]
Shuran Song[5], Thomas Funkhouser[5], Silvio Savarese[1], Frank Dellaert[6]

[1]Stanford University, [2]Facebook, [3]Virginia Tech, [4]DeepScale, [5]Princeton University, [6]Georgia Tech

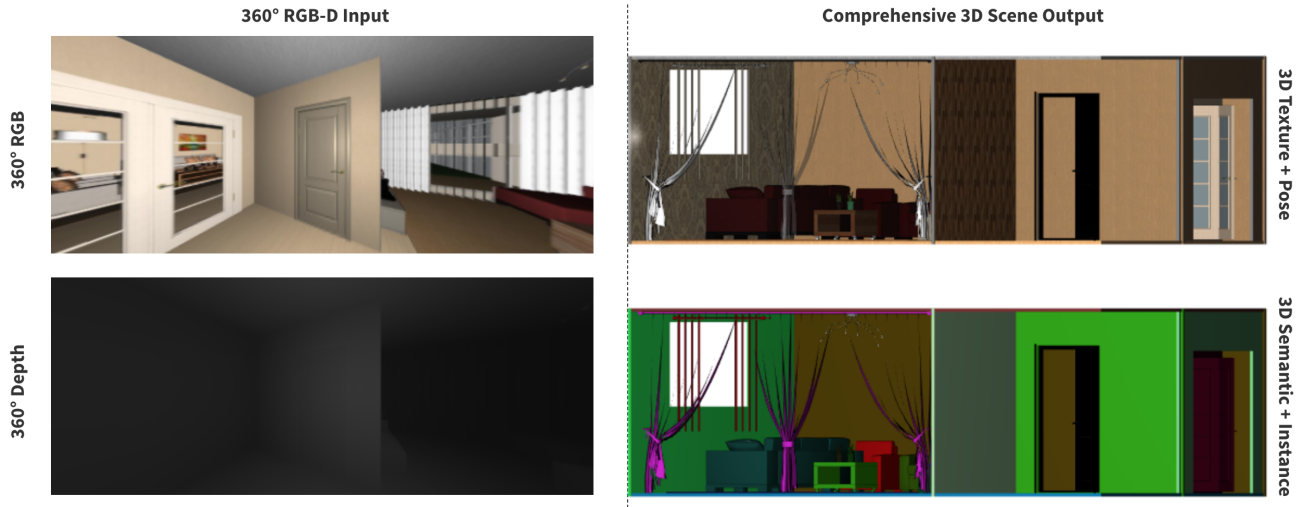

Figure 1: SUMO: 3D **S**cene **U**nderstanding and **MO**deling from 360° panoramas.[The dashed line in this picture is not in the center)]

Abstract

We introduce the Scene Understanding and Modeling (SUMO) challenge with the goal of evaluating the performance of 3D holistic scene understanding algorithms. With the rise of deep learning algorithms in computer vision, the need for comprehensive datasets started to increase as well as the dependency to benchmark results on those datasets. Our challenge enables evaluating scene understanding approaches on a compelling dataset of synthetically generated 360° RGB-D panoramas, with the accurate ground-truth semantic annotations. Challenge participants are asked to derive a complete, instance-level 3D estimation of a scene. Submitted algorithms are evaluated at three levels of complexity corresponding to 3 tracks of the challenge: oriented 3D bounding boxes, oriented 3D voxel grids, and oriented 3D meshes. This paper describes the details of the challenge tasks, including characteristics of scene elements, data format, and evaluation metrics. We also provide baselines for each task as a proxy for the participants.

1. Introduction

Semantic understanding of scenes and recovering three-dimensional structure of the environment are core tasks in computer vision. In recent years, researchers continue to improve the state-of-the-art algorithms for object detection [9], semantic segmentation [8], and image retrieval tasks [citation]. The success of these data-driven approaches is largely due to well-annotated large scale datasets, such as ImageNet [23], DAVIS [16], and multiple tracks of SHREC[1]. Moreover, the developments are also carried over to 3D domain, thanks to comprehensive 3D datasets such as SUNCG [29] and ShapeNet [1]. In those synthetically generated datasets, annotations are accurate, object styles are diverse, and the dataset itself is scalable; facilitating research in 3D vision algorithms.

We explore the less-studied problem of understanding 360° images, which is well-suited to act as a test bed for deep learning algorithms, considering its resemblance to several existing 3D vision tasks. Since panoramas capture the global appearance of the environment, they can poten-

[1]http://www.shrec.net

tially allow computer vision systems to holistically understand the 3D geometry of the scene. Ren: I don't fully understand "The inherent ambiguity in 2D-to-3D generation/-conversion problems can be thought similar to other multi-dimensional problems that learning algorithms are proven to succeed with high accuracy results."

In order to bring more attention to this domain and evaluate algorithms on a fair ground, we introduce The 2019 SUMO Challenge — 360° Indoor **S**cene **U**nderstanding and **MO**deling. The SUMO Challenge (i) released a comprehensive collection of 360° RGB-D imagery with the corresponding 3D scenes, (ii) organized three public competitions structured around scene understanding, in the format of *bounding boxes*, *voxels*, and *meshes*, and (iii) gathered researchers from different fields to collaborate on 3D scene understanding problems in a workshop. The diverse and extensive samples in our dataset enables evaluations for robust and scalable vision algorithms. SUMO competition tracks are defined below.

- **Bounding Box Track:** Localizing objects using 3D bounding boxes in an unstructured scene is an important and challenging task. This track expects participants to propose 3D bounding boxes to localize objects, estimate 6D poses, and classify object categories.

- **Voxel Track:** Representing a scene in the voxel domain is an essential step for several scene understanding and rendering algorithms. Participants in this track will extract voxel-based representations from the RGB-D panoramas, in addition to the bounding box information discussed above.

- **Mesh Track:** Representing detailed 3D environments as meshes is crucial for simulations and mixed reality applications. This is also the ultimate goal of complete scene understanding in our challenge. Our Mesh Track aims to reveal solutions that generate 3D textured meshes of objects from the 360° RGB-D imagery.

The competitions are hosted in the EvalAI platform [34]. Our dataset contains thousands of panoramas as discussed in Section 2. In Section 3, we introduce the data representations and submission formats to introduce the challenge dataset. We then formulate evaluation metrics for each track in Section 4. Finally, we introduce baselines in Section 5 to act as a proxy for future submissions. The results of the competitions will be presented in The 2019 SUMO Workshop in association with 2019 International Conference on Computer Vision and Pattern Recognition Conference in Long Beach, CA on June 17th 2019.

2. Dataset

In this section we describe the dataset used for the SUMO challenge and provide details about data generation and class distribution.

2.1. 360° RGBD Panoramas

The SUMO dataset is derived from the SUNCG[29] dataset, a large-scale synthetic dataset of over 45000 3D scenes with manually created room and furniture layouts. We used [X] houses from the SUNCG dataset, and for each house, we generate 360° panoramas at different rooms. The task we propose is to generate a 3D scene model of a room given a 360° panorama taken in that room. We generate ground-truth for each input panorama by keeping all 3D models in the scene that are visible from a given panorama. We select a set of [1500] input panoramas for the training set and set aside [500] panoramas for testing.

2.2. Dataset Statistics

[how many panoramas per scene? how many objects? how many object categories? how many instances per object? can we add a plot for the last question? how many objects per scene on the average? how are they rendered? what is the resolution? what is the depth accuracy? is there any approximation?] [possibly copy texts from 1-introduction.tex?]

3. Annotations and Submission Format

In the SUMO challenge, 3D scenes are composed by a collection of instances, known as elements hereafter. Each element represents one object in the scene (e.g., a wall, the floor, or a chair) in one of three increasingly descriptive representations: oriented bounding box, oriented voxel grid, or oriented surface mesh. All aspects of a scene are modeled using the same representation. Each representation is evaluated with its own set of metrics described in Section 4.

Each element's geometric description (bounding box, voxel grid, or mesh) is represented in the local coordinate frame. The element's pose is represented by a rigid body transformation (translation t and 3×3 rotation matrix R), which maps points from the element's local coordinate frame to the scene coordinate frame. The ground truth of the training data is provided in the scene coordinate frame. It is right handed with the +Y axis approximately up and its origin coincident with the camera used to obtain the data.

[Add figure showing coordinate frames]

The origin of an element's local coordinate frame is the center of the element's bounding box, and the axes are oriented so that +Y is "up," +Z is the "front," and +X is defined according to a right-handed coordinate system. The direction of "up" and "front" is defined on a per-category basis. (see Appendix A).

Some objects do not have an intuitive "up" or "front" direction. For example, a rectangular table has a 2-fold rotational symmetry about the Y axis. For simplicity, we only support 2-fold (e.g., rectangular table), 4-fold (e.g., square table), circular (e.g., plate), or spherical (e.g., basketball) symmetries and only about the major axes (X, Y, or Z). Furthermore, only a subset of the possible symmetries may be used in conjunction with one another: 2-fold + 2-fold + 2-fold (e.g., rectangular box), 4-fold + 4-fold + 4-fold (e.g., cube).

Mathematically, an element e is a tuple $(c, P, \mathcal{S}, \mathcal{X})$, where c is the category of the element, P is the pose, $\mathcal{S}$ is the shape and appearance, and $\mathcal{X}$ is the symmetry specification. In pseudo-code an element is represented as:

Ren: In all the pseudo-code in this paper, is it more appropriate to add " ; " in each line?

```
SymmetryType =
  {twoFold | fourFold | circular | spherical}

ElementBase {
  string category
  Pose3 pose {
    Rot3 rotation
    Vec3 translation
  }
  Symmetries {
    SymmetryType xSymmetry
    SymmetryType ySymmetry
    SymmetryType zSymmetry
  }
  float detectionScore
}
```

Next, we describe the three formats for shape and appearance representation in submissions for each track.

3.1. Oriented Bounding Boxes

In SUMO challenge, Oriented Bounding Boxes (OBBs) offer the coarsest representation of a scene. An OBB augments the basic element with a bounding box with bounds min_corner, and max_corner. The box is represented in the element's local coordinate frame, not in the scene coordinate frame. In pseudocode,

```
BoundingBox {
  float minCorner[3]
  float maxCorner[3]
}

OrientedBoundingBoxObject {
  ElementBase elementBase
  BoundingBox boundingBox
}
```

Note that the bounding box may extend beyond the observed data corresponding to an object, as would happen, for example, when a chair is partially occluded. The box indicates the extent of the chair if it were fully observed.

3.2. Oriented Voxel Grids

Oriented Voxel Grids provide an intermediate level of description for a scene. Each element includes the base element and oriented bounding box as defined above, along with a voxelized 3D volume with a given voxel size and a matrix of voxel centers with 3D location (x, y, z) and color (RGB). The voxel grid is represented in the element's local coordinate frame. In pseudocode,

```
OrientedVoxelGridObject {
  ElementBase elementBase
  BoundingBox obb
  float voxelSize
  float voxelCenters[N,6]
}
```

3.3. Oriented Surface Meshes

Oriented surface meshes allow for the most precise representation of a scene. Each element is represented as a base element and oriented bounding box as defined above, combined with a textured triangle surface mesh. A mesh is composed of a set of 3D vertices, face indices, UV texture coordinates, and a texture map.

```
OrientedMeshObject {
  ElementBase elementBase
  OrientedBoundingBox obb
  // x,y,z coordinates
  float vertices[3,N]
  // indices of triangle corners
  float indices[3*M]
  // u,v per-vertex
  float textureCoords[2,3*M]
  // RGB texture map
  float baseColor[W,H,3]
 }
```

3.4. Submission Format

Participants for the SUMO challenge should submit their results in zip file format. The zip file should be named `<room_name.zip>` (e.g., `living_room.zip`) and should contain a single folder `<room_name>` (e.g., `living_room`). The submission folder should contain an xml file named `<room_name>.xml` (e.g., `living_room.xml`). The xml lists the elements in the scene, and, for each element, provides its category, bounding box, and pose. Each element's voxel grid or mesh is

provided in a separate file for efficiency. The format for the xml is given in Appendix C. Additionally, an xsd specification for the format is provided as part of the SUMO GitHub repository.

Oriented bounding boxes (OBB) format: The scene is described by a single xml file:

```
living_room
    > living_room.xml
```

Oriented voxels format: In addition to the xml file used in the OBB format, each element's voxel centers are saved as an $(N \times 6)$ matrix in hdf5 file format [10]. The base name for each file should match the <id> tag for the corresponding element within the xml file.

```
living_room
    > living_room.xml
    > bottle_1.h5
    > bottle_2.h5
    > chair_1.h5
    > wall_1.h5
    > floor_2.h5
```

Oriented meshes format: In addition to the xml file used in the OBB format, each element's mesh reperesnation is saved in glb format [6]. The base name for each file should match the <id> tag for the corresponding element within the xml file.

```
living_room
    > living_room.xml
    > bottle_1.glb
    > bottle_2.glb
    > chair_1.glb
    > wall_1.glb
    > floor_2.glb
```

4. Evaluation Metrics

SUMO submissions are evaluated according to four categories of metrics: **Geometry**, **Appearance**, **Semantics** and **Perceptual**, which we refer to as the **GASP** methodology. The perceptual evaluation is a novel evaluation metric derived from user studies to emphasize the aspects of modeling that are more perceptually relevant to people, and the other metrics are based on best practices from state-of-the-art recognition and modeling algorithms.

A submission is evaluated using the following process. First, a data-association process matches ground-truth scene elements to the submission's scene elements (Section **??**). The process identifies correct matches as well as false negatives (missed elements) and false positives (extra elements). The resulting association is then evaluated using metrics targeting geometry, appearance, semantics, and perceptual features (Sections 4.2 through 4.5). Lastly, the metrics are combined using a weighted average for a final evaluation score. The methods are summarized in Table 1.

		Track		
Metric		Bounding Boxes	Voxels	Meshes
Data association		Greedy Shape Similarity		
Geometry	Shape	Category Agnostic mAP		
	Pose	Average Geodestic Distance (rotation) Average Translation Error (translation)		
Appearance		N/A	RMS color distance Voxel RMSSSD	RMS color distance Surface point RMSSSD
Semantics Perceptual		Mean Average Precision (mAP) Average of Weighted Gaussians		

Table 1: Summary of main algorithms and metrics for the SUMO challenge

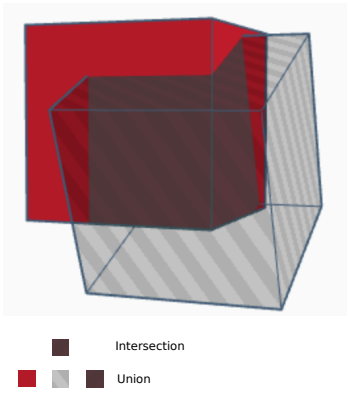

Figure 2: Bounding Box Intersection and Union

4.1. Scene Element Data Association

To associate submitted scene elements to ground-truth elements, we apply a greedy algorithm similar to the method used in the COCO and PASCAL VOC challenges [4]. We define a similarity measure S for comparing the shape of detected elements to their ground-truth counterparts in each of the data representations: bounding boxes (S_{bb}), voxels (S_{vox}), and meshes (S_{mesh}). For a given scene, detections with similarity exceeding a threshold τ_i are sorted in descending order of detection score, and each is matched with the unmatched ground-truth element with which it has the highest S. For the geometric and appearance evaluation metrics, this matching is performed independently of element category, whereas for semantic evaluation metrics, the matching is performed on a per-category basis. Following the COCO challenge methodology [15, 2], our metrics are averaged across multiple similarity thresholds, $\tau = \{0.5, 0.55, 0.6, ...0.95\}$, which rewards methods with better localization. Data association produces a set of matches $M_i = m_1, m_2, \ldots, m_J$, where $m_j = (e_j^{\mathrm{DET}}, e_j^{\mathrm{GT}})$ is a pair of matched elements (detection, ground truth).

Bounding Box Shape Similarity. To compare the similarity (S_{bb}) between bounding boxes m and n, we use the

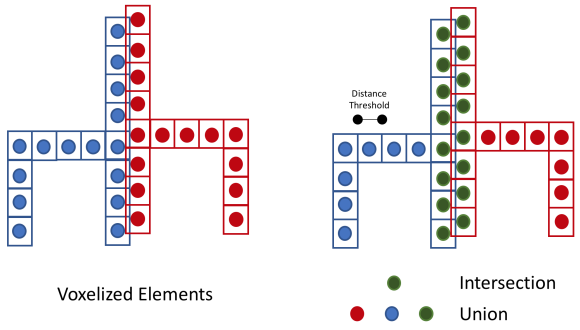

Figure 3: Voxel Intersection and Union

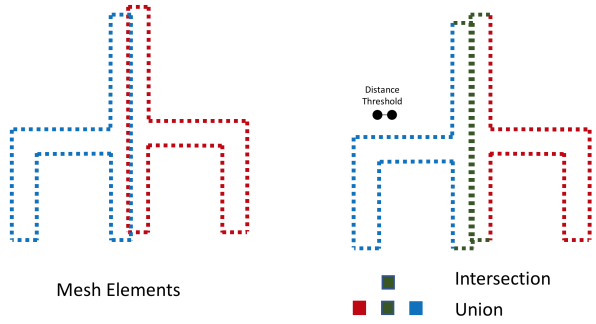

Figure 4: Mesh Intersection and Union

conventional bounding box intersection-over-union (IoU):

$$S_{\text{bb}}(m,n) = \frac{V_{\text{ov}_{mn}}}{V_m + V_n - V_{\text{ov}_{mn}}}, \tag{1}$$

where $V_{\text{ov}_{mn}}$ is the volume of the intersection between m and n, and V_m and V_n are the volumes of m and n respectively (Figure 2).

Voxel Shape Similarity. To compare the similarity (S_{vox}) between voxelized elements m and n, we define a voxel intersection-over-union metric as the ratio of overlapping occupied voxels to the total number of voxels. An overlapping voxel in m is a voxel whose center is within a small distance threshold of at least one other voxel center in n. The threshold is set to twice the voxel size and accounts for quantization effects. change notation!

$$S_{\text{vox}}(m,n) = \frac{|V_{c_{mn}}|}{|V_{c_m}| + |V_{c_n}|}, \tag{2}$$

where $V_{c_{mn}}$ is the set of overlapping voxels between m and n, and V_{c_m} and V_{c_n} are the voxel centers of m and n respectively (Figure 3).

Surface Mesh Shape Similarity. To compare the similarity (S_{mesh}) between surface meshes m and n we first extract a uniform random sample of surface points on each mesh with sampling density x. We define a mesh intersection-over-union metric as the ratio of overlapping surface points to the total number of surface points. An overlapping point in m is a sampled point that is within a small distance threshold of at least one sampled point in n. change notation?

$$S_{\text{mesh}}(m,n) = \frac{|B_{mn}|}{|B_m| + |B_n|}, \tag{3}$$

Above B_{mn} is the set of overlapping points between m and n, and B_m and B_n are the set of sampled points from m and n respectively (Figure 4). Note that for uniformly sampled meshes, the number of points on a given surface is approximately proportional to the surface area.

4.2. Geometric Evaluation

Geometry is evaluated using two primary metrics: a **shape similarity score** (SSc) , which is the category-agnostic average precision, and a **pose error**, which consists of a translation error and a rotation error. In addition, for voxel and mesh representations, we compare shapes using a Root Mean Squared Symmetric Surface Distance (SD_{rms}), which bears similarity with the Chamfer distance used to compare two point clouds [17].

Shape similarity score (SSc): To evaluate predicted shape, we use the category-agnostic mean average precision which is an indication of the fraction of elements for which a sufficiently geometrically similar shape was found as a match in the ground-truth irrespective of object category. Average precision (AP) is computed similarly to the COCO [2] and PASCAL VOC challenges [4]. Specifically, for a given similarity threshold τ_i, the precision-recall (P-R) curve is computed using the detection scores of predicted elements in the scene, and AP is computed by averaging the precision at 11 points on the P-R curve,

$$AP(\tau_i) = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} p_{\text{interp}}(r, \tau_i), \tag{4}$$

where $p_{\text{interp}}(r, \tau_i)$ is the *interpolated* precision at recall level r for similarity threshold τ_i,

$$p_{\text{interp}}(r, \tau_i) = \max_{\tilde{r}:\tilde{r} \geq r} p(\tilde{r}, \tau_i), \tag{5}$$

where $p(\tilde{r}, \tau_i)$ is the measured precision at recall $\tilde{r}$ for similarity threshold τ_i.

The shape similarity score (SSc) is computed by averaging the AP over all shape similarity thresholds $\tau = \{0.5, 0.55, \ldots, 0.95\}$.

$$SSc = \frac{1}{|\tau|} \sum_{i=1}^{|\tau|} AP(\tau_i) \tag{6}$$

Pose error for rotation. To evaluate rotation error across **matched pairs**, we compute the average geodesic

distance between the detected and ground truth rotations [30]:

$$\Delta R = \frac{1}{|\tau|}\sum_{i=1}^{|\tau|}\frac{1}{|M_i|}\sum_{j=1}^{|M_i|}\Delta r(R_{i,j}^{\mathrm{GT}}, R_{i,j}^{\mathrm{DET}}) \quad (7)$$

where $R_{i,j}^{\mathrm{GT}}$ and $R_{i,j}^{\mathrm{DET}}$ are the rotation matrices of e_j^{GT} and e_j^{DET} from M_i, and $\Delta r(R_a, R_b)$ is the geodesic distance between rotations R_a and R_b:

$$\Delta r(R_a, R_b) \equiv \frac{1}{\sqrt{2}}||\log(R_a^T R_b)||_F, \quad (8)$$

...clarification needed... Rotational symmetries are handled as follows: For 2-fold and 4-fold symmetries, the ground truth pose is rotated by each of the possible values, and the minimum rotation error is reported. For cylindrical symmetry, the error is computed on a reduced dimensionality rotation matrix, with the symmetry axis eliminated. For spherical symmetry, rotation error is zero.

Pose error for translation. To evaluate translation error across matched pairs, we compute the average translation error as

$$\Delta T = \frac{1}{|\tau|}\sum_{i=1}^{|\tau|}\frac{1}{|M_i|}\sum_{j=1}^{|M_i|}\Delta t(t_{i,j}^{\mathrm{GT}}, t_{i,j}^{\mathrm{DET}}), \quad (9)$$

where $t_{i,j}^{\mathrm{GT}}$ and $t_{i,j}^{\mathrm{DET}}$ are the translation vectors of e_j^{GT} and e_j^{DET} from M_i, and $\Delta t(t_a, t_b)$ is the translation error:

$$\Delta t(t_a, t_b) = ||t_a - t_b|| \quad (10)$$

Average Root Mean Squared Symmetric Surface Distance (SD_{rms}). For voxel and mesh representations, we follow [35] to compare matched scene elements by computing a measure of the distance from the points (or voxel centers) of one element to the points (or voxel centers) of the other. We average this value across matches and thresholds to obtain the average SD_{rms} measure as follows:

$$SD_{rms} = \frac{1}{|\tau|}\sum_{i=1}^{|\tau|}\frac{1}{|M_i|}\sum_{j=1}^{|M_i|}\Delta d(j, jm_i) \quad (11)$$

where,

$$\Delta d(j, jm_i) = \frac{1}{\sqrt{|B_j| + |B_{jm_i}|}}\sqrt{\sum_{x\in B_j} d^2(x, B_{jm_i}) + \sum_{y\in B_{jm_i}} d^2(y, B_j)} \quad (12)$$

where Δd is the RMS error, B_j is the set of points (or voxel centers) of matched ground-truth element j, B_{jm_i} is the set of points (or voxel centers) for the scene element matched with j, and $d(a, B)$ is the distance between point a and set B – if b is the nearest neighbor of a in set B, then $d(a, B) = ||a - b||$.

4.3. Appearance evaluation

Average Color Root Mean Squared Symmetric Surface distance (CD_{rms}). For voxel and mesh representations, we use a variant of the SD_{rms} measure in which we compare RGB color instead of point location as follows:

$$CD_{rms} = \frac{1}{|\tau|}\sum_{i=1}^{|\tau|}\frac{1}{|M_i|}\sum_{j=1}^{|M_i|}\Delta d_{\mathrm{RGB}}(j, jm_i) \quad (13)$$

where,

$$\Delta d_{\mathrm{RGB}}(j, jm_i) = \frac{1}{\sqrt{|B_j| + |B_{jm_i}|}}\sqrt{\sum_{x\in B_j} d_{\mathrm{RGB}}^2(x, B_{jm_i}) + \sum_{y\in B_{jm_i}} d_{\mathrm{RGB}}^2(y, B_j)} \quad (14)$$

where Δd_{RGB} is the color RMS error, B_j is the set of points (or voxel centers) of matched ground-truth element j, B_{jm_i} is the set of points (or voxel centers) for the scene element matched with j at threshold i. If b is the nearest neighbor of a in set B, then $d_{RGB}(a, B) = ||a_{RGB} - b_{RGB}||$ where a_{RGB} is the color vector of point a.

4.4. Semantics evaluation

We evaluate semantics using mean Average Precision (mAP) which we obtain by averaging precision across categories and across shape similarity threshold using the same method as in the COCO challenge [15].

$$AP(\tau_i, C_j) = \frac{1}{11}\sum_{r\in\{0, 0.1, ..., 1\}} p_{\tau_i, C_j}(r), \quad (15)$$

where $p_{\tau_i, C_j}(r)$ is the precision at recall level r for class j and similarity threshold i. The mAP is computed by averaging the AP over all shape similarity thresholds $\tau = \{0.5, 0.55, 0.6, ...0.95\}$ and object classes C_j.

$$mAP = \frac{1}{|\tau|}\frac{1}{|C|}\sum_{i=1}^{|\tau|}\sum_{c\in C} AP(\tau_i, c) \quad (16)$$

where C is the set of categories and $AP(\tau_i, c)$ is the category-specific average precision, which is computed as in Equation 4, except only using detections and ground truth elements for category c.

4.5. Perceptual Evaluation

Perceptual evaluation is performed based on the results of a series of user studies performed over the course of one year. X participants experienced Y sets of real and virtual rooms. They were expected to identify which characteristics of a room were most noticeably different and how they can manipulate the rooms to make them feel more realistic. The designs and analyses of these experiments will be published in a paper soon.

Perceptual evaluation depends heavily on the results of the scene elements' association discussed in detail in Section 4.1. Once objects are associated, we assign a score (and/or a penalty) to each object based on its perceptual importance as observed in the experiments. These scores are computed as the result of a weighted Gaussian formulated as

$$P(x) = ae^{-(x-\mu)^2/2\sigma^2} \tag{17}$$

where the weight a is driven by the relative importance of a given object property (i.e., the overall scale of the room was found to be more important to participants than the specific position of an object within it), the mean μ can vary to enable asymmetric behavior (i.e. an object floating above a surface is generally more perceptually obvious than one sinking into it), and the variance σ can vary to allow more or less tolerance with respect to the specific property (i.e., while room scale was perceived to be most important, participants were willing to accept approximately 20% error; we account for this in the scoring with a "wider" Gaussian, essentially giving full—or close to full—credit, even for properties that may exhibit some error).

The specific properties of the room (or of individual objects in it) that are addressed by the perceptual evaluation metric are: room scale, presence or absence of an individual object (weighted by object size; larger objects are more perceptually important to participants), scale of an individual object (relative to the scale of the room), scale of an individual object (relative to the "true" scale of that object), horizontal translation of an individual object compared to its true value (whether the object is correctly located on the surface), and vertical elevation of an object compared to its true value (whether the object is actually *on* the surface, as opposed to sinking into it or floating above it).

5. Baseline Methods

In this section, we discuss state-of-the-art approaches that can be applied to SUMO, and provide baseline methods for each track of the competition.

5.1. Mesh Track: Mask R-CNN + Template Mesh

3D shape reconstruction from single images is a challenging task. However, when assuming the input image contains a single object, we have witnessed several advances in recent years [22, 32, 33, 24, 13, 31]. When it comes to predicting the full geometry of the scene, the task becomes significantly more challenging. Tulsiani *et al*. [30] factorize scenes into objects and layouts, and estimate each part individually. Shin *et al*. [25] factorize scenes into layers, and synthesize features in multiple views using novel transformer networks. Zhi *et al*. [36] utilize multi-view images to perform dense semantic 3D reconstruction. There is also a line of research that utilizes 3D exemplar CAD models to reconstruct the scene [11, 37]. Our baseline method is also similar in this method.

As shown in Figure 5, we extend the Mask R-CNN instance segmentation framework [8], and perform 3D reconstruction. Using Mask R-CNN, we perform instance segmentation on the side faces of an input SUMO cube-map. We add an additional head to Mask R-CNN that predicts the 3D orientation of each instance. Together, this gives us a list of 2D bounding boxes, each contain information about the segmentation mask, the semantic category, and the 3D orientation.

We predict the translation for each instance by cropping the depth map with the predicted mask and finding the median of the resulting point cloud. We predict the 3D shape using a crude shape retrieval strategy. We find the most commonly occurring mesh in the training set for each semantic category, and treat those as representative meshes. For each instance, we retrieve the representative mesh for the predicted category and present it as the shape prediction. Although this brute force approach does not recover the exact shape, we achieve XXX percent accuracy on the evaluation site for the XX track.

5.2. Voxel Track: ?

cite SSCNet [29]

5.3. Bounding Box Track: Faster R-CNN + 3D Bounding Box Recovery

Localizing object in 3D cuboids has become a standard task in autonomous driving applications [5]. In recent years, many algorithms are proposed and improved the performance on indoor scene datasets [26] as well. Data-driven approaches [27, 7] align 3D CAD models to objects in the scene. Novel 3D descriptors and representations [20, 21] robustly links 3D object pose to 2D image boundaries. Methods using 3D convolutional neural networks [28, 12] significantly improved the speed of the detection. Another general approach is utilizing 2D detection outputs for recovering 3D bounding boxes [18, 14, 3]. Our baseline algorithm also utilizes this idea.

Our approach for the Bounding Box Track consists of a 2D object detection algorithm followed by a 3D bounding box recovery step (see Figure 6). We first train a standard

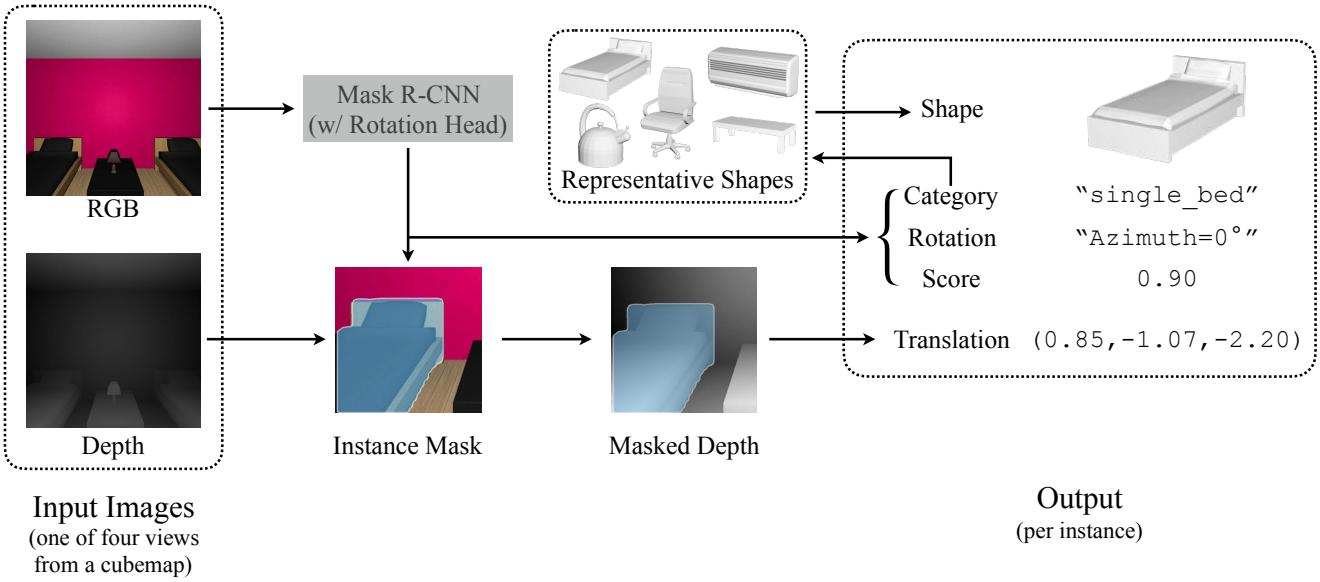

Figure 5: **Mask RCNN + Template Mesh for Scene Modeling**

Faster R-CNN network [19] using a pre-trained VGG-16 model, and predict 2D bounding boxes from 4 of the frames views of the 360° RGB scene (Back, Front, Left, Right). We then extract point cloud that correspond to pixels enclosed in each of those boxes, and fit a tight 3D bounding box around the predicted object. Finally, we trim the predicted bounding box along each axis and keep the $10-90$th percentile. This is done to account for the outliers of the point cloud.

To train the network, we first separated the RGB images of the cube-map into the respective Back, Left, Front, and Right frame views. Next, we projected the 3D bounding box coordinates into their respective frames in 2D, and formulate the 2D bounding box annotations. We use them to train the Faster R-CNN network [19]. The projected coordinates that spanned multiple frames were unused.

A major drawback of this approach is that the network is unable to learn the true depths of objects. Although the boundaries may be reasonable in 2D, the recovery of a bounding box in 3D varies wildly. Additionally, detecting objects in the Back, Left, Front, and Right frames individually leaves many objects in the scene undetected. Our performance is shown in Table 2.

Table 2: Results from the Contest Phase for the Bounding Box Track using Method 5.3. Values scaled up by a factor of 100.

Shape	Rotation	Translation	Semantics	Perceptual
1	92	20	1	0

6. Conclusions

As the end-to-end scene understanding approaches start to tackle with realistic scenes using deep learning, we believe that datasets and challenges to enable such data-driven methods will revolutionize 3D computer vision domain. In this paper and in the SUMO Challenge, we aim to provide a comprehensive dataset, extensive evaluation metrics, and novel baseline solutions to motivate the community and provide tools to invest more into RGBD-to-3D problems.

As the second version of the SUMO workshop, the participation to our competitions is already promising. We foresee SUMO to become a valuable testbed for robust 3D scene understanding methodologies, as well as 2D-3D translation, segmentation and recognition from 360, and semantic reconstruction tasks. We would also like to encourage researchers and the academic community to use the SUMO dataset as a flexible synthetic dataset for generative modeling.

Acknowledgments

We would like to thank [people], [research funding], [workshop/challenge funding], [program and technical committees], [CVPR chairs and IEEE].

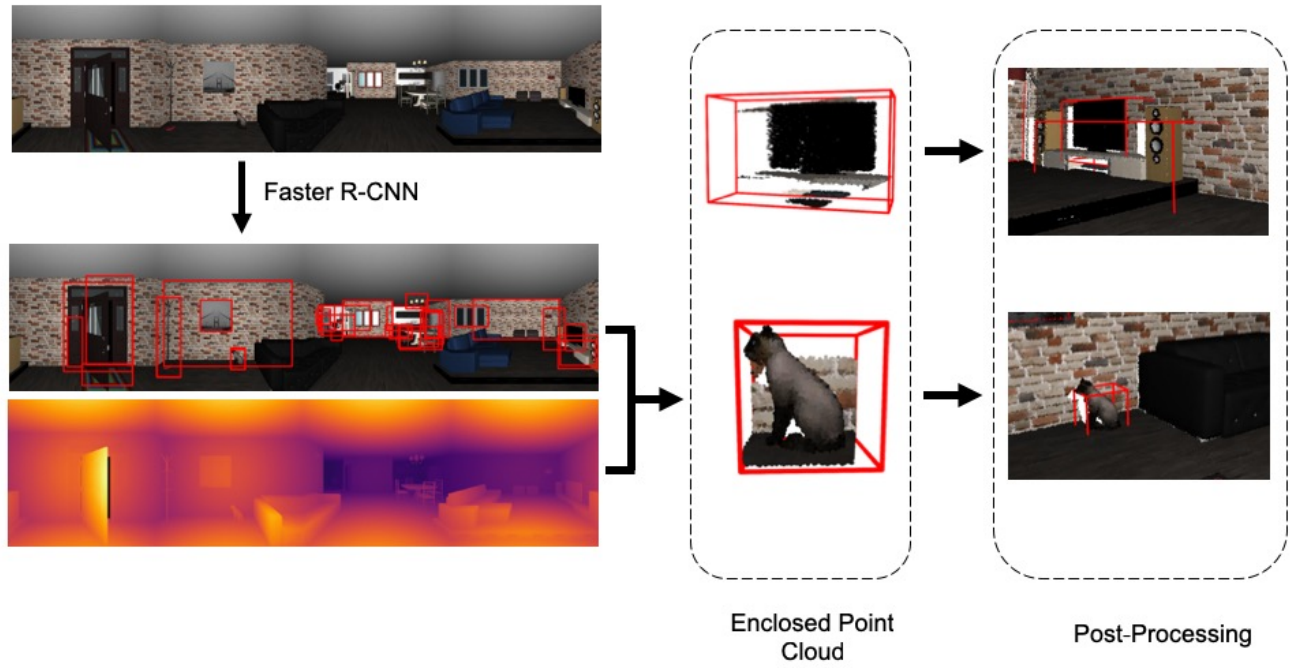

Figure 6: **Faster R-CNN + 3D Bounding Box Recovery.**

Appendices

Appendix A. Object Categories

The categories for the SUMO challenge elements are a subset of those in the SUN-CG fine-grained class list. Three types of categories have been removed:

1. animate objects (e.g., human, pet)
2. categories with than 100 instances in the training data
3. ”unknown” category. Instances in the unknown category are primarily box-shaped objects, which may be used to represent instances from a variety of categories. In the underlying annotations, these objects are unlabeled, and in this challenge, those objects are not evaluated.

Included Categories			
air_conditioner	cutting_board	kitchen_set	single_bed
arch	desk	knife_rack	sink
armchair	dining_table	laptop	soap_dispenser
atm	dishwasher	loudspeaker	sofa
baby_bed	door	magazines	stairs
basketball_hoop	double_bed	microwave	stand
bathtub	dresser	mirror	stationary_container
beer	dressing_table	motorcycle	stereo_set
bench_chair	dryer	office_chair	switch
book	fence	ottoman	table_and_chair
books	fireplace	outdoor_lamp	table_lamp
bookshelf	fish_tank	outdoor_seating	telephone
bottle	fishbowl	pan	television
bunker_bed	floor	partition	toilet
candle	floor_lamp	pedestal_fan	towel_hanger
car	food_processor	person	towel_rack
cart	food_tray	pet	toy
ceiling	fruit_bowl	piano	trash_can
ceiling_fan	game_table	picture_frame	trinket
chair	garage_door	pillow	tv_stand
chair_set	glass	place_setting	umbrella
chandelier	goal_post	plant	utensil_holder
chessboard	grill	plates	vacuum_cleaner
clock	gym_equipment	playstation	vase
cloth	hanger	pool	wall
coffee_kettle	hanging_kitchen_cabinet	range_hood	wall_lamp
coffee_machine	heater	range_oven	wardrobe_cabinet
coffee_table	household_appliance	refrigerator	washer
column	iron	roof	water_dispenser
computer	ironing_board	rug	whiteboard
containers	jug	shelving	window
cup	kettle	shoes_cabinet	workplace
curtain	kitchen_cabinet	shower	xbox

Excluded Categories			
accordion	food	microphone	surveillance_camera
amplifier	fork	mortar_and_pestle	table
bicycle	gramophone	outdoor_spring	teapot
bread	guitar	poker_chips	theremin
cake	hair_dryer	range_hood_with_cabinet	toaster
camera	headphones_on_stand	range_oven_with_hood	toilet_paper
cellphone	headstone	rifle_on_wall	toilet_plunger
coffin	helmet	safe	toiletries
container	ipad	shoes	tricycle
cooker	keyboard	slot_machine	tripod
decoration	knife	slot_machine_and_chair	unknown
drinkbar	ladder	soap_dish	weight_scale
drumset	lawn_mower	spoon	wood_board
empty	mailbox	storage_bench	

- borderline cases. For objects where it is not clear whether a particular type of symmetry exists along a given axis, the more liberal label is applied. For example, a 6-sided mirror would have a rotational symmetry about the Z axis.

Appendix C. Scene File Format (XML)

Below, is a simple example of the xml portion of a SUMO scene. The file consists of a header section (version and categories tags) and a list of elements. Each element contains an id, category, detection score, bounds, pose, and symmetry specification. The detection score is only meaningful for detection results (i.e., your output scenes), but ground truth scenes also contain the detectionScore field, and it is set to -1.

```
<?xml version="1.0" encoding="UTF-8"?>
<scene xmlns="https://www.sumo-challenge.org">
    <version>2.0</version>

    <categories>
        <id>fb_categories_v1_0</id>
        <url>https://sumochallenge.org/en/categories-1_0.json</url>
    </categories>

    <elements>
        <element>
            <id>1</id>
            <category>floor</category>
            <detectionScore>0.8</detectionScore>
            <bounds>
                <corner1> 0, 0, 0 </corner1>
                <corner2> 1.0, 1.0, 1.0 </corner2>
            </bounds>
            <pose>
                <translation>
                    -131.596614, -39.9279011, 92.1260558
                </translation>
                <rotation>
                    <c1> 1.0, 0, 0 </c1>
                    <c2> 0, 1.0, 0 </c2>
                    <c3> 0, 0, 1.0 </c3>
                </rotation>
            </pose>
            <symmetry>
                <x>twoFold</x>
                <y>twoFold</y>
                <z>fourFold</z>
            </symmetry>
        </element>

        <element>
            <id>2</id>
```

```
            <category>bookshelf</category>
            <detectionScore>0.4</detectionScore>
            <bounds>
                <corner1> 0, 0, 0 </corner1>
                <corner2> 1.0, 1.0, 1.0 </corner2>
            </bounds>
            <pose>
                <translation>
                    -131.596614,
                    -39.9279011,
                    92.1260558
                </translation>
                <rotation>
                    <c1> 1.0, 0, 0 </c1>
                    <c2> 0, 1.0, 0 </c2>
                    <c3> 0, 0, 1.0 </c3>
                </rotation>
            </pose>
            <symmetry>
                <x>none</x>
                <y>none</y>
                <z>none</z>
            </symmetry>
        </element>

        <element>
            <id>3</id>
            <category>wall_art</category>
            <detectionScore>0.2</detectionScore>
            <bounds>
                <corner1> 0, 0, 0 </corner1>
                <corner2> 1.0, 1.0, 1.0 </corner2>
            </bounds>
            <pose>
                <translation>
                    -131.596614, -39.9279011, 92.1260558
                </translation>
                <rotation>
                    <c1> 1.0, 0, 0 </c1>
                    <c2> 0, 1.0, 0 </c2>
                    <c3> 0, 0, 1.0 </c3>
                </rotation>
                <symmetry>
                    <x>none</x>
                    <y>none</y>
                    <z>none</z>
                </symmetry>
            </pose>
        </object>
    </element>
</scene>
```

References

[1] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1

[2] *COCO (Common Objects in Context) Detection Evaluation*, 2018 (accessed May 31, 2018). http://cocodataset.org/#detections-eval. 4, 5

[3] Z. Deng and L. J. Latecki. Amodal detection of 3D objects: Inferring 3D bounding boxes from 2d ones in rgb-depth images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 7

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision (IJCV)*, 88(2):303–338, Jun 2010. 4, 5

[5] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361. IEEE, 2012. 7

[6] *glTF (GL Transmission Format)*, 2018 (accessed May 31, 2018). https://en.wikipedia.org/wiki/GlT. 4

[7] S. Gupta, P. A. Arbeláez, R. B. Girshick, and J. Malik. Aligning 3D models to RGB-D images of cluttered scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 7

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969, 2017. 1, 7

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 1

[10] *Hierarchical Data Format*, 2018 (accessed May 31, 2018). https://en.wikipedia.org/wiki/Hierarchical_Data_Format. 4

[11] H. Izadinia, Q. Shan, and S. M. Seitz. Im2cad. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 7

[12] H. Ji, A. Dai, and M. Nießner. 3D-sis: 3D semantic instance segmentation of rgb-d scans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 7

[13] H. Kato, Y. Ushiku, T. Harada, A. Shin, L. Crestel, H. Kato, K. Saito, K. Ohnishi, M. Yamaguchi, M. Nakawaki, et al. Neural 3D mesh renderer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 7

[14] J. Lahoud and B. Ghanem. 2D-driven 3D object detection in RGB-D images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 7

[15] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014. 4, 6

[16] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016. 1

[17] C. Qi, H. Su, M. Kaichun, and L. 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 (CVPR)*, pages 77–85, July 2017. 5

[18] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 7

[19] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. 8

[20] Z. Ren and E. B. Sudderth. Three-dimensional object detection and layout prediction using clouds of oriented gradients. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1525–1533, 2016. 7

[21] Z. Ren and E. B. Sudderth. 3D object detection with latent support surfaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 7

[22] J. Rock, T. Gupta, J. Thorsen, J. Gwak, D. Shin, and D. Hoiem. Completing 3D object shape from one depth image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2484–2493, 2015. 7

[23] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015. 1

[24] D. Shin, C. C. Fowlkes, and D. Hoiem. Pixels, voxels, and views: A study of shape representations for single view 3D object shape prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 7

[25] D. Shin, Z. Ren, E. B. Sudderth, and C. C. Fowlkes. Multi-layer depth and epipolar feature transformers for 3D scene reconstruction, 2019. 7

[26] S. Song, L. Samuel, and J. Xiao. SUN RGB-D: A RGB-D scene understanding benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015. 7

[27] S. Song and J. Xiao. Sliding shapes for 3D object detection in depth images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 634–651. Springer, 2014. 7

[28] S. Song and J. Xiao. Deep sliding shapes for amodal 3D object detection in RGB-D images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 7

[29] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 7

[30] S. Tulsiani, S. Gupta, D. Fouhey, A. A. Efros, and J. Malik. Factoring shape, pose, and layout from the 2D image of a 3D scene. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6, 7

[31] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3D mesh models from single rgb images. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 7

[32] J. Wu, Y. Wang, T. Xue, X. Sun, B. Freeman, and J. Tenenbaum. Marrnet: 3D shape reconstruction via 2.5 d sketches. In *Advances in Neural Information Processing Systems (NIPS)*, pages 540–550, 2017. 7

[33] J. Wu, C. Zhang, X. Zhang, Z. Zhang, W. T. Freeman, and J. B. Tenenbaum. Learning 3D Shape Priors for Shape Completion and Reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 7

[34] D. Yadav, R. Jain, H. Agrawal, P. Chattopadhyay, T. Singh, A. Jain, S. B. Singh, S. Lee, and D. Batra. Evalai: Towards better evaluation systems for ai agents. *arXiv preprint arXiv:1902.03570*, 2019. 2

[35] V. Yeghiazaryan and I. Voiculescu. An overview of current evaluation methods used in medical image segmentation. *CS-RR-15-08*, 2015. 6

[36] S. Zhi, M. Bloesch, S. Leutenegger, and A. J. Davison. Scenecode: Monocular dense semantic reconstruction using learned encoded scene representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 7

[37] C. Zou, R. Guo, Z. Li, and D. Hoiem. Complete 3D scene parsing from single rgbd image. *International Journal of Computer Vision (IJCV)*, 2018. 7