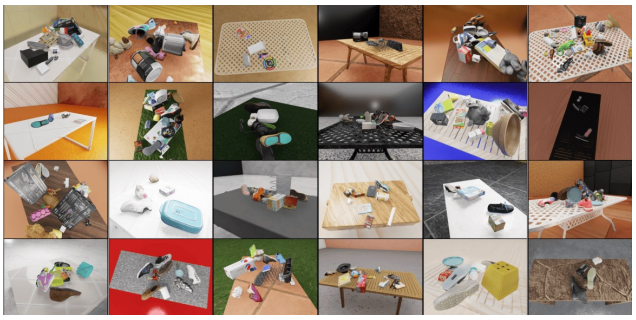
SynTable: A Synthetic Data Generation Pipeline for Unseen Object Amodal Instance Segmentation of Cluttered Tabletop Scenes

Anonymous CVPR submission

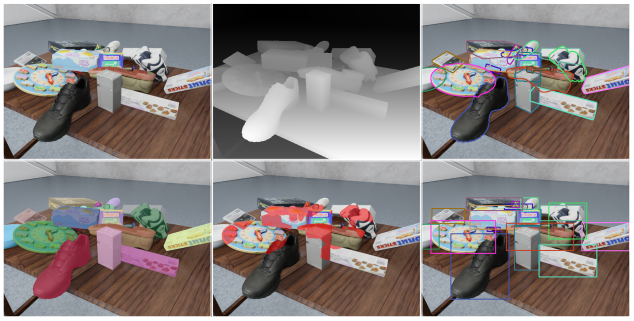
Paper ID 44

Abstract

In this work, we present SynTable, a unified and flexible Python-based dataset generator built using NVIDIA's Isaac Sim Replicator Composer for generating high-quality synthetic datasets for unseen object amodal instance segmentation of cluttered tabletop scenes. Our dataset generation tool can render complex 3D scenes containing object meshes, materials, textures, lighting, and backgrounds. Metadata, such as modal and amodal instance segmentation masks, object amodal RGBA instances, occlusion masks, depth maps, bounding boxes, and material properties can be automatically generated to annotate the scene according to the users' requirements. Our tool eliminates the need for manual labeling in the dataset generation process while ensuring the quality and accuracy of the dataset. In this work, we discuss our design goals, framework architecture, and the performance of our tool. We demonstrate the use of a sample dataset generated using SynTable for training a state-of-the-art model, UOAIS-Net. Our state of the art results show significantly improved performance in Sim-to-Real transfer when evaluated on the OSD-Amodal dataset. We offer this tool as an open-source, easy-to-use, photorealistic dataset generator for advancing research in deep learning and synthetic data generation.



(a)



(b)

Figure 1. (a) RGB outputs of photorealistic cluttered tabletop scenes generated by SynTable pipeline. (b) Visualization of RGB Images, Depth Images, Object Amodal Masks, Object Visible Masks, Object Occlusion Masks, and Object Visible Bounding Boxes.

1. Introduction

Amodal completion is a perceptual ability that enables the perception of whole objects, even when they are partially occluded [1, 16]. It encompasses three key tasks: amodal shape completion, amodal appearance completion and occlusion order. Amodal shape completion involves predicting the complete structure of an object beyond its visible portion, typically represented as a binary segmentation mask that includes both visible and occluded regions. Amodal appearance completion refers to the process of inferring the likely apperance of the hidden regions of an object based on its visible parts (RGB values of hidden pixels). Occlusion Order considers the occlusion relationship between objects, distinguishing between occluders (objects that obscure others) and occludees (objects being occluded), which can involve no occlusion or bi-directional occlusion. Humans are capable of "filling in" the occluded appearance of invisible objects, owing to their vast experience in perceiving countless objects in various contexts and scenes. This ability to infer an object's complete structure from its partial appearance is critical for systems requiring holistic scene understanding, such as augmented or virtual reality, and robotics and automation. In modern vision systems, accurately comprehending occluded objects in cluttered en-

vironments is essential for tasks ranging from object interaction to environment reconstruction.

There are three key challenges in amodal instance segmentation: Firstly, the lack of large-scale, high-quality datasets for unseen object amodal instance segmentation (UOAIS) limits the performance of vision systems in real-world applications [3]. While datasets exist for object detection and segmentation [6, 10, 14, 17, 30], only a few address UOAIS [2]. This is largely due to the difficulty of manually annotating amodal data, as human annotators must estimate occluded regions, leading to inherent subjectivity and inconsistencies in ground-truth annotations [2, 22, 32].

Secondly, synthetic datasets often suffer from visual domain mismatch due to non-photorealistic rendering or insufficient domain randomization [29], resulting in poor Sim-to-Real transfer. Existing tools prioritize rendering speed over photorealism, limiting their utility for training robust vision models, which results in a poor Sim-to-Real transfer that will inevitably reduce the performance of algorithms in real-world applications.

Thirdly, the lack of automated tools for generating amodal annotations and evaluating occlusion relationships hinders progress in this domain. Existing evaluation metrics focus on visible object regions but do not assess a model's ability to infer occlusion order — a critical capability for systems operating in cluttered scenes. For example, understanding occlusion hierarchies enables sequential task planning and reduces errors caused by overlapping objects. However, manual annotation of such relationships is prohibitively time-consuming, necessitating simulation tools as a more cost-effective and accurate solution.

In this work, we address these challenges by developing SynTable, a unified Python-based tool for generating customizable, photorealistic datasets for UOAIS in cluttered scenes. While our experiments focus on tabletop environments (common in interaction tasks), our framework generalizes to diverse settings. SynTable integrates rendering and annotation into a single pipeline, allowing users to control scene complexity, object variety, and annotation types. Built on NVIDIA's Isaac Sim Replicator Composer, it leverages high-fidelity ray tracing and domain randomization to bridge the Sim-to-Real gap.

Our key contributions are summarized as follows:

1. We develop a pipeline to automatically render photorealistic cluttered tabletop scenes and generate ground truth amodal instance segmentation masks, eliminating manual labeling in dataset generation. Our designed dataset generation tool creates photorealistic and accurately-labeled custom datasets for UOAIS (refer to Figure 1(a)).
2. Our tool provides a rich set of annotations related to amodal instance segmentation (refer to Figure 1(b)): modal (visible) and amodal instance segmentation masks, RGBA object instances, occlusion masks, occlusion rates, and occlusion order adjacency matrix. Users can easily select which annotations to include in their dataset based on the requirements of their application.
3. We proposed a novel method to evaluate how accurately an amodal instance segmentation model can determine object occlusion ordering in a scene by computing the scene's Occlusion Order Accuracy (ACC_{OO}).
4. We generated an open-sourced large-scale sample synthetic dataset using our tool consisting of amodal instance segmentation labels for users to train and evaluate amodal segmentation models on 1075 novel objects, designed to benchmark amodal segmentation in occlusion-rich scenarios.

2. Related Works

2.1. Amodal Instance Segmentation in Vision Systems

Recent advances in amodal instance segmentation aim to enhance object detection and tracking in complex scenes. However, challenges such as limited training data and Sim-to-Real gaps persist, particularly in cluttered environments where occlusion reasoning is critical.

Lack of Large-scale High-quality Training Data. While datasets like [7, 9, 31] have advanced amodal segmentation for indoor scenes, few address occlusion-rich scenarios in everyday interaction tasks. Existing efforts often focus on narrow domains: for example, [12] introduced a benchmark for multi-object interaction in industrial settings, but its limited scene and object diversity restrict broader applicability. Similarly, the Object Segmentation Database (OSD) [24] and Object Cluttered Indoor Dataset (OCID) [25] pioneered tools for segmentation in cluttered scenes but lack amodal annotations. Recent work by Back *et al.* [2] manually added amodal masks to OSD, yet this approach remains labor-intensive and prone to human error.

Sim-to-Real Problem. Synthetic datasets like the Tabletop Object Dataset (TOD) [29] and UOAIS-Sim [2] struggle with photorealism and domain randomization, leading to significant Sim-to-Real gaps. For instance, TOD's non-photorealistic rendering limits its utility for training models deployed in real-world applications such as augmented reality or autonomous navigation.

2.2. Tools for Generating Synthetic Datasets

With the rapid development of deep learning, the demand of researchers for synthetic datasets has increased in recent years, leading to the increased development of various tools for generating these datasets [28]. For robotics and computer vision applications, PyBullet and MuJoCo [27] are commonly used physical simulators to generate synthetic data. Xie *et al.* [29] pre-trained an RGB-D unseen object

instance segmentation model using PyBullet. Tobin *et al.* [26] used MuJoCo to generate synthetic images with domain randomization, which can bridge the Sim-to-Real gap by realistically randomizing 3D content. Simulation tools such as PyBullet and MuJoCo typically come with renderers that are accessible and flexible, but they lack physically based light transport simulation, photorealism, material definitions, and camera effects.

To obtain better rendering capabilities, researchers also explored the use of video game-based simulation tools, such as Unreal Engine (UE4) or Unity 3D. For example, Qiu and Yuille [23] exported specific metadata by adding a plugin to UE4. Besides, Unity 3D can generate metadata and produce scenes for computer vision applications using the official computer vision package. Although game engines provide the most advanced rendering technology, they prioritize frame rate over image quality and offer limited capabilities in light transport simulation.

Ray-tracing technology has gained significant traction in creating photorealistic synthetic datasets, as it enables the simulation of light behavior with high accuracy. Software applications such as Blender, NVIDIA OptiX, and NVIDIA Isaac Sim have all incorporated ray-tracing techniques into their functionality. The Replicator Composer, a component of NVIDIA Isaac Sim, constitutes an excellent tool for creating tailored synthetic datasets to meet various requirements in robotics. In this work, we leverage this platform to design a customized pipeline to generate a synthetic dataset tailored to the specific demands of UOAIS for cluttered tabletop scenes.

3. Method

Our dataset generation pipeline is illustrated in Figure 2. Parameters and configurations of the scenes to be rendered are defined in a parameter file. Objects, materials, and light sources used in our pipeline are referred to as assets. The scene is prepared by rendering a tabletop scene with floating objects in Isaac Sim. A physical simulation is run to drop the rendered objects onto the table. For every view within a scene, camera viewpoints and lighting conditions are re-sampled. Subsequently, the annotations are captured to create the dataset. We provide additional details about each step of our data generation pipeline in Section 8 of our supplementary materials.

3.1. Preparing Each Scene

To prepare each scene, a table is randomly sampled and rendered in the center of a room, as shown in Figure 4. The texture and materials of the table, ceiling, wall, and floor are randomized for domain randomization while objects are added with randomized coordinates and orientations. We randomly sample (with replacement) N_{lower} to N_{upper} number of objects for each scene. Objects are initialized with real-life dimensions, mass, collision properties, randomized rotations and coordinates, ensuring diverse object arrangements across scenes. Additional details about our scene preparation method can be found in Section 8.1.

3.2. Physical Simulation of Each Scene

Rendered objects are dropped onto the table through a physics simulation to ensure the random placement of objects in the scene. Objects that rebound off the tabletop surface and land beyond the spatial coordinate region of the tabletop surface are removed, excluding extraneous objects from annotations. We provide more details about our physical simulation in Section 8.2.

3.3. Sampling of Camera Viewpoints

To capture annotations for each scene from multiple viewpoints, we enhance the approach of Gilles *et al.* [18] (which only uses fixed viewpoints) by capturing the V number of viewpoints at random positions within custom radii of two concentric hemispheres of custom radii. The calculation of the Cartesian coordinates of each viewpoint can be found in 8.3 of our supplementary materials. Each viewpoint is oriented such that the camera looks directly at the center of the tabletop surface.

3.4. Sampling of Lighting Conditions

To simulate various indoor lighting conditions for each viewpoint, we resample L spherical light sources using a method similar to Section 8.3. Please refer to Section 8.4 in our supplementary materials for more details. In contrast to Back *et al.*'s [2] approach of using point light sources, we use spherical light sources emitting light in all directions to mimic light bulbs. Furthermore, we uniformly sample the temperatures and intensity of the light sources. Users can customize the number of spherical light sources, as well as their intensities and temperatures.

3.5. Capturing of Ground Truth Annotations

The process of capturing the annotations for a scene is illustrated in Figure 3. In each view, the RGB and depth images of the tabletop scene will be captured (Figure 3(a)). The built-in segmentation function in Isaac Sim Replicator Composer is used to capture the scene's instance segmentation mask from a viewpoint (Figure 3(b)). Subsequently, the visible mask of each object is cropped from the scene's segmentation mask.

For object amodal mask generation, we have developed the following steps. Initially, all objects' visibility are disabled. For each object o in the scene, its visibility is enabled and the instance segmentation function is utilized to capture its amodal mask and the amodal RGBA instance (Figure 3(c)). We compute the object's occlusion mask and occlusion rate, as presented in (Figure 3(d)). After capturing all

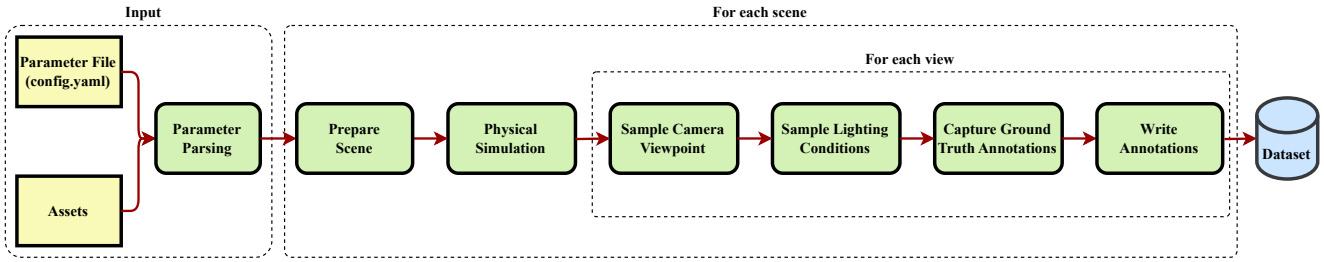


Figure 2. High-level overview of synthetic data generation pipeline.

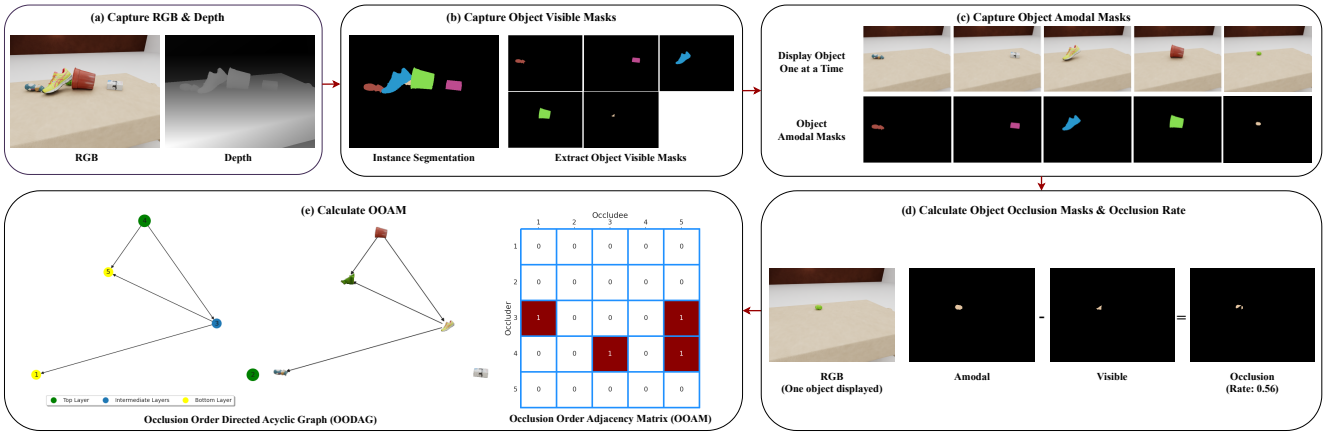


Figure 3. The process of capturing annotations for a scene. For each viewpoint, (a) RGB and depth with all objects (b) object visible masks & bounding box, (c) object amodal masks (including object amodal RGBA instances), (d) object occlusion masks and occlusion rate, (e) occlusion order adjacency matrix are captured.

object masks, we use Algorithm 1 to generate the Occlusion Order Adjacency Matrix (OOAM) for this viewpoint (Figure 3(e)). For a scene with M objects, the OOAM contains $M \times M$ elements, where the element (i, j) is a binary value in the matrix that indicates whether the object i occludes the object j. Given the OOAM, we can easily construct the Occlusion Order Directed Graph (OODG) to visualize the occlusion order in the viewpoint (Figure 3(e)). We provide a detailed explanation of the OODG in Section 11 of our supplementary materials. After that, the visibility of all objects is enabled to prepare for the capturing of annotations from the next viewpoint of the scene.

Algorithm 1 A function to generate the OOAM of objects in a viewpoint.

Input: Arrays of *visibleMasks* and *occlusionMasks* of objects in a scene
Output: The OOAM of objects in a viewpoint

```
function GENERATE_OOAM(visibleMasks, occlusionMasks)
    Initialize OOAM as matrix of zeros
    for each object i in length(VisibleMasks) do
        for each object j in length(OcclusionMasks) do
            if (i != j): then
                intersect = sum(visibleMasks[i] ∩ occlusionMasks[j])
                if (intersect > 0) : then
                    OOAM[i][j] = 1
    return OOAM
Note: object i occludes object j if OOAM[i][j] = 1
```

4. Dataset Details

To demonstrate the capabilities of SynTable, we generated a sample synthetic dataset of cluttered tabletop scenes, SynTable-Sim, using our pipeline, to train and evaluate UOAIS models. Note that users can also generate other custom datasets that meet the specific requirements of their application using the SynTable pipeline.

4.1. Object Models Used in Generating SynTable-Sim

We use 1075 object CAD models from the Google Scanned Objects dataset [8] and the Benchmark for the 6D Object Pose Estimation (BOP) [13] to generate our train dataset. The Google Scanned Objects dataset features more than 1030 photorealistic 3D scanned household objects with real-life dimensions, and BOP features 3D object models from household and industrial objects. Upon inspection of the Google Scanned Objects dataset, we filter out invalid objects that contain more than two instances in each model and keep the remaining 891 valid objects for our training dataset. From the BOP, we exclude 21 objects from the YCB-Video dataset that we include in our validation dataset and use the remaining 184 objects for our training dataset. We also create a synthetic validation set using 78 novel ob-

Table 1. A comparison of publicly available unseen object instance segmentation datasets for cluttered tabletop scenes. # indicates the number of items. **VI**: Visible Instances. **OI**: Occluded Instances. **Avg. OR %**: Avg. Occlusion Rate %, i.e., the fraction of occluded pixels to amodal pixels across all object instances in the dataset. **AM**: Availability of amodal masks. **OM**: Availability of occlusion masks. **Order**: Availability of occlusion order relation information between objects. **R/S**: Real or Synthetic. - indicates that the data was not available in the literature. * indicates that the values were not provided in the original literature, but we were able to compute the values.

Dataset	#Images	#Objects	#Scenes	#VI	#OI	Avg. OR (%)	AM	OM	Order	R/S
OCID [25]	2,390	89	96	19,097*	-	-	✗	✗	✗	R
OSD [24]	111	-	111	474*	-	-	✗	✗	✗	R
OSD-Amodal [2]	111	-	111	474*	237*	**24.11***	✓	✓	✗	R
UOAIS-Sim (Tabletop) [2]	25,000	375	500	356,885*	127,129*	11.16*	✓	✓	✗	S
SynTable-Sim (Ours)	**50,000**	**1075**	**1000**	**744,454**	**482,921**	17.56	✓	✓	✓	S

jects from the YCB dataset [4]. We sample a table object from 10 Omniverse Nucleus table assets to provide randomization for each scene. To load the 3D object models into Isaac Sim, we converted the OBJ and texture files to the Universal Scene Description (USD) format.

4.2. Dataset Configuration

With 50 viewpoints for each scene, we generated 900 scenes to create 45,000 RGB-D images for the training dataset and 100 scenes to create 5,000 RGB-D images for the validation dataset. $N_{lower} = 1$ to $N_{upper} = 40$ objects are rendered in randomly textured tabletop planes in each scene. We used 130 materials from Omniverse Nucleus material assets to be randomly applied on the walls, floor, and table for domain randomization purposes. $L_{lower} = 0$ to $L_{upper} = 2$ spherical lights are sampled for each scene. The viewpoint and lighting hemisphere parameters are automatically sampled based on the table dimensions. The camera parameters used are horizontal aperture: 2.63, vertical aperture: 1.96, and focal length: 1.88 to mimic the configuration of the RealSense LiDAR Camera L515. The rest of the parameters follow the default configurations of the pipeline.

4.3. Syntable-Sim Versus Other Cluttered Tabletop Datasets

We compare our SynTable-Sim dataset with several existing cluttered tabletop datasets in Table 1. Our tabletop dataset is the only one that provides complete annotations for all aspects of amodal instance segmentation. Furthermore, our dataset contains the most extensive variety of objects, the highest number of occlusion instances, and the second highest average occlusion rate — critical factors that significantly enhance the complexity and realism of training scenarios. These characteristics make our dataset very challenging for amodal instance segmentation tasks.

Additionally, SynTable-Sim exhibits a significantly higher proportion of heavily occluded objects in its training set compared to UOAIS-Sim, aligning more closely with the OSD-Amodal dataset, as shown in Figure 7 in the supplementary materials. This high occlusion density ensures that models trained on our dataset generalize better to real-world cluttered environments. Moreover, the weakly connected component size, which quantifies the number of mutually overlapped regions per OODG and serves as a metric for scene complexity [15], is consistently larger in SynTable-Sim compared to UOAIS-Sim (Figure 8 in the supplementary materials). This indicates that our dataset presents significantly more intricate occlusion patterns, enabling amodal segmentation models to learn more robust occlusion reasoning capabilities.

5. Experiments

In this section, we present the results of our experiments aimed at evaluating the effectiveness of our dataset generation pipeline in producing synthetic datasets with good Sim-to-Real transfer performance. We used our SynTable-Sim sample dataset to train a state-of-the-art (SOTA) UOAIS model, UOAIS-Net [2]. UOAIS-Net is evaluated on the SynTable-Sim validation set and the OSD-Amodal [2] test set. To verify consistency of our results and further demonstrate the capability of SynTable to improve the performance of a variety of different UOAIS models, we also train and evaluate three other UOAIS models—Amodal MRCNN [11], ORCNN [11], ASN [21]—on the SynTable-Sim and OSD-Amodal datasets respectively.

5.1. Training Strategy

We train UOAIS-Net on the UOAIS-Sim tabletop and SynTable-Sim datasets using an NVIDIA Tesla V100 GPU with 16 GB of memory. For both datasets, we used 90% of the images for training and 10% for validation. To train UOAIS-Net using the UOAIS-Sim tabletop dataset, we use the same hyperparameters as Back *et al.* [2]. To train UOAIS-Net with SynTable-Sim, we modified the depth range hyperparameter, which is used to preprocess input depth images. Specifically, we changed the range from the 2500 mm to 40000 mm range set by Back *et al.* to a nar-

rower range of 250 mm to 2500 mm. This adjustment is required because our dataset reflects real-world proportions and has a smaller depth range than the UOAIS-Sim dataset. We also use a similar training strategy to train Amodal MRCNN, ORCNN, and ASN.

5.2. Evaluation Metrics

We measure the performance of UOAIS-Net on the following traditional metrics [5, 19, 29]: Overlap P/R/F, Boundary P/R/F, and *F@.75* for the amodal, visible, and invisible masks. Overlap P/R/F and Boundary P/R/F evaluate the whole area and the sharpness of the prediction, respectively, where P, R, and F are the precision, recall, and F-measure of instance masks after the Hungarian matching, respectively. *F@.75* is the percentage of segmented objects with an Overlap F-measure greater than 0.75. We also report the accuracy ($ACC_{\mathscr{O}}$) and F-measure ($F_{\mathscr{O}}$) of occlusion classification, where $ACC_{\mathscr{O}} = \frac{\delta}{\alpha}$, $F_{\mathscr{O}} = \frac{2P_oR_o}{P_o+R_o}$, $P_o = \frac{\delta}{\beta}$, $R_o = \frac{\delta}{\gamma}$. α is the number of the matched instances after the Hungarian matching. β, γ, and δ are the number of occlusion predictions, ground truths, and correct predictions, respectively. We provide more details about the evaluation metrics in Section 9 of our supplementary materials.

Due to the subjectivity of the invisible masks of objects, the evaluation of the performance of the UOAIS model solely based on the overlap and boundary *P/R/F* of segmented objects may be inaccurate. The current UOAIS occlusion evaluation metrics measure how well the model can predict whether individual objects are occluded. However, these metrics neglect hierarchical occlusion relationships, which are crucial for systems requiring structured scene understanding. The Occlusion Order Adjacency Matrix (OOAM) encodes these relationships, and the derived Occlusion Order Directed Graph (OODG) enables applications such as sequencing interactions in cluttered environments (for example, retrieving obscured items) or rendering occluded objects in augmented reality. To quantify a model's ability to infer occlusion hierarchies, we propose the Occlusion Order Accuracy Occlusion Order Accuracy (ACC_{OO}) metric as defined in Equation 1.

$$ACC_{OO} = \frac{sum(similarityMatrix) - gtOOAMDiagonalSize}{gtOOAMSize - gtOOAMDiagonalSize} \tag{1}$$

In Equation 1, *similarityMatrix* is the element-wise equality comparison between the ground truth OOAM, *gtOOAM*, and the predicted OOAM, *predOOAM*. As an object cannot occlude itself, the diagonal of any OOAM is always 0. Thus, we subtract the number of elements along the diagonal of *gtOOAM*, *gtOOAMDiagonalSize*, from the calculation of ACC_{OO}. ACC_{OO} is used to evaluate the model's ability to accurately determine the order of occlusions in a clutter of objects by comparing the OOAM generated by the model to the ground truth OOAM using Algorithm 2. We give a specific example of how to compute ACC_{OO} in Sections 10 and 11 of our supplementary materials.

Algorithm 2 Evaluating Occlusion Ordering Accuracy

Input: The arrays of the ground truth and predicted visible and occlusion masks (*gtVisible*, *gtOcclusion*, *predVisible*, *predOcclusion*)
Output: Scene occlusion order accuracy ACC_{oo}

```
gtOOAM = GENERATE_OOAM(gtVisible, gtOcclusion)
Get groundtruth-prediction assignment pairs after Hungarian matching
Extract predVisible and predOcclusion masks from assignment pairs
predOOAM = GENERATE_OOAM(predVisible, predOcclusion)
similarityMatrix = (predOOAM == gtOOAM)        ▷ Compare the similarity between the predicted and ground truth OOAMs
Calculate ACC_oo using Equation 1
```

5.3. Results

Table 2 compares the performance of UOAIS-Net on the OSD-Amodal dataset after training on the UOAIS-Sim tabletop dataset and our SynTable-Sim sample dataset. We conducted four sets of experiments. In each set of experiments, we vary the amount of data augmentation used and the size of the dataset we use for training.

In our first set of experiments, we can see that the UOAIS-Net trained on the SynTable-Sim dataset significantly outperforms the UOAIS-Net trained on the UOAIS-Sim tabletop dataset in all metrics. Even when we train UOAIS-Net using a dataset of the same size as UOAIS-Sim (SynTable-Sim-0.5X), the performance is still remarkably better than the UOAIS-Net trained on the UOAIS-Sim tabletop dataset across all metrics. A detailed breakdown of the precision P, recall R, and F-measure F, and *F@.75* scores for the amodal, invisible and visible masks for our first set of experiments is shown in Table 3. We observe that except for the Boundary precision scores of the invisible masks, UOAIS-Net achieves substantial improvements in all other metrics.

In the next three sets of experiments, we observe that even when we include data augmentation, the performance of UOAIS-Net trained on the UOAIS-Sim tabletop dataset is still worse than that trained on the SynTable-Sim dataset without using any data augmentation. We also provide images of the inference results on the OSD-Amodal dataset in Section 12 of our supplementary materials.

Similarly, from Table 4, the UOAIS-Net model trained on the SynTable-Sim dataset outperforms the one trained on UOAIS-Sim tabletop dataset in all metrics when both models are benchmarked on SynTable-Sim validation dataset.

We evaluated the effectiveness of SynTable-Sim across different UOAIS models comprising distinct architectures. Table 5 compares the performance of UOAIS models—Amodal MRCNN, ORCNN, ASN, and UOAIS-Net—on the OSD-Amodal dataset after training on the UOAIS-Sim tabletop dataset and our SynTable-Sim sample dataset. For each model result in our experiments, we used seed 7 for training. Generally, across most metrics, the UOAIS mod-

Table 2. The performance of UOAIS-Net on the **OSD-Amodal dataset** after training on the UOAIS-Sim and SynTable-Sim datasets. UOAIS-Net is trained with RGB-D images. **CR**: Crop Ratio lower bound. **HF**: Horizontal Flip. **CA**: Colour Augmentation. **PD**: Perlin Distortion. **OV**: Overlap F-measure, **BO**: Boundary F-measure, **F@.75**: Percentage of segmented objects with an Overlap F-measure greater than 0.75, $\boldsymbol{F}_{\emptyset}$: Occlusion F-Measure, $\mathbf{ACC_{OO}}$: Occlusion Order Accuracy

No.	Training Set	Augmentation				Amodal Mask			Invisible Mask			Occlusion		Visible Mask			ACC_{OO}
		CR	HF	CA	PD	OV	BO	*F@.75*	OV	BO	*F@.75*	$F_{\emptyset}$	$ACC_{\emptyset}$	OV	BO	*F@.75*	
1	UOAIS-Sim (Tabletop)	✗	✗	✗	✗	42.4	34.1	47.1	21.6	15.2	18.5	43.1	61.8	42.5	32.3	37.1	12.7
1	SynTable-Sim (Ours)	✗	✗	✗	✗	**80.9**	61.8	**78.1**	**52.4**	**31.2**	41.3	**75.7**	**86.7**	**81.1**	64.3	**74.4**	**82.9**
1	SynTable-Sim-0.5X (Ours)	✗	✗	✗	✗	80.7	**63.8**	77.3	51.9	30.2	**42.9**	**75.7**	84.1	80.5	**65.4**	71.7	82.7
2	UOAIS-Sim (Tabletop)	0.8	✓	✗	✗	26.1	33.1	66.7	15.5	7.7	20.4	60.8	78.1	25.9	27.6	51.8	42.7
2	SynTable-Sim (Ours)	0.8	✓	✗	✗	67.7	56.0	81.2	49.4	30.1	**48.6**	72.5	89.8	71.8	61.3	78.2	86.6
2	SynTable-Sim-0.5X (Ours)	0.8	✓	✗	✗	**75.6**	**61.2**	**83.5**	**53.6**	**31.2**	48.5	**75.5**	**90.1**	**76.8**	**64.5**	**78.3**	**87.0**
3	UOAIS-Sim (Tabletop)	0.8	✓	✓	✓	71.8	**62.8**	81.4	**55.6**	**31.3**	**44.6**	**75.1**	86.2	70.2	**63.2**	73.2	79.6
3	SynTable-Sim (Ours)	0.8	✓	✓	✓	**78.3**	58.8	81.9	54.0	29.7	43.9	66.6	93.2	**79.2**	60.4	77.2	**87.7**
3	SynTable-Sim-0.5X (Ours)	0.8	✓	✓	✓	74.0	57.5	**83.3**	49.2	23.9	41.0	65.7	**93.4**	74.2	59.2	**79.2**	87.6
4	UOAIS-Sim (Tabletop)	0.5	✓	✓	✓	49.0	50.3	82.7	42.3	23.9	40.3	**68.9**	84.0	47.3	50.0	70.6	80.4
4	SynTable-Sim (Ours)	0.5	✓	✓	✓	**64.4**	**51.5**	84.3	**47.3**	**24.2**	47.4	60.0	**91.9**	**65.3**	**53.7**	**78.2**	87.0
4	SynTable-Sim-0.5X (Ours)	0.5	✓	✓	✓	55.0	47.2	**85.9**	43.2	22.0	**48.4**	55.4	91.5	55.3	46.6	76.9	**87.8**

Table 3. A breakdown of the evaluation results of UOAIS-Net on the **OSD-Amodal dataset** for the first set of experiments after training on the UOAIS-Sim and SynTable-Sim dataset. **P**: Precision, **R**: Recall, **F**: F-measure, **F@.75**: Percentage of segmented objects with an Overlap F-measure greater than 0.75, $\boldsymbol{F}_{\emptyset}$: Occlusion F-Measure, $\mathbf{ACC_{OO}}$: Occlusion Order Accuracy

Training Set	Amodal Mask							Invisible Mask							Visible Mask							Occlusion		ACC_{OO}
	Overlap			Boundary			*F@.75*	Overlap			Boundary			*F@.75*	Overlap			Boundary			*F@.75*	$F_{\emptyset}$	$ACC_{\emptyset}$	
	P	R	F	P	R	F		P	R	F	P	R	F		P	R	F	P	R	F				
UOAIS-Sim (Tabletop)	35.9	65.4	42.4	31.4	42.8	34.1	47.1	55.9	24.5	21.6	**45.3**	19.3	15.2	18.5	36.2	61.3	42.5	30.8	39.2	32.3	37.1	43.1	61.8	12.7
SynTable-Sim (Ours)	**81.0**	**82.5**	**80.9**	**59.1**	**66.8**	**61.8**	**78.1**	**69.3**	**51.8**	**52.4**	34.6	**42.6**	**31.2**	**41.3**	**80.1**	**83.2**	**81.1**	**62.4**	**68.1**	**64.3**	**74.4**	**75.7**	**86.7**	**82.9**

Table 4. The performance of UOAIS-Net on the **SynTable-Sim validation** dataset after training on the UOAIS-Sim and SynTable-Sim datasets. UOAIS-Net is trained with RGB-D images. **OV**: Overlap F-measure, **BO**: Boundary F-measure, **F@.75**: Percentage of segmented objects with an Overlap F-measure greater than 0.75, $\boldsymbol{F}_{\emptyset}$: Occlusion F-Measure, $\mathbf{ACC_{OO}}$: Occlusion Order Accuracy

Training Set	Amodal Mask			Invisible Mask			Occlusion		Visible Mask			ACC_{OO}
	OV	BO	*F@.75*	OV	BO	*F@.75*	$F_{\emptyset}$	$ACC_{\emptyset}$	OV	BO	*F@.75*	
UOAIS-Sim (Tabletop)	38.0	37.8	35.9	14.1	12.9	7.6	47.2	72.9	40.4	38.9	34.8	31.6
SynTable-Sim (Ours)	**84.5**	**78.4**	**75.6**	**41.4**	**37.7**	**21.5**	**76.1**	**82.4**	**86.8**	**81.8**	**74.4**	**77.5**

els trained on SynTable-Sim outperform the same models trained on the UOAIS-Sim tabletop dataset. There is also a significant improvement in the results of ACC_{oo} for Amodal MRCNN, ORCNN, and ASN when trained on our SynTable-Sim as compared to the UOAIS-Sim tabletop dataset. This is consistent with the performance trend observed for UOAIS-Net and, therefore, demonstrates that SynTable is an effective tool for generating high-quality datasets that can improve the performance of UOAIS models. A detailed breakdown of the precision P, recall R, and F-measure F, and *F@.75* scores for the amodal, invisible, and visible masks are shown in Table 6.

As shown in Table 7, the UOAIS models trained on the SynTable-Sim dataset outperform the same models trained on the UOAIS-Sim tabletop dataset in all metrics when they are benchmarked on the SynTable-Sim validation dataset.

Our experiments demonstrate the effectiveness of our proposed dataset generation pipeline, SynTable, in improving the Sim-to-Real transfer performance of SOTA deep learning computer vision models for UOAIS. These results highlight the potential of SynTable for addressing the challenge of annotating amodal instance segmentation masks.

6. Conclusion

In conclusion, we present SynTable, a novel synthetic data generation pipeline for generating photorealistic datasets that facilitated amodal instance segmentation of cluttered tabletop scenes. SynTable enables the creation of complex 3D scenes with automatic annotation of diverse metadata, eliminating the need for manual labeling while ensuring dataset quality and accuracy. We demonstrate the effectiveness of the SynTable pipeline by generating a photorealistic amodal instance segmentation dataset and using it to

Table 5. The performance of Amodal MRCNN, ORCNN, ASN, and UOAIS-Net on the **OSD-Amodal dataset** after training on the UOAIS-Sim and SynTable-Sim datasets. UOAIS-Net is trained with RGB-D images. **OV**: Overlap F-measure, **BO**: Boundary F-measure, **F@.75**: Percentage of segmented objects with an Overlap F-measure greater than 0.75, **ACC$_{OO}$**: Occlusion Order Accuracy

Training Set	Method	Amodal Mask			Invisible Mask			Occlusion		Visible Mask			ACC_{OO}
		OV	BO	*F@.75*	OV	BO	*F@.75*	$F_{\mathcal{O}}$	$ACC_{\mathcal{O}}$	OV	BO	*F@.75*	
UOAIS-Sim (Tabletop)	Amodal MRCNN	36.7	26.9	45.7	8.8	4.8	7.7	39.2	54.8	38.7	26.3	32.2	15.6
	ORCNN	36.3	25.4	47.0	12.2	6.7	9.0	43.8	59.2	30.5	21.8	29.6	21.5
	ASN	40.5	33.6	49.8	17.4	12.1	15.0	47.0	63.2	39.3	31.6	36.8	17.8
	UOAIS-Net	49.0	50.3	82.7	42.3	23.9	40.3	68.9	84.0	47.3	50.0	70.6	80.4
SynTable-Sim (Ours)	Amodal MRCNN	74.5	57.5	77.2	41.3	23.5	37.6	69.3	79.4	73.8	57.7	66.1	79.2
	ORCNN	74.2	58.2	77.1	44.7	24.3	33.8	72.9	82.2	72.0	58.3	67.7	79.1
	ASN	78.2	60.2	75.3	46.4	27.7	35.8	72.6	83.0	78.1	61.8	68.9	80.2
	UOAIS-Net	64.4	51.5	84.3	47.3	24.2	47.4	60.0	91.9	65.3	53.7	78.2	87.0

Table 6. A breakdown of the precision, recall, and F-measure of the amodal, invisible, and visible mask predictions by Amodal MRCNN, ORCNN, ASN, and UOAIS-Net on the **OSD-Amodal dataset** after training on the UOAIS-Sim and SynTable-Sim dataset. **P**: Precision, **R**: Recall, **F**: F-measure

Training Set	Method	Amodal Mask							Invisible Mask							Visible Mask						
		Overlap			Boundary			*F@.75*	Overlap			Boundary			*F@.75*	Overlap			Boundary			*F@.75*
		P	R	F	P	R	F		P	R	F	P	R	F		P	R	F	P	R	F	
UOAIS-Sim (Tabletop)	Amodal MRCNN	27.9	66.7	36.7	22.5	39.8	26.9	45.7	20.2	24.9	8.8	16.4	19.9	4.8	7.7	30.1	60.5	38.7	22.0	37.8	26.3	32.2
	ORCNN	26.3	71.1	36.3	19.8	42.4	25.4	47.0	41.5	22.7	12.2	33.9	17.9	6.7	9.0	21.4	63.4	30.5	16.6	38.2	21.8	29.6
	ASN	31.7	67.8	40.5	28.6	45.4	33.6	49.8	47.6	23.4	17.4	38.8	20.2	12.1	15.0	32.0	63.5	39.3	28.2	41.5	31.6	36.8
	UOAIS-Net	37.2	85.5	49.0	41.1	71.3	50.3	82.7	50.9	54.0	42.3	24.8	41.1	23.9	40.3	35.4	81.6	47.3	41.3	69.3	50.0	70.6
SynTable-Sim (Ours)	Amodal MRCNN	72.3	81.6	74.5	54.6	64.5	57.5	77.2	54.9	48.0	41.3	30.3	38.8	23.5	37.6	72.1	78.4	73.8	55.1	63.7	57.7	66.1
	ORCNN	73.7	80.8	74.2	55.6	64.5	58.2	77.1	61.0	47.1	44.7	31.1	38.6	24.3	33.8	69.8	79.1	72.0	55.7	64.3	58.3	67.7
	ASN	78.2	80.3	78.2	57.8	64.9	60.2	75.3	65.2	46.2	46.4	32.9	38.7	27.7	35.8	77.5	80.1	78.1	60.4	65.4	61.8	68.9
	UOAIS-Net	53.9	86.3	64.4	40.9	74.6	51.5	84.3	53.0	60.0	47.3	20.5	48.3	24.2	47.4	55.0	86.2	65.3	43.2	75.3	53.7	78.2

Table 7. The performance of Amodal MRCNN, ORCNN, ASN, and UOAIS-Net on the **SynTable-Sim validation dataset** after training on the UOAIS-Sim and SynTable-Sim datasets. UOAIS-Net is trained with RGB-D images. **OV**: Overlap F-measure, **BO**: Boundary F-measure, **F@.75**: Percentage of segmented objects with an Overlap F-measure greater than 0.75, **ACC$_{OO}$**: Occlusion Order Accuracy

Training Set	Method	Amodal Mask			Invisible Mask			Occlusion		Visible Mask			ACC_{OO}
		OV	BO	*F@.75*	OV	BO	*F@.75*	$F_{\mathcal{O}}$	$ACC_{\mathcal{O}}$	OV	BO	*F@.75*	
UOAIS-Sim (Tabletop)	Amodal MRCNN	27.1	25.2	23.8	6.7	6.2	3.5	35.8	66.0	29.1	26.2	23.5	19.0
	ORCNN	30.9	29.0	28.1	12.5	11.4	8.0	39.9	68.4	31.8	30.2	27.3	23.1
	ASN	33.3	34.4	35.3	10.3	9.1	5.0	47.6	72.3	35.0	36.0	34.1	31.6
	UOAIS-Net	39.9	40.5	38.6	17.0	15.5	9.6	49.6	74.9	41.6	40.7	35.9	31.6
SynTable-Sim (Ours)	Amodal MRCNN	83.5	76.2	72.5	35.4	31.8	16.4	73.2	80.3	85.7	79.1	71.1	72.8
	ORCNN	83.4	76.0	72.2	34.4	29.3	15.3	67.2	73.7	85.3	78.9	70.9	73.0
	ASN	83.6	76.9	73.9	38.5	35.1	18.5	74.8	81.5	86.1	80.0	72.8	75.8
	UOAIS-Net	83.7	77.5	75.1	40.3	36.7	20.2	75.5	82.0	86.2	80.1	73.3	77.4

train UOAIS-Net. As a result, UOAIS-Net achieves significantly improved Sim-to-Real transfer performance on the OSD-Amodal dataset, particularly in determining the object occlusion order of objects in a cluttered tabletop scene. SynTable advances amodal segmentation for systems that require occlusion-aware perception, such as robotics, augmented reality. By automating annotation of amodal masks and appearance via photorealistic rendering, and scene occlusion order, our pipeline addresses a key bottleneck in training robust vision models with amodal perception capabilities.

References

[1] Jiayang Ao, Krista A Ehinger, and Qiuhong Ke. Image amodal completion: A survey. *arXiv preprint arXiv:2207.02062*, 2022. 1

[2] Seunghyeok Back, Joosoon Lee, Taewon Kim, Sangjun Noh, Raeyoung Kang, Seongho Bak, and Kyoobin Lee. Unseen object amodal instance segmentation via hierarchical occlusion modeling. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 5085–5092. IEEE, 2022. 2, 3, 5, 6

[3] Shehan Caldera, Alexander Rassau, and Douglas Chai. Review of deep learning methods in robotic grasp detection. *Multimodal Technologies and Interaction*, 2(3), 2018. 2

[4] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 510–517, 2015. 5

[5] Achal Dave, Pavel Tokmakov, and Deva Ramanan. Towards segmenting anything that moves. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 1493–1502, 2019. 6, 3

[6] Amaury Depierre, Emmanuel Dellandréa, and Liming Chen. Jacquard: A large scale dataset for robotic grasp detection. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3511–3516, 2018. 2

[7] Helisa Dhamo, Nassir Navab, and Federico Tombari. Object-driven multi-layer scene decomposition from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 2

[8] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. *arXiv preprint arXiv:2204.11918*, 2022. 4

[9] Kiana Ehsani, Roozbeh Mottaghi, and Ali Farhadi. Segan: Segmenting and generating the invisible. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[10] Hao-Shu Fang, Chenxi Wang, Minghao Gou, and Cewu Lu. Graspnet-1billion: A large-scale benchmark for general object grasping. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11441–11450, 2020. 2

[11] Patrick Follmann, Rebecca König, Philipp Härtinger, Michael Klostermann, and Tobias Böttger. Learning to see the invisible: End-to-end trainable amodal instance segmentation. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1328–1336, 2019. 5

[12] Maximilian Gilles, Yuhao Chen, Tim Robin Winter, E. Zhixuan Zeng, and Alexander Wong. Metagraspnet: A large-scale benchmark dataset for scene-aware ambidextrous bin picking via physics-based metaverse synthesis. In *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, pages 220–227, 2022. 2

[13] Tomáš Hodaň, Frank Michel, Eric Brachmann, Wadim Kehl, Anders Glent Buch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, Caner Sahin, Fabian Manhardt, Federico Tombari, Tae-Kyun Kim, Jiří Matas, and Carsten Rother. BOP: Benchmark for 6D object pose estimation. *European Conference on Computer Vision (ECCV)*, 2018. 4

[14] Yun Jiang, Stephen Moseson, and Ashutosh Saxena. Efficient grasping from rgbd images: Learning using a new rectangle representation. In *2011 IEEE International Conference on Robotics and Automation*, pages 3304–3311, 2011. 2

[15] Hyunmin Lee and Jaesik Park. Instance-wise Occlusion and Depth Orders in Natural Scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 5

[16] Ke Li and Jitendra Malik. Amodal instance segmentation. In *Computer Vision – ECCV 2016*, pages 677–693, Cham, 2016. Springer International Publishing. 1

[17] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In *Robotics: Science and Systems (RSS)*, 2017. 2

[18] Gilles Maximilian, Yuhao Chen, Tim Robin Winter, E. Zhixuan Zeng, and Alexander Wong. MetaGraspNet: A large-scale benchmark dataset for scene-aware ambidextrous bin picking via physics-based metaverse synthesis. In *IEEE International Conference on Automation Science and Engineering (CASE)*, 2022. 3, 1

[19] Peter Ochs, Jitendra Malik, and Thomas Brox. Segmentation of moving objects by long term video analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6): 1187–1200, 2014. 6, 3

[20] 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 *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 724–732, 2016. 3

[21] Lu Qi, Li Jiang, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Amodal instance segmentation with kins dataset. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3009–3018, 2019. 5

[22] Lu Qi, Li Jiang, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Amodal instance segmentation with kins dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[23] Weichao Qiu and Alan Yuille. Unrealcv: Connecting computer vision to unreal engine. In *Computer Vision – ECCV 2016 Workshops*, pages 909–916, Cham, 2016. Springer International Publishing. 3

[24] Andreas Richtsfeld, Thomas Mörwald, Johann Prankl, Michael Zillich, and Markus Vincze. Segmentation of unknown objects in indoor environments. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4791–4796, 2012. 2, 5

[25] Markus Suchi, Timothy Patten, David Fischinger, and Markus Vincze. Easylabel: A semi-automatic pixel-wise object annotation tool for creating robotic rgb-d datasets. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6678–6684, 2019. 2, 5

[26] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on*

Intelligent Robots and Systems (IROS), pages 23–30, 2017. 3

[27] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. 2

[28] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. Sapien: A simulated part-based interactive environment. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11094–11104, 2020. 2

[29] Christopher Xie, Yu Xiang, Arsalan Mousavian, and Dieter Fox. The best of both modes: Separately leveraging rgb and depth for unseen object instance segmentation. In *Proceedings of the Conference on Robot Learning*, pages 1369–1378. PMLR, 2020. 2, 6, 3

[30] Xinchen Yan, Jasmined Hsu, Mohammad Khansari, Yunfei Bai, Arkanath Pathak, Abhinav Gupta, James Davidson, and Honglak Lee. Learning 6-dof grasping interaction via deep geometry-aware 3d representations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3766–3773, 2018. 2

[31] Chuanxia Zheng, Duy-Son Dao, Guoxian Song, Tat-Jen Cham, and Jianfei Cai. Visiting the invisible: Layer-by-layer completed scene decomposition. *International Journal of Computer Vision*, 2021. 2

[32] Yan Zhu, Yuandong Tian, Dimitris Metaxas, and Piotr Dollar. Semantic amodal segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

SynTable: A Synthetic Data Generation Pipeline for Unseen Object Amodal Instance Segmentation of Cluttered Tabletop Scenes

Supplementary Material

7. Overview

This supplementary material offers dataset visualization, qualitative results, and additional technical details to support the main paper. Section 9 provides a comprehensive elaboration of the evaluation metrics employed. Section 10 illustrates how occlusion order accuracy is calculated and the validity of the metric. Furthermore, Section 11 delineates the process of generating an occlusion order directed acyclic graph from the occlusion order adjacency matrix to classify objects in three distinct order layers. Lastly, Section 12 showcases some qualitative inference results of UOAIS-Net on the OSD-Amodal dataset.

8. Additional Details About the Dataset Generation Process

8.1. Preparing Each Scene

The method to prepare each scene is shown in Figure 4. A table is randomly sampled from the assets in Omniverse Nucleus and is rendered at the center of a room. The texture and materials of the table, ceiling, wall, and floor are randomized for every scene to ensure domain randomization. The objects are added to the scene with randomized x, y, and z coordinates and orientations. We randomly sample (with replacement) N_{lower} to N_{upper} objects to render for each scene. By default, $N_{lower} = 1$, $N_{upper} = 40$. Each object is initialized with real-life dimensions, randomized rotations and coordinates, allowing for diverse object arrangements across scenes. Each object also has mass and collision properties so that they can be dropped onto the tabletop in our physics simulation.

8.2. Physical Simulation of Each Scene

Upon completing the scene preparation, the rendered objects are dropped onto the table surface using a physics simulation. The simulation is paused after t seconds ($t = 5$ by default), halting any further movement of the objects. During the simulation, any objects that rebound off the tabletop surface and fall outside the spatial coordinate region of the tabletop surface (i.e., either below the table or beyond the width and length of the table) are automatically removed. This is necessary to prevent the inclusion of extraneous and irrelevant objects outside the specified tabletop region during the annotation process from different viewpoints.

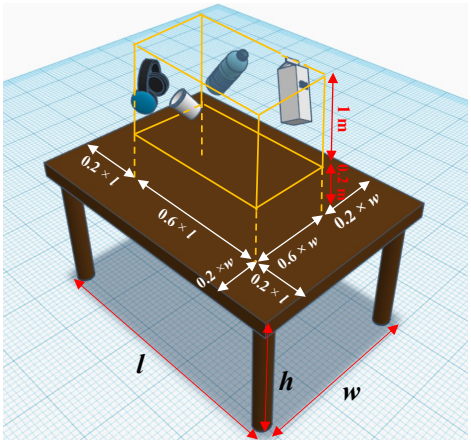


Figure 4. Initialization of objects with randomized coordinates and rotations. The initial position of the objects in the scene is randomized but constrained to be within the dimensions of the 3D orange box. The orange box is 0.2 m above the tabletop. The roll, pitch, and yaw of each object are also randomly sampled within the range of 0° to 360°.

8.3. Sampling of Camera Viewpoints

To capture annotations for each scene from multiple viewpoints, we enhance the approach by Gilles *et al.* [18]—which only uses fixed viewpoint positions—by introducing a feature that captures V number of viewpoints at random positions within two concentric hemispheres, as illustrated in Figure 5. V can be set by the user. The radii of the two concentric hemispheres are uniformly sampled within the range r_{view_lower} m to r_{view_upper} m, where r_{view_lower} and r_{view_upper} are defined in Equations 2 and 3. Users may also set fixed values for r_{view_lower} and r_{view_upper} should they wish to do so.

$$r_{view\_lower} = max\left(\frac{w}{2}, \frac{l}{2}\right) \tag{2}$$

$$r_{view\_upper} = 1.7 \times r_{view\_lower} \tag{3}$$

The hemisphere's spherical coordinates are parameterized using three variables r_{view}, u, and v. To generate the camera coordinates in the world frame, we first obtain the radius of the hemisphere r_{view} by uniform sampling between r_{view_lower} and r_{view_upper}. Next, we uniformly sample $u, v \in [0,1]$, then substitute all the sampled values into Equations 4, 5 and 6 to compute the cartesian coordinates of the camera.

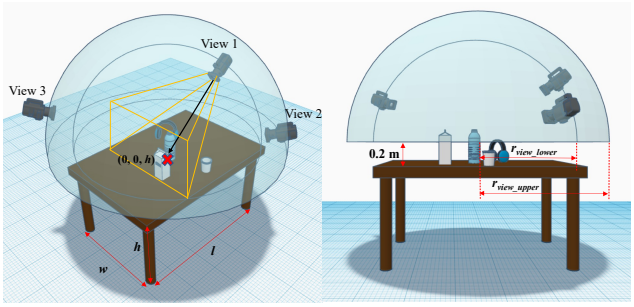


Figure 5. Sampling of camera viewpoints within concentric hemispheres (shown in blue). The two concentric hemispheres' origins are centered at the tabletop surface's center coordinate with an offset of 0.2 m in the positive z direction in the world frame. This allows the camera viewpoints to minimally have a direct line of sight to the tabletop surface to capture part of the tabletop plane. This figure is best viewed zoomed in.

$$x = r_{view} \sin(\arccos(1-v)) \cos(2\pi u) \tag{4}$$

$$y = r_{view} \sin(\arccos(1-v)) \sin(2\pi u) \tag{5}$$

$$z = r_{view} \cos(\arccos(1-v)) \tag{6}$$

Once the camera coordinates are set, the orientation of each camera is set such that each viewpoint looks directly at the center of the tabletop surface $(0, 0, h)$.

8.4. Sampling of Lighting Conditions

To simulate different indoor lighting conditions, we resample L spherical light sources between L_{lower} to L_{upper} for each viewpoint (Figure 6). By default, we set L_{lower} and L_{upper} to be 0 and 2, respectively. To position L spherical light sources for a viewpoint, we adopt a similar approach to the camera viewpoint sampling method discussed in Section 8.3. In contrast to the approach by Back *et al.* [2], we use spherical light sources that emit light in all directions. Furthermore, we uniformly sample light source temperatures between 2,000 K to 6,500 K. The default light intensity of each light source is uniformly sampled between 100 lx to 20,000 lx, and the default light intensity of ceiling lights in the scene is also sampled uniformly between 100 lx to 2,000 lx. To achieve diverse indoor lighting conditions for tabletop scenes, users have the flexibility to adjust the number of spherical light sources, as well as their intensities and temperatures.

Similar to the sampling method for the camera viewpoint coordinates, we have designed a feature that samples the lower and upper radii bounds for the light sources based on the camera hemisphere's upper bound radius, r_{view_upper}. The sampled lower and upper bound radii constraints for the lighting hemisphere r_{light_lower} and r_{light_upper} are as follows:

$$r_{light\_lower} = r_{view\_upper} + 0.1m \tag{7}$$

$$r_{light\_upper} = r_{light\_lower} + 1m \tag{8}$$

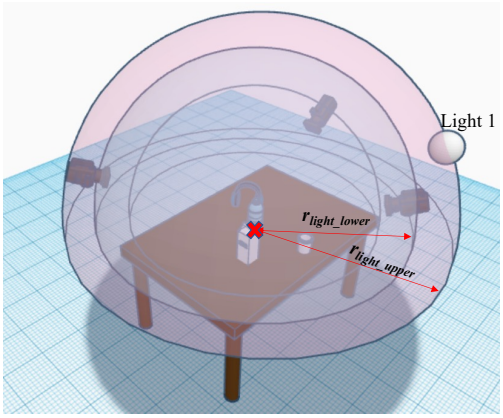


Figure 6. Sampling of lighting within concentric hemispheres (shown in pink). Each spherical light source lies within the constraints of two concentric hemispheres of arbitrary radius between r_{light_lower} to r_{light_upper}. Note that the radii constraints for the spherical light source concentric hemispheres are larger than those for the camera viewpoints' and are customizable by the user.

8.5. Capturing of Ground Truth Annotations

The process of capturing the ground truth annotations for a scene is illustrated in Figure 3. At each view, the RGB and depth images of the tabletop scene will be captured (Figure 3(a)). The built-in instance segmentation function in Isaac Sim Replicator Composer is employed to capture the instance segmentation mask of the entire scene from a viewpoint (Figure 3(b)). Subsequently, each object's visible mask is cropped from the instance segmentation mask of the scene. To obtain the amodal mask of each object on the simulated tabletop scene, we have developed the subsequent steps.

Initially, all objects' visibility is disabled. For each object o within the scene, its visibility is enabled, and the instance segmentation function is utilized to capture its amodal mask (Figure 3(c)). Following this, we compute the object's occlusion mask and occlusion rate, as presented in (Figure 3(d)). The occlusion mask of an object o can be acquired by subtracting its visible mask from its amodal mask.

The occlusion rate of the object o can be computed by dividing the number of pixels in the occlusion mask by the number of pixels in the amodal mask. If the occlusion rate of the object o is equal to 1, it implies that object o is completely obscured from the viewpoint, thus we do not save the object o's annotation for this view. The visibility of object o is then disabled to capture the masks of the next object. Following the preservation of all objects' masks, we use Algorithm 1 to generate the Occlusion Order Adjacency Matrix (OOAM) for this viewpoint (Figure 3(e)). For a scene with M objects, the OOAM contains $M \times M$ elements, where the element (i, j) is a binary value in the matrix which indicates whether object i occludes object j. Given the OOAM, we can easily construct the Occlusion

Order Directed Graph (OODG) to visualize the occlusion order in the viewpoint (Figure 3(e)). We provide a detailed explanation of the OODG in our supplementary materials. After that, the visibility of all objects is enabled to prepare for the capturing of annotations from the next viewpoint of the scene.

8.6. Saving of Ground Truth Annotations

We saved the RGB and depth images as PNG images. The OOAM of the objects in each image is saved as a NumPy file. The amodal, visible, and occlusion masks are saved as Run-length Encoding (RLE) in COCO JSON format to optimize disk space used by the generated datasets. We also recorded each object's visible bounding box, image ID, and object name in the generated COCO JSON file.

9. Details about Evaluation Metrics

In this paper, we employ the precision/recall/F-measure (P/R/F) metrics, as defined in [5, 19, 29]. This metric favors methods that accurately segment the desired objects while penalizing those that produce false positives. Specifically, the precision, recall, and F-measure are calculated between all pairs of predicted and ground truth objects. The Hungarian method, employing pairwise F-measure, is utilized to establish a match between predicted objects and ground truth. Given this matching, the Overlap P/R/F is computed by:

$$P = \frac{\sum_i |c_i \cap g(c_i)|}{\sum_i |c_i|}, \quad R = \frac{\sum_i |c_i \cap g(c_i)|}{\sum_j |g_j|} \tag{9}$$

$$F = \frac{2PR}{P+R} \tag{10}$$

where c_i denotes the set of pixels belonging to predicted object i, $g(c_i)$ is the set of pixels of the matched ground truth object of c_i after Hungarian matching, and g_j is the set of pixels for ground truth object j.

Although the aforementioned metric provides valuable information, it fails to consider the boundaries of the objects. Therefore, Xie *et al*. [29] proposed the Boundary P/R/F measure to supplement the Overlap P/R/F. The calculation of Boundary P/R/F involves the same Hungarian matching as used in the computation of Overlap P/R/F. Given these matchings, the Boundary P/R/F is computed by:

$$P = \frac{\sum_i |c_i \cap D[g(c_i)]|}{\sum_i |c_i|}, \quad R = \frac{\sum_i |D[c_i] \cap g(c_i)|}{\sum_j |g_j|} \tag{11}$$

$$F = \frac{2PR}{P+R} \tag{12}$$

Here, overloaded notations are used to represent the sets of pixels belonging to the boundaries of the predicted object i and the ground truth object j as c_i and g_j, respectively. The dilation operation is denoted by $D[\cdot]$, which allows for some tolerance in the prediction. The metrics we use are a combination of the F-measure described in [20] and the Overlap P/R/F as defined in [5].

In our work, we use the Overlap and Boundary P/R/F evaluation metrics to evaluate the accuracy of the predicted visible, invisible, and amodal masks. In the context of the Overlap P/R/F metrics, c_i denotes the set of pixels belonging to the predicted visible, invisible, and amodal masks, $g(c_i)$ denotes the set of pixels belonging to the matched ground-truth visible, invisible and amodal masks annotations, and g_j is the ground-truth visible, invisible and amodal mask. The meaning of c_i, $g(c_i)$, and g_j are similar in the context of the Boundary P/R/F metrics.

An additional vital evaluation metric used in our paper is the *F@.75*. This metric represents the proportion of segmented objects with an Overlap F-measure greater than 0.75. It is important not to confuse this metric with the F-measure computed for the Overlap and Boundary P/R/F. The F-measure for Overlap and Boundary is a harmonic mean of a model's average precision and average recall, while *F@.75* indicates the percentage of objects from a dataset that can be segmented with high accuracy. The *F* in F@.75 refers to the F-measure computed for a ground truth object after the Hungarian matching of the ground truth mask j with the predicted mask i as defined in [5] and stated in Equation (14).

$$P_{ij} = \frac{|c_i \cap g_j|}{|c_i|}, \quad R_{ij} = \frac{|c_i \cap g_j|}{|g_j|} \tag{13}$$

$$F_{ij} = \frac{2P_{ij}R_{ij}}{P_{ij}+R_{ij}} \tag{14}$$

The notation c_i denotes the set of pixels that belong to a predicted region i, while g_j represents all the pixels that belong to a non-background ground truth region j. In addition, P_{ij} represents the precision score, R_{ij} represents the recall score, and F_{ij} represents the F-measure score that corresponds to this particular pair of predicted and ground truth regions.

10. Occlusion Order Accuracy ACC_{oo} metric

Given an image v that depicts a typical cluttered tabletop scene, we get the ground truth-prediction assignment pairs after Hungarian matching as illustrated in Figure 9. The predicted masks will then be re-indexed to match the ids of the ground truth masks. Following that, the *predVisible* and *predOcclusion* masks that belong to the assigned pairs will be extracted. After that, the ground truth OOAM (*gtOOAM*) and the predicted OOAM (*predOOAM*) will be obtained using Algorithm 1.

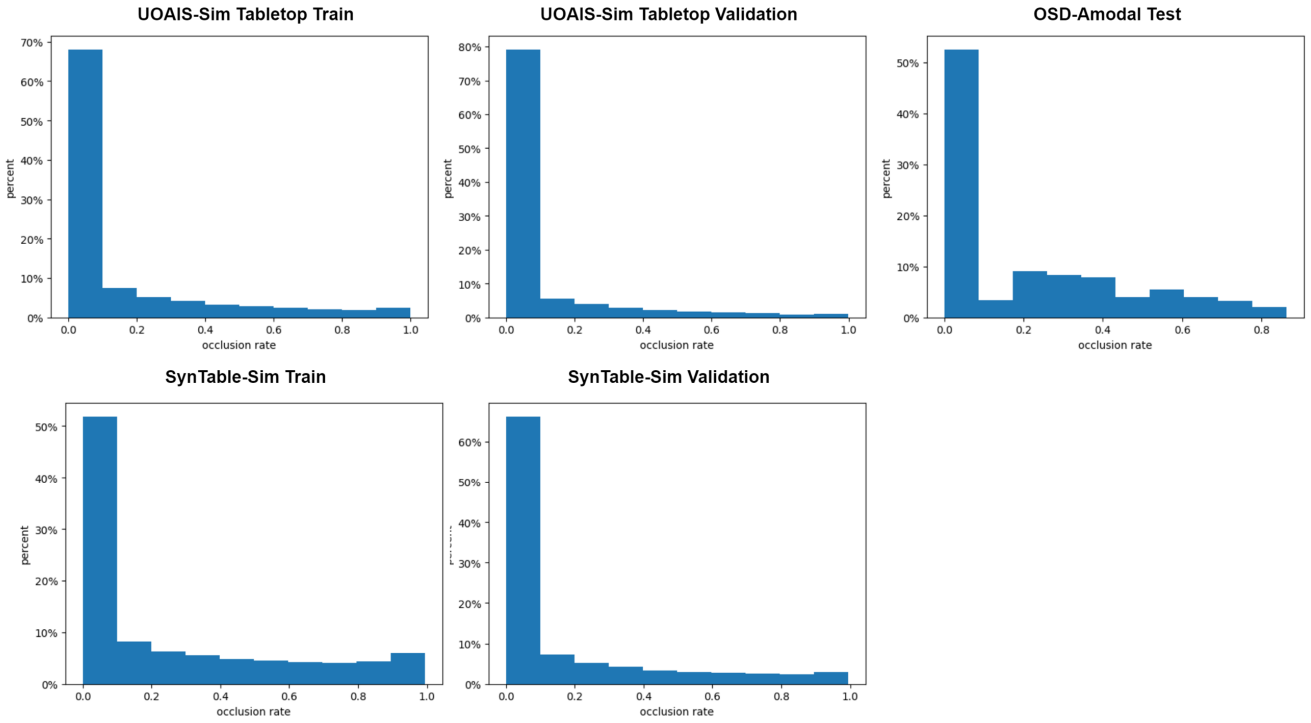


Figure 7. Histogram of occlusion rate for UOAIS-Sim tabletop, SynTable-Sim and OSD-Amodal datasets

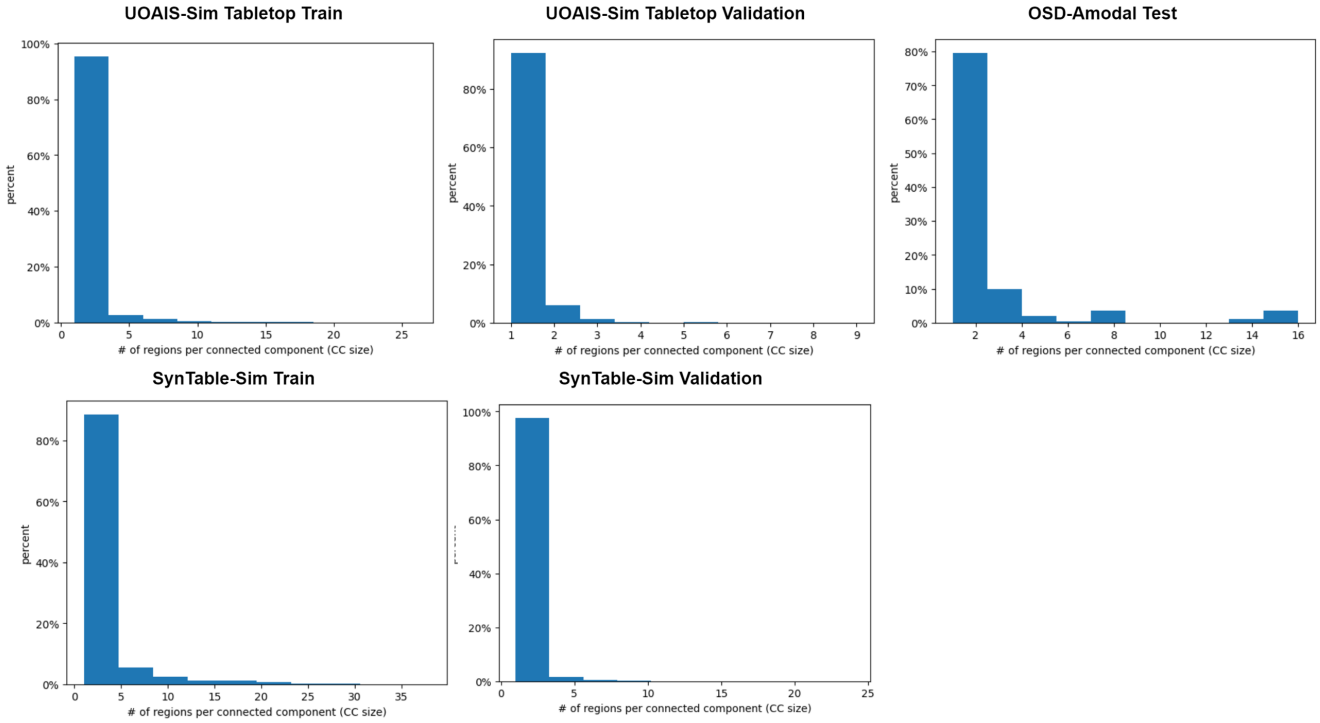


Figure 8. Histogram for number of regions per connected component (connected component size) for UOAIS-Sim tabletop, SynTable-Sim and OSD-Amodal datasets

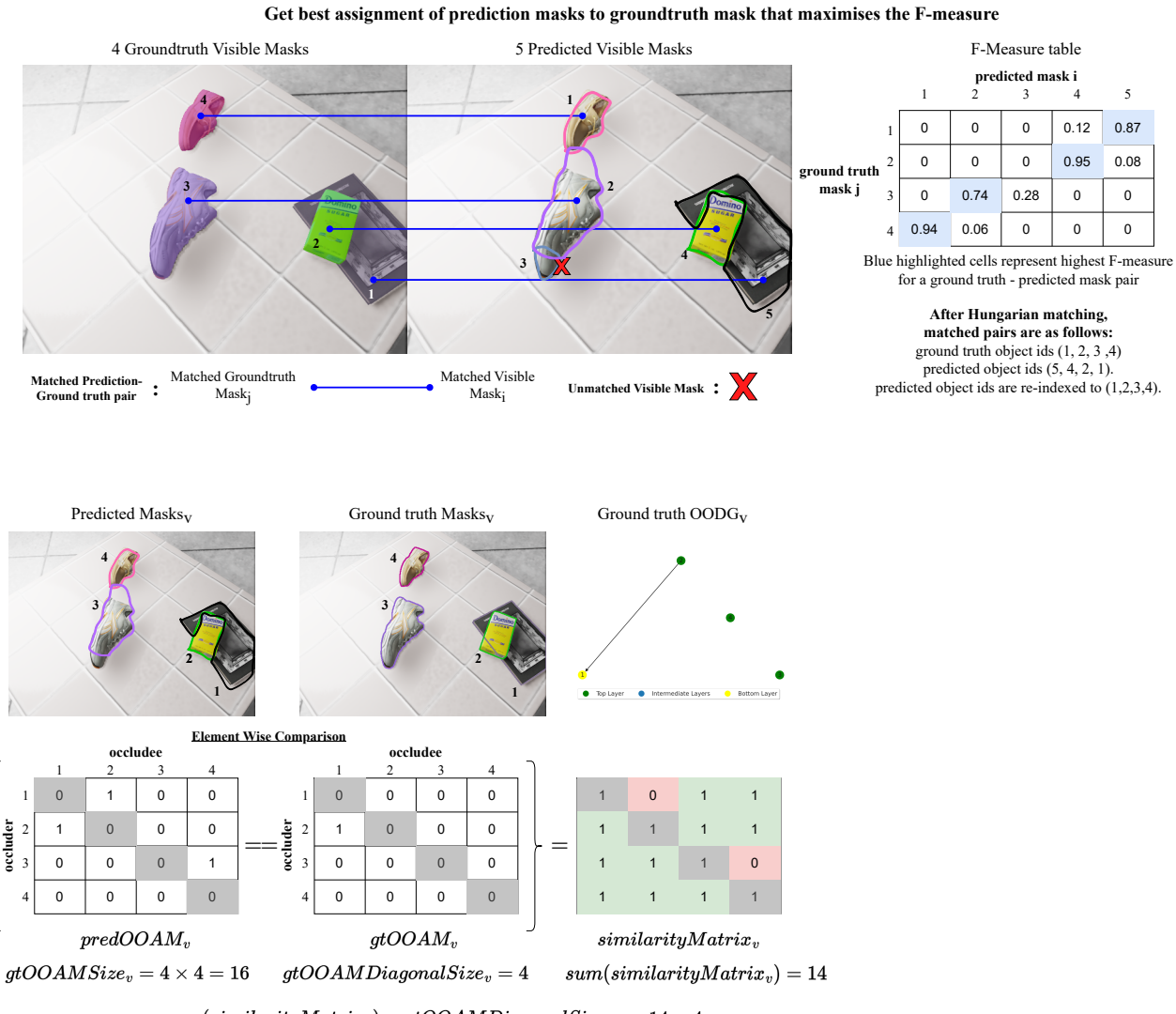


Figure 9. Hungarian Matching and calculating Occlusion Order Accuracy of image v

Figure 9 also illustrates the calculation of occlusion order accuracy in an image v. The similarity matrix (denoted as *similarityMatrix* in Figure 9) is obtained by conducting an element-wise equality comparison between the *gtOOAM* and *predOOAM*. After that, ACC_{oo} can be calculated using Equation 1.

In Equation 1, the ACC_{oo} represents the ratio of the number of correct predicted occlusion nodes over the number of ground truth occlusion nodes. Let *#correctPredictedOcclusionNodes* denote the number of correct occluder and occludee predictions for all objects in a viewpoint (represented by green highlighted cells in *similarityMatrix* in Figure 9).

A summation of all the elements in the similarity matrix is carried out to obtain *#correctPredictedOcclusionNodes*. Let *#groundtruthOcclusionNodes* denote the number of ground truth occluder and occlude nodes in a viewpoint. To obtain *#groundtruthOcclusionNodes*, we count the number of elements (*gtOOAMSize*) in the ground truth OOAM. As an object cannot occlude itself, the diagonal of any OOAM is always 0, and the diagonal of any similarity matrix is always 1 (depicted as grey highlighted cells in Figure 9). Thus, we subtract the number of elements along the diagonal of the gtOOAM (denoted by *gtOOAMDiagonalSize*) from the calculation of *#correctPredictedOcclusionNodes* and *#groundtruthOcclusionNodes*.

Correct occlusion order predictions occur when the pre-

dicted occlusion relationship for each object matches the ground truth. Incorrect occlusion order predictions can result from erroneous predictions or missing visible mask predictions of object instances. When there are missing predictions, setting the corresponding row and column of the missing object instance in the similarity matrix to 0 penalizes the model for the missing object predictions. The smaller element-wise sum of the similarity matrix leads to a smaller ACC_{oo}. This demonstrates the appropriate assignment of penalties by ACC_{oo} to different error types for measuring object occlusion ordering in a scene.

11. Occlusion Order Directed Acyclic Graph (OODAG)

After obtaining the Occlusion Order Adjacency Matrix (OOAM), we can generate the occlusion order directed graph from it. For each non-zero entry (i, j) in the OOAM, we draw a directed edge from node i to node j. If the entry is zero, we do not draw an edge. A non-zero entry at (i, j) represents that object i is occluding object j.

For example, the OOAM generated in Figure 10 shows that $(i, j) = (1, 12)$ where i and j are the object indices (the bounding box labels) in the image. This means that object 1 occludes object 12, and a directed edge will point from object 1 to 12. From the generated Directed Occlusion Graph, we can also check if the graph is cyclic or acyclic using graph cyclic detection methods such as Depth First Search (DFS) and Breadth First Search (BFS). Only if the graph has no directed cycles (Directed Acyclic Occlusion Graph) can topological sorting be implemented.

In the generated Occlusion Order graph, we further classify objects in three different order layers - Top, Intermediate, and Bottom. Objects at the top layer represent objects that are not occluded by any other object. Objects in the intermediate layers mean that they are occluded but they also occlude other objects. For objects in the bottom layer, they are occluded but they do not occlude other objects.

12. Qualitative Inference Results of UOAIS-Net on the OSD-Amodal Dataset

After training the UOAIS-Net model [2] on both SynTable-Sim and UOAIS-Sim (tabletop) datasets [2], we present some of our qualitative results in Figure 11. As discussed in the main text of our paper, the UOAIS-Net trained on the SynTable-Sim dataset exhibits superior performance in contrast to the UOAIS-Net trained on the UOAIS-Sim tabletop dataset. This observation is further supported by the inference results presented in Figure 11. Furthermore, as the scene becomes more and more cluttered, the UOAIS-Net model trained on the SynTable-Sim dataset evidently outperforms that of the UOAIS-Net trained on the UOAIS-Sim tabletop dataset.

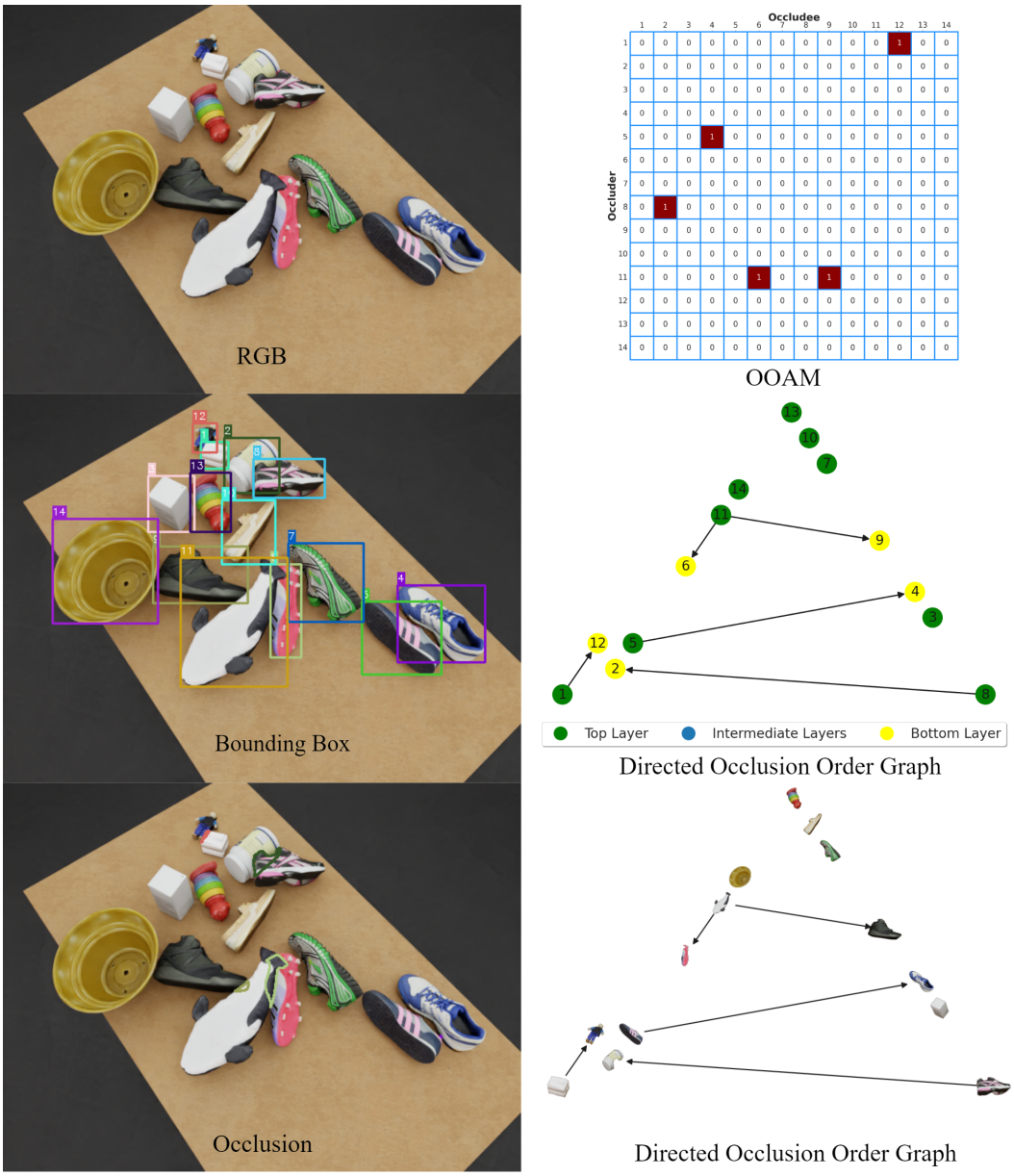


Figure 10. A visualisation of annotations for a cluttered tabletop image generated by SynTable

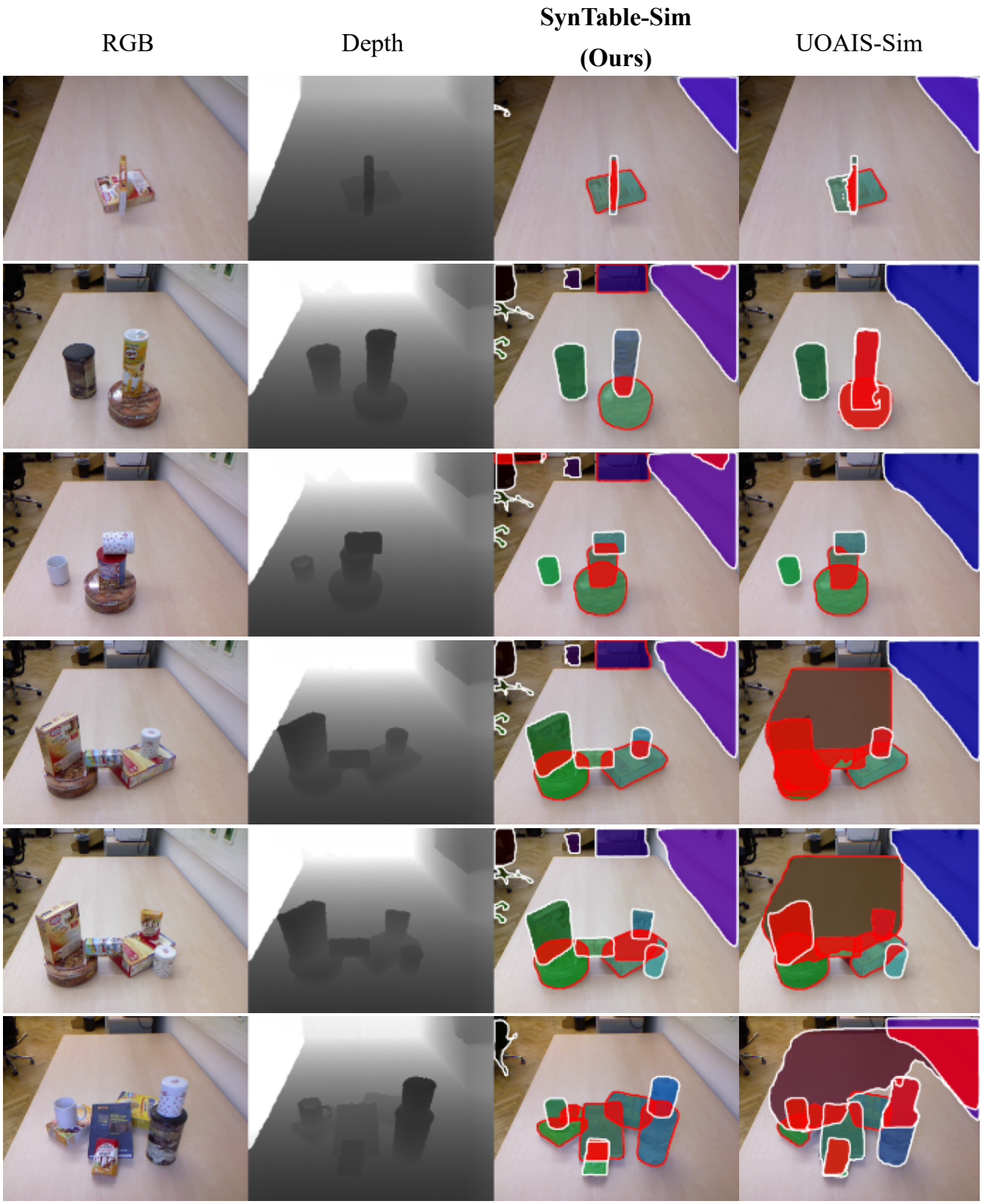


Figure 11. Comparison of the inference results on the OSD-Amodal dataset. **SynTable-Sim (Ours):** the performance of UOAIS-Net on the OSD-Amodal dataset after training on the SynTable-Sim dataset. **UOAIS-Sim**: the performance of UOAIS-Net on the OSD-Amodal dataset after training on the UOAIS-Sim tabletop dataset.