SRDA: Generating Instance Segmentation Annotation via Scanning, Reasoning and Domain Adaptation

Wenqiang Xu, Yonglu Li, and Cewu Lu(✉)

Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
{vinjohn,yonglu_li,lucewu}@sjtu.edu.cn

Abstract. Instance segmentation is a problem of significance in computer vision. However, preparing annotated data for this task is extremely time-consuming and costly. By combining the advantages of 3D scanning, reasoning, and GAN-based domain adaptation techniques, we introduce a novel pipeline named SRDA to obtain large quantities of training samples with very minor effort. Our pipeline is well-suited to scenes that can be scanned, i.e. most indoor and some outdoor scenarios. To evaluate our performance, we build three representative scenes and a new dataset, with 3D models of various common objects categories and annotated real-world scene images. Extensive experiments show that our pipeline can achieve decent instance segmentation performance given very low human labor cost.

Keywords: 3D scanning · Physical reasoning · Domain adaptation

1 Introduction

Instance segmentation [6,21] is one of the fundamental problems in computer vision, which provides many more details in comparison to object detection [28], or semantic segmentation [23]. With the development of deep learning, significant progress has been made in instance segmentation. Many annotated datasets of large quantity were proposed [5,22]. However, in practice, when meeting a new environment with many new objects, large-scale training data collection and annotation is inevitable, which is cost-prohibitive and time-consuming.

Researchers have longed for a means of generating numerous training samples with minor effort. Computer graphics simulation is a promising way, since a 3D

W. Xu and Y. Li—Equal contributions.

C. Lu—Member of MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, and SJTU-SenseTime AI lab.

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-01258-8_8) contains supplementary material, which is available to authorized users.

V. Ferrari et al. (Eds.): ECCV 2018, LNCS 11216, pp. 124–140, 2018.
https://doi.org/10.1007/978-3-030-01258-8_8

scene can be a source of unlimited photorealistic images paired with ground truths. Besides, modern simulation techniques are capable of synthesizing most indoor and outdoor scenes with perceptual plausibility. Nevertheless, these two advantages are double-edged, rendered images would be painstaking to make the simulated scene visually realistic [31,38,43]. Moreover, for new environment, it is very likely some of the objects in reality are not in the 3D model database.

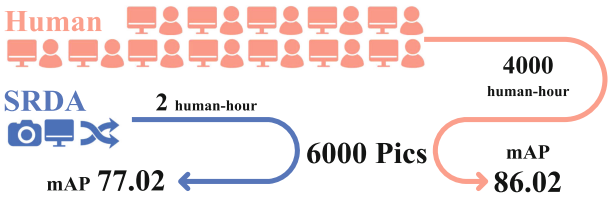

Fig. 1. Compared with human labeling (red), our pipeline (blue) can significantly reduce human labor cost by nearly 2000 folds and achieve reasonable accuracy in instance segmentation. 77.02 and 86.02 are average mAP@0.5 of 3 scenes. (Color figure online)

We present a new pipeline that attempts to address these challenges. Our pipeline comprises three stages: *scanning*, *physics reasoning*, *domain adaptation* (**SRDA**) as shown in Fig. 1. At the first stage, new objects and environmental background from a certain scene are scanned into 3D models. Unlike other CG based methods that do simulation with existing model datasets, images synthesized by our pipeline can ensure realistic effect and well describe the targeting environment, since we use real-world scanned data. At the reasoning stage, we proposed a reasoning system to generate proper layout for each scene by fully considering physically and commonsense plausible. Physics engine is used to ensure physics plausible and commonsense plausible is checked by commonsense likelihood (CL) function. For example, "a mouse on the mouse pad and they on the table" would have a large output in CL function. In the last stage, we proposed a novel *Geometry-guided GAN* (GeoGAN) framework. It integrates geometry information (segmentation as edge cue, surface normal, depth) which helps to generate more plausible images. In addition, it includes a new component *Predictor* which can serve as a useful auxiliary supervision, and also a criterion to score the visual quality of images.

The major advantage of our pipeline is time-saving. Compared with conventional exhausting annotation, we can reduce labor cost by nearly 2000 folds, in the meantime, achieve decent accuracy, preserving 90% performance. (See Fig. 1). The most time-consuming stage is scanning, which is easy to accomplish in most of indoor and some of outdoor scenarios.

Our pipeline can be widely adaptive to many scenarios. We choose three representative scenes, namely a shelf from a supermarket (for a self-service super-

market), a desk from an office (for home robot), a tote similar in Amazon Robotic Challenge[1].

To the best of our knowledge, no current datasets consist of compact 3D object/scene models and real scene images with instance segmentation annotations. Hence, we build a dataset to prove the efficacy of our pipeline. This dataset have two parts, one for scanned object models (SOM dataset) and one for real scene images with instance level annotations (Instance-60K).

Our contributions have two folds:

- The main contribution is the novel three-stage SRDA pipeline. We added a reasoning system to the feasible layout building and proposed a new domain adaptation framework named GeoGAN. It is time-saving and the output images are close to real ones according to the evaluation experiment.
- To demonstrate the effectiveness, we build up a database which contains 3D models of common objects and corresponding scenes (SOM dataset) and scene images with instance level annotations (instance-60K).

We will first review some of the related concepts and works in Sect. 2 and depict the whole pipeline from Sect. 3 on. We describe the scanning process in Sect. 3, reasoning system in Sect. 4, and GAN-based domain adaptation in Sect. 5. In Sect. 6, we illustrate how Instance-60K dataset is built. Extensive evaluation experiments are carried out in Sect. 7. And finally, we discuss the limitation of our pipeline in Sect. 8.

2 Related Works

Instance Segmentation. Instance segmentation has become a hot topic in recent years. Dai et al. [6] proposed a complex multiple-stage cascaded network that does detection, segmentation, and classification in sequence. Li et al. [21] combined a segment proposal system and object detection system, simultaneously producing object classes, bounding boxes, and masks. Mask R-CNN [14] supports multiple tasks including instance segmentation, object detection, human pose estimation. Whereas exhausting labeling is required to guarantee a satisfactory performance, if we apply these methods to a new environment.

Generative Adversarial Networks. Since introduced by Goodfellow [12], GAN-based methods have fruitful results in various fields, such as image generation [27], image-to-image translation [42], 3D model generation [40], etc. The former paper on image-to-image translation inspired our work, it indicates GAN has the potential to bridge the gap between simulation domain and real domain.

[1] https://www.amazonrobotics.com/#/roboticschallenge.

Image-to-Image Translation. A general image-to-image translation framework was first introduced by Pix2Pix [16], but it required a great amount of paired data. Chen [4] proposed a cascaded refinement network free of adversarial training, which gets high-resolution results, but still demands paired data. Taigman et al. [36] proposed an unsupervised approach to learn cross-domain conversion, however it needs a pre-trained function to map samples from two domains into an intermediate representation. Dual learning [17,41,42] is soon imported for unpaired image translation, but currently, dual learning methods encounter setbacks when camera viewpoint or object position varies. On the contrary to CycleGAN, Benaim et al. [2] learned one-side mapping. Refining rendered image using GAN is also not unknown [3,32,33]. Our work is a complementary to these approaches, where we deal with more complex data and tasks. We will compare [3,32] with our GeoGAN in Sect. 7.

Synthetic Data for Training. Some researchers attempt to generate synthetic data for vision tasks such as viewpoint estimation [35], object detection [11], semantic segmentation [30]. In [1], Alhaija et al. addressed generation of instance segmentation training data for street scenes with technical effort in producing realistically rendered and positioned cars. However, they focus on street scenes and do not use an adversarial formulation.

Scene Generation by Computer Graphics. Scene generation by CG techniques is a well-studied area in the computer graphics community [9,13,25,26,34]. These methods are capable of generating plausible layout of indoor or outdoor scene, but they have no intention to transfer the rendered images to real domain.

3 Scanning Process

In this section, we describe the scanning process. Objects and scene backgrounds are scanned in two ways due to the scale issue.

We choose the multi-view environment (MVE) [10] to perform dense reconstruction for objects, since it is image-based and thus requires only a RGB sensor. Objects are first videotaped, which can be easily done by most RGB sensors. In the experiment, we use an iPhone5s. The videos are sliced into images with multiple viewpoints, and fed into MVE to generate 3D models. We can videotape multiple objects (at least 4) and generate corresponding models per time, which can alleviate the scalability issue when new objects are too many to scan one by one. MVE is capable of generating dense meshes with a fine texture. As for the texture-less objects, we scan the object with hand holding, and the hand-object interaction can be a useful cue for reconstruction, as indicated in [39].

For the environmental background, scenes without targeting objects were scanned by Intel RealSense R200 and reconstructed by ReconstructMe[2]. We follow the official instruction to operate reconstruction.

[2] http://reconstructme.net/.

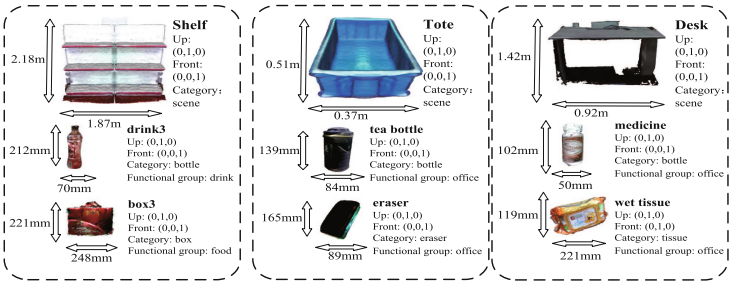

Fig. 2. Representative environmental backgrounds, object models, and corresponding label information.

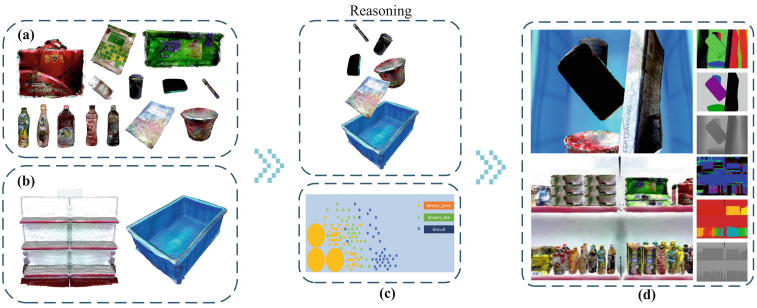

Fig. 3. The scanned objects (a) and background (b) are put into a rule-based reasoning system (c) to generate physics plausible layouts. The upper of (c) is the random scheme, while the bottom is the rule-based scheme. In the end, system output rough RGB images and corresponding annotations (d).

Resolution for iPhone5s is 1920×1080 and for R200 is 640×480 at 60 FPS. Remaining settings are by default.

4 Layout Building with Reasoning

4.1 Scene Layout Building with Knowledge

With 3D models of objects and environmental background at hand, we are ready to generate scenes by our reasoning system. A proper scene layout must obey physics laws and human conventions. To make scene physics plausible, we select an off-the-shelf physics engine, Project Chrono [37]. However, it is not as direct to make object layout convincing, some commonsense knowledge should be incorporated. To produce a feasible layout, we need to make object pose and location reasonable. For example, a cup has the pose of "standing up", but not "lying down", meanwhile, it is always on the table not the ground. This prior falls in daily knowledge that cannot be achieved by physics reasoning. Therefore, we present how to annotate the pose and location prior in what follows (Figs. 2 and 3).

Pose Prior: For each object, we show annotators its 3D model in 3D graphics environment, and ask annotators to draw all its possible poses that she/he can imagine. For each possible pose, the annotator should suggest a probability that this pose would happen. We record the probability of i^{th} object in pose k as $D_p[k|i]$. We use interpolation to ensure most poses has a probability value.

Location Prior: The same as pose prior, we show annotators the environmental background in 3D graphics environment, thus annotators label all its possible locations that an object may be placed. For each possible location, the annotator should suggest a probability that this object would be placed. We denoted the probability of i^{th} object in location k as $D_l[k|i]$. We use interpolation to make most of location has correponding probability value.

Relationship Prior: Some objects have strong co-occurrence prior. For example, mouse is always close to laptop. Given an object name list, we use language prior to select a set of object pair that have high co-occurrence probability, we call them as occurrence object pair (OOP). For each OOP, annotator suggests a probability of occurrence of corresponding object pairs. For object i^{th} and j^{th}, their probability of occurrence is denoted as $D_r[i,j]$ and a suggested distance (by annotators) is $H_r[i,j]$.

Note that the annotation maybe subjective, but we found that we only need a prior for layout generation guidance. Extensive experiments show that roughly subjective labeling is sufficient for producing satisfactory results. We will report the experiment details in supplementary file.

4.2 Layout Generation by Knowledge

We generate layout by considering both physics laws and human conventions. First, we randomly generate a layout and check its physics plausible by Chrono. If it is not physically reasonable, we reject this layout. Second, we check its commonsense plausible by three priors above. In detail, all object pairs are extracted in layout scene. We denote $(\{c_1(i), c_2(i)\}$, $(\{p_1(i), p_2(i)\}$ and $(\{l_1(i), l_2(i)\}$ as category, pose and 3D location of i^{th} extracted object pair in scene layout. The likelihood of pose is expressed as

$$K_p[i] = D_p[p_1(i)|c_1(i)]D_p[p_2(i)|c_2(i)]. \tag{1}$$

The likelihood of location for i^{th} object pair is written as,

$$K_l[i] = D_l[l_1(i)|c_1(i)]D_l[l_2(i)|c_2(i)]. \tag{2}$$

The likelihood of occurrence for i^{th} object pair is presented as

$$K_r[i] = \begin{cases} G_\sigma(|l_1(i) - l_2(i)| - D_r[c_1(i), c_2(j)]) & \text{if } H_r[i,j] > \gamma \\ 1, & \text{otherwise.} \end{cases} \tag{3}$$

where G_σ is a Gaussian function with parameter σ ($\sigma = 0.1$ in our paper). We compute occurrence prior in the case where the probability $H_r[i, j]$ is larger than a threshold γ ($\gamma = 0.5$ in our paper).

We denote commonsense likelihood function of a scene layout as

$$K = \prod_i K_l[i]K_l[i]K_r[i] \propto \sum_i \log(K_l[i]) + \log(K_p[i]) + \log(K_r[i]) \tag{4}$$

Thus, we can judge commonsense plausible by K. If K is smaller than a threshold ($K \leq 0.6$ in our experiments), we reject its corresponding layout. In this way, we can generate large quantities of layouts that is both physics and commonsense plausible.

4.3 Annotation Cost

We annotate scanned model one by one. So, the annotation cost is linear scale with respect to scanned object model number M. Note that only a small set of object have strong object occurrence assumption (e.g. laptop and mouse). So, the complexity of object occurrence annotation is close to $O(M)$. We carry out experiment to find that 10 s is taken to label knowledge for a scanned object model in average, which is minor (one hour for hundreds of objects).

5 Domain Adaptation with Geometry-Guided GAN

Now, we have collection of the rough (RGB) image $\{I_i^r\}_{i=1}^M \in \mathcal{I}^r$ and its corresponding ground truths, instance segmentation $\{I_i^{s\text{-}gt}\}_{i=1}^M \in \mathcal{I}^{s\text{-}gt}$, surface normal $\{I_i^{n\text{-}gt}\}_{i=1}^M \in \mathcal{I}^{n\text{-}gt}$, depth image $\{I_i^{d\text{-}gt}\}_{i=1}^M \in \mathcal{I}^{d\text{-}gt}$. Besides, the real image captured from targeting environment is denoted as $\{I_j\}_{j=1}^N$. M, N are the sample sizes for rendered samples and real samples. With these data, we can embark on training GeoGAN.

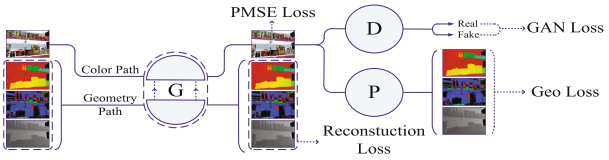

Fig. 4. The GDP structure consists of three components: a generator (G), a discriminator (D), and a predictor (P), along with four loss: LSGAN loss (GAN loss), Structure loss, Reconstruction loss (L1 loss), Geometry-guided loss (Geo loss).

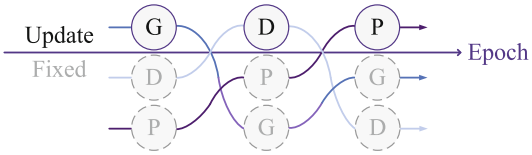

Fig. 5. Iterative optimization framework. As the epoch goes, G, D and P are updated as presented. While one component is updating, the other two are fixed.

5.1 Objective Function

GeoGAN has a "GDP" structure, as sketched in Fig. 4, which comprises a generator (G), a discriminator (D) and a predictor (P) which serves as a geometry prior guidance. Such structure leads to the design of the objective function, which consists of four loss functions that will be presented in what follows.

LSGAN Loss. We adopt a least-square generative adversarial objective (LSGAN) [24] to help G and D training stable. The LSGAN adversarial loss can be written as

$$\mathcal{L}_{GAN}(G, D) = \mathbb{E}_{y \sim p_{data}(y)}[(D(y) - 1)^2] + \mathbb{E}_{x \sim p_{data}(x)}[(D(G(x)))^2], \quad (5)$$

x and y stand for a sample from the rough image and the real image domain respectively.

We denote the output of the generator with parameter Φ_G for i^{th} rough image as I_i^*, i.e. $I_i^* \triangleq G(I_i^r|\Phi_G)$.

Structure Loss. A structure loss is introduced to ensure I_i^* maintains the original structure of I_i^r. A Pairwise Mean Square Error (PMSE) loss is imported from [7], expressed as:

$$\mathcal{L}_{PMSE}(G) = \frac{1}{N}\sum_i (I_i^r - I_i^*)^2 - \frac{1}{N^2}(\sum_{i,j}(I_i^r - I_i^*))^2. \quad (6)$$

Reconstruction Loss. To ensure the geometry information successfully encoded in the network. We also use ℓ^1 as a reconstruction loss for the geometric images.

$$\mathcal{L}_{rec}(G) = ||[I^r, I^s, I^n, I^d|\Phi_G]_{rec}, [I^r, I^s, I^n, I^d]||_1 \quad (7)$$

Geometry-Guided Loss. Given an excellent geometry predictor, a high-quality image should be able to produce desirable instance segmentation, depth map and normal map. It is a useful criterion that judges whether I_i^* is qualified or not. An unqualified image (with artifacts, distorted structure) will induce large geometry-guided loss (Geo Loss).

To achieve this goal, we pretrained the predictor with following formula:

$$[I^s, I^n, I^d] = P(I|\Phi_P), \quad (8)$$

It means given an input image I, with the parameter Φ_P, the predictor can output instance segmentation I^s, normal map I^n and depth map I^d respectively. In the first few iterations, the predictor is pretrained with the rough image, that is, $I = I^r$. When the generator starts to produce reasonable results, Φ_P can be updated with $I = I^*$. And then, the predictor is ready to supervise the generator, and Φ_G will be updated as follow:

$$\mathcal{L}_{Geo}(G, P) = ||P(I_i^*|\Phi_P), [I_i^{s\text{-}gt}, I_i^{n\text{-}gt}, I_i^{d\text{-}gt}]||_2^2. \tag{9}$$

In this equation, Φ_P is not updated, and it is a ℓ^2 loss.

Overall Objective Function. In sum, our objective function can be expressed as:

$$\begin{aligned} &\min_{\Phi_G} \max_{\Phi_D} \lambda_1 \mathcal{L}_{GAN}(G, D) + \lambda_2 \mathcal{L}_{PMSE}(G) + \lambda_3 \mathcal{L}_{rec}(G) + \lambda_4 \mathcal{L}_{Geo}(G, P), \\ &\min_{\Phi_P} \mathcal{L}_{Geo}(G, P). \end{aligned} \tag{10}$$

It reveals the iterative optimization, as shown in Fig. 5.

5.2 Implementation

Dual Path Generator (G). Our generator has dual forward data paths (color path and geometry path), which help to integrate the color and geometry information. For color path, input rough image will firstly pass three convolutional layers, and then downsample to 64×64 and pass 6 resnet blocks [15]. After that, output feature maps are upsampled to 256×256 with bilinear upsampling. During upsampling, color information path will concatenate feature maps from geometry information path.

Geometry information are firstly convolutioned to feature maps and concatenated together, resulting in a 3-dimensional 256×256 feature map before passing to geometry path described below. After the last layer, we split the output of the last layer into three parts, and produce three reconstruction images for three kinds of geometric images.

Let $3n64s1$ denote 3×3-Convolution-InstanceNorm-ReLU layer with 64 filters and stride 1. *Rk* denotes a residual block that contains two 3×3 convolutional layers with the same number of filters on both. *upk* denotes a bilinear upsampling layer followed with a 3×3 Convolution-InstanceNorm-ReLU layer with k filters and stride 1.

The generator architecture is:

color path: 7n3s1-3n64s2-3n128s2-R256-R256-R256-R256-R256-R256-up512-up256
geometry path: 7n3s1-3n64s2-3n128s2-R256-R256-R256-R256-R256-R256-up256-up128.

Markovian Discriminator (D). The discriminator is a typical PatchGAN or Markovian discriminator described in [16,19,20]. We also found 70×70 is a proper receptive field size, hence the architecture is exactly like [16].

Geometry Predictor (P). FCN-like networks [23] or UNet [29] are good candidates for the geometry predictor. In implementation, we choose a UNet architecture. *downk* denotes a 3×3 Convolution-InstanceNorm-LeakyReLU layer with k filters and stride 2, the slope of leaky ReLU is 0.2. *upk* denotes a bilinear upsampling layer followed with a 3×3 Convolution-InstanceNorm-ReLU layer with k filters and stride 1. k in *upk* is 2 times larger than that in *downk*, since a skip connection between corresponding layers. After the last layer, feature maps are split into three parts and convolution to a three dimension layer separately, activated by tanh function.

The predictor architecture is: down64-down128-down256-down512-down512-down512-up1024-up1024-up1024-up512-up256-up128.

Training Details. Adam optimizer [18] is used for all three "GDP" components, with batch size of 1. G, D and P are trained from scratch. We firstly trained geometry predictor with 5 epochs to get a good initialization, then began the iterative procedures. In the iterative procedures, learning rate for the first 100 epochs are 0.0002 and linearly decay to zero in the next 100 epochs. All training images are of size 256×256.

All models are trained with $\lambda_1 = 2$, $\lambda_2 = 5$, $\lambda_3 = 10$, $\lambda_4 = 3$ in Eq. 10. The generator is trained twice before the discriminator updates once.

6 Instance-60K Building Process

As we found no existing Instance segmentation datasets [5,8,22] can benchmark our task, we have to build a new dataset to benchmark our method.

Instance-60K is an ongoing effort to annotate instance segmentation for scenes can be scanned. Currently it contains three representative scenes, namely supermarket shelf, office desk and tote. These three scenes are chosen since they potentially benefit real-world applications in the future. Supermarket cases are well-suited to self-service supermarkets like Amazon Go[3]. Home robots will always meet the scene of an office desk. The tote is in the same setting as Amazon Robotic Challenge.

To note that our pipeline does not restrict to these three scenes, technically any scenes can be simulated are suitable to our pipeline.

Shelf scene has objects of 30 categories, which items such as soft drinks, biscuits, and tissues. 15 categories for desk scene and tote scene. All are common objects in the corresponding scenes. Objects and scenes are scanned for building SOM dataset as described in Sect. 3.

For instance-60K dataset, these objects are placed in corresponding scenes and then videotaped by iPhone5s under various viewpoints. We arranged 10 layouts for the shelf, and over 100 layouts for desk and tote. Videos are then sliced into 6000 images in total, 2000 for each scene. The number of labeled instance is 60894, that is the reason why we call it instance-60K. We have average

[3] https://www.amazon.com/b?node=16008589011.

Fig. 6. Representative images and manual annotations in the Instance-60K dataset.

966 instances per category. This scale is about three times larger than PASCAL VOC [8] level (346 instances per category), so it is qualified to benchmark this problem. Again, we found instance segmentation annotation is laborious, it took more than 4000 man-hours on building this dataset. Some representative real images and annotation are shown in Fig. 6. As we can see, annotating them is time-consuming.

7 Evaluation

In this section, we evaluate our generated instance segmentation samples quantitatively and qualitatively (Fig. 7).

Table 1. mAP results on real, rough, fake, fake$_{plus}$ models of different scenes with Mask R-CNN.

	Shelf				Desk				Tote			
	real	rough	fake	fake$_{plus}$	real	rough	fake	fake$_{plus}$	real	rough	fake	fake$_{plus}$
mAP@0.5	79.75	18.10	49.11	**66.31**	88.24	43.81	57.07	**82.07**	90.06	28.67	61.40	**82.69**
mAP@0.7	67.02	10.53	37.56	**47.25**	73.75	35.14	45.44	**71.82**	85.10	16.87	50.13	**76.84**

7.1 Evaluation on Instance-60K

We employed instance segmentation tasks to evaluate on generated samples. To prove that the proposed pipeline generally works, we will report results using Mask R-CNN [14]. We train segmentation model on resulting images produced by our GeoGAN. The trained model is denoted as "fake-model". Likewise, model trained on rough images is denoted as "rough-model". One question we should ask is that how "fake-model" compare to models train on real images. To answer this question, we train models on training set of instance-60K dataset, which is denoted as "real-model". It is pre-trained on COCO dataset [22].

Training procedures on real images strictly follow the procedures mentioned in [14]. We find the learning rate for real images is not workable to rough and

Fig. 7. Refinement of GAN. Refined column is the result of GeoGAN and rough column is the rendered image. Apparent improvement on lighting conditions and texture can be observed.

GAN generated images, so we lower the learning rate and make it decay earlier. All models are trained with 4500 images, though we can generate endless training sample for "rough-model" and "fake-model", since "real-model" only can train on 4500 images in the training set of instance-60K dataset. Finally, all models are evaluated on testing set of instance-60K dataset.

Experiment results shown in Table 1. Overall mAP of the rough image is generally low, while "fake-model" significantly outperformed it. Noticeably, it still has a clear gap between "fake-model" results and real one, though the gap has been bridged a lot. Naturally, we would like to know how many refined training images is sufficient to achieve comparable results with "real-model". Hence, we conducted experiments on 15000 GAN generated images, and named model as "fake$_{plus}$-model". As we can see from Table 1, "fake$_{plus}$" and "real" is really close. We try to augment more training samples to "fake$_{plus}$-model", but, the improvement is marginal. In this way, our synthetic "images + annotation" is comparable with "real image + human annotation" for instance segmentation (Fig. 8).

Fig. 8. Qualitative results visualization of rough, fake, fake$_{plus}$ and real model respectively.

The results for real-model may imply that our instance-60K is not that difficult for Mask R-CNN. Extension of the dataset is on-going. However, it is undeniable that the dataset is capable of proving the ability of GeoGAN.

In contrast to exhausting annotation using over 1000 human-hours per scene, our pipeline takes 0.7 human-hours per scene. Admittedly, the results suffer from performance loss, but save the whole task 3-order of human-hours.

7.2 Comparison with Other Domain Adaptation Framework

Previous domain adaptation framework focus on different tasks, such as gaze and hand pose estimation [32], object classification and 6D pose estimation [3]. To the best of our knowledge, we are the first to propose a GAN-based framework to do instance segmentation. Comparison with each other is indirect. We reproduced the work of [32] and [3]. For [3], we substituted the task component with our P. The experiments are conducted on the scenes same in the paper. Results are shown in Fig. 9 and Table 2.

Fig. 9. Qualitative comparison of our pipeline and [3], [32]. The background of generated images from [3] are damaged since they use a masked-PMSE loss.

7.3 Ablation Study

Ablation study is carried out by removing geometry-guided loss and structure loss separately. Extended ablation study on the specific geometric information in the geometry path is reported in the supplementary file. We applied Mask R-CNN to train the segmentation models on resulting images from GeoGAN without geometry-guided loss (denoted as "fake$_{plus,\text{w/o-}geo}$-model") or structure loss (denoted as "fake$_{plus,\text{w/o-}pmse}$-model"). As we can see, it suffers a significant performance loss when removing geometry-guided loss or structure loss. Besides, we also need to prove the necessity of reasoning system. After removing reasoning system, resulting in unrealistic images and performance loss. Results are shown in Table 3 (Fig. 10).

Table 2. Quantitative comparison of our pipeline and [3], [32].

mAP			0.5	0.7
Mask R-CNN	Shelf	$\text{fake}_{plus,ours}$	66.31	47.25
		$\text{fake}_{plus,[25]}$	31.46	20.88
		$\text{fake}_{plus,[13]}$	56.16	36.04
	Desk	$\text{fake}_{plus,ours}$	82.07	71.82
		$\text{fake}_{plus,[25]}$	44.33	29.93
		$\text{fake}_{plus,[13]}$	69.54	57.27
	Tote	$\text{fake}_{plus,ours}$	82.69	76.84
		$\text{fake}_{plus,[25]}$	42.50	33.61
		$\text{fake}_{plus,[13]}$	70.73	62.68

Table 3. mAP results of ablation study on Mask R-CNN.

mAP			0.5	0.7
Mask R-CNN	Shelf	fake_{plus}	66.31	47.25
		$\text{fake}_{plus,\text{w/o-}geo}$	48.52	31.17
		$\text{fake}_{plus,\text{w/o-}pmse}$	27.33	19.24
		$\text{fake}_{plus,\text{w/o-}reason}$	15.21	8.44
	Desk	fake_{plus}	82.07	71.82
		$\text{fake}_{plus,\text{w/o-}geo}$	63.99	55.23
		$\text{fake}_{plus,\text{w/o-}pmse}$	45.05	34.51
		$\text{fake}_{plus,\text{w/o-}reason}$	18.36	9.71
	Tote	fake_{plus}	82.69	76.84
		$\text{fake}_{plus,\text{w/o-}geo}$	64.22	53.31
		$\text{fake}_{plus,\text{w/o-}pmse}$	46.44	35.62
		$\text{fake}_{plus,\text{w/o-}reason}$	20.05	12.43

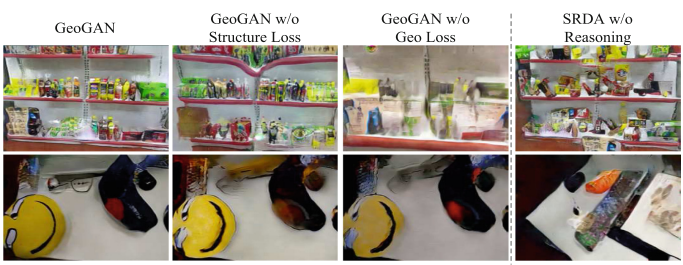

Fig. 10. Samples to illustrate the efficacy of structure loss, geometry-guided loss in GeoGAN and reasoning system in our pipeline.

8 Limitations and Future Work

If the environmental background changes dynamically, we should scan a large number of environmental backgrounds to cover this variance and take much effort. Due to the limitations of the physics engine, it is hard to handle highly non-rigid objects such as a towel. For another limitation, our method does not consider illumination effects in rendering, since it is much more complicated. GeoGAN that transfers illumination conditions of the real image may partially address this problem, but it is still imperfect. In addition, the size of our benchmark dataset is relatively small in comparison with COCO. Future work is necessary to address these limitations.

Acknowledgement. This work is supported in part by the National Key R&D Program of China, No. 2017YFA0700800, National Natural Science Foundation of China under Grants 61772332 and SenseTime Ltd.

References

1. Alhaija, H.A., Mustikovela, S.K., Mescheder, L., Geiger, A., Rother, C.: Augmented reality meets deep learning for car instance segmentation in urban scenes. In: Proceedings of the British Machine Vision Conference, vol. 3 (2017)
2. Benaim, S., Wolf, L.: One-sided unsupervised domain mapping. In: Advances in Neural Information Processing Systems (2017)
3. Bousmalis, K., Silberman, N., Dohan, D., Erhan, D., Krishnan, D.: Unsupervised pixel-level domain adaptation with generative adversarial networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017
4. Chen, Q., Koltun, V.: Photographic image synthesis with cascaded refinement networks. In: IEEE International Conference on Computer Vision (ICCV) (2017)
5. Cordts, M., et al.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
6. Dai, J., He, K., Sun, J.: Instance-aware semantic segmentation via multi-task network cascades. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3150–3158, June 2016. DOI: https://doi.org/10.1109/CVPR.2016.343
7. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: International Conference on Neural Information Processing Systems, pp. 2366–2374 (2014)
8. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html
9. Fisher, M., Ritchie, D., Savva, M., Funkhouser, T., Hanrahan, P.: Example-based synthesis of 3D object arrangements. ACM Trans. Graph. **31**(6), 135 (2012)
10. Fuhrmann, S., Langguth, F., Goesele, M.: MVE-A multi-view reconstruction environment. In: GCH, pp. 11–18 (2014)
11. Georgakis, G., Mousavian, A., Berg, A.C., Kosecka, J.: Synthesizing training data for object detection in indoor scenes. arXiv preprint arXiv:1702.07836 (2017)

12. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
13. Handa, A., Ptucean, V., Stent, S., Cipolla, R.: SceneNet: an annotated model generator for indoor scene understanding. In: IEEE International Conference on Robotics and Automation, pp. 5737–5743 (2016)
14. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. arXiv preprint arXiv:1703.06870 (2017)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Computer Vision and Pattern Recognition, pp. 770–778 (2016)
16. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017
17. Kim, T., Cha, M., Kim, H., Lee, J., Kim, J.: Learning to discover cross-domain relations with generative adversarial networks. In: IEEE International Conference on Computer Vision (ICCV) (2017)
18. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Computer Science (2014)
19. Ledig, C., et al.: Photo-realistic single image super-resolution using a generative adversarial network (2016)
20. Li, C., Wand, M.: Precomputed real-time texture synthesis with Markovian generative adversarial networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 702–716. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_43
21. Li, Y., Qi, H., Dai, J., Ji, X., Wei, Y.: Fully convolutional instance-aware semantic segmentation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
22. Lin, T.Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
23. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
24. Mao, X., Li, Q., Xie, H., Lau, R.Y.K., Wang, Z., Smolley, S.P.: Least squares generative adversarial networks (2016)
25. Mccormac, J., Handa, A., Leutenegger, S., Davison, A.J.: SceneNet RGB-D: 5M photorealistic images of synthetic indoor trajectories with ground truth (2017)
26. Merrell, P., Schkufza, E., Li, Z., Agrawala, M., Koltun, V.: Interactive furniture layout using interior design guidelines. In: ACM SIGGRAPH, p. 87 (2011)
27. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks
28. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems (NIPS) (2015)
29. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
30. Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.: The SYNTHIA dataset: a large collection of synthetic images for semantic segmentation of urban scenes. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)

31. Rusu, A.A., Vecerik, M., Rothrl, T., Heess, N., Pascanu, R., Hadsell, R.: Sim-to-real robot learning from pixels with progressive nets (2016)
32. Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., Webb, R.: Learning from simulated and unsupervised images through adversarial training. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
33. Sixt, L., Wild, B., Landgraf, T.: RenderGAN: generating realistic labeled data. arXiv preprint arXiv:1611.01331 (2016)
34. Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image (2016)
35. Su, H., Qi, C.R., Li, Y., Guibas, L.J.: Render for CNN: viewpoint estimation in images using CNNs trained with rendered 3D model views. In: The IEEE International Conference on Computer Vision (ICCV), December 2015
36. Taigman, Y., Polyak, A., Wolf, L.: Unsupervised cross-domain image generation (2016)
37. Tasora, A., et al.: Chrono: an open source multi-physics dynamics engine. In: Kozubek, T., Blaheta, R., Šístek, J., Rozložník, M., Čermák, M. (eds.) HPCSE 2015. LNCS, vol. 9611, pp. 19–49. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-40361-8_2
38. Tzeng, E., et al.: Towards adapting deep visuomotor representations from simulated to real environments. In: Computer Science (2015)
39. Tzionas, D., Gall, J.: 3D object reconstruction from hand-object interactions. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 729–737 (2015)
40. Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J.: Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In: Advances in Neural Information Processing Systems, pp. 82–90 (2016)
41. Yi, Z., Zhang, H., Gong, P.T., et al.: DualGAN: unsupervised dual learning for image-to-image translation. In: IEEE International Conference on Computer Vision (ICCV) (2017)
42. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: IEEE International Conference on Computer Vision (ICCV) (2017)
43. Zhu, Y., et al.: Target-driven visual navigation in indoor scenes using deep reinforcement learning (2016)