Quasi-Balanced Self-Training on Noise-Aware Synthesis of Object Point Clouds for Closing Domain Gap

Yongwei Chen[1,2*], Zihao Wang[1*], Longkun Zou[1], Ke Chen[1,3†], and Kui Jia[1,3†]

[1] South China University of Technology
[2] DexForce Co. Ltd.
[3] Peng Cheng Laboratory
{eecyw,eezihaowang,eelongkunzou}@mail.scut.edu.cn,
{chenk,kuijia}@scut.edu.cn

Abstract. Semantic analyses of object point clouds are largely driven by releasing of benchmarking datasets, including synthetic ones whose instances are sampled from object CAD models. However, learning from synthetic data may not generalize to practical scenarios, where point clouds are typically incomplete, non-uniformly distributed, and noisy. Such a challenge of Simulation-to-Reality (Sim2Real) domain gap could be mitigated via learning algorithms of domain adaptation; however, we argue that generation of synthetic point clouds via more physically realistic rendering is a powerful alternative, as systematic non-uniform noise patterns can be captured. To this end, we propose an integrated scheme consisting of physically realistic synthesis of object point clouds via rendering stereo images via projection of speckle patterns onto CAD models and a novel quasi-balanced self-training designed for more balanced data distribution by sparsity-driven selection of pseudo labeled samples for long tailed classes. Experiment results can verify the effectiveness of our method as well as both of its modules for unsupervised domain adaptation on point cloud classification, achieving the state-of-the-art performance. Source codes and the SpeckleNet synthetic dataset are available at https://github.com/Gorilla-Lab-SCUT/QS3.

1 Introduction

As raw observations of 3D sensors, object point clouds are popularly used for a variety of 3D semantic analysis tasks, including shape classification [39,40,51,15,32], part segmentation of object surface [39,40,28], estimation of object poses in indoor scenes [8,14,30], and 3D detection in autonomous driving [26,55,11]. The current progress along this line has largely been driven by the publicly released synthetic benchmarks (*e.g.*, the ModelNet [53] and the ShapeNet [7]). Object point cloud instances in those datasets are sampled from object CAD models,

*Equal contribution
†Corresponding authors

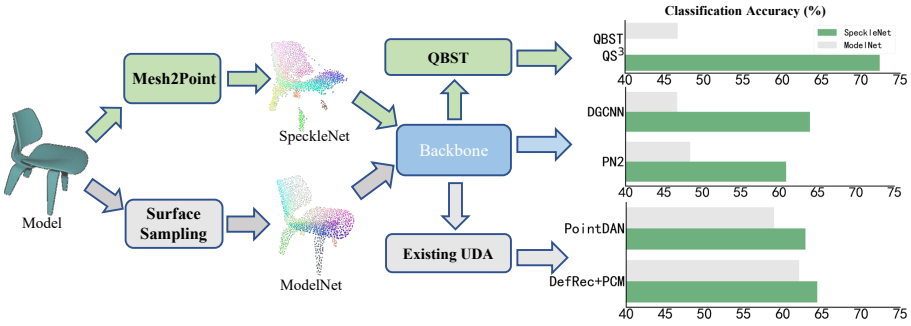

Fig. 1. We propose an integrated scheme of Quasi-balanced Self-training on Speckle-projected Synthesis (QS^3) to cope with shape and density shift between synthetic and real point clouds. Given identical CAD models, we generate a point cloud SpeckleNet dataset simulating realistic noises in stereo imaging and matching; while point clouds in existing ModelNet dataset [53] are sampled from object surface of those models. Moreover, we design a novel quasi-balanced self-training (QBST) strategy to further boost the UDA performance. In comparison with two representative UDA methods (DefRec+PCM [1], PointDAN [41]) and two representative point cloud classification networks (PointNet++ [40], DGCNN [51]), our integrated QS^3 can perform consistently better, when evaluating on real world data – an adapted DepthScanNet dataset.

which are ideally noise-free and uniformly distributed on the complete object surface. Very high accuracies (*e.g.*, 96.2% on classification accuracy of the DGCNN [51] with the ModelNet10 dataset [41]) can be gained by a large number of recent point classifiers [39,51,40,32,15,31]. However, point clouds collected in real-world applications are typically incomplete, non-uniformly distributed and noisy, due to unavoidable sensor noises and interaction with contextual objects. In this way, classification of real-world object point clouds can be difficult, as pointed out in recent works [50,54] (*e.g.*, 78.4% on classification accuracy by the DGCNN with the realistic ScanNet10 [41]).

Classification of object point clouds is made more challenging when considering Sim2Real domain discrepancy of semantic patterns (*i.e.*, training point classifiers on synthetic data and testing on real data), which has been investigated in recent works [2,41,1,58,57]. Such a problem can be formulated into a practical unsupervised domain adaptation (UDA) problem, where supervision signals are only available for source synthetic data. Existing methods either adapt effective UDA algorithms on 2D images to point clouds directly as one application [2] or capture domain-invariant geometry patterns in self-supervised learning style [58,1]. Although these methods have gained remarkable success on alleviating the suffering of Sim2Real domain gap in the style of point distribution shift, our paper attempts to address the challenge in a new perspective – an integrated scheme of physically realistic synthesis of object point clouds for domain-adapted feature learning. Our motivation is based on the following observation: point distribution shift can be composed of two types of point defor-

mation from ideal synthetic point clouds: 1) shape variations along the normal directions of surface; and 2) density variations along the tangent plane of object surface.

On one hand, in view of data-driven nature of deep models, we argue that generation of synthetic point clouds to physically simulate realistic noises is a powerful alternative for coping with the former shape variations, which can intuitively be interpreted as input transformation to align geometric patterns of point clouds. Recently, in addition to the straightforward simulation by adding random noises, data augmentation on synthetic point clouds can only mimic partiality and non-uniformness of real point clouds by deformation on a (sub)set of points, whose output is termed as augmented point clouds in this paper. However, only a few works [17,38,37] focus on simulating realistic non-uniform systematic sensor noises in photorealistic rendering and depth image generation, whose output is utilized to convert point clouds. In view of this, we introduce a Mesh-to-Point (Mesh2Point) pipeline to generate synthetic point clouds based on a photorealistic rendering of a stereo camera. Specifically speaking, by using physically based rendering to simulate the workflow of a real-world depth scanner, our Mesh2Point can approach non-uniform systematic noises of depth sensors affected by projection patterns and the method of depth computation.

On the other hand, physically realistic synthesis of object point clouds via a virtual depth sensor cannot ensure that similar point densities to those of realistic target data, which encourages us to use UDA techniques to further bridge Sim2Real domain gap. Inspired by positive effects of balanced data distribution to suppress negative transfer [52,21,49], we design a novel quasi-balanced self-training (QBST) with updating the sample selector with pseudo labeled target samples only, which can effectively model geometric patterns of real point clouds in target domain. Note that, the class balanced sampling strategy to ease the long-tailed data distribution has been widely adopted in different fields, but we argue that it remains non-trivial in the context of UDA. Its main challenge lies in learning to construct a class-balanced self-training set with diverse samples via selection and pseudo annotation on unlabeled target samples, whose data distribution is unknown. Our QBST method is simple yet effective, owing to 1) sparsity-sensitive weight sampling filtered by a confidence threshold; and 2) self-training only with pseudo labeled target samples, thus resulting in robust performance against long tailed distributions. Experiments on a raw benchmark – DepthScanNet10 of the ScanNet, without any pre-processing on object samples except resizing, can confirm that our integrated scheme of Quasi-balanced Self-training on Speckle-projected Synthesis (QS^3) can significantly improve performance on the challenging Sim2Real UDA task, as shown in Fig. 1.

Main contributions of this paper are as follows.

- We propose a unique QS^3 scheme of integrating a physically realistic synthesis of object point clouds and a UDA point classifier, to jointly cope with shape and density shift between synthetic and real point clouds.
- A new synthetic dataset – the SpeckleNet is constructed via the Mesh2Point generation, which can be readily scaled given sufficient CAD models.

- A novel quasi-balanced self-training method is proposed to inhibit negative transfer, owing to balanced data distribution via sparsity-driven selection of pseudo-labeled target samples.
- Experiment results of the proposed QS^3 can achieve the state-of-the-art performance on the challenging Sim2Real domain adaptation task.

2 Related Work

UDA on Point Cloud Classification – Recently, a few works [41,1,58] propose the problem of UDA on an irregular point-based representation, which inherits the challenge of semantic gap as other UDA problems on images and also has its specific challenge of domain-agnostic feature encoding from local geometries of point clouds. Qin *et al.* [41] explore the first attempt of UDA on point cloud classification by explicitly aligning local features across domains. Achituve *et al.* [1] propose to incorporate domain-insensitive local geometric patterns, in a self-supervised reconstruction from partially distorted point clouds, into a global representation, with a simple yet effective data augmentation of point cloud mixup to inherently alleviate imbalanced data distribution. Zou *et al.* [58] combine self-paced self-training for preserving intrinsic target discrimination and self-supervised learning for domain invariant geometric features, which can capture both global and local geometric patterns invariant across domains. Different from existing methods concerning superior cross-domain generalization via feature alignment, in our paper, we propose an integrated scheme consisting of the synthesis of realistic point clouds and a quasi-balanced self-training, which can be interpreted as alignment in both input and feature space, to reduce negative effects of domain gap.

Generation of Synthetic Data – Different from benchmarking image classifiers on real images, recent progress of geometric deep learning on point clouds has been largely driven by synthetic datasets [7,53,12,20,18,46,19,56], in view of difficulties in acquiring and annotating real-world 3D data. Recent works [12,56,27] mainly focus on utilizing Physically Based Rendering (PBR) for the synthesis of photorealistic RGB images, while depth images as a byproduct are typically noise-free or simply perturbed by Gaussian noises, which are evidently different from non-uniform systematic noises in practice. Such an observation encourages a number of works to explore physical simulation of 3D sensors to approach realistic noises. Early attempts [19,4,6,35] in a theoretical style fail to cover all kinds of realistic noises. Recently, a number of works including ours prefer the Physically Based Rendering owing to its capability of replicating realistic systematic noises. Specifically, based on PBR, Blensor [17] as well as [13,34,48,42] are able to simulate time-of-flight based 3D sensors while other works [16,24,38,37] mainly focus on simulation of structured light based ones. The most relevant works to ours are DepthSynth [38] and DDS [37], as all of these methods employ speckle pattern projection in PBR for realistic depth acquisition during simulation. However, both of DepthSynth [38] and DDS [37] concern about realistic simulation of depth sensors only, while the goal of our

synthesis in the context of UDA is to mitigate domain gap, coupled with a new quasi-balanced self-training in a unified QS^3 scheme.

Self-training – Self-training utilizes pseudo labels generated from predictions of a model learned on labeled data, as supervision signals for unlabeled data, which is widely used in semi-supervised learning [25,45,3] and UDA [44,43]. Sohn *et al.* [45] use pseudo labels predicted from a weakly-augmented input image as supervision of its strongly-augmented counterpart; for semantic segmentation on 2D images, while Zou *et al.* [59] propose a self-paced learning based self-training framework (SPST), which is formulated as a self-paced learning with latent variable objective optimization [23]. However, these methods directly choose pseudo-labels with high prediction confidence, which will result in model bias towards easy classes and thus ruin the transforming performance for the hard classes. To solve this problem, a class-balanced self-training (CBST) scheme is proposed in [59] for semantic segmentation, which shows comparable domain adaptation performance to the best adversarial training based methods. The method in [29] proposes a self-motivated pyramid curriculum domain adaptation method using self-training. More recently, CRST [60] further integrates a variety of confidence regularizers to CBST [59], producing better domain adaption results. Our QBST method shares similar spirit with CBST [59] to neglect negative transfer caused by imbalanced data distribution, but can encourage a more balanced self-training dataset of high confident samples, regardless of source data distribution.

3 Methodology

For unsupervised domain adaptation on point cloud classification, given a labeled source domain $\mathcal{S} = \{\mathcal{P}_i^s\}_{i=1}^{n_s}$ with their corresponding class labels $\{y^s\}_{i=1}^{n_s} \in \mathcal{Y}$ and an unlabeled target domain $\mathcal{T} = \{\mathcal{P}_i^t\}_{i=1}^{n_t}$, where n_s and n_t denote the size of samples in source and target domains respectively and point cloud $\mathcal{P} \in \mathcal{X}$ consists of a set of 3d coordinates covering object surface, the semantic label space $\mathcal{Y}$ is shared between both domains. The objective of UDA on point sets is to learn a domain-adapted mapping function $\Phi : \mathcal{X} \to \mathcal{Y}$ that classifies any testing sample $\mathcal{P}$ from target domain $\mathcal{T}$ correctly into one of $K = |\mathcal{Y}|$ object categories. In the context of deep learning, the mapping function Φ can be decomposed into a cascade of a feature encoder $\Phi_{\text{fea}} : \mathcal{X} \to \mathcal{F}$ for any input $\mathcal{P}$ and a classifier $\Phi_{\text{cls}} : \mathcal{F} \to \mathcal{Y}$ as follows: $\Phi(\mathcal{P}) = \Phi_{\text{cls}}(\boldsymbol{F}) \circ \Phi_{\text{fea}}(\mathcal{P})$, where the feature output $\boldsymbol{F} \in \mathcal{F}$ of $\Phi_{\text{fea}}(\mathcal{P})$ and $\mathcal{F}$ denotes feature space.

Existing UDA classifiers on point sets [41,1,58] concern on reducing domain discrepancy of feature encoding Φ_{fea}, while our paper starts from the origin of Sim2Real domain gap, which is caused by point distribution shift of the shape representations of CAD models. In other words, given an identical CAD model, the procedure of generating synthetic point clouds determines Sim2Real domain gap. As a result, the problem of this paper can be reformulated as the following: given a set of labeled CAD models $\mathcal{M} = \{M_i, y_i\}_{i=1}^{n_s}$ as source domain $\mathcal{S}$ and an unlabeled set of real point clouds $\{\mathcal{P}_i^t\}_{i=1}^{n_t}$ as target domain $\mathcal{T}$, the point cloud of source domain $\{\mathcal{P}_i^s\}$ are generated from each CAD model M. The goal is to

learn a feature mapping from point clouds $\mathcal{P} \in \mathcal{X}$ to the labels $y \in \mathcal{Y}$. Most of existing synthetic point clouds are directly sampled from CAD models' surface and optionally with moderate pre-defined noises, while the remaining ones can be obtained via physically realistic simulation of rendering of depth images.

In a geometric perspective, we argue that point distribution shift can be decomposed into 1) shape changes of each point along the normal direction of surface due to all kinds of noises and 2) points' density changes along their tangent plane (*i.e.*, approximately equivalent to the object surface). Such a geometric interpretation of Sim2Real domain gap encourages us to propose an integrated scheme for domain-adapted feature learning – Quasi-balanced Self-training on Speckle-projected Synthesis (QS^3), which consists of physically realistic 3D synthesis of realistically systematic noises causing shape changes and feature encoding with domain-invariant patterns on density variations. Specifically, we introduce 1) a Mesh-to-Point data generation based on photorealistic rendering of stereo images with projection of speckle patterns, which generates a new synthetic point cloud dataset – SpeckleNet (see Sec. 3.1), and 2) a novel self-training method on our SpeckleNet data to mitigate the suffering from unknown data distribution of target domain (see Sec. 3.2).

3.1 Synthesis of Realistic Point Clouds via Physical Simulation

Previous works [19,4,6,35] have shown neither theoretical noisy models nor data augment strategies can cover a diversity of noises appearing in real-world depth sensor scanning, *e.g.*, axial noise, shadow noise and structural noise, which are induced by scene illumination, object material, hardware of sensors and composition of the scanned scene (see the survey [33] for more details). Different from existing UDA methods on point cloud classification only enforcing feature alignment across domains, we argue that the synthesis of object point clouds sharing similar statistical distribution with target data is a powerful alternative, via simulating realistic non-uniform noises, which can be viewed as a special alignment in the input space. Inspired by recent works [17,38,37] utilizing Physical Based Rendering (PBR) for depth sensor simulation, we leverage photorealistic rendering to reproduce realistic noises via a virtual depth sensor. As consuming RGB-D sensors (*e.g.*, Microsoft Kinect V1, Intel RealSense) actively projecting speckle patterns are widely used for indoor scene scanning, we build a virtual active stereo based depth scanner based on PBR. Note that, physical simulation of the depth sensor is not limited to projection of speckle patterns adopted in our paper, which can be replaced by other styles of depth sensors such as time-of-flight depth sensors or structured light based depth sensors via fringe pattern projection.

For replicating a typical active stereo based depth sensor, we organize a stereo camera and a projector, as an active illuminant to project pre-defined speckle pattern, in a simulation platform, *e.g.*, Blender [9] adopted in this paper. The whole pipeline of the Mesh-to-Point synthesis is shown in Fig. 2. In details, we set two identical optical imaging sensors (*i.e.*, left camera O_L and right camera O_R in Fig. 2) for a stereo camera, where the image planes of two cameras are coplanar

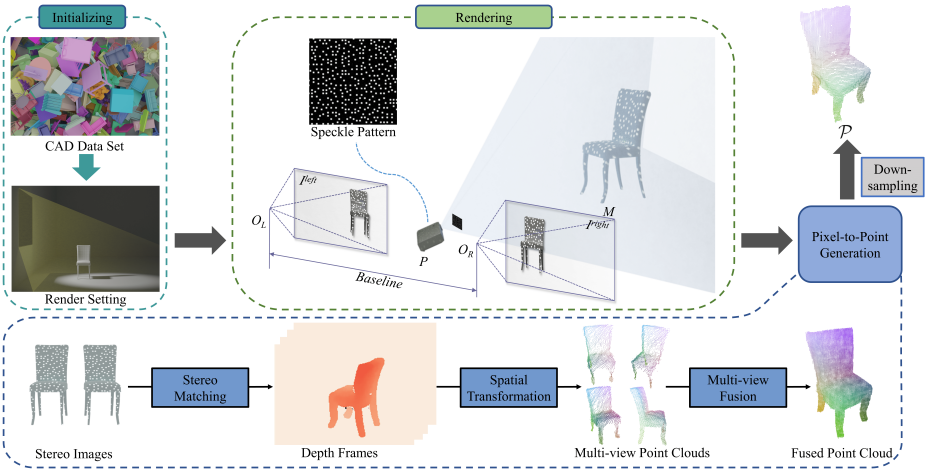

Fig. 2. Pipeline of the Mesh-to-Point method. During data pre-processing, we first scale the mesh-based model $\mathcal{M}$ to a unit-cube with an arbitrary rotation along the z-axis, which are fed into the Rendering module together with the settings of scene illumination and object reflection. In the Rendering module, we virtually organize a stereo camera and a projector. Given the model M, the projector P actively projects a speckle pattern to M, then the stereo camera (O_L, O_R) outputs stereo images I^{left} and I^{right} to the next Pixel-to-Point Generation module. Depth images are converted via stereo matching on stereo images I^{left} and I^{right}, which are further projected into 3D space using intrinsic parameters and extrinsic poses of the camera. Synthetic point clouds under multiple views are fused to produce a dense one carrying richer geometric information, which is then down-sampled to a sparse point set $\mathcal{P}$ as the final output.

and translation between two cameras only along the x-axis of the left camera. The intrinsic and extrinsic parameters of the stereo camera are defined as prior knowledge in our virtual depth sensor. A projector P is positioned in the middle of stereo cameras, actively projecting pre-defined speckle patterns on object surface to provide visual appearance, which thus benefits for discovering pixel correspondence in stereo matching. Given the CAD set $\mathcal{M}$ together with their labels $y \in \mathcal{Y}$ as input, the bidirectional scattering distribution function (BSDF) material [5] is adopted to model the scattered pattern of light by a surface, by following the default setting of BSDF function in [9] to initialize object models' material. We further place one area light source on top of the object model to be rendered, which can thus provide a uniform scene illumination. Note that, our paper concerns on simulating systematic noises due to modules' precision, with an empirically simplified rendering condition.

With the aforementioned settings, pairs of RGB images $\{I_i^{\text{left}}, I_i^{\text{right}}, y_i\}_{i=1}^{n_s}$ for each CAD model M can be generated via photorealistic rendering. For generating a point cloud $\mathcal{P}$, we firstly perform stereo matching [47] on the stereo RGB images I^{left} and I^{right} to gain disparity maps, each element d of which can be used to compute its depth distance as follows [47]: $(f \cdot b)/d$, where f and b are

the focal length and the baseline of the camera. We further project depth images into 3D space to generate point clouds using intrinsic parameters and extrinsic poses of the camera. To mimic generation of realistic point clouds from RGB-D videos scanning object in multiple poses in practice, synthetic point clouds under multiple camera poses are fused together to produce dense sets, which are then down-sampled to the final point set $\mathcal{P}$.

SpeckleNet10 − For a comparative evaluation, we use the same 10 categories of the ModelNet object models as PointDAN [41], and a new synthetic dataset, namely SpeckleNet10, can be generated via simulation of the above virtual active stereo based depth sensor. Given the same size of CAD models as the ModelNet10, the only difference between ModelNet10 and our SpeckleNet10 lies in the procedure of the synthesis of object point clouds, *i.e.*, reality of simulated noises. Parameter settings of our virtual depth sensor are empirically selected for typical real-world indoor scenarios instead of fitting a specific type depth sensor, based on the fact that practical noises of depth sensors are mainly affected by the types of projection patterns and the corresponding methods to compute depth, rather than extrinsic settings. In our implementation, given an object model located in the scene, a stereo camera is randomly placed 3 to 5 meters away from the model, with an arbitrary elevation angle within $[20^{\text{o}}, 50^{\text{o}}]$ in the simulator, where the baseline distance b between two imaging sensors in the stereo camera is set to 10 cm. For multi-view fusing under different viewing angles, one camera pose is randomly selected as an anchor to satisfy the aforementioned constraints, while the remaining as the variants of the anchored pose, lying in $(-10cm, 10cm)$ in translation and $(-0.1, 0.1)$ in the Euler rotation. With the setting of rendering, we can obtain stereo images with 1080×1080 resolution and utilize the block matching algorithm provided by the OpenCV to compute disparity from stereo images, which are then converted to depth images with a down-sampling operation to $270 * 270$. As point clouds transformed from depth images remain rather dense, sparse point sets of 2048 points are down-sampled via the Farthest Point Sample (FPS), as the input of UDA on point classification.

3.2 Quasi-Balanced Self-Training

Given the generated synthetic point clouds $X_s = \{\mathcal{P}^s\} \in \mathcal{S}$ with their corresponding class labels $Y_s = \{y^s\} \in \mathcal{Y}$ and real point clouds $X_t = \{\mathcal{P}^t\} \in \mathcal{T}$ respectively, the remaining part of domain-adapted feature learning shares the same setting as existing 3D UDA methods. In this section, we propose a simple yet effective quasi-balanced self-training (QBST) method to dynamically select target instances for data balanceness when assigning pseudo labels in self-training. An overview of our QBST is shown in Fig. 3, which consists of three steps below, with iteratively updating by the latter two.

(a) In the *warm-up*, a supervised classification uses labeled source data (i.e., $\{X_s, Y_s\}$) to train a model $\boldsymbol{\Phi_o}$ as an initialized pseudo-label generator $\boldsymbol{G_o}$.
(b) In the *target instance selection with pseudo labels*, $\boldsymbol{G}$ is used to obtain class prediction of unlabeled target data, and confident prediction will be assigned as pseudo labels for selected target samples.

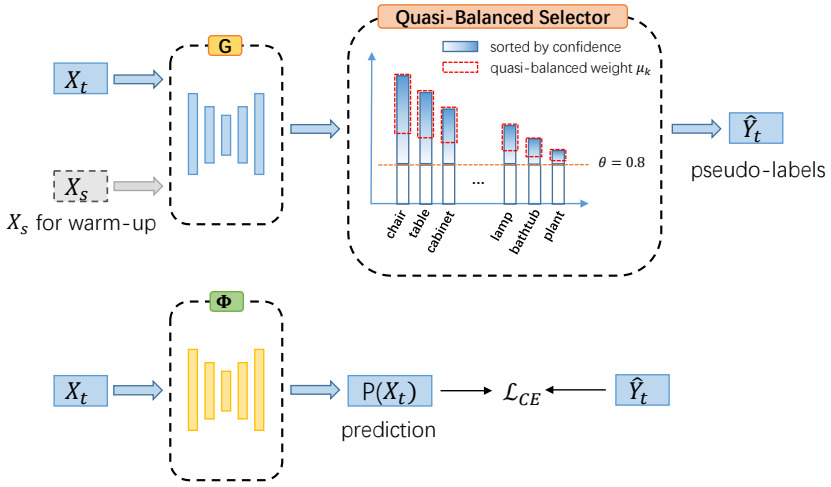

Fig. 3. Overview of our proposed *Quasi-Balanced Self-Training*. **Top:** illustration of a data sparsity-driven strategy for pseudo label generation – a quasi-balanced selector. During self-training, labeled source data is only used for training a model as an initialized pseudo-label generator and does not participate in self-training. In the legend, the blue box indicates predicated confidence (the darker, the higher); the red dashed box denotes selection range by the quasi-balanced weight μ_k, *i.e.*, selecting a μ_k proportion of samples from the target samples classified as class k with predicted confidence greater than θ (*e.g.*, $\theta = 0.8$) as pseudo-label samples. **Bottom:** Training with selected pseudo-labels, where $\mathcal{L}_{CE}$ refers to the cross-entropy loss function.

(c) In the *self-training*, instead of fine-tuning the Φ_o as [58], an initial model Φ_{init} is trained from scratch by selected target point clouds with pseudo-labels.

Pseudo-label Generation – The generic pseudo-label generation strategy can be simplified to the following form when the model parameter w is fixed:

$$\begin{aligned} \min_{\widehat{Y}_t} \quad & \mathcal{L}_{CE}(w, \widehat{Y}_t) = -\frac{1}{|X_t|} \sum_{\mathcal{P}_t \in X_t} \sum_{k=1}^{K} \hat{y}_t^{(k)} \log \frac{p(k|\mathcal{P}_t, w)}{\theta} \\ \text{s.t.} \quad & \hat{y}_t \in \{[\text{onehot}]^K\} \cup 0, \forall \hat{y}_t \in \widehat{Y}_t \end{aligned}$$

where θ indicates the confidence threshold, and $\hat{y}_t = [\hat{y}_t^{(1)}, \hat{y}_t^{(2)}, ..., \hat{y}_t^{(K)}]$ is required to be a one-hot vector or all-zero vector. Therefore, $\hat{y}_t^{(K)}$ can be solved:

$$\hat{y}_t^k = \begin{cases} 1, & \text{if } \ k = \arg\max_k p(k|\mathcal{P}_t, w) \ \text{ and } \ p(k|\mathcal{P}_t, w) > \theta \\ 0, & \text{otherwise.} \end{cases}$$

Inspired by self-paced self-training (SPST) [59], we adopt a threshold θ that gradually increases with self-training iterations evolve (*i.e.*, each iteration increases by a constant ϵ, with more details in Alg. 1 of supplementary material).

Target instances to be assigned with pseudo-labels are selected by following a class-balanced rule [59]: the more sparsity of data in a class k is, the higher weight. To this end, we generate target data distribution from the pseudo-labeled samples selected according to data sparsity. In other words, unlabeled target samples that are predicted as class k are sorted by confidence and selected in descending order of confidence according to the weight of $\mu_k = (1 - \frac{L_k}{L})$, where L_k denotes the number of unlabeled target samples classified into class k, and L is the total number of all unlabeled target samples whose predicted confidence greater than θ. In contrast, CBST [59] selects samples with the same proportion for all object classes, which cannot avoid label noises caused by low-confident mis-classified samples of long-tailed classes. To avoid negative effects of imbalanced data distribution of source data, only selected target samples with pseudo labels are self-trained in our QBST, which is again different from the CBST.

4 Experiments

4.1 Data and Settings

Benchmarking Data in the Sim2Real UDA setting – We construct a new and challenging benchmark for evaluating point cloud classification on Sim2Real domain adaptation, where a synthetic point cloud data is generated as source domain while a real-world one is set as target domain. Inspired by PointDA-10 [41], a pioneering Sim2Real benchmark, we directly use the CAD models of ModelNet10 from the PointDA-10 to generate synthetic data as source domain. Different from point clouds uniformly sampled from the CAD models of object classes as the ModelNet10, realistically-simulated point clouds generated by our Mesh-to-Point are collected as the SpeckleNet presented in Sec. 3.1. The PointDA-10 builds a real ScanNet10 as target domain via extracting the same 10 classes instances from ScanNet [10], where point clouds are directly generated from mesh vertices of the reconstructed surface. As meshing in shape reconstruction can alleviate realistic sensor noises, the point clouds in original ScanNet10 cannot reflect the true challenge of severe noises in practical classification of object point clouds. Moreover, meshing of partial and noisy point clouds is challenging and demands extra processing, which is typically unsuitable for real-time perception. To this end, we follow the setting in the PointDA-10 and choose the same scenes from the ScanNet to generate a more challenging real-world dataset – DepthScanNet10, directly from the raw outputs of depth sensors. Specifically, to gain the point cloud of a single object instance, 2D instance segmentation is first applied to crop a depth image patch of the instance from the whole frame, which can then be converted to a point cloud. Multi-view fusion is employed to gather point clouds from several different viewpoints, followed by down-sampling to produce a fixed size of 2048 points. Note that, only those more than 1000 points will be kept for multi-view fusion whose size of viewpoints is set to 10.
Settings – On the Sim2Real benchmarking data, our proposed pipeline is compared with recent methods. ModelNet10 (**M**) and SpeckleNet10 (**S**) are employed

Table 1. Effects of simulation of realistic noises with real data DepthScanNet (**D**) on classification accuracy (%).

Method	M→D	S→D
Pointnet++ [40]	48.4 ± 1.3	**60.9 ± 0.8**
DGCNN [51]	46.7 ± 1.4	**64.0 ± 1.0**
RSCNN [32]	49.7 ± 1.1	**53.9 ± 0.2**
SimpleView [15]	54.6 ± 0.7	**62.3 ± 1.3**

as source domain, while we choose testing split of DepthScanNet10 (**D**)) as target domain, following the setting of the PointDA-10. Specifically, all object point clouds are normalized within a unit ball and down-sampled to 1024 points using the FPS algorithm. Random rotation along the z-axis and jittering as [39] is employed for data augmentation during training. Moreover, the same data split of the ScanNet10 is used in the DepthScanNet10.

Comparative Methods – We evaluate several UDA classification methods, including Point Domain Adaptation Network (**PointDAN**) [41], Deformation Reconstruction Network with Point Cloud Mixup (**DefRec+PCM**) [1] and Geometry-Aware Self-Training (**GAST**) [58], by following the settings of the best performance in their paper, for simulation-to-reality domain adaptation.

Implementation Details – All the networks are implemented based on PyTorch [36]. Beyond the SpeckleNet10 generated by our Mesh2Point simulator, we utilize DGCNN [51] with Point Cloud Mixup (PCM) [1] as our network baseline, followed by our proposed QBST as the self-training strategy to perform feature alignment on the SpeckleNet10. To implement recent UDA methods on point classification, except for PointDAN [41] using official released source codes, other UDA methods are implemented based on DGCNN [51] as the backbone network. We choose the ADAM [22] as our optimizer with an initial learning rate of 0.001 and weight decay of 0.00005 and an epoch-wise cosine annealing learning rate scheduler for the UDA methods as [58]. With the cross-entropy loss, we train all the methods for 200 epochs with a batch size 16 on an NVIDIA GTX-1080 Ti GPU. In each iteration of self-training, we adopt the ADAM with learning rate 1×10^{-3} and batch size 32 for 10 epochs. The initial threshold θ_0 is set to 0.8, and the constant ϵ is set to 5×10^{-3}. The mean accuracy and standard error of the mean (SEM) is reported on three trials of random seeds.

4.2 Results

Effects of the Synthesis of Realistic Noises with Ordinary Point Cloud Classifiers – We report comparative evaluation on real data **D** of ordinary point cloud classification with four popular classifiers in columns of M→D and S→D of Table 1. Following recent SimpleView [15], the following four classification networks, *i.e.*, PointNet++ [40], DGCNN [51], RSCNN [32] and SimpleView [15], are evaluated and compared. It is evident that all the methods training on synthetic point clouds (from the **S**) of the Mesh2Point method can perform

Table 2. Comparative evaluation in classification accuracy (%) averaged over 3 seeds (± SEM) on the Sim2Real data with recent UDA methods.

Method	M→D	S→D
Supervised	90.4 ± 0.4	90.4 ± 0.4
DGCNN [51] (w/o Adapt)	46.7 ± 1.4	64.0 ± 1.0
PointDAN [41]	58.9 ± 0.9	62.9 ± 1.6
DefRec [1]	57.8 ± 1.1	60.8 ± 0.6
DefRec+PCM [1]	62.1 ± 0.8	64.4 ± 0.7
GAST w/o SPST [58]	62.4 ± 1.1	61.8 ± 1.0
GAST [58]	64.8 ± 1.4	64.4 ± 0.2
QBST (ours)	**66.4** ± 1.1	–
QS^3 (ours)	–	**72.4** ± 0.8

better than those on simply-augmented synthetic data (from the **M**), *i.e.*, results in the column of **S→D** are significantly superior to those in the column of **M→D** in Table 1, which demonstrate our motivation of using physical simulation to alleviate Sim2Real domain gap, owing to capturing realistic noises underlying in real data.

Rationale of Balanced Data Distribution for UDA Point Cloud Classification – In Table 2, a number of recent UDA methods are evaluated and compared on simulation-to-reality tasks. The **Supervised** method trains DGCNN [51] backbone with labeled target data only, and the **DGCNN w/o Adapt** method trains the identical net with only labeled source samples, treated as a reference of the upper and lower performance bound respectively. On one hand, models training on the physically simulated instances (from the **S**) cannot be consistently superior to those on augmented ones (from the **M**). Specifically, superior performance of DefRec, GAST and its degenerated GAST w/o SPST with the **M** to the proposed **S** when evaluation on real data **D**, where domain gap of **M→D** is verified to be larger than those of **S→D** as shown in Table 1. Such a phenomena can be explained by that these self-supervised methods (*i.e.,* DefRec and GAST) rely on capturing domain-invariant geometric patterns from local regions, where synthetic data in the **S** are generated from partially visible surfaces and thus suffers more than those complete ones in the **M** to learn cross-domain local geometric patterns. DefRec+PCM can significantly improve performance on **S→D** over DefRec, which can be credited to the extra PCM data augmentation to enrich data diversity and inherently balance data distribution by increasing samples shared with long-tailed classes. On the other hand, it is observed that all the UDA methods on **M→D** can outperform the backbone DGCNN, while very marginal improvement or even negative transfer performance of most of methods can be gained on the remaining task. With the PCM and SPST modules respectively, negative transfer on DefRec and GAST can be alleviated, which can confirm their effectiveness. As [58], the SPST cannot guarantee the correctness of pseudo labels assigned to target instances but under the assumption that they are mostly correct, which are largely affected by per-

Table 3. Effects of our QBST with classification accuracy (%) averaged over 3 seeds (± SEM) on the sim2real data.

Method	M→D	S→D
Supervised	90.4 ± 0.4	90.4 ± 0.4
DGCNN [51] (w/o Adapt)	46.7 ± 1.4	64.0 ± 1.0
DGCNN+CBST	41.8 ± 2.0	58.6 ± 1.7
DGCNN+SPST	45.2 ± 0.5	63.3 ± 0.5
DGCNN+QBST	45.6 ± 0.4	63.8 ± 0.1
DGCNN+PCM	61.2 ± 0.6	68.5 ± 1.1
DGCNN+PCM+CBST	62.5 ± 3.0	62.9 ± 0.9
DGCNN+PCM+SPST	65.4 ± 0.6	71.9 ± 0.7
DGCNN+PCM+QBST	**66.4 ± 1.1**	**72.4 ± 0.8**

formance of instance selector. Inspired by the success of PCM , it is encouraged to propose a self-training method to achieve balanced data distribution.

Effects of Quasi-balanced Self-Training – With the DGCNN as the identical backbone, we compare our QBST with its direct competitors – SPST and CBST as well as PCM, whose results are reported in Table 3. As aforementioned, the PCM and SPST can effectively inhibit negative transfer, it is observed that both SPST and CBST can work better together with the PCM owing to improvement on selection of confident target samples gained by the PCM. In light of this, based on the backbone DGCNN and the PCM, superior performance of the proposed QBST can confirm its effectiveness by consistently beating its direct competitor SPST and CBST as well as the state-of-the-art DefRec+PCM and GAST (refer to Table 2) for both Sim2Real tasks. Superior performance of our QBST to SPST and CBST can be credited to using data sparsity-sensitive weight sampling on high-confident target samples, while comparative methods (i.e., SPST and CBST) cannot alleviate negative effects from mis-classified samples of long-tailed classes with low confidence.

Comparison with the State-of-the-art Methods – Our QS^3 scheme and the degenerated QBST are compared with recent UDA methods, whose results are shown in Table 2. It is evident that classification accuracy of our QS^3 scheme can reach 72.4%, a large marginal improvement over other UDA methods, when testing on the DepthScanNet. Using our QBST on the ideal synthetic point clouds from the ModelNet (*i.e.,* M→D), superior performance to the state-of-the-art UDA methods can still be achieved, which can again verify the effectiveness of the proposed QBST.

Ablation Studies about Simulation of Realistic Data with Ordinary Point Cloud Classifiers – Motivation of our physically rendering is to simulate non-neglected systematic noises, which is considered as the key factor to reduction of shape shift in Sim2Real domain gap. To verify it, we generate a clean depth image, whose pixels' depth value are directly measured according to distance between the corresponding point on object surface and the camera along its optical axis, to avoid depth computation via stereo imaging and matching. Synthetic data from these ideally clean depth frames are termed as

Table 4. Ablation studies about simulation of realistic noises of stereo rendering and matching with real dataset **D** on classification accuracy (%).

Method	S_c→D	B→D	M_d→D	S→D
Pointnet++ [40]	50.2 ± 2.0	52.4 ± 1.3	57.9 ± 1.2	**60.9 ± 0.8**
DGCNN [51]	53.2 ± 1.6	56.6 ± 2.1	50.4 ± 1.0	**64.0 ± 1.0**
RSCNN [32]	48.8 ± 0.1	51.7 ± 2.0	**56.5 ± 1.0**	53.9 ± 0.2
SimpleView [15]	56.2 ± 1.2	**65.1 ± 0.2**	57.4 ± 0.7	62.3 ± 1.3

a SpeckleNet10-Clean ($\mathbf{S}_c$). Some works [38,37,9] on sensor simulation can be adopted in our scheme to replace the Mesh2Point method. Therefore, we compare one of the representative simulators – Blensor [17], an open-source depth sensor simulator based on Blender [9], which output a set of point clouds (namely Blensor10, **B**) using the same setting as our Specklenet10. Moreover, we also provide a better baseline for the ModelNet10 where we perform random region dropout on point clouds to simulate missing parts typically encountered in real data as further data augmentation. Such an augmented dataset is named as ModelNet10-Dropout ($\mathbf{M}_d$). Results in Table 4 can demonstrate superior performance of most of models training on the SpeckleNet10 (**S**), which can be credited to simulation of realistic noises. Good performance of the RSCNN on M_d →D and the SimpleView on B→D can verify our claim that density shift should not be omitted beyond systemic noises of the depth sensor.

5 Conclusion

This paper investigates an effective pipeline to mitigate Sim2Real domain gap in a novel perspective of physically realistic synthesis of object point clouds. The synthesis of realistic data can inherently be robust against imbalanced data distribution, as we can observe negative transfer of existing UDA methods in our experiments, which encourages us to propose the quasi-balanced self-training for further suppression. Experiment results can verify the effectiveness of the unified QS^3 scheme as well as both physical simulation of realistic noises and also our quasi-balanced self-training, achieving the state-of-the-art performance on the challenging Sim2Real domain adaptation tasks. Moreover, the proposed Mesh2Point method is not differentiable and parameter settings of the synthesis pipeline are empirically selected, which could cause sub-optimal generation for specific tasks. Inspired by DDS [37], incorporating differentiable physics-based rendering into our QS^3 scheme in an end-to-end learning manner can be a promising direction in future.

Acknowledgements This work is supported in part by the National Natural Science Foundation of China (Grant No.: 61902131), the Program for Guangdong Introducing Innovative and Enterpreneurial Teams (Grant No.: 2017ZT07X183), the Guangdong Provincial Key Laboratory of Human Digital Twin (Grant No.: 2022B1212010004).

References

1. Achituve, I., Maron, H., Chechik, G.: Self-supervised learning for domain adaptation on point clouds. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 123–133 (2021)
2. Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M.: Domain-adversarial neural networks. Stat **1050**, 15 (2014)
3. Arazo, E., Ortego, D., Albert, P., O'Connor, N.E., McGuinness, K.: Pseudo-labeling and confirmation bias in deep semi-supervised learning. In: International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2020)
4. Barron, J.T., Malik, J.: Intrinsic scene properties from a single rgb-d image. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 17–24 (2013)
5. Bartell, F.O., Dereniak, E.L., Wolfe, W.L.: The theory and measurement of bidirectional reflectance distribution function (brdf) and bidirectional transmittance distribution function (btdf). In: Radiation Scattering in Optical Systems (RSOS). vol. 257, pp. 154–160. SPIE (1981)
6. Bohg, J., Romero, J., Herzog, A., Schaal, S.: Robot arm pose estimation through pixel-wise part classification. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 3143–3150. IEEE (2014)
7. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
8. Chen, W., Jia, X., Chang, H.J., Duan, J., Shen, L., Leonardis, A.: Fs-net: Fast shape-based network for category-level 6d object pose estimation with decoupled rotation mechanism. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1581–1590 (2021)
9. Community, B.O.: Blender - a 3D modelling and rendering package. Blender Foundation, Stichting Blender Foundation, Amsterdam (2018), `http://www.blender.org`
10. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5828–5839 (2017)
11. Deng, S., Liang, Z., Sun, L., Jia, K.: Vista: Boosting 3d object detection via dual cross-view spatial attention. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8448–8457 (2022)
12. Denninger, M., Sundermeyer, M., Winkelbauer, D., Olefir, D., Hodan, T., Zidan, Y., Elbadrawy, M., Knauer, M., Katam, H., Lodhi, A.: Blenderproc: Reducing the reality gap with photorealistic rendering. In: Robotics: Science and Systems (RSS) (2020)
13. Fang, J., Zhou, D., Yan, F., Zhao, T., Zhang, F., Ma, Y., Wang, L., Yang, R.: Augmented lidar simulator for autonomous driving. IEEE Robotics and Automation Letters (RA-L) **5**(2), 1931–1938 (2020)
14. Gao, G., Lauri, M., Wang, Y., Hu, X., Zhang, J., Frintrop, S.: 6d object pose regression via supervised learning on point clouds. IEEE International Conference on Robotics and Automation (ICRA) pp. 3643–3649 (2020)
15. Goyal, A., Law, H., Liu, B., Newell, A., Deng, J.: Revisiting point cloud shape classification with a simple and effective baseline. In: International Conference on Machine Learning (ICML). pp. 3809–3820. PMLR (2021)

16. Grans, S., Tingelstad, L.: Blazer: Laser scanning simulation using physically based rendering. arXiv preprint arXiv:2104.05430 (2021)
17. Gschwandtner, M., Kwitt, R., Uhl, ., Pree, W.: Blensor: Blender sensor simulation toolbox. In: International Symposium on Visual Computing (ISVC). pp. 199–208. Springer (2011)
18. Handa, A., Patraucean, V., Badrinarayanan, V., Stent, S., Cipolla, R.: Understanding real world indoor scenes with synthetic data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4077–4085 (2016)
19. Handa, A., Whelan, T., McDonald, J., Davison, A.J.: A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 1524–1531. IEEE (2014)
20. Heindl, C., Brunner, L., Zambal, S., Scharinger, J.: Blendtorch: A real-time, adaptive domain randomization library. In: International Conference on Pattern Recognition (ICPR). pp. 538–551. Springer (2021)
21. Kim, J., Hur, Y., Park, S., Yang, E., Hwang, S.J., Shin, J.: Distribution aligning refinery of pseudo-label for imbalanced semi-supervised learning. Advances in Neural Information Processing Systems (NeurIPS) **33** (2020)
22. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2015)
23. Kumar, M.P., Packer, B., Koller, D.: Self-paced learning for latent variable models. In: Advances in Neural Information Processing Systems (NeurIPS). pp. 1189–1197 (2010)
24. Landau, M.J., Choo, B.Y., Beling, P.A.: Simulating kinect infrared and depth images. IEEE Transactions on Cybernetics (IEEE T CYBERNETICS) **46**(12), 3018–3031 (2015)
25. Lee, D.H.: Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. In: ICML Workshop on Challenges in Representation Learning (WREPL) (2013)
26. Li, B., Zhang, T., Xia, T.: Vehicle detection from 3d lidar using fully convolutional network. In: Robotics: Science and Systems (RSS) (2016)
27. Li, W., Saeedi, S., McCormac, J., Clark, R., Tzoumanikas, D., Ye, Q., Huang, Y., Tang, R., Leutenegger, S.: Interiornet: Mega-scale multi-sensor photo-realistic indoor scenes dataset. In: British Machine Vision Conference (BMVC) (2018)
28. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: Pointcnn: Convolution on x-transformed points. Advances in Neural Information Processing Systems (NeurIPS) **31**, 820–830 (2018)
29. Lian, Q., Lv, F., Duan, L., Gong, B.: Constructing self-motivated pyramid curriculums for cross-domain semantic segmentation: A non-adversarial approach. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6758–6767 (2019)
30. Lin, J., Wei, Z., Li, Z., Xu, S., Jia, K., Li, Y.: Dualposenet: Category-level 6d object pose and size estimation using dual pose network with refined learning of pose consistency. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 3560–3569 (2021)
31. Lin, X., Chen, K., Jia, K.: Object point cloud classification via poly-convolutional architecture search. In: Proceedings of the ACM International Conference on Multimedia (MM). pp. 807–815 (2021)
32. Liu, Y., Fan, B., Xiang, S., Pan, C.: Relation-shape convolutional neural network for point cloud analysis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8895–8904 (2019)

33. Mallick, T., Das, P.P., Majumdar, A.K.: Characterizations of noise in kinect depth images: A review. IEEE Sensors Journal (IEEE Sens. J) **14**(6), 1731–1740 (2014)
34. Manivasagam, S., Wang, S., Wong, K., Zeng, W., Sazanovich, M., Tan, S., Yang, B., Ma, W., Urtasun, R.: Lidarsim: Realistic lidar simulation by leveraging the real world. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11167–11176 (2020)
35. Nguyen, C.V., Izadi, S., Lovell, D.: Modeling kinect sensor noise for improved 3d reconstruction and tracking. In: International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission (3DIMPVT). pp. 524–530. IEEE (2012)
36. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems (NeurIPS) **32**, 8026–8037 (2019)
37. Planche, B., Singh, R.V.: Physics-based differentiable depth sensor simulation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 14387–14397 (2021)
38. Planche, B., Wu, Z., Ma, K., Sun, S., Kluckner, S., Lehmann, O., Chen, T., Hutter, A., Zakharov, S., Kosch, H., et al.: Depthsynth: Real-time realistic synthetic data generation from cad models for 2.5 d recognition. In: International Conference on 3D Vision (3DV). pp. 1–10. IEEE (2017)
39. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 652–660 (2017)
40. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in Neural Information Processing Systems (NeurIPS) **30** (2017)
41. Qin, C., You, H., Wang, L., Kuo, C.C.J., Fu, Y.: Pointdan: A multi-scale 3d domain adaption network for point cloud representation. Advances in Neural Information Processing Systems (NeurIPS) **32** (2019)
42. Reitmann, S., Neumann, L., Jung, B.: Blainder—a blender ai add-on for generation of semantically labeled depth-sensing data. Sensors **21**(6), 2144 (2021)
43. Roy, S., Siarohin, A., Sangineto, E., Bulò, S.R., Sebe, N., Ricci, E.: Unsupervised domain adaptation using feature-whitening and consensus loss. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9463–9472 (2019)
44. Saito, K., Ushiku, Y., Harada, T.: Asymmetric tri-training for unsupervised domain adaptation. In: International Conference on Machine Learning (ICML). vol. 70, pp. 2988–2997 (2017)
45. Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Raffel, C.A., Cubuk, E.D., Kurakin, A., Li, C.L.: Fixmatch: Simplifying semi-supervised learning with consistency and confidence. Advances in Neural Information Processing Systems (NeurIPS) **33**, 596–608 (2020)
46. Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J.J., Mur-Artal, R., Ren, C., Verma, S., et al.: The replica dataset: A digital replica of indoor spaces. arXiv preprint arXiv:1906.05797 (2019)
47. Szeliski, R.: Computer vision: algorithms and applications. Springer Science & Business Media (SSBM) (2010)
48. Tallavajhula, A., Meriçli, Ç., Kelly, A.: Off-road lidar simulation with data-driven terrain primitives. In: IEEE International Conference on Robotics and Automation (ICRA). pp. 7470–7477. IEEE (2018)

49. Tang, H., Chen, K., Jia, K.: Unsupervised domain adaptation via structurally regularized deep clustering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
50. Uy, M.A., Pham, Q., Hua, B., Nguyen, T., Yeung, S.: Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 1588–1597 (2019)
51. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics (TOG) **38**(5), 1–12 (2019)
52. Wei, C., Sohn, K., Mellina, C., Yuille, A., Yang, F.: Crest: A class-rebalancing self-training framework for imbalanced semi-supervised learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10857–10866 (2021)
53. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1912–1920 (2015)
54. Xu, Z., Chen, K., Liu, K., Ding, C., Wang, Y., Jia, K.: Classification of single-view object point clouds. arXiv preprint arXiv:2012.10042 (2020)
55. Yang, B., Luo, W., Urtasun, R.: Pixor: Real-time 3d object detection from point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7652–7660 (2018)
56. Zhang, Y., Song, S., Yumer, E., Savva, M., Lee, J., Jin, H., Funkhouser, T.: Physically-based rendering for indoor scene understanding using convolutional neural networks. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
57. Zhang, Y., Lin, J., He, C., Chen, Y., Jia, K., Zhang, L.: Masked surfel prediction for self-supervised point cloud learning. arXiv preprint arXiv:2207.03111 (2022)
58. Zou, L., Tang, H., Chen, K., Jia, K.: Geometry-aware self-training for unsupervised domain adaptation on object point clouds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 6403–6412 (2021)
59. Zou, Y., Yu, Z., Kumar, B.V., Wang, J.: Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 289–305 (2018)
60. Zou, Y., Yu, Z., Liu, X., Kumar, B.V., Wang, J.: Confidence regularized self-training. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2019)