The SUMO Challenge

Lyne Tchapmi (Stanford University) and Daniel Huber (Facebook, Inc.)

1 Introduction

We introduce the Scene Understanding and Modeling (SUMO) challenge with the goal of studying and evaluating the performance of RGBD-to-3D semantic scene modeling algorithms. Challenge participants are tasked to derive a complete, instance-based 3D representation of a scene given an RGB-D representation. The complete representation should highlight geometric instances, appearance, and semantics. This document describes the details of the challenge tasks, including scene elements of interest, data format, and metrics.

2 Data Representation and Submission Format

In the SUMO challenge, 3D scenes are a modeled by a collection of instances, known as elements hereafter. Each element models 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 3.

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

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

```
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 options for shape and appearance representation in submissions.

2.1 Oriented Bounding Boxes

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.

2.2 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]
```

2.3 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
  float vertices[3,N]          // x,y,z coordinates
  float indices[3*M]           // indices of triangle corners
  float textureCoords[2,3*M]   // u,v per-vertex
  float baseColor[W,H,3]       // RGB texture map
 }
```

2.4 Submission Format

Submissions for the SUMO challenge should be provided 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 $(NX6)$ matrix in hdf5 file format [4]. 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 [3]. 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
```

3 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 3.1). 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 3.2 through 3.5). The methods are summarized in Figure 1.

3.1 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 [2]. 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.

		Track		
Metric		Bounding Boxes	Voxels	Meshes
Data Association		greedy shape similarity		
Geometry	Shape	category agnostic mAP	category agnostic mAP	category agnostic mAP
	Pose	• average geodesic distance (rotation) • average translation error (translation)		
Appearance		N/A	• RMS color distance (RMSCD) • Voxel RMSSSD	• RMS color distance (RMSCD) • Surface point RMSSSD
Semantics		mean Average Precision (mAP)	mean Average Precision (mAP)	mean Average Precision (mAP)
Perceptual		average of weighted Gaussians		

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

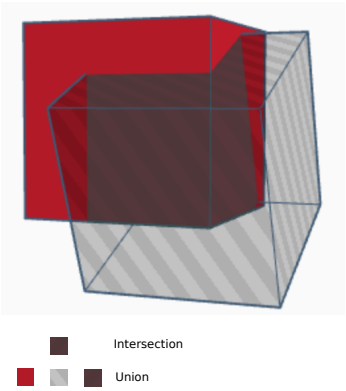

Figure 2: Bounding Box Intersection and Union

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 [5, 1], 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 conventional bounding box intersection-over-union (IoU):

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

where $V_{\mathrm{ov}_{mn}}$ 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.

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

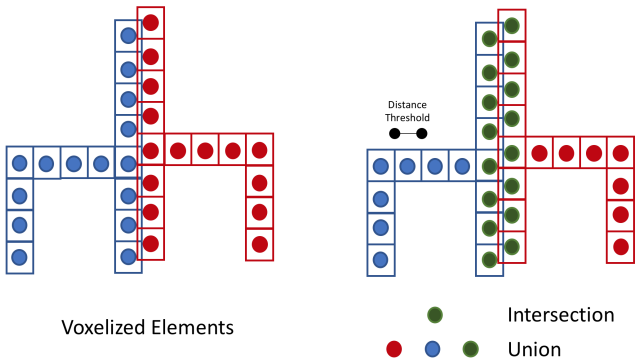

Figure 3: Voxel Intersection and Union

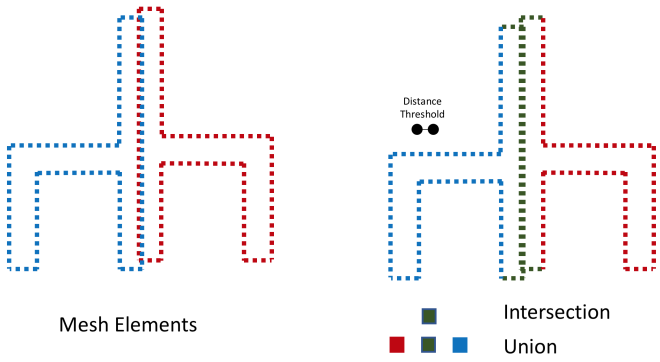

Figure 4: Mesh Intersection and Union

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. 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.

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

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.

3.2 Geometric Evaluation

Geometry is evaluated using two primary metrics: a **shape similarity** score, which is the category-agnostic average precision (AP), and a **pose error**, which consists of a translation error and a geodesic distance. In addition, for voxel and mesh representations, we compute a Root Mean Squared Symmetric Surface Distance (RMSSSD), which bears similarity with the Chamfer distance used to compare two point clouds [6].

Shape similarity: the category-agnostic mean average precision (mAP). The category-agnostic mAP is an indication of the fraction of elements for which a sufficiently geometrically similar shape was found as a match in a submission irrespective of object category. Average precision (AP) is computed similarly to the COCO and PASCAL VOC challenges [2]. Specifically, for a given value

of τ, the precision-recall (P-R) curve is computed using the detection scores of all detections in all scenes, averaging the precision at 11 points on the P-R curve:

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

Above $p_{\text{interp}}(r, \tau_i)$ 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 mAP is computed by averaging the AP over all shape similarity thresholds $\tau = \{0.5, 0.55, \ldots, 0.95\}$.

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

Pose error: rotation. To evaluate rotation error across matched pairs, we compute the average geodesic distance between the detected and ground truth rotations [7]:

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

where $R_{i,j}^{\text{GT}}$ and $R_{i,j}^{\text{DET}}$ 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:

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

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

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

where $t_{i,j}^{\text{GT}}$ and $t_{i,j}^{\text{DET}}$ 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:

$$\Delta t(t_a, t_b) = ||t_a - t_b|| \tag{10}$$

Average Root Mean Squared Symmetric Surface Distance (RMSSSD). For voxel and mesh representations, we follow [8] 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 RMSD measure as follows:

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

where,

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

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 $d(a, B) = ||a - b||$.

3.3 Appearance evaluation

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

$$RMSCD = \frac{1}{|\tau|}\sum_{i=1}^{|\tau|}\frac{1}{|M_i|}\sum_{j=1}^{|M_i|}\Delta d_{RGB}(j, jm_i) \tag{13}$$

where,

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

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.

3.4 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 [5].

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

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.

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

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.

3.5 Perceptual Evaluation

Perceptual evaluation is performed based on the results of a series of user studies performed over the course of the last year, in which participants experienced sets of real and virtual rooms, and were given the opportunity to identify which characteristics of the room were most noticeably different and also to manipulate the rooms in such a way as to make them feel most real. The designs and analyses of these experiments will be summarized in a paper that will be made available on arXiv.

Perceptual evaluation depends heavily on the results of the scene elements association discussed in detail in Section 3.1. 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

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

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 position of an object within it), the mean μ can be varied 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 be varied 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).

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	

In order to allow consistent modeling across disparate object instances within the same category, each object is represented in is canonical pose, with +Y being up and +Z being the front. The following rules were used in assigning canonical pose for each object:

- Objects with an intuitive front (e.g., chair, bookshelf, toilet). The front is the direction from which a normal person would typically approach the object. Typically, the back of an object would be placed against a wall.
- Objects with no intuitive front (e.g., coffee cup, blender). The front is chosen arbitrarily. Symmetry about the Y axis is set to twoFold, fourFold, or cylindrical.
- Walls. Walls are assumed to be vertical. The front is perpendicular to the largest vertical plane. Top (+Y) is up.
- Floors and ceilings. Floors and ceilings may be horizontal or angled. The front is perpendicular to the largest surface. Top is arbitrary, but perpendicular to the front. Symmetry about the Y axis is set to cylindrical.

The annotations may not agree with everyone's definition of "front," but our goal is that every object within a category is consistently labeled with the same canonical pose with respect to this somewhat arbitrary definition.

Appendix B Symmetry Annotation

The symmetry annotations do not adhere to the strict sense of the definition of symmetry. Symmetry is used to indicate ambiguities in the definition of top or front. We employ the symmetry specification in the evaluation metrics to allow freedom in the pose rotation without increasing error. Here are a few illustrative examples:

- blender. The front of a blender is not obvious, but is is primarily square-shaped with respect to the Y axis. Therefore, we set the Y axis symmetry to 4-fold.
- coffee cup. Cups have no obvious front but are primarily cylindrical. Therefore, we set the Y axis symmetry to cylindrical.
- rectangular table. For rectangular furniture like tables, we label the long side as the front and specify a 2-fold symmetry about the Y axis.
- bookshelves. Some furniture would look equally correct if posed upside down. For these objects, we specified a 2-fold symmetry about the Z axis.
- 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] *COCO (Common Objects in Context) Detection Evaluation*, 2018 (accessed May 31, 2018). http://cocodataset.org/\#detections-eval.

[2] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, Jun 2010.

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

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

[5] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, 2014.

[6] Charles Qi, H. Su, M. Kaichun, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85, July 2017.

[7] Shubham Tulsiani, Saurabh Gupta, David Fouhey, Alexei A. Efros, and Jitendra Malik. Factoring shape, pose, and layout from the 2d image of a 3d scene. In *Computer Vision and Pattern Regognition (CVPR)*, 2018.

[8] Varduhi Yeghiazaryan and Irina Voiculescu. An overview of current evaluation methods used in medical image segmentation. *CS-RR-15-08*, 2015.