# DCL: Differential Contrastive Learning for Geometry-Aware Depth Synthesis

Yuefan Shen, Yanchao Yang, Youyi Zheng, C. Karen Liu, and Leonidas J. Guibas

***Abstract*—We describe a method for unpaired realistic depth synthesis that learns diverse variations from the real-world depth scans and ensures geometric consistency between the synthetic and synthesized depth. The synthesized realistic depth can then be used to train task-specific networks facilitating label transfer from the synthetic domain. Unlike existing image synthesis pipelines, where geometries are mostly ignored, we treat geometries carried by the depth scans based on their own existence. We propose differential contrastive learning that explicitly enforces the underlying geometric properties to be invariant regarding the real variations been learned. The resulting depth synthesis method is task-agnostic, and we demonstrate the effectiveness of the proposed synthesis method by extensive evaluations on real-world geometric reasoning tasks. The networks trained with the depth synthesized by our method consistently achieve better performance across a wide range of tasks than state of the art, and can even surpass the networks supervised with full real-world annotations when slightly fine-tuned, showing good transferability.[1]**

***Index Terms*—Deep learning methods, deep learning for visual perception, transfer learning.**

## I. Introduction

UNPAIRED realistic depth synthesis is important in transferring annotations for geometric reasoning tasks from simulation, where labels can be automatically generated, while label generation in the real world is expensive. There exist many works on realistic image synthesis based on generative adversarial networks, yet, there are only a few on realistic depth synthesis. Traditional methods on realistic depth synthesis either model the real depth variations with an empirical noise model or add random noise and dropout to corrupt the synthetic depth. Thus, their capability to capture diverse real variations is limited. On the other hand, learning-based real depth synthesis methods add noise and missing regions to the synthetic depth maps by transformation networks usually trained in an adversarial manner. Despite the ability to reduce the distributional shift between the synthetic and real domains, the underlying geometric properties are not well preserved and are always subject to undesired distortions, affecting the label transfer efficiency for downstream geometric reasoning tasks (see Fig. 1).

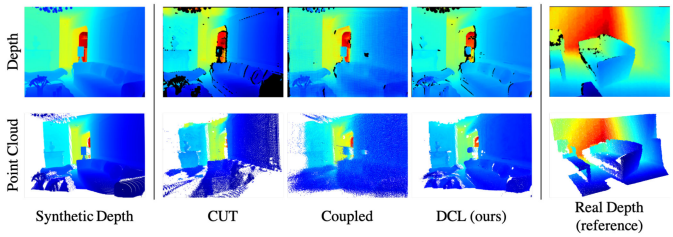


Fig. 1. Synthetic to real depth synthesis. Left: clean depth map; Middle: synthesized depth map by CUT [2], Coupled [4] and DCL (ours); Right: reference real-world depth from ScanNet [5]. Corresponding point clouds are displayed in the second row. Note our method captures missing regions and sensor noise similar to that exhibited in the real-world one. Moreover, our method preserves the underlying geometry as demonstrated in the point clouds, e.g., ours has much fewer out of surface points and distortions.

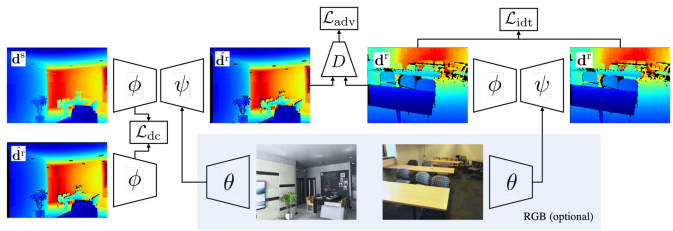


Fig. 2. Overview of our method. The depth synthesis network comprises $\phi$ (encoder) and $\psi$ (decoder). Optionally, $\psi$ can take auxiliary information from the aligned RGB image through an image encoder $\theta$. The three loss terms are described in Section III.

We treat geometric properties as first-class citizens since depth maps are 2.5D representations of the scene, and the geometric properties carried in depth maps deserve their own existence. Moreover, we propose differential contrastive learning to learn the real-world depth variations to minimize the distributional shift between domains and explicitly enforce the underlying geometry to be consistent for efficient transfer between domains. As illustrated in Fig. 3, the proposed differential contrastive learning first computes differences between features extracted at different spatial locations within each feature map of the depth (the synthetic one and its transformed version). The resulting differential features are then arranged into positive and negative samples following the terminology used in [1], [2]. More explicitly, two differential features computed at the same pair of spatial locations are considered positive; otherwise, those computed at different pairs of spatial locations are considered

Manuscript received September 9, 2021; accepted January 3, 2022. Date of publication February 7, 2022; date of current version March 4, 2022. This letter was recommended for publication by Associate Editor M. Liu and Editor C. C. Lerma upon evaluation of the reviewer's comments. This work was supported in part by National Key Research & Development Program of China under Grant 2018YFE0100900, in part by a Hoffman-Yee Research Grant, in part by NSF under Grant IIS-1763268, and in part by a Vannevar Bush Faculty fellowship. *(Yuefan Shen and Yanchao Yang contributed equally to this work.) (Corresponding authors: Yanchao Yang; Youyi Zheng.)*

Yuefan Shen and Youyi Zheng are with the State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310000, China (e-mail: jhonve@zju.edu.cn; youyizheng@zju.edu.cn).

Yanchao Yang, C. Karen Liu, and Leonidas J. Guibas are with Computer Science Department, Stanford University, Stanford, CA 94305 USA (e-mail: yanchao0yang@gmail.com; karenliu@cs.stanford.edu; guibas@cs.stanford.edu).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2022.3148788, provided by the authors.

Digital Object Identifier 10.1109/LRA.2022.3148788

[1]https://github.com/Jhonve/DCL-DepthSynthesis

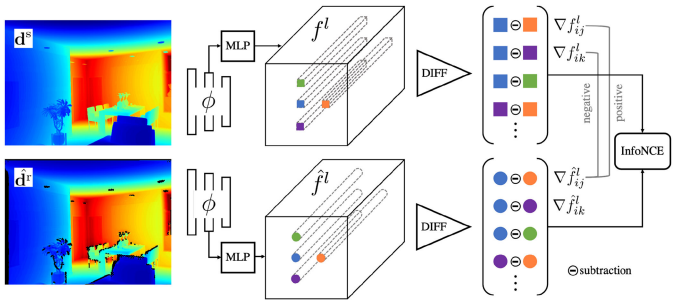


Fig. 3. Differential Contrastive Learning. Given a synthetic depth map $\mathbf{d}^{\mathrm{s}}$ and the transformed noisy depth map $\hat{\mathbf{d}}^{\mathrm{r}}$, feature maps $f^l$ and $\hat{f}^l$ are extracted from the $l$-th layer of the encoder $\phi$. Differential features are then computed and arranged into positive and negative samples depending on their pairwise spatial locations, e.g., differential features with the same row number are positive samples.

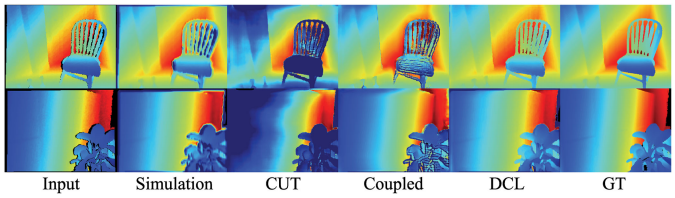


Fig. 4. Visual comparison on depth enhancement. From left to right: input real depth scan, Simulation [38], CUT [2], Coupled [4], DCL (ours) and ground-truth.

negative. Positive and negative samples are then used to compute the InfoNCE loss [3], which serves as the training loss, namely, the differential contrastive loss for learning depth synthesis.

The motivation of our approach derives from the observation that geometric characteristics can always be captured in the differential forms [6], e.g., surface normal. So we explicitly ask the corresponding differential features computed from the depth to be as invariant as possible against the synthesis procedure, which is then ensured by how we select positive and negative samples in the proposed differential contrastive loss. The resulting unpaired depth synthesis framework can be used for learning realistic variations from any depth sensor of any type, while preserving the geometric properties. Moreover, our approach is task agnostic, so the synthesized realistic depth maps, together with synthetic labels, can be used for training any downstream tasks.

We evaluate the quality of the synthesized depth across a broad spectrum of downstream tasks, including depth enhancement, normal estimation, pose estimation, grasping, and semantic segmentation. The task models trained with the depth map synthesized by our method consistently achieve the best performance when tested on real-world data without fine-tuning. Our work makes the following contributions: 1) a framework that explicitly models geometric consistency for depth synthesis; 2) a mechanism that prevents geometric distortion by contrasting the feature differences instead of the features themselves; 3) an extensive study of state-of-the-art synthesis methods on a wide range of downstream tasks while achieving top performance.

## II. Related Work

*Image generation and translation:* Image generation maps a random noise sampled from a prior distribution to images satisfying a predefined distribution. Many works have been proposed to generate diverse and realistic images based on generative adversarial networks (GAN) [7]. Please refer to [8] for a detailed overview. Image translation aims to transform images from one domain to another, in either paired [9] or unpaired [10] settings. Image translation can help reduce domain gaps [11] when it is enforced to preserve task-relevant information [12]. Cycle-consistency is widely used as a regularizer for style transfer [10]. Recently, contrastive unpaired translation (CUT) [2] proposes contrastive losses on the patch-level features to enforce the similarity of the input and output image features. Besides the vast development on image-to-image translation [13]–[15], little effort has been devoted to depth-to-depth synthesis, where the translation has to align not only the domain noise but also preserve the underlying geometry that is crucial for downstream geometric inferences based on depth.

*Contrastive learning:* Based on the InfoNCE loss [3], contrastive learning has been shown effective for self-supervised representation learning [1]. The critical ingredient of contrastive learning is the selection of variations to which we would like the learned representations to be invariant [16]. Our primary task is not to learn representations that share the invariance of the downstream tasks. Instead, we learn realistic variations of depth and utilize contrastive learning to take care of geometric properties that should be invariant to the synthesis process.

*Point cloud generation:* 3D point cloud generation is closely related to 2.5D depth map synthesis. A variational auto-encoder with multi-resolution tree networks is used in [17] to generate point clouds, and [18] studies various GANs and proposes a Gaussian mixture model in the latent space of an autoencoder. Instead of transforming random noise, [19] maps a set of 2D grid points to the target point cloud through deep grid deformation, and [20] proposes hierarchical modeling of shapes and points using continuous normalizing flows. At the scene level, [21] generates synthetic point clouds via a virtual lidar in simulation.

*Depth synthesis:* To inject realistic noise into synthetic depth maps, [22] proposes an empirical noise model of the depth sensor's transmitter/receiver system, which captures sensor-specific noise and may not generalize to different ones. Similarly, [23] explicitly models sensor noise, material properties, and surface geometry for depth synthesis, but is limited to single CAD models. One can also add random Gaussian noise and dropout to synthesize additive sensor noise and missing regions [24], [25]. Furthermore, [26] applies adversarial training to synthesize realistic hand pose images, and [27] synthesizes holes with a network trained to predict missing regions from RGB images. Similarly, [4] learns the hole prediction from real RGBD images, but relies on image translation to bring synthetic images to the real domain such that the learned hole prediction model can be applied on synthetic RGB images. One can also apply domain adaptation for depth-based predictive tasks [28], [29]. However, these methods do not generate realistic depth maps. Our method focuses on realistic depth synthesis, and the synthesized depth maps can be used for any tasks that take depth as input. Even though our method can be used in conjunction with domain adaptation methods when specific tasks are known, we treat realistic depth synthesis as our primary goal and evaluate the quality of the synthesized depth maps using specific geometric reasoning tasks.

## III. Method

Let $\mathbf{d} \in \mathbb{R}^{\mathrm{H}\times\mathrm{W}}$ be a depth map, and optionally $\mathbf{I} \in \mathbb{R}^{\mathrm{H}\times\mathrm{W}\times 3}$ be the corresponding color image. Suppose we have a synthetic

(clean) dataset $\mathcal{D}^{\mathrm{s}} = \{(\mathbf{d}^{\mathrm{s}}, \mathbf{I}^{\mathrm{s}})\}$ and a real-world (noisy) dataset $\mathcal{D}^{\mathrm{r}} = \{(\mathbf{d}^{\mathrm{r}}, \mathbf{I}^{\mathrm{r}})\}$. Both of them contain pairs of aligned depth maps and color images. However, there is no pairing between $\mathcal{D}^{\mathrm{s}}$ and $\mathcal{D}^{\mathrm{r}}$, a typical setting of unpaired image synthesis or translation. Our goal is to learn a mapping between the synthetic depth map $\mathbf{d}^{\mathrm{s}}$ and the real depth map $\mathbf{d}^{\mathrm{r}}$, using only unpaired datasets $\mathcal{D}^{\mathrm{s}}$ and $\mathcal{D}^{\mathrm{r}}$. The overall architecture of our method is illustrated in Fig. 2. Note that the aligned color images are auxiliary, without which our method still runs, and we elaborate in the following.

### A. Depth-to-Depth Synthesis

We first illustrate our method using only depth maps for simplicity. We then detail how color images can be incorporated as auxiliary signals to facilitate the synthesis procedure. Let $\phi$ be an encoder, and $\psi$ be a decoder, together they constitute the transformation network:

$$\hat{\mathbf{d}}^{\mathrm{r}} = \psi(\phi(\mathbf{d}^{\mathrm{s}})) \tag{1}$$

with $\hat{\mathbf{d}}^{\mathrm{r}}$ be the synthesized (noisy) depth map conditioned on the clean depth map $\mathbf{d}^{\mathrm{s}}$. To enforce statistical similarity between the synthesized depth map $\hat{\mathbf{d}}^{\mathrm{r}}$ and the real depth map $\mathbf{d}^{\mathrm{r}}$, we can apply a discriminator network $D$ to minimize the domain discrepancy by adversarial training:

$$\mathcal{L}_{\mathrm{adv}} = \mathbb{E}_{\mathbf{d}^{\mathrm{r}} \sim \mathcal{D}^{\mathrm{r}}} \log D(\mathbf{d}^{\mathrm{r}}) + \mathbb{E}_{\mathbf{d}^{\mathrm{s}} \sim \mathcal{D}^{\mathrm{s}}} \log(1 - D(\hat{\mathbf{d}}^{\mathrm{r}})) \tag{2}$$

Since Eq. (2) only helps to reduce the distributional shift, but does not guarantee the consistency of the generated content [26], [30], CycleGAN [10] resorts to cycle consistency to constrain the transformation. On the other hand, contrastive unpaired translation (CUT) [2] eliminates the cycle consistency by applying contrastive loss on features from multiple layers of the encoder to preserve the image content. Even though it works for image-to-image translation, we observe heavy distortions on the underlying geometric structures of the synthesized depth maps, which hinder the transfer from synthetic domains to real domains.

### B. Differential Contrastive Learning

We aim for a transformation $\psi \circ \phi$ that can capture the complex noise phenomenon in real depth, and, at the same time, preserve the underlying geometry of the clean depth maps for better transfer on geometric reasoning tasks.

Since we ask for geometric invariants of the synthetic depth maps, we choose to work with the InfoNCE loss [3] due to its effectiveness in capturing invariants for self-supervised representation learning [1], [31]. However, the type of invariants that will be learned with the InfoNCE loss depends mainly on the mechanism to choose positive and negative samples. For example, in [1], an image and its color distorted version are considered as a pair of positive samples, whereas this same image and another different image are considered as negative samples. With this sampling strategy, the features learned will be invariant to color distortions but still be discriminative for image identities.

Inspired by the fact that geometric properties can always be captured by their differential forms [6], we propose to impose an explicit constraint on the underlying geometry of the scene through differential contrastive learning shown in Fig. 3. Let $f^l = \phi^l(\mathbf{d}^{\mathrm{s}})$, $\hat{f}^l = \phi^l(\hat{\mathbf{d}}^{\mathrm{r}})$ be the feature maps extracted from the $l$-th layer of the encoder $\phi$ applied on the synthetic depth map $\mathbf{d}^{\mathrm{s}}$ and the synthesized depth map $\hat{\mathbf{d}}^{\mathrm{r}}$. Also, let $f_i^l$ be the feature vector from $f^l$ at the spatial location $i$. We apply the following sampling mechanism to collect positive and negative pairs:

$$\textbf{positive} : (\nabla \hat{f}_{ij}^l = \hat{f}_i^l - \hat{f}_j^l, \nabla f_{ij}^l = f_i^l - f_j^l) \tag{3}$$

$$\textbf{negative} : (\nabla \hat{f}_{ij}^l = \hat{f}_i^l - \hat{f}_j^l, \nabla f_{ik}^l = f_i^l - f_k^l) \tag{4}$$

where $j, k$ are different spatial locations sampled around $i$ following a Gaussian. Note that each sample consists of two differential vectors (synthetic and synthesized) computed either at the same pair-wise locations (positive) or different pair-wise locations (negative) (see Fig. 3). Given the InfoNCE loss:

$$\mathcal{L}_{\mathrm{nce}}(\nabla \hat{f}_{ij}^l, \nabla f_{ij}^l, \{\nabla f_{ik}^l\}_{k \neq j})$$

$$= -\log \frac{\exp(\nabla \hat{f}_{ij}^l \cdot \nabla f_{ij}^l / \tau)}{\exp(\nabla \hat{f}_{ij}^l \cdot \nabla f_{ij}^l / \tau) + \sum_k \exp(\nabla \hat{f}_{ij}^l \cdot \nabla f_{ik}^l / \tau)}$$

Our **differential contrastive loss** is defined as:

$$\mathcal{L}_{\mathrm{dc}} = \mathbb{E}_{\mathbf{d}^{\mathrm{s}} \sim \mathcal{D}^{\mathrm{s}}} \sum_l \sum_{i,j} \mathcal{L}_{\mathrm{nce}}(\nabla \hat{f}_{ij}^l, \nabla f_{ij}^l, \{\nabla f_{ik}^l\}_{k \neq j}) \tag{5}$$

here $(i, j, k's)$ can be randomly sampled to avoid enumerating the entire grid. The key insight is that, we want the differential features to be similar (invariant) before and after the transformation, i.e., the depth values may be altered due to the noise or missing regions learned from real depth maps; however, the underlying geometric structures captured by the differentials should be similar.[2] In other words, the proposed differential contrastive loss explicitly enforces the consistency between geometric structures of the synthetic and synthesized depth maps, while leaving enough flexibility for the transformation network to learn real variations. Note, differential contrastive losses over feature maps from multiple layers of the encoder $\phi$ are also computed, making it possible to capture both local and global geometric properties.

Given that a global shift in the depth values might be differentiated away and thus can not be detected with the proposed differential contrastive loss, we apply an identity loss on the real depth map $\mathbf{d}^{\mathrm{r}}$ to prevent potential global shifts in the range of the synthesized depth:

$$\mathcal{L}_{\mathrm{idt}} = \mathbb{E}_{\mathbf{d}^{\mathrm{r}} \sim \mathcal{D}^{\mathrm{r}}} \|\psi(\phi(\mathbf{d}^{\mathrm{r}})) - \mathbf{d}^{\mathrm{r}}\|_1 \tag{6}$$

which is the $L1$ loss between a real depth map and its transformed version. The **final training loss** of the proposed differential contrastive learning for synthesizing realistic depth maps from synthetic ones is:

$$\mathcal{L} = \mathcal{L}_{\mathrm{adv}} + \alpha \mathcal{L}_{\mathrm{dc}} + \beta \mathcal{L}_{\mathrm{idt}} \tag{7}$$

where $\alpha, \beta$ are the scalar weights, which are set to 1.5 and 1.0 for all experiments.

## IV. Experiments

We evaluate the proposed depth synthesis method on multiple downstream tasks, including depth enhancement, normal estimation, pose estimation, grasping, and semantic segmentation. Our goal is to have a comprehensive understanding of

[2]For differentials involved in a negative pair but sampled nearby, their features can remain similar while the differentials are pushed away.

the capability of our method to learn the noise exhibited in the real-world depth scans by checking the performance of the task-specific networks trained using the synthesized depth on real-world geometric reasoning tasks.

Specifically, the clean synthetic depth maps are first transformed into noisy realistic ones in a task-agnostic manner. We then perform evaluations on each downstream task in three settings: 1) train a task-specific network using only the labels from the synthetic domain, and test the network directly on a test set from the real domain; 2) apply task-specific domain adaptation methods, e.g., [11], [32], [33] using depth from the synthetic and real domains, then test on the real validation set as in 1). 3) fine-tune the previously trained task-specific networks using a small portion of the annotations from the real domain, and then test on the same real test set as in 1). In the first setting, we like to check how the synthesized depth maps mimic the real ones. In the second, we check whether our synthesis helps to reduce the domain gaps when compared to task-specific domain adaptation methods. In the third, we check the usefulness of the weights from the first setting in terms of reducing the number of labels compared to the one supervised with full real annotations.

### A. Training Depth Synthesis

*Datasets:* For depth enhancement, normal estimation, and semantic segmentation, we use InteriorNet [34] as the source of synthetic data, and ScanNet [5] as the source of realistic data. InteriorNet provides depth maps rendered from 1.7 M interior layouts for different scenes created by professional designers. We randomly sample 30 K depth maps from InteriorNet to form the synthetic dataset. We further split them into a subset of 24 K depth maps for training the depth synthesis network and 6 K for training task-specific networks. Each sample in the synthetic dataset consists of a clean depth map and the corresponding annotations for the downstream tasks. Similarly, we randomly sample 24 K real scans from ScanNet for depth synthesis, which contains real-world depth maps from 1.5 K indoor scenes. The raw scans are manually annotated. For task-specific networks, we follow the filtering scheme in [35] to guarantee the data quality and collect 7 K real depth maps, where 4 K are used for supervised training and 3 K are for testing all baselines. Since our goal for generating the real-world (ScanNet) training and test sets is to check both the upperbound on real scans and how effective the transfer is, we sample the 7 K depth maps uniformly from a common set of scenes such that the task-specific training and test splits do not have domain gaps caused by different scene layouts.

For pose estimation and grasping, we use the large-scale GraspNet [36] as our primary dataset, whose raw scans are used as the real-world depth, and the synthetic depth maps are rendered with the corresponding camera parameters and CAD models. GraspNet [36] contains 48.6 K depth maps along with annotated object labels, object poses, and ground-truth grasps (center, gripper orientation, and width), which are used for training pose estimation and grasp proposal networks. The LineMOD dataset [37] is also used for pose estimation, which contains 20 K depth maps and ground-truth object poses. The detailed split and pre-processing of the data are described in the evaluation sections.

*Baselines:* A natural depth synthesis baseline is the identity mapping, i.e., the clean depth scans (Baseline). We also compare with the commonly used empirical noise model [38] (Simulation), and two state-of-the-art image translation methods [10] (CycleGAN) and [2] (CUT). Moreover, we compare to [4] (Coupled), which is the current state-of-the-art on synthetic to real depth synthesis that separately models missing regions and sensor noise based on both depth and color images.

TABLE I
DEPTH ENHANCEMENT

| Method | RMSE ↓ | MAE ↓ | PSNR ↑ | SSIM ↑ |
|---|---|---|---|---|
| Baseline | 0.1871 | 0.0716 | 33.670 | 0.7822 |
| Simulation [38] | 0.1618 | 0.0817 | 36.561 | 0.7955 |
| CycleGAN [10] | 0.1448 | 0.0911 | 38.771 | 0.8469 |
| CUT [2] | 0.1757 | 0.1186 | 34.870 | 0.8066 |
| Coupled [4] | 0.1273 | 0.0568 | 41.522 | 0.8524 |
| DCL | 0.1233 | **0.0564** | 42.340 | 0.8692 |
| DCL w/ rgb | **0.1198** | 0.0650 | **42.980** | **0.8754** |
| DCL* | 0.0996 | 0.0407 | 47.424 | 0.9096 |
| Supervised | 0.1062 | 0.0437 | 45.426 | 0.9079 |

Scores are computed with the networks trained using only the synthesized (noisy) depth maps and the corresponding clean depth maps. Top-performing ones are marked as bold and * means fine-tuning with 10% of the real-world annotations.

*Training details:* Our network architecture follows that of CUT [2], which employs an image transformation network consisting of ResNet blocks [39]. We extract multi-scale features from five evenly distributed layers of the encoder to compute the differential contrastive loss. Like CUT, we apply a two-layer MLP with 128 output units and $L2$ normalization on the extracted features before passing them to the loss function. For the competing methods, aligned color images are always available to the transformation network, except the empirical noise model proposed in [38], which only relies on geometry to synthesize noise. We use an Adam optimizer for the training of depth synthesis with differential contrastive loss ($\beta_1 = 0.5$ and $\beta_2 = 0.999$) with an initial learning rate of 0.0002 (same to all downstream tasks). We set the batch size to 16 and train up to 50 epochs. Our method can optionally incorporate RGB images as auxiliary signals during synthesis (DCL w/ rgb). However, due to the lack of background textures in GraspNet and LineMOD datasets, we cannot render corresponding RGB images as auxiliaries, thus we report the depth-only results on them. Now we detail the evaluations on each downstream task and report the qualitative and quantitative results.

### B. Depth Enhancement

After training the depth synthesis network, we convert the 6 K synthetic depth maps to realistic (noisy) ones. We train a depth enhancement network [35] for each of the synthesized datasets from different synthesis methods. We also apply the depth enhancement training loss proposed in [35] for all models, and the training runs for 50 epochs.

When the training converges, we test each enhancement network on a preserved real-world test set consists of 300 raw scans and the corresponding clean scans generated from reconstructed meshes. We report the scores under multiple evaluation metrics: root mean square error (RMSE), mean absolute error (MAE), peak signal-to-noise ratio (PSNR), and structural similarity index measure (SSIM) following [4]. The first two measure the accuracy of the enhanced depth maps, and the latter two measure the structural similarity of the enhanced depth maps compared to the ground truth.

TABLE II
NORMAL ESTIMATION

| Method | Median ↓ | Mean ↓ | 16 ↑ | 22.5 ↑ | 30 ↑ |
|---|---|---|---|---|---|
| Baseline | 21.127 | 24.907 | 0.392 | 0.541 | 0.679 |
| Simulation [38] | 23.725 | 27.473 | 0.349 | 0.485 | 0.623 |
| CycleGAN [10] | 22.321 | 25.454 | 0.367 | 0.501 | 0.653 |
| CUT [2] | 22.492 | 25.082 | 0.361 | 0.507 | 0.665 |
| Coupled [4] | 20.049 | 24.629 | 0.420 | 0.557 | 0.677 |
| DCL | 19.343 | **22.724** | 0.407 | 0.579 | **0.743** |
| DCL w/ rgb | **18.920** | 23.805 | **0.444** | **0.579** | 0.693 |
| Cycada [11] | 20.694 | 24.763 | 0.402 | 0.550 | 0.681 |
| Cycada [11]+DCL | **17.120** | **21.567** | **0.460** | **0.649** | **0.787** |
| DCL* | 7.907 | 13.881 | 0.739 | 0.815 | 0.865 |
| Supervised | 8.950 | 14.420 | 0.725 | 0.814 | 0.868 |

Normal estimation on real depth scans. Networks are trained with synthesized depth and corresponding normal maps. Top-performing ones are marked as bold and * means fine-tuning with 10% of the annotated real-world data.

As shown in Table I, our method consistently achieves smaller error and higher structural similarity compared to other methods. As expected, the empirical noise model (Simulation) [38] generally performs worse than the learning-based methods. Note that CUT [2] performs even worse than Simulation.[3] We conjecture that CUT may capture biased real noise. Hence, its performance is not even as good as the empirical noise model that randomly adds noise without looking at the real scans. We include the score from a purely supervised model that is trained on a separate training set of 3000 real depth scans to provide a reference on the desired real domain performance.

We also fine-tune the networks trained with the synthesized depth using 10% of the annotations from the real-world training set that is used to train the supervised baseline in Table I. As observed, the model trained with the synthesized depth from DCL surpasses the supervised baseline on all metrics, which confirms that the network parameters learned using our synthesized depth maps transfer efficiently to the real world. Please see Fig. 4 for visual comparisons.

### C. Surface Normal Estimation

In this part, we evaluate the quality of the synthesized depth maps by checking the performance of the task-specific networks on normal estimation. Performance on normal estimation can measure how well the synthesized depth maps preserve the underlying geometry since the surface normal is the cross-product of partial derivatives. We train the same architecture used for depth enhancement using the L1 loss between predicted normal maps and the ground-truth for 50 epochs. We report the angular errors and accuracy.

As shown in Table II, the empirical noise model based method [38] (Simulation) is now consistently worse than the baseline trained with clean depth maps (Baseline), which signals that randomly adding noise could destroy the underlying geometry. The same phenomenon is also observed for two other learning-based methods (CycleGAN and CUT). The normal estimation network trained using the depth synthesized by DCL outperforms the second-best by $3.52\%, 7.73\%$ in terms of the median and mean angular errors, respectively. Involving RGB images in depth synthesis (DCL w/ rgb) has slight improvements for both depth enhancement and normal estimation. Compared to the domain adaptation method [11], our method also achieves higher performance. When replacing the source dataset with the synthesized dataset from DCL in Cycada [11], we can observe a significant improvement, demonstrating the effectiveness of our method. We further apply fine-tuning on the pre-trained network using 10% of the real-world annotations from the training set of the supervised baseline. Similarly, the normal estimation network trained using synthesized depth from DCL achieves comparable performance with the supervised baseline. This confirms the quality of the synthesized depth measured by the normal estimation performance directly related to the surface geometry. Please refer to Fig. 5 for visual results.

[3]We have tuned CUT and other competing methods using grid search for their optimal hyper-parameters.

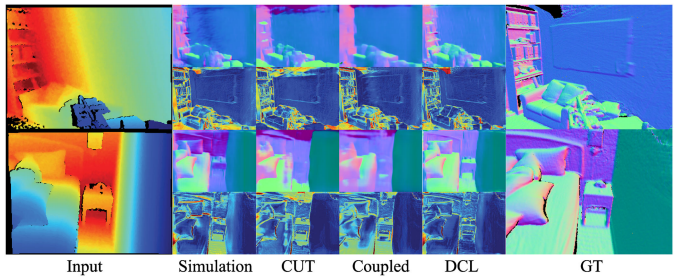


Fig. 5. Visual comparison on normal estimation with error maps inserted. Left to right: input real scan, the result of Simulation [38], CUT [2], Coupled [4], DCL (ours) and the ground-truth.

### D. Pose Estimation

We train task-specific networks that predict poses of objects directly from the depth scans to check how the synthesized depth maps preserve geometric information useful for inferring poses. To comply with the evaluation protocol proposed in [37], [40], we crop the scans from GraspNet [36] using object masks to obtain the input to the pose estimation network. Similarly, the commonly used object-centric version of LineMOD [37] is used for further evaluations. For both datasets, 10 K depth scans of 10 randomly chosen objects are preserved for training the pose estimation networks, and the remaining 6 K depth scans are used for training the depth synthesis networks.

We use ResNet [39] as the backbone for all pose estimation networks. Following the literature [40], the training loss contains a classification error, measured by a cross-entropy loss, and a rotation regression error, measured by an MSE loss. In addition to the rotation error, which directly reflects how the fine geometric properties are preserved for pose estimation, we also report the classification accuracy for reference.

The results are shown in Table III. As observed, the pose estimation network trained with the synthesized depth from DCL achieves the smallest rotation error on both datasets, e.g., it outperforms the second-best by 6.63% and 16.39% on GraspNet and LineMOD, respectively. In terms of classification accuracy, our method is comparable with Coupled [4] on the LineMOD dataset despite that Coupled uses aligned RGB images to facilitate the synthesis procedure. Moreover, DCL outperforms Coupled by 7.02% in classification accuracy on the GraspNet dataset. Overall, DCL is more efficient in learning the real-world variations while maintaining the fine geometries for inferring poses. Different from dense prediction tasks, here we choose to compare with latent space adversarial domain adaptation methods [32], [33].

TABLE III
POSE ESTIMATION

| | GraspNet [36] | | LineMOD [37] | |
|---|---|---|---|---|
| Method | Acc ↑ | Error ↓ | Acc ↑ | Error ↓ |
| Baseline | 57.153 | 55.861 | 44.722 | 62.169 |
| Simulation [38] | 83.333 | 42.305 | 50.417 | 54.910 |
| CycleGAN [10] | 78.056 | 43.616 | 63.819 | 51.845 |
| CUT [2] | 81.458 | 39.722 | 68.125 | 55.260 |
| Coupled [4] | 82.153 | 38.869 | **72.778** | 50.965 |
| DCL | **87.917** | **36.290** | 71.042 | **42.607** |
| DANN [32] | 92.500 | 32.058 | 86.548 | 43.376 |
| UAN [33] | 94.090 | 28.353 | 87.431 | 44.077 |
| UAN [33]+DCL | **94.444** | **23.469** | **93.750** | **38.488** |
| DCL* | 98.264 | 16.906 | 97.986 | 25.087 |
| Supervised | 99.653 | 16.765 | 99.792 | 23.678 |

Results of object pose estimation on GraspNet [36] and LineMOD [37]. The prediction error is calculated using the rotation component. The classification accuracy is also reported, and * means fine-tuning with 10% of the annotated real-world data.

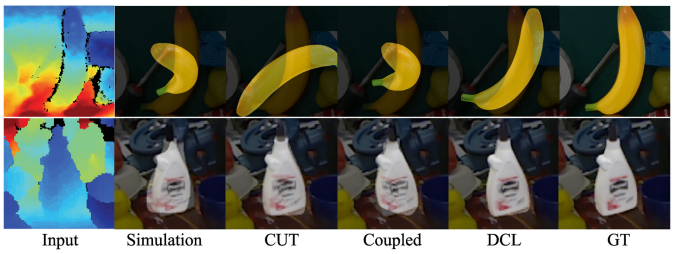


Fig. 6. Visual comparison on object pose estimation. Left to right: input real-world scan, object models overlaid on the corresponding color images using the estimated pose. The first row is from GraspNet [36] and the second is from LineMOD [37].

Again, domain adaptation using synthesized depth from DCL achieves the highest score. Further, after fine-tuning with 10% of the annotations from the real-world domain, the pose estimation network trained using our synthesized depth achieves similar performance on both datasets compared to the fully supervised baseline. Please see Fig. 6 for visual comparisons.

### E. Grasping

Grasping also relies on an accurate understanding of the geometries from the depth scans to enable object manipulations. We first apply the depth synthesis methods on the training subset from GraspNet [36], which contains 25.6 K depth maps. Then we train GG-CNN [41] on the synthesized depth to predict the ground-truth grasp proposals. To evaluate, we choose the three most confident grasps from the predicted proposals following [41]. A grasp is considered successful if its intersection-over-union with the ground truth is larger than 0.5. Furthermore, we report the success rate on three different testing subsets of GraspNet, e.g., the test sets for objects seen during training, objects similar to the training objects, and novel objects.

As shown in Table IV, the network trained using the depth scans synthesized by DCL achieves the highest success rate among the competing methods. It outperforms the second-best by 1.91%, 0.74%, and 4.86% on the test split of seen, similar, and novel objects, respectively. This shows that DCL learns the realistic variations exhibited in the real-world depth scans

TABLE IV
GRASPING

| Method | $SR^{Seen}$ ↑ | $SR^{Sim}$ ↑ | $SR^{Novel}$ ↑ |
|---|---|---|---|
| Baseline | 65.0260 | 61.1328 | 56.7448 |
| Simulation [38] | 80.2866 | 75.0260 | 73.3854 |
| CycleGAN [10] | 86.6146 | 81.3802 | 77.5391 |
| CUT [2] | 92.5000 | 88.5286 | 78.2813 |
| Coupled [4] | 83.5938 | 74.4531 | 73.1250 |
| DCL | **94.2708** | **89.1797** | **82.0833** |
| Cycada [11] | 92.3047 | 89.3100 | 82.5651 |
| Cycada [11]+DCL | **95.1823** | **90.7812** | **83.1771** |
| DCL* | 97.7865 | 95.8594 | 87.0313 |
| Supervised | 96.1849 | 94.7135 | 84.1016 |

Grasping performance on GraspNet [36]. $SR^{Seen}$, $SR^{Sim}$ and $SR^{Novel}$ stand for success rates on seen, similar and novel objects, respectively. And * means fine-tuning with 10% of the annotated real-world data.

for seen objects and generalizes better by preventing potential overfitting to distorted geometries in the synthesized depth. Similarly, when the grasp proposal network trained using the depth synthesized by DCL is fine-tuned with 10% of the real-world annotations, it outperforms the supervised baseline trained with full annotations. Moreover, compared with directly applying Cycda [11], the synthesized depth scans from DCL help achieve better performance. The results on grasping confirm again that the geometry which enables physical manipulations of the objects is well preserved by our method.

### F. Semantic Segmentation

We train DeepLabv3+ [42] for 50 epochs with the cross-entropy loss for semantic segmentation. Due to the class mismatch between InteriorNet and ScanNet, we choose a common set of fifteen classes for training since our goal is to validate the synthesis networks' performance for learning realistic sensor noise but not to adapt for class imbalance. We also report the performance of a purely supervised model trained with 6000 manually annotated real depth scans from ScanNet.

We use the intersection-over-union (IoU) score to measure the quality of the predicted semantic masks. We also report the mean intersection-over-union (mIoU) scores across different subsets of classes in Table V. Coupled [4] now lags compared to other learning-based methods. Our method still achieves the best performance on most of the classes and the mean IoUs. However, due to the imbalanced class distributions, we can still observe a gap compared to the supervised baseline.

To check how the pre-trained weights help reduce the number of real-world annotations needed for semantic segmentation, like previous tasks, we also fine-tune the pre-trained network with 25% of the annotations of the supervised baseline. We report the scores in Table V. As observed, the network pre-trained with the depth synthesized by our method now surpasses the supervised baseline. Moreover, we find that the fine-tuned model has higher scores in classes that appear more frequently in the synthetic domain, e.g., window and curtain. Please see Fig. 7 for visual comparisons. Again, the depth scans synthesized by DCL help achieve better domain adaptation performance than Cycada [11].

TABLE V
SEMANTIC SEGMENTATION

| Method | wall | floor | cabinet | bed | chair | sofa | table | door | window | desk | curtain | ceiling | fridge | tv | others | mIoU-15 | mIoU-12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Baseline | 39.52 | 41.90 | 2.96 | 0.97 | 0.04 | 5.45 | 1.97 | 1.25 | 2.50 | 0.0 | 2.90 | 4.75 | 0.0 | 0.0 | 9.78 | 7.60 | 9.50 |
| Simulation [38] | 38.50 | 47.67 | 10.47 | 22.40 | 8.88 | 10.30 | 9.49 | 2.31 | 4.77 | 3.05 | 3.97 | 28.72 | 0.03 | 0.0 | 13.63 | 13.61 | 16.27 |
| CycleGAN [10] | 39.11 | 47.47 | 10.39 | 21.48 | 6.49 | 9.14 | 9.99 | 1.56 | 4.22 | 2.47 | 4.37 | 30.12 | 0.04 | 0.0 | 14.31 | 13.41 | 16.22 |
| CUT [2] | 45.23 | 60.49 | 12.84 | 13.03 | 4.34 | 9.37 | 5.72 | 4.55 | 6.76 | 1.66 | 8.43 | 22.46 | 0.83 | 0.84 | 11.98 | 13.90 | 16.88 |
| Coupled [4] | 38.85 | 25.47 | 8.52 | 13.57 | 3.33 | 2.28 | 1.02 | 4.44 | 2.99 | 0.84 | 8.56 | 21.11 | 0.14 | 0.09 | 11.15 | 9.49 | 11.57 |
| DCL | 45.25 | 61.35 | 11.60 | 17.67 | 8.20 | 11.10 | 7.14 | 7.72 | 8.13 | 5.15 | 6.92 | 19.49 | 1.81 | 3.43 | 11.83 | **15.12** | **17.78** |
| DCL w/ rgb | 46.11 | 67.53 | 12.48 | 10.04 | 8.16 | 11.65 | 4.52 | 3.16 | 4.68 | 2.81 | 8.72 | 26.67 | 1.06 | 0.13 | 15.30 | 14.74 | 17.64 |
| Cycada [11] | 54.81 | 72.97 | 13.18 | 26.05 | 9.76 | 12.45 | 13.45 | 8.80 | 10.33 | 7.85 | 7.29 | 39.63 | 5.77 | 4.89 | 16.36 | 20.02 | 23.60 |
| Cycada [11]+DCL | 58.62 | 74.84 | 12.83 | 32.11 | 10.04 | 16.15 | 12.83 | 11.81 | 10.35 | 6.94 | 10.85 | 32.93 | 5.54 | 4.56 | 15.83 | **21.08** | **24.67** |
| DCL* | 75.62 | 87.56 | 49.50 | 69.84 | 58.01 | 59.81 | 60.91 | 43.22 | 51.78 | 42.09 | 47.90 | 83.53 | 38.51 | 44.75 | 42.15 | 57.01 | 59.49 |
| Supervised | 72.43 | 88.45 | 52.57 | 70.97 | 54.15 | 55.60 | 52.83 | 36.30 | 34.31 | 43.25 | 37.66 | 81.64 | 22.65 | 56.18 | 39.25 | 53.22 | 55.44 |

Results on semantic segmentation. We report the intersection-over-union (IoU) scores for each class, and the mean intersection-over-union (mIoU) scores over subsets of the semantic classes (mIoU-15, mIoU-12). The network trained with the depth synthesized by DCL performs well on most classes and consistently outperforms the others in terms of mIoUs. * means fine-tuning with 25% of the real-world annotations.

TABLE VI
ABLATION STUDIES

| row # | $\alpha$ | $\beta$ | RMSE ↓ | MAE ↓ | PSNR ↑ | SSIM ↑ |
|---|---|---|---|---|---|---|
| (a) | 1.5 | 0.0 | 0.1813 | 0.0701 | 34.31 | 0.7888 |
| (b) | 0.0 | 1.0 | 0.2624 | 0.2104 | 26.97 | 0.7181 |
| (c) | 1.0 | 1.0 | 0.1213 | 0.0653 | 42.423 | **0.8767** |
| (d) | 2.0 | 1.0 | 0.1399 | 0.0664 | 39.65 | 0.8441 |
| (e) | 1.5 | 1.0 | **0.1198** | **0.0650** | **42.98** | 0.8754 |

Ablation study on the effectiveness of each term in Eq. (7). Evaluations for different sets of the coefficient are performed on the depth enhancement task with the same metrics as in Table I

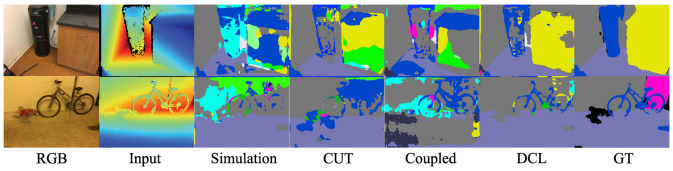


Fig. 7. Visual comparison on semantic segmentation from real depth. Left to right: the reference RGB image, input real scan, the result of Simulation [38], CUT [2], Coupled [4], DCL (ours), and the ground-truth.

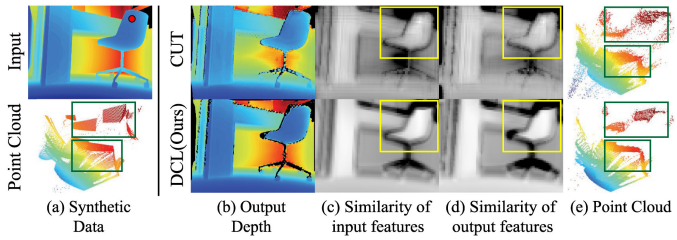


Fig. 8. Visualization of the similarities between learned features from DCL and CUT [2]. Given a query point shown as a red dot in the input depth ((a) top) and the output depth maps ((b), top: CUT, bottom: DCL), we visualize the feature similarity to the query point feature both for the features extracted from the input depth (c) and the output depth (d). We also show point cloud visualizations from the top view to examine the geometric distortions (e).

### G. Analysis and Ablation Studies

To further understand how our proposed DCL affects the feature learning in the synthesis procedure, we study the feature space from the middle layer of $\phi$. As shown in Fig. 8, given an input and output depth map, we calculate the similarities between feature maps and a query point feature in the input feature map. One can see that features learned with CUT [2] tend to be dissimilar in different positions even they have the same underlying geometries. However, with the proposed DCL, features have higher similarities where the underlying geometries are similar. This confirms that DCL prevents undesired distortions by not directly pushing away the features in the differential contrastive learning process.

We perform ablation studies on each loss term in Eq. (7) with the depth enhancement task. We train the depth synthesis network using five different combinations of the coefficients $\alpha$ (weight on the differential contrastive loss) and $\beta$ (weight on the identity loss). After training the depth synthesis network, we train a depth enhancement network using the synthesized depth maps from each depth synthesis network. Specific coefficients and their corresponding scores are reported in Table VI. These results confirm that the two terms in our proposed loss Eq. (7) for depth synthesis are necessary and effective.

## V. DISCUSSION

We have proposed an effective synthetic-to-real depth synthesis method based on differential contrastive learning, which enforces the differential features to be invariant through the synthesis procedure. Extensive evaluations on downstream geometric reasoning tasks indicate that our method learns diverse variations from the real-world scans and preserves the geometric information crucial for real-world applications. Our method is end-to-end trainable and does not need to deal with hole generation and value degeneration separately. However, since our method is GAN-based, it currently does not guarantee to synthesize material-specific noise, which may need extra investigations to show the impact. Currently, we perform experiments with depth scans from existing datasets, yet a specifically designed dataset with disentangled variations is needed for more detailed analysis. Moreover, since our synthesis method is task-agnostic,

we like to see it be used with task-specific domain adaptation techniques to further improve real-world performance.

## REFERENCES


[1] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[2] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu, "Contrastive learning for unpaired image-to-image translation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 319–345.

[3] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv:1807.03748*.

[4] X. Gu, Y. Guo, F. Deligianni, and G.-Z. Yang, "Coupled real-synthetic domain adaptation for real-world deep depth enhancement," *IEEE Trans. Image Process.*, vol. 29, pp. 6343–6356, 2020.

[5] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5828–5839.

[6] M. D. Spivak, "A comprehensive introduction to differential geometry," *Bull. Amer. Math. Soc.*, vol. 79, pp. 303–306, 1973.

[7] I. J. Goodfellow *et al.*, "Generative adversarial nets," *Adv. Neural Inf. Process. Syst.*, vol. 27, 2014.

[8] Z. Pan, W. Yu, X. Yi, A. Khan, F. Yuan, and Y. Zheng, "Recent progress on generative adversarial networks (GANs): A survey," *IEEE Access*, vol. 7, pp. 36 322–36 333, 2019.

[9] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays, "Scribbler: Controlling deep image synthesis with sketch and color," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5400–5409.

[10] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2223–2232.

[11] J. Hoffman *et al.*, "Cycada: Cycle-consistent adversarial domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1989–1998.

[12] Y. Yang and S. Soatto, "FDA: Fourier domain adaptation for semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4085–4095.

[13] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8789–8797.

[14] M.-Y. Liu *et al.*, "Few-shot unsupervised image-to-image translation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 10551–10560.

[15] A. Gokaslan, V. Ramanujan, D. Ritchie, K. I. Kim, and J. Tompkin, "Improving shape deformation in unsupervised image-to-image translation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 649–665.

[16] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 9929–9939.

[17] M. Gadelha, R. Wang, and S. Maji, "Multiresolution tree networks for 3D point cloud processing," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 103–118.

[18] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3D point clouds," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 40–49.

[19] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point cloud auto-encoder via deep grid deformation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 206–215.

[20] G. Yang, X. Huang, Z. Hao, M.-Y. Liu, S. Belongie, and B. Hariharan, "Pointflow: 3D point cloud generation with continuous normalizing flows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 4541–4550.

[21] X. Yue, B. Wu, S. A. Seshia, K. Keutzer, and A. L. Sangiovanni-Vincentelli, "A lidar point cloud generator: From a virtual world to autonomous driving," in *Proc. ACM Int. Conf. Multimedia Retrieval*, 2018, pp. 458–464.

[22] M. J. Landau, B. Y. Choo, and P. A. Beling, "Simulating Kinect infrared and depth images," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 3018–3031, Dec. 2016.

[23] B. Planche *et al.*, "Depthsynth: Real-time realistic synthetic data generation from cad models for 2.5D recognition," in *Proc. Int. Conf. 3D Vis.*, 2017, pp. 1–10.

[24] L. Seoud, J. Boisvert, M.-A. Drouin, M. Picard, and G. Godin, "Increasing the robustness of CNN-based human body segmentation in range images by modeling sensor-specific artifacts," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2018, pp. 729–743.

[25] Y. Litvak, A. Biess, and A. Bar-Hillel, "Learning pose estimation for high-precision robotic assembly using simulated depth images," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 3521–3527.

[26] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2207–2116.

[27] A. Atapour-Abarghouei, S. Akcay, G. P. de La Garanderie, and T. P. Breckon, "Generative adversarial framework for depth filling via Wasserstein metric, cosine transform and domain transfer," *Pattern Recognit.*, vol. 91, pp. 232–244, 2019.

[28] K.-C. Liu, Y.-T. Shen, J. P. Klopp, and L.-G. Chen, "What synthesis is missing: Depth adaptation integrated with weak supervision for indoor scene parsing," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 7345–7354.

[29] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 4376–4382.

[30] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised cross-domain image generation," 2016, *arXiv:1611.02200*.

[31] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9729–9738.

[32] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by back-propagation," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1180–1189.

[33] K. You, M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Universal domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2720–2729.

[34] W. Li *et al.*, "InteriorNet: Mega-scale multi-sensor photo-realistic indoor scenes dataset," 2018, *arXiv:1809.00716*.

[35] J. Jeon and S. Lee, "Reconstruction-based pairwise depth dataset for depth image enhancement using CNN," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 422–438.

[36] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "GraspNet-1billion: A large-scale benchmark for general object grasping," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11444–11453.

[37] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother, "Learning 6D object pose estimation using 3D object coordinates," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 536–551.

[38] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla, "Understanding real world indoor scenes with synthetic data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4077–4085.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[40] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3722–3731.

[41] D. Morrison, J. Leitner, and P. Corke, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach," in *Proc. Robot., Sci. Syst. XIV*, 2018, pp. 1–10.

[42] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 801–818.