A. Appendix

A.1. Architecture details

Encoder ϕ and SG Predictor h We use ResNet 101 [22] pretrained on ImageNet as the encoder neural network. We use the Faster R-CNN [44] and Graph Convolution Network based architecture from Graph R-CNN [65] to implement the SG Predictor h.

GRL For appearance alignment σ^a, we use a 2-layer 2D convolution neural network based discriminator with ReLU activation. For prediction alignment $\sigma^{c,pred}$ we use 2-layer MLP based discriminator. We also scale the gradients to the encoder network ϕ from the discriminator by a factor of 4 in above cases.

A.2. Experiments

A.2.1 CLEVR

Setup The source and target domains of the CLEVR [25] environment leverage Blender [9] to render 320 x 240 images and corresponding ground truth scene graphs. The source and target domains were generated using disjoint object colors, object materials, margin between objects, and number of objects, to ensure a significant domain gap. We use 4 objects of colors (blue, green, magenta, yellow) and material (metal) for source domain. We use 2–3 objects of colors (pink, brown, white) and material (rubber) for target domain. Additionally, we transform the target by using a style transfer network [2]. For both domains, we sample each class and their size (small, medium & large) with equal probability. The environment has three lights and a fixed camera. We add a small random jitter to their initial positions during the rendering process. Some samples of source and target domain are shown in Figure 7.

Training Details We train our model for 10^5 iterations with label alignment $\sigma^{c,label}$ and appearance alignments σ^a. We optimize the model using a SGD optimizer with learning rate of 10^{-4} and momentum of 0.9. We train our model using a batch size 4 on NVIDIA DGX workstations. We report saturation peak performance in all our tables. We give equal regularization weights to source task loss σ_s, appearance alignment σ^a and label alignment $\sigma^{c,label}$.

Results More qualitative results of Sim2SG evaluated on the target domain for CLEVR are shown in Figure 8. We see better recall and fewer false positive object detections leading to more accurate scene graphs. Label alignment $\sigma^{c,label}$ improves object recall, but occasionally introduces some false positive detections. Our appearance alignment σ^a helps in reducing such false positives as shown in Figure 9. The full quantitative results of ablation are present in Table 4.

A.2.2 Dining-Sim

Setup The Dining-Sim environment is written using Pixar's USD API[3] and rendered with a proprietary renderer. The source domain is rendered with 2 spp (samples per pixels) followed by denoiser. We select 1 chair (cantilever chair), 1 table (workshop table) and 1 laptop (PC). We randomly place chair and table on the floor and laptop on the floor as well as on the table with a random orientation. The asset for each subcategory is randomly chosen from a list of subcategory specific ShapeNet [5] assets. We also ensure that objects do not overlap by applying collision avoidance with simple box collision volumes. A subset of 4 to 5 simple materials that vary only in diffuse colour is created for each of the walls, floor, chair and table. Laptops use the original asset texture.

The target domain is rendered using path tracing with 20 spp (samples per pixels) followed by denoiser. We use 4 chairs (Windsor chair), 1 table (kitchen table) and 2 laptops (MacBook). We first place the table with a random orientation and position on the floor. We then place the four chairs at each side of the table, oriented towards the table centre. Two laptops are then placed randomly on the table surface with a random rotation. The asset for each subcategory is randomly chosen from a list of subcategory specific ShapeNet [5] assets. For materials, we use a subset of 4 to 6 physically based, highly detailed materials for each of the walls, floor, chair and table. As with the source domain, laptops use the original asset texture.

Both domains share room parameters: a fixed camera (60 degree field of view, positioned at far side of the room) and 3 fixed spherical lights. Samples from the source and the target domains are shown in Figure 10. There are five kinds of relationships - front, behind, left, right, and on with table as subject. We use 5000 labeled images from source, 5000 unlabeled images from target for training and 1000 labeled images from both source & target domains for evaluation. We use 1024 x 768 image resolution for training and evaluation.

Details of Synthesis Step We describe how we generate synthetic data by inferring scene graphs from target domain in detail. We filter the objects and relation-

[2] https://github.com/pytorch/examples/tree/master/fast_neural_style

[3] https://graphics.pixar.com/usd/docs/index.html

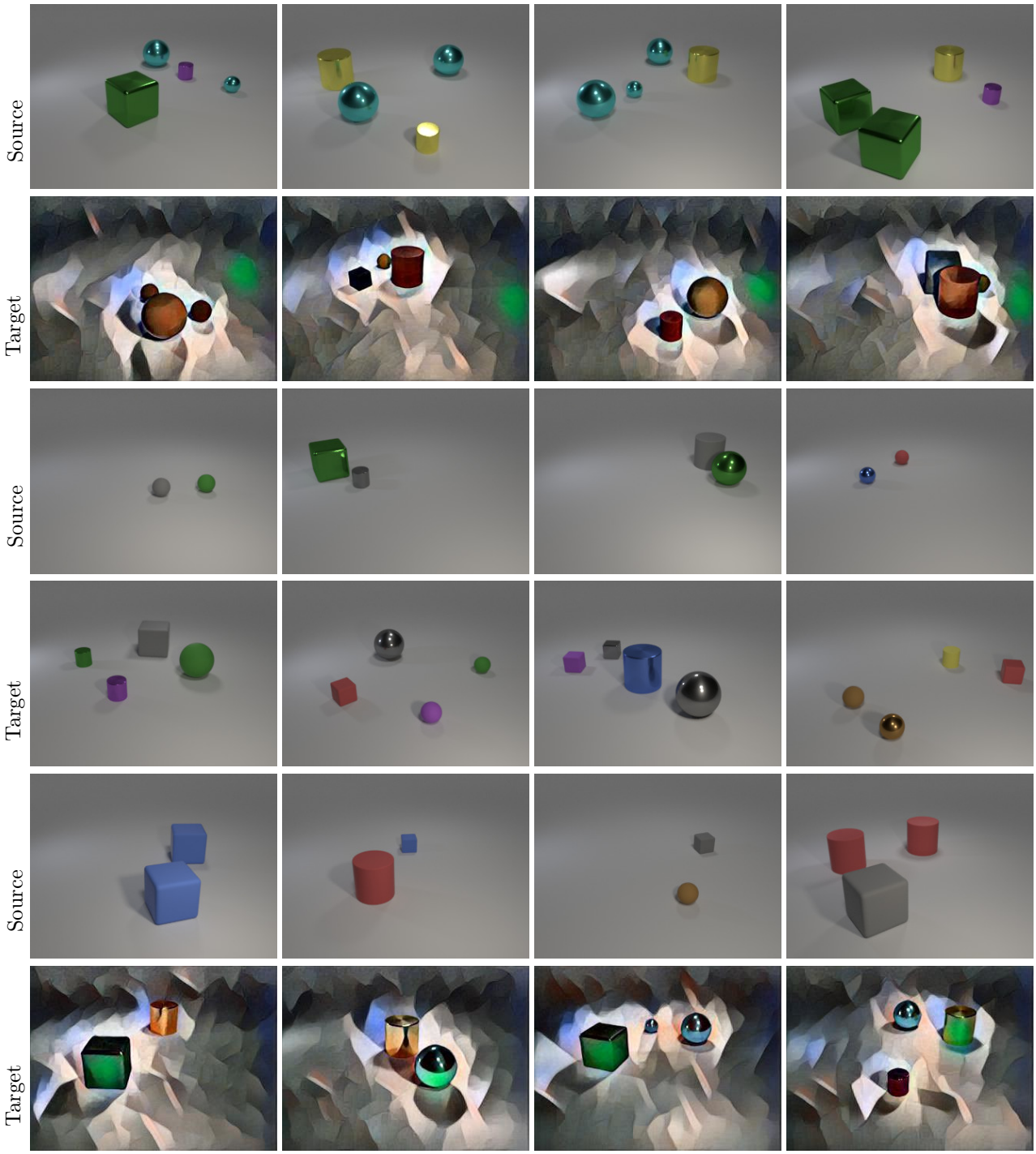

Figure 7. Samples from source and target distributions from Clevr environment. Row 1-2: Source and Target differ in both appearance and content. Row 3-4: Source and Target differ in content but have same appearance. Row 5-6: Source and Target differ in appearance but have same content.

ships among them using an adaptive threshold (details in the next paragraph) for the generation. We assume access to camera parameters (intrinsic and extrinsic both). We project a ray from the camera through the pixel corresponding to the bounding box bottom-centre. The 3D coordinate of the object is then the intersection of the ray and the ground plane, which we assume to be flat at elevation 0. We place each object in the 3D

Method	mAP@0.5 IoU	Recall@20
SDR [41]	0.675	0.339
Ours ($\sigma^{c,label}$)	0.923	0.646
Ours (σ^{a})	**0.938**	**0.938**

Method	mAP@0.5 IoU	Recall@20
SDR [41]	**1.000**	0.760
Ours (σ^{a})	0.970	0.722
Ours ($\sigma^{c,label}$)	**1.000**	**0.996**

Table 4. Left (resp. right): Source and target domains have different (resp. similar) appearance but similar (resp. different) content distribution. All the evaluations are on the target domain.

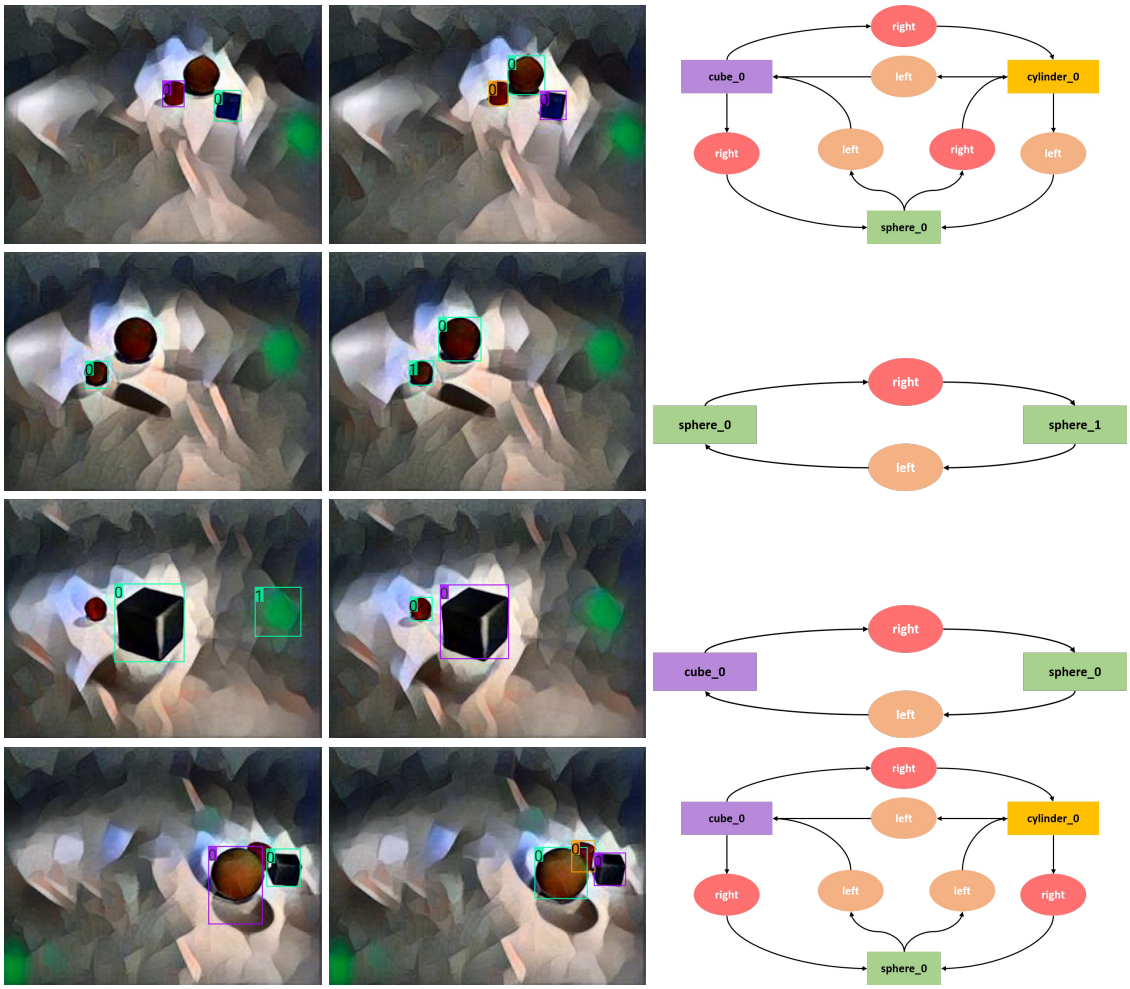

Figure 8. Qualitative results of Sim2SG on the target domain for CLEVR. First column shows that the SDR [41] fails to either detect objects or have high number of false positives (mislabels) leading to poor scene graph. Our method detects objects better, has fewer false positives and ultimately generates more accurate scene graphs as shown in second and third column respectively. Objects are color coded.

scene by picking a random 3D asset according its type (class) and assigning random pose in the range 0°–360°. We assume context like ground, wall as described in the previous paragraph. We refine the 3D scene further according to the predicted relationships among objects. For example, we use "on" relationship to refine object placements by adjusting the object (laptop or chair) elevation to match the table top. We then render the 3D scene.

Training Details We optimize the model using a SGD optimizer with learning rate of 10^{-4} and momentum of 0.9. We train our model using a batch size 2 on NVIDIA DGX workstations. We report saturation peak performance in all our tables. We give equal weights to source task loss σ_s, appearance alignment σ^a, prediction alignment $\sigma^{c,pred}$ and label alignment $\sigma^{c,label}$.

We first train the model with label alignment $\sigma^{c,label}$) for 6 epochs each with 10^4 iterations and score threshold of 0.5. Then we add appearance alignment σ^a and

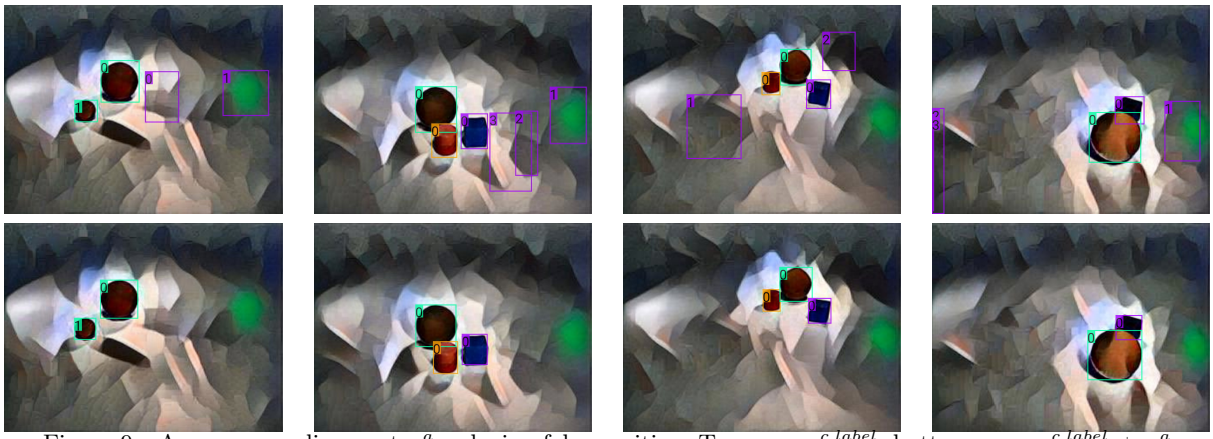

Figure 9. Appearance alignment σ^a reducing false positive. Top row: $\sigma^{c,label}$, bottom row: $\sigma^{c,label} + \sigma^a$

prediction alignment $\sigma^{c,pred}$ and train for an additional 2×10^4 iterations. It takes 12 hours for full training including rendering time.

Results We present the full quantitative results in Table 5 and qualitative results in the Figure 11. We observe that the combination of all alignment terms $\sigma^{c,label}$, σ^a and $\sigma^{c,pred}$ gives the best relationship triplet recall of 0.547@50. In order to keep our approach as general as possible, we do not enforce strict rules on object placements and prefer to randomize parameters that are not predicted such as orientation as illustrated in the qualitative results of label alignment $\sigma^{c,label}$ in Figure 12. When target domain assets are too dissimilar from the assets in the source domain, it often results in incorrect reconstructions as shown in Figure 12 (last column). We also observe that after label alignment $\sigma^{c,label}$, the model occasionally has false positive detections, particularly in areas of the floor that have intricate patterns. We qualitatively show that these false positives disappear with the addition of appearance alignment σ^a term (Figure 13).

Ablations We conduct two sets of experiments on the Dining-Sim environment. The first experiment studies appearance gap: source domain has different appearance but similar content from the target domain. The source domain is generated using the target generation scheme but using source dataset materials as shown in Figure 10. The appearance gap because of photo-realistic texture in target domain is from 0.846 Recall@50 to 0.625 Recall@50. We observe that the appearance alignment σ^a helps reduce the appearance gap, increasing relationship triplet recall from 0.625@50 to 0.821@50. For reference, the oracle performance on the target domain is 0.846 Recall@50. Similarly, the second experiment studies content gap where the source and target use the same materials but have different assets, object positions and number of objects. We accomplish this by modifying the source generation scheme to select materials from the target dataset. Samples of source and target are shown in the third and fourth rows of Figure 10. We observe that the label alignment $\sigma^{c,label}$ term helps reduce the content gap and increase relationship triplet recall from 0.468@50 to 0.539@50. The relatively modest improvement makes sense as the two domains still differ in content (source and target domain assets differ). Full results are in Table 6.

A.2.3 Drive-Sim

Setup As mentioned in Section 5.3, we use an Unreal Engine 4[4] based driving simulator akin to [41] to generate synthetic data. We have cars (1-2 per lane), trees(1-3), houses/buildings(1-3), pedestrians(0-2), sidewalk(2), roads(2-6). We do not have poles, street signs or any other objects. We have straight roads. We use realistic random placements, e.g. cars can only be placed on a lane, pedestrians on sidewalk, houses on ground and trees on both sidewalk and ground. We randomize the time of the day, cloud density and use directional light. We assume real world scale. We place our camera at a car height on a random right lane with fixed camera parameters (0 yaw, 0 pitch, 90 fov). We add realistic texture and color to each object similar to [41]. We use 1242 x 375 image resolution for training and evaluation.

Details on Synthesis Step We describe how we generate synthetic data by inferring scene graphs from KITTI [20] in detail. During synthesis stage, we infer the scene graphs from KITTI and further filter the objects and relationships among them using a confidence threshold of 0.2. We do not have access to KITTI camera parameters and we use the camera parameters

[4] https://www.unrealengine.com/

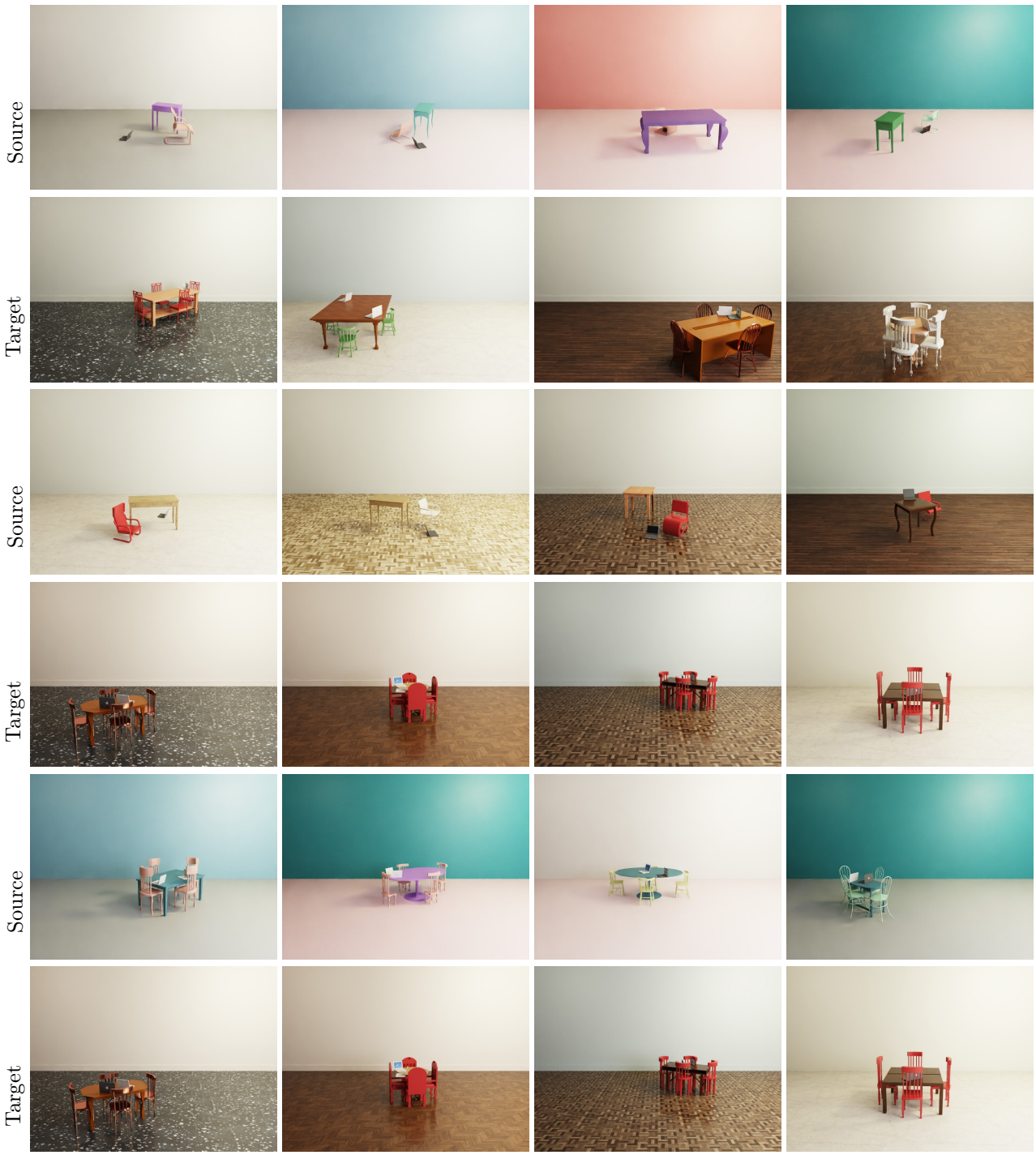

Figure 10. Samples from source and target distributions for Dining-Sim. Row 1-2: Source and Target domains differ in both appearance and content. Row 3-4: Source and Target differ in content but have same appearance. Row 5-6: Source and Target differ in appearance but have same content.

Method	Chair AP	Table AP	Laptop AP	mAP @0.5 IoU	Recall@50
SDR [41]	**0.842** ±0.038	0.519 ±0.088	0.392 ±0.051	0.584 ±0.049	0.331 ±0.064
Ours ($\sigma^{c,label}$)	0.737 ±0.043	0.724 ±0.030	0.608 ±0.047	0.713 ±0.038	0.501 ±0.044
Ours ($\sigma^{c,label}$, σ^{a}, $\sigma^{c,pred}$)	0.770 ±0.022	**0.757** ±0.037	**0.659** ±0.005	**0.729** ±0.015	**0.547** ±0.015

Table 5. Quantitative results of Sim2SG on a target domain in Dining-Sim environment.

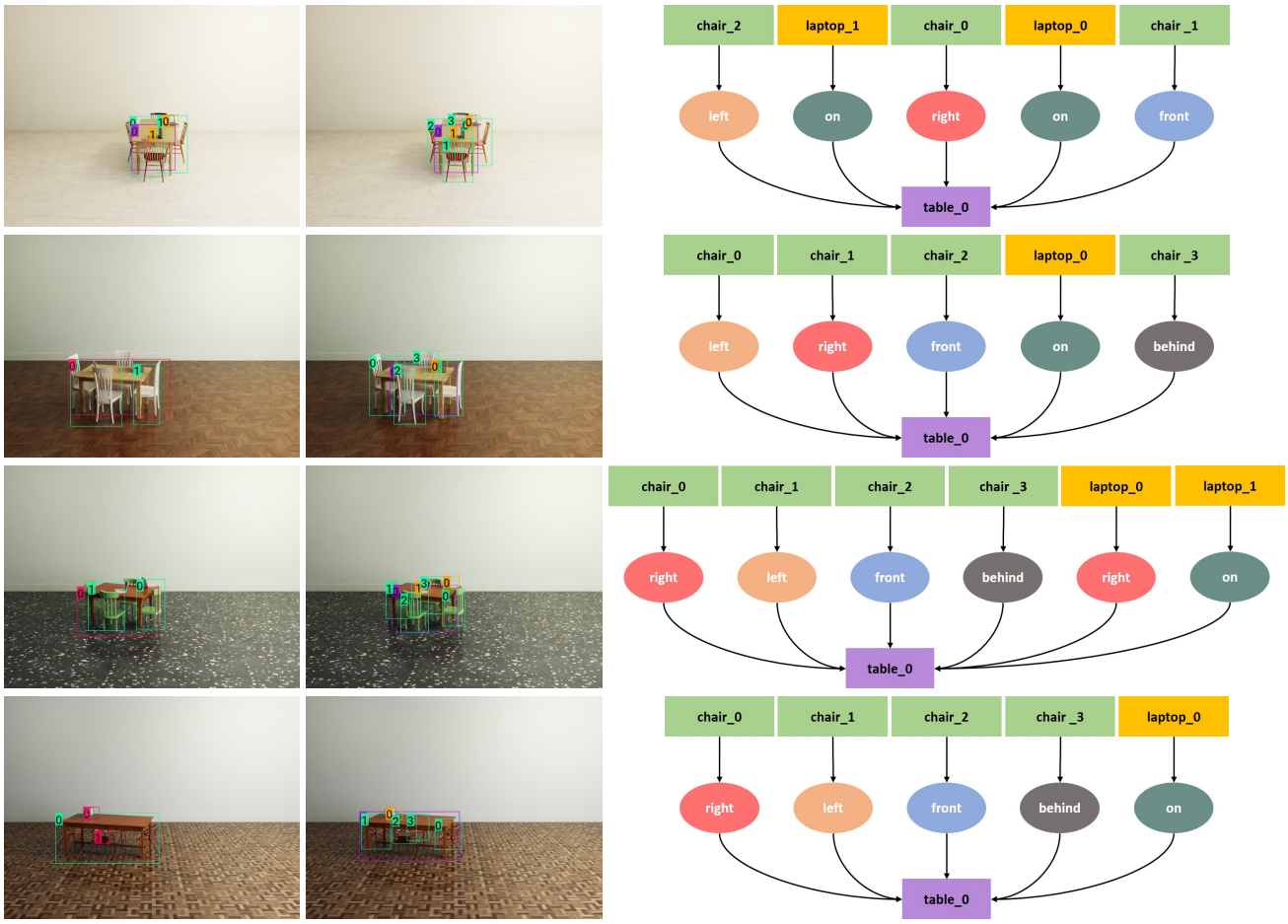

Figure 11. Qualitative results of Sim2SG on the target domain for Dining-Sim. First column shows that the SDR [41] fails to either detect objects or have high number of false positives (mislabels) leading to poor scene graph. Our method detects objects better, has fewer false positives and ultimately generates more accurate scene graphs as shown in second and third column respectively. Objects are color coded.

Figure 12. Scenes generated by our method (top) for target samples (bottom) in Dining-Sim environment.

described in the previous paragraph. Using the assumed camera parameters (both intrinsic and extrinsic) we project a ray from the camera through the pixel corresponding to the bounding box bottom-centre. The 3D coordinate of the object is then the intersection of the ray and the ground plane, which we assume to be

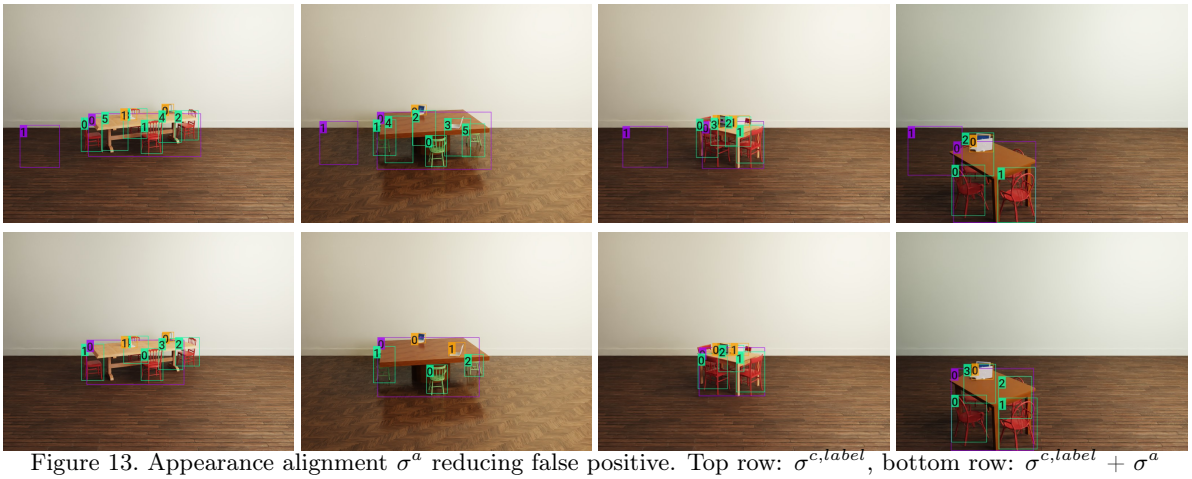

Figure 13. Appearance alignment σ^a reducing false positive. Top row: $\sigma^{c,label}$, bottom row: $\sigma^{c,label} + \sigma^a$

Method	mAP @0.5 IoU	Recall@50
SDR [41]	0.772 ±0.043	0.625 ±0.076
Ours (σ^a)	**0.878** ±0.001	**0.821** ±0.006

Method	mAP @0.5 IoU	Recall@50
SDR [41]	0.676 ±0.011	0.468 ±0.006
Ours ($\sigma^{c,label}$)	**0.737** ±0.024	**0.539** ±0.006

Table 6. Dining-Sim ablations. Left (resp. right): Source and target domains have different (resp. similar) appearance but similar (resp. different) content distribution. All the evaluations are on the target domain.

flat at elevation 0. We place each object in the 3D scene by picking a random 3D asset according its type (class) and assigning random pose in the range 0°–360° (except cars that are aligned to the lane). We assume contexts like road, ground, sky, sidewalk as described in the previous paragraph. We refine the 3D scene further according to the predicted relationships among objects. We also assume a consistent lane width, and number of roads are determined by positions of the detected vehicles in the scene. We place multiple Trees (*i.e. Vegetation*) if the projected 3D volume permits. We then render the 3D scene.

Training Details We optimize the model using a SGD optimizer with learning rate of 10^{-4} and momentum of 0.9. We train our model using a batch size 2 on NVIDIA DGX workstations. We report saturation peak performance in all our tables. We give equal regularization weights to source task loss σ_s, appearance alignment σ^a, prediction alignment $\sigma^{c,pred}$ and label alignment $\sigma^{c,label}$.

We first train our model using label alignment $\sigma^{c,label}$) for 3 epochs each with 10^4 iterations. We then add appearance alignment σ^a and prediction alignment $\sigma^{c,pred}$ and train for an additional 60k iterations. This makes sense as σ^a works better when content/labels are aligned between the two domains. The total training takes 12 hours including the rendering time.

Baselines: We adapt domain adaptation baselines [7, 64] to our framework by using the same backbone (Resnet 101) and SG Predictor (GraphRCNN [65]) network as Sim2SG, but their loss function. We do not adapt SAPNet [30]. We train these baselines on 6000 images from the source domain [41] using the same optimizer and learning rate as Sim2SG for 6×10^4 iterations. We found GPA [64] and SAPNet [30] detection performance to be lower than that reported in their work especially for pedestrian, vegetation and house classes. It is worth noting that their reported class-wise performance numbers only overlap with some of the classes in our work.

We train [27] for 40 epochs with a batch size of 16 and learning rate 0.001 as per the authors. We then obtain 6000 images and train it on Sim2SG framework (Resnet 101 backbone and GraphRCNN SG predictor) for 6×10^4 iterations using the same optimizer and learning rate as Sim2SG. For self-learning based on pseudo labels [72], we obtain the pseudo labels on KITTI images using the most confident predictions by synthetic pretrained GraphRCNN network (as per the authors). We then train these labeled KITTI images on Sim2SG framework for 60k iterations using the same optimizer and learning rate as Sim2SG.

KITTI Annotation We use the existing bounding box annotations of Vehicle and Pedestrians. We annotate Trees and Houses/Buildings of all sizes, occlusion and truncation in KITTI. We use the available camera parameters to project the 2D bounding box into 3D space to help us annotate spatial relationships–front, behind, left and right.

Method	Car	Pedes.	House	Veg.	mAP	Recall@50
SDR [41]	0.488 ±0.007	0.214 ±0.025	0.223 ±0.022	0.177 ±0.010	0.276 ±0.005	0.112 ±0.009
Meta-Sim [27]	0.575 ±0.008	0.227 ±0.024	0.252 ±0.008	0.174 ±0.033	0.307 ±0.003	0.143 ±0.007
Self-learning [72]	0.466 ±0.009	0.215 ±0.016	0.189 ±0.025	0.265 ±0.008	0.284 ±0.006	0.129 ±0.006
DA-FasterRCNN [7]	0.523 ±0.036	0.209 ±0.038	0.203 ±0.037	0.171 ±0.012	0.277 ±0.017	0.119 ±0.022
GPA [64]	0.248 ±0.053	0.016 ±0.026	0.097 ±0.030	0.063 ±0.031	0.109 ±0.022	0.028 ±0.009
SAPNet [30]	0.420 ±0.052	0.124 ±0.035	0.018 ±0.004	0.042 ±0.010	0.151 ±0.010	–
Ours ($\sigma^{c,label}$)	0.566 ±0.033	**0.310** ±0.029	0.261 ±0.009	0.242 ±0.040	0.345 ±0.002	0.193 ±0.010
Ours ($\sigma^{c,label}$, σ^{a})	0.606 ±0.021	0.309 ±0.013	0.272 ±0.008	0.260 ±0.021	0.362 ±0.007	0.220 ±0.010
Ours ($\sigma^{c,label}$, σ^{a}, $\sigma^{c,pred}$)	**0.623** ±0.033	0.301 ±0.018	**0.283** ±0.007	**0.274** ±0.015	**0.370** ±0.005	**0.240** ±0.003
SDR	0.412 ±0.006	0.174 ±0.018	0.215 ±0.022	0.177 ±0.011	0.245 ±0.002	0.085 ±0.008
Meta-Sim	0.455 ±0.040	0.203 ±0.026	0.242 ±0.009	0.176 ±0.029	0.269 ±0.007	0.093 ±0.005
Self-learning	0.377 ±0.007	0.174 ±0.014	0.197 ±0.002	0.263 ±0.006	0.253 ±0.004	0.077 ±0.005
DA-FasterRCNN	0.472 ±0.028	0.181 ±0.031	0.203 ±0.041	0.168 ±0.013	0.256 ±0.012	0.091 ±0.019
GPA	0.201 ±0.043	0.016 ±0.026	0.106 ±0.031	0.065 ±0.036	0.096 ±0.026	0.018 ±0.007
SAPNet	0.419 ±0.068	0.098 ±0.028	0.017 ±0.005	0.038 ±0.008	0.143 ±0.017	–
Ours ($\sigma^{c,label}$)	0.471 ±0.033	**0.273** ±0.027	0.244 ±0.010	0.233 ±0.036	0.305 ±0.006	0.128 ±0.008
Ours ($\sigma^{c,label}$, σ^{a})	0.511 ±0.002	0.266 ±0.014	0.251 ±0.013	0.256 ±0.021	0.321 ±0.004	0.155 ±0.005
Ours ($\sigma^{c,label}$, σ^{a}, $\sigma^{c,pred}$)	**0.529** ±0.029	0.249 ±0.017	**0.262** ±0.011	**0.270** ±0.015	**0.328** ±0.007	**0.170** ±0.004
SDR	0.382 ±0.029	0.168 ±0.017	0.211 ±0.023	0.174 ±0.010	0.234 ±0.006	0.070±0.007
Meta-Sim	0.413 ±0.009	0.197 ±0.027	0.236 ±0.009	0.164 ±0.023	0.253 ±0.003	0.075 ±0.005
Self-learning	0.312 ±0.006	0.167 ±0.015	0.191 ±0.003	0.263 ±0.006	0.233 ±0.004	0.062 ±0.003
DA-FasterRCNN	0.424 ±0.028	0.170 ±0.029	0.200 ±0.041	0.169 ±0.014	0.241 ±0.014	0.074 ±0.015
GPA	0.174 ±0.040	0.011 ±0.016	0.106 ±0.031	0.059 ±0.027	0.087 ±0.020	0.015 ±0.005
SAPNet	0.362 ±0.054	0.085 ±0.051	0.116 ±0.021	0.067 ±0.022	0.157 ±0.024	–
Ours ($\sigma^{c,label}$)	0.410 ±0.009	**0.262** ±0.025	0.240 ±0.010	0.229 ±0.036	0.285 ±0.003	0.104 ±0.006
Ours ($\sigma^{c,label}$, σ^{a})	0.493 ±0.004	0.252 ±0.014	0.247 ±0.012	0.253 ±0.020	0.311 ±0.311	0.127 ±0.004
Ours ($\sigma^{c,label}$, σ^{a}, $\sigma^{c,pred}$)	**0.501** ±0.006	0.241 ±0.018	**0.254** ±0.010	**0.269** ±0.014	**0.316** ±0.004	**0.139** ±0.004

Table 7. Evaluation on three modes of KITTI : easy (top), moderate (middle), hard (bottom) when training on the labeled synthetic data and unlabeled real data. The class specific AP and mAP are reported at 0.5 IoU.

Method	Recall@20	Recall@50	Recall@100
SDR	0.098	0.131	0.146
Meta-Sim	0.109	0.149	0.164
Ours ($\sigma^{c,label}$, σ^{a}, $\sigma^{c,pred}$)	**0.184**	**0.235**	**0.252**
SDR	0.067	0.088	0.099
Meta-Sim	0.071	0.094	0.104
Ours ($\sigma^{c,label}$, σ^{a}, $\sigma^{c,pred}$)	**0.132**	**0.167**	**0.181**
SDR	0.053	0.071	0.079
Meta-Sim	0.058	0.076	0.085
Ours ($\sigma^{c,label}$, σ^{a}, $\sigma^{c,pred}$)	**0.107**	**0.137**	**0.150**

Table 8. Recall on three modes of KITTI : easy (top), moderate (middle), hard (bottom) when training on the labeled synthetic data and unlabeled real data. The evaluation is performed once.

Results Full quantitative evaluations results are in Table 7 on all KITTI [20] evaluation criteria– easy, moderate and hard. In all three criteria, Sim2SG is able to achieve significantly better results (higher detection mAP @0.5 IoU and relationship triplet recall @ 50) than all the baselines. We also report recall @20, 100 for few baselines and our approach in Table 8. More qualitative results of label alignment $\sigma^{c,label}$ is in Figure 14. We show qualitative improvements (better object recall and fewer false positive detections) over SDR [41] and Meta-Sim [27] in Figure 15 and the corresponding accurate and full scene graphs in Figure 16.

Figure 14. Scenes generated by our method (left) for target KITTI samples (right).

Figure 15. Qualitative results of objects detected on three different KITTI images. Top: SDR fails to detect many objects and yields a large number of false positives (mislabels), leading to poor scene graphs (not shown). Middle: Meta-Sim improves on false-positives, but still fails to detect some objects. Bottom: Our method detects objects correctly with fewer false positives, thus generating more accurate scene graphs. (Cars in green, vegetation in yellow, buildings in purple.)

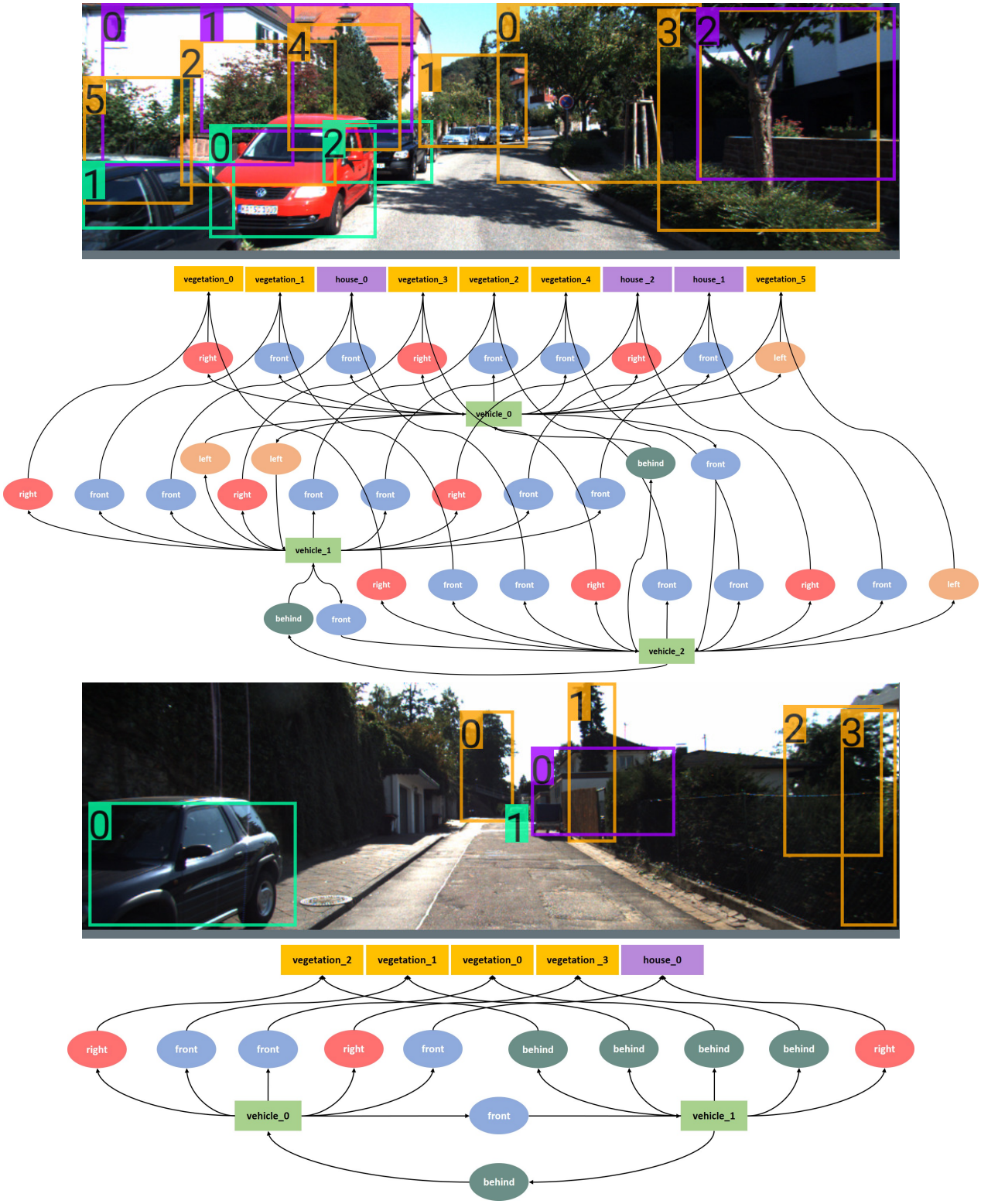

Figure 16. Qualitative results of Sim2SG on KITTI. Sim2SG generates accurate scene graphs.