Project to Adapt: Domain Adaptation for Depth Completion from Noisy and Sparse Sensor Data

Adrian Lopez-Rodriguez[1][0000−0003−3984−5126], Benjamin Busam[2,3][0000−0002−0620−5774], and Krystian Mikolajczyk[1][0000−0003−0726−9187]

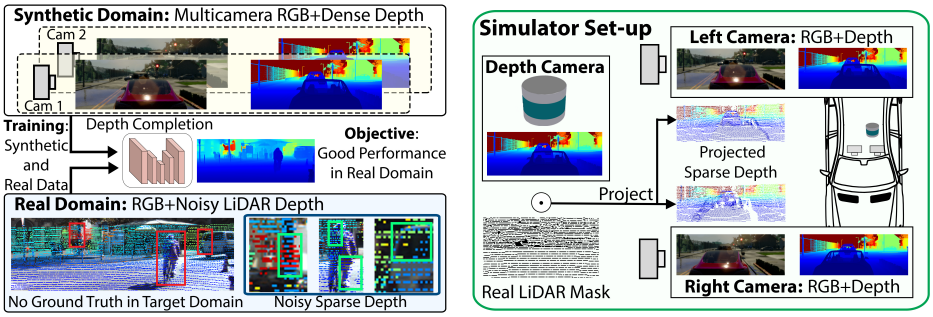
[1] Imperial College London
{al4415, k.mikolajczyk}@imperial.ac.uk
[2] Huawei Noah's Ark Lab
[3] Technical University of Munich
b.busam@tum.de

Abstract. Depth completion aims to predict a dense depth map from a sparse depth input. The acquisition of dense ground truth annotations for depth completion settings can be difficult and, at the same time, a significant domain gap between real LiDAR measurements and synthetic data has prevented from successful training of models in virtual settings. We propose a domain adaptation approach for sparse-to-dense depth completion that is trained from synthetic data, without annotations in the real domain or additional sensors. Our approach simulates the real sensor noise in an RGB + LiDAR set-up, and consists of three modules: simulating the real LiDAR input in the synthetic domain via projections, filtering the real noisy LiDAR for supervision and adapting the synthetic RGB image using a CycleGAN approach. We extensively evaluate these modules against the state-of-the-art in the KITTI depth completion benchmark, showing significant improvements.

1 Introduction

Motivation. Active sensors such as LiDAR determine the distance of objects within a specified range via a sparse sampling of the environment whose density decreases quadratically with the distance. RGB cameras densely capture their field of view, however, monocular depth estimation from RGB is an ill-posed problem that can be solved only up to a geometric scale. The combination of RGB and depth modalities form a rich source for mutual improvements where each sensor can benefit from the advantage of the other.

Many pipelines have been proposed for a fusion of these two inputs [1–6]. Ground truth annotations for this task, however, require elaborate techniques, manual adjustments and are subject to hardware noise or costly and time-consuming labeling. The most prominent publicly available data for this task [7] creates a ground truth by aligning consecutive raw LiDAR scans that are cleaned from measurement errors, occlusions, and motion artifacts in a post-processing step involving classical stereo reconstruction. Even after the use of this additional data and tedious processing, the signal is not noise-free as discussed in [7]. To



(a) Overview of the problem (b) Multicamera set-up in CARLA

Fig. 1: We investigate the depth completion problem without ground truth in the real domain, which contains paired noisy and sparse depth measurements and RGB images. We highlight some noise present in the real data: see-through on the tree trunk and bicycle, self-occlusion on the bicycle and missing points on the van. We leverage synthetic data with multicamera dense depth and RGB images. An overview of the multicamera set-up in CARLA used to simulate the real projection LiDAR artifacts is included. The *Depth Camera* acts as a virtual LiDAR and collects a dense depth, which is sparsified using real LiDAR binary masks and projected to either the *Left Camera* or *Right Camera* reference frame. Both the *Left Camera* and the *Right Camera* collect RGB information, used as part of the input data, and a dense depth map, used for supervision.

avoid such annotations, some methods perform self-supervision [4, 5, 8], where a photometric loss is employed with stereo or video data. The dependence on additional data such as stereo or temporal sequences brings other problems such as line-of-sight issues and motion artifacts from incoherently moving objects. Modern 3D engines can render highly realistic virtual environments [9–11] with perfect ground truth. However, a significant domain gap between real and virtual scenes prevents from successful training on synthetic data only.

Contributions and Outline. In contrast to the self-supervised methods [4,5,8], we propose to use a domain adaptation approach to address the depth completion problem without real data ground truth as shown in Figure 1. We train our method from the synthetic data generated with the driving simulator CARLA [11] and evaluate it on the real KITTI depth completion benchmark [7]. The real LiDAR data is noisy with the main source of noise being the see-through artifacts that occur after projecting the LiDAR's point cloud to the RGB cameras. We aim to generate synthetic LiDAR data with a noise distribution similar to the real LiDAR data. Hence, we propose an approach to simulate the see-through artifacts by generating data in CARLA using a multicamera set-up, employing random masks from the real LiDAR to sparsify the virtual LiDAR sensor, and projecting from the virtual LiDAR camera to the RGB reference frame. We further improve the model by filtering the noisy input in the real domain, thus obtaining a set of reliable points that are used as supervision. Finally, to reduce

the domain gap between the RGB images, we use a CycleGAN [12] to transfer the image style from the real domain to the synthetic one.

We compare our approach to other state-of-the-art depth completion methods and provide a detailed analysis of the proposed components. The proposed domain adaptation for RGB-guided sparse-to-dense depth completion is a novel approach for the task of depth completion, which leads to significant improvements as demonstrated by the results. To this end, our main contributions are:

1. A novel domain adaptation method for depth completion that includes geometric and data-driven sensor mimicking, noise filtering and image style adaptation. We demonstrate that adapting the synthetic sparse depth is crucial for improving the performance, whereas RGB adaptation is secondary.
2. An improvement of 6.4% RMSE, and 9.2% RMSE when combining our pipeline with video self-supervision, over the state-of-the-art in the KITTI depth completion benchmark amongst ground truth free methods.

2 Related Work

We first review related works on depth estimation using either RGB or LiDAR, and then discuss depth completion methods from both RGB and LiDAR.

2.1 Unimodal Approaches

RGB Images. RGB based depth estimation has a long history [13–15] reaching from temporal Structure from Motion (SfM) [16, 17] and SLAM [18–20] to recent approaches that estimate depth from a static image [21–25]. Networks are either trained with full supervision [21, 26] or use additional cameras to exploit photometric consistency during training [22, 27]. Some monocular depth estimators leverage a pre-computation stage with an SfM pipeline to provide supervision for both camera pose and depth [28, 29] or incorporate hints from stereo algorithms [30]. These approaches are in general tailored for a specific use case and suffer from domain shift errors, which has been addressed with stereo proxies [23] or various publicly available pre-training sources [24].

Sparse Depth. While recent advantages in depth super-resolution [31, 32] show good performance, they are not directly applicable to LiDAR data which is sparsely and irregularly distributed within the image. Similar to super-resolution, a rectangular grid for the sampling was assumed in [33]. The sampling grid of the sparse depth signal is crucial for the depth completion task [7], which can be provided as a mask to the network, thus helping to densify the input. While classical image processing techniques are used in in [34], an encoder-decoder architecture is applied for this task in [35]. Other approaches [36, 37] design more efficient architectures to improve the runtime performance.

2.2 Depth Completion from RGB and LiDAR

Most recent solutions to depth completion leverage deep neural networks. These can be divided into supervised and self-supervised approaches.

Supervision and Ground Truth. Usually, an encoder-decoder network is used to encode the different input signals into a common latent space where feature fusion is possible and a decoder reconstructs an output depth map [1–3, 6, 38]. Different additional random sampling strategies can increase the density of the input signal [1] while fusing 2D and 3D representations [39] can improve depth boundaries. The noise problem has been targeted with local and global information in [2]. Other methods [3, 6] leverage different input modalities such as surface normals to increase the amount of diversity in the input data. The publicly available dataset KITTI [7,40] includes real driving scenes where a stereo RGB camera system is fixed on the roof of a car along with a LiDAR scanner that acquires data while the car is driving. A post-processing stage fuses several LiDAR scans and filters outliers with the help of stereo vision to provide labeled ground truth. While this process is intricate and time-consuming, further error is accumulated from calibration and alignment [7].

Self-Supervised Approaches. Another view either from a second camera or a video sequence can be used for self-supervision. Temporal information and mutually predicted poses between RGB frames were used in [4] for self-supervision with a photometric loss on the reprojected image. A probabilistic formulation was proposed in [5] with a conditional prior within a maximum a posteriori (MAP) estimation, which also leverages stereo information. A non-learning method was used in [8] to form a spatially dense but coarse depth approximation from the sparse points, where the coarse approximation was then refined using another network. A photometric loss was also used in [8], where a separate network predicted the poses between RGB frames obtained from a video sequence.

Synthetic Data. For monocular depth estimation, two domain adaptation approaches used style-transfer methods [41,42]. Sparse-to-dense methods, however, have used synthetic data without any adaptation [3, 5, 43] so far. Training on synthetic data requires a high rendering quality [44]. To this end, synthesizing driving scenarios has also been researched: SYNTHIA [9] provides synthetic urban images together with semantic annotations, while Virtual KITTI [10] gives synthetic renderings that closely match the videos of the KITTI dataset [40] including semantic and depth ground truth. A LiDAR simulator using ray-casting and a learning process to drop points was proposed in [45], which was tested in detection and segmentation tasks, but is not publicly available. The CARLA simulator [11] allows for photo-realistic simulations of driving scenarios, which we utilize to generate realistic RGB images. While LiDAR scans can be simulated with CARLA via ray-casting, the car shapes are approximated with cuboids, thus losing detail. We leverage the simulator z-buffer to obtain fine-granular depth and then sparsify the signal to simulate LiDAR scans.

3 Method

Our method, shown in Figure 2, consists of two main components that include an adaptation of the synthetic data to make it similar to the real data, as well as a retrieval of reliable supervision from the real but noisy LiDAR signal.

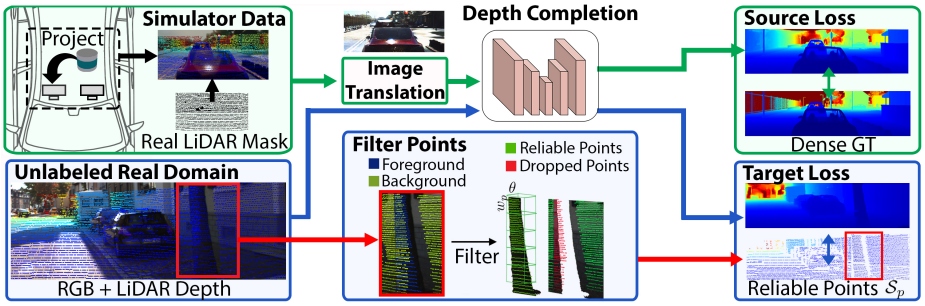


Fig. 2: Overview of our method. We use a simulator with a multicamera set-up and real LiDAR binary masks to transform the synthetic dense depth map into a noisy and sparse depth map. We train in two steps: green blocks are used in the 1st and 2nd step of training while blue blocks in the 2nd step only. In the filtering block, green points are the reliable points $\mathcal{S}_p$ and red points are dropped. The *Image Translation* network is pretrained using a CycleGAN approach [12].

3.1 Data Generation via Projections

Supervised depth completion methods strongly rely on the sparse depth input, achieving good performance without RGB information [2,4]. To train a completion model from synthetic data that works well in the real domain we need to generate a synthetic sparse input that reflects the real domain distribution. Instead of simulating a LiDAR via ray-casting, which is computationally expensive and hard to implement [46], we leverage the z-buffer of our synthetic rendering engine to provide a dense depth ground truth at first. Now, we aim to transform this synthetic dense data into a sparse depth resembling a real LiDAR sparse input.

Previous approaches used synthetic sparse data to evaluate a model in indoor scenes or synthetic outdoor scenes [5, 35, 47]. To sparsify the data a Bernoulli distribution per pixel is used in some works [1,35,47] which, given a probability p_B and a dense depth image x_D, samples each of the pixels $x_{D,k}$ by either keeping the value $x_{D,k}$ with probability p_B or setting its value to 0 with probability $(1 - p_B)$, thus generating the sparse depth $x_D^{s_B}$. We argue that a model trained with $x_D^{s_B}$ does not perform well in the real domain, and our results in Section 4 support this observation. There are two reasons for the drop of performance in the real LiDAR data. Firstly, the distribution of the points $x_D^{s_B}$ does not follow the LiDAR sparse distribution. Secondly, there is no noise in the sampled points, as we directly sample from the ground truth. We now address these two issues.

Mimicking LiDAR Sampling Distribution. To simulate a pattern similar to a real LiDAR, we propose to sample at random the real LiDAR inputs $x_{R,D}^s$ from the real domain similarly to [43]. We use $x_{R,D}^s$ to generate a binary mask M_L, which is 1 in $M_{L,k}$ if $x_{R,D,k}^s > 0$ and 0 if $x_{R,D,k}^s = 0$. We then apply the masks to the dense synthetic depth data by $x_D^{s_M} = M_L \odot x_D$. This approach adapts the synthetic data directly to the sparsity level in the real domain without the need to tune it depending on the LiDAR used.

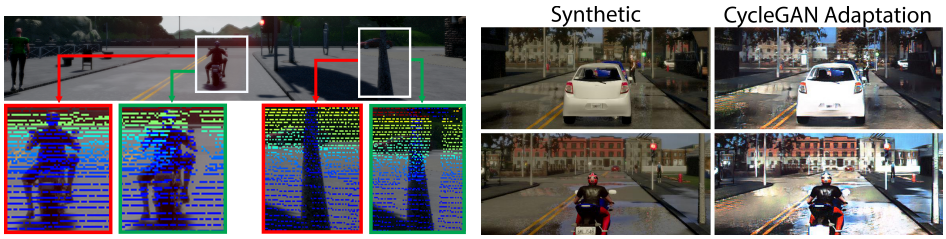


Fig. 3: *Left:* Example of the generated projection artifacts in the simulator. The zoomed-in areas marked with red rectangles correspond to $x_D^{s_M}$ and the zoomed-in areas marked with green rectangles to $x_D^{s_P}$, where we can see simulated projection artifacts, *e.g.*, see-through points on the left side of the motorcyclist. *Right:* We reduce the domain gap in the RGB modality using a CycleGAN approach. We show synthetic CARLA images and the resulting adapted images.

Generating Projection Artifacts. Previous works use noise-free sparse data to pre-train [3] or evaluate a model [35] with synthetic data. However, simulating the noise of real sparse data can reduce the domain gap and improve the adaptation result. Real LiDAR depth contains noise from several sources including the asynchronous acquisition due to the rotation of lasers, dropping of points due to low surface reflectance and projection errors. Simulating a LiDAR sampling process by modelling all of these noise sources can be costly and technically difficult as a physics-based rendering engine with additional material properties is necessary to simulate the photon reflections individually. We propose a more pragmatic solution and use the z-buffer of a simulator by assuming that the dominating noise is a consequence of the point cloud projection to the RGB camera reference frame. For such a simulation, the error becomes twofold. Firstly, the 3D points are not exactly projected on the pixel center which produces a minor quantization error. Secondly, as we are projecting a sparse point cloud arising from another viewpoint, we do not have a way to filter the overlapping points by depth. This creates the see-through patterns that do not respect occlusions as shown in Figure 3 which is also observed in the real domain [48]. Therefore, a simple point drawing from a depth map at the RGB reference cannot recreate this effect and such method does not perform well in the real domain.

To recreate this pattern, we use the CARLA simulator [11], which allows us to capture multicamera synchronized synthetic data. Our CARLA set-up mimics the camera distances in KITTI [40], as our benchmark is the KITTI depth completion dataset [7]. Instead of a LiDAR, we use a virtual dense depth camera. The set-up is illustrated in Figure 1. As the data is synthetic, the intrinsic and extrinsic parameters needed for the projections are known. After obtaining the depth from the virtual LiDAR camera, we sparsify it using the LiDAR masks resulting in $x_D^{s_M}$, which is then projected onto the RGB reference with

$$x_D^{s_P} = K_{RGB} P_{RGB}^{L} K_L^{-1} x_D^{s_M} \tag{1}$$

where K_L, K_{RGB} are the LiDAR and RGB camera intrinsics and P^L_{RGB} is the rigid transformation between the LiDAR and RGB reference frame. The resulting x^{sp}_D is the projected sparse input to either left or right camera.

3.2 RGB Adaptation

Similarly to domain adaptation for depth estimation methods [41, 42, 49], we address the domain gap in the RGB modality with style translation from synthetic to real images. Due to the complexity of adapting high-resolution images, we first train a model to translate from synthetic to real using a CycleGAN [12] approach. The generator is not further trained and is used to translate the synthetic images to the style of real images, thus reducing the domain gap as shown in Figure 3.

3.3 Filtering Projection Artifacts for Supervision

In a depth completion setting, the given LiDAR depth can also be used as supervision data, as in [4]. The approach in [4] did not take into account the noise present in the data. The given real-domain LiDAR input is precise in most points with an error of only a few centimeters. However, due to the noise present, some points cannot be used for supervision, such as the see-through points, which have errors in the order of meters. Another method [48] also used the sparse input as guidance for LiDAR-stereo fusion while filtering the noisy points using stereo data. We propose to filter the real-domain noisy input without using additional data such as a stereo pair as this may not always be available.

Our goal is to find a set of reliable sparse points $\mathcal{S}_p$, likely to be correct, for supervision in the real-domain based on the assumption used in Section 3.1, *i.e.*, the main source of error are the see-through points after projection. We assume that in any given local window there are two modes of depth distribution, approximated by a closer and a further plane. We show an overview of the idea in Figure 2. The points from the closer plane are more likely to be correct as part of the occluding objects. To retrieve $\mathcal{S}_p$ we apply a minimum pooling with window size w_p yielding a minimum depth value d_m per window. Then, we include in $\mathcal{S}_p$ the points $s \in [d_m, d_m + \theta]$ where θ is a local thickness parameter of an object. The number of noisy points not filtered out depends on the window w_p and object thickness θ, *e.g.*, larger windows remove more points but the remaining points are more reliable. We use the noise rate η, which is the fraction of noisy points as introduced in noisy labels literature [50–52], to select w_p and θ in the synthetic validation set, thus not requiring any ground truth in the real domain. Section 4 shows that using a large object thickness parameter θ or a small window size w_p leads to a higher noise rate due to an increased tolerance of the filter.

After the filtering step, a certain number of false positives remains. The noisy points in $\mathcal{S}_p$ are more likely to be further away from the dense depth prediction $\hat{y}$, hence the Reverse Huber (BerHu) loss [53] used in the synthetic domain will give more weight to those outliers. To provide extra robustness against these false positives, we use in the real domain a Mean Absolute Error (MAE) loss, as MAE weights all values equally, showing more robustness to the noise.

3.4 Summary of Losses

Our proposed loss is

$$\mathcal{L} = \lambda_S \mathcal{L}_S + \lambda_R \mathcal{L}_R \tag{2}$$

where $\mathcal{L}_S$ is the loss used for the synthetic data, $\mathcal{L}_R$ the loss used for the real data and λ_S and λ_R are hyperparameters.

We use a two-step training approach similar to past domain adaptation works using pseudo-labels [54, 55], aiming first for good performance in the synthetic data before introducing noise in the labels. First, we set $\lambda_S = 1.0$ and $\lambda_R = 0.0$, to train only from the synthetic data. For $\mathcal{L}_S$ we use a Reverse Huber loss [53], which works well for depth estimation problems [21]. Hence, we define $\mathcal{L}_S$ as

$$\mathcal{L}_S = \frac{1}{b_S} \sum_i \frac{1}{n_i} \sum_k \mathcal{L}_{bh}(\hat{y_k}, y_k) \tag{3}$$

where b_S is the synthetic batch size, n_i the number of ground truth points in image i, $\hat{y}$ is the predicted dense depth, y is the ground truth depth and $\mathcal{L}_{bh}$ is the Reverse Huber loss.

In the second step we set $\lambda_S = 1.0$ and $\lambda_R = 1.0$ as we introduce real domain data into the training process using $\mathcal{S}_p$ for supervision. We define $\mathcal{L}_R$ as

$$\mathcal{L}_R = \frac{1}{b_R} \sum_i \frac{1}{\#(\mathcal{S}_{p,i})} \sum_k |\hat{y}_k - y_k| \tag{4}$$

where b_R is the real domain batch size and $\#(\mathcal{S}_{p,i})$ is the cardinality of the set of reliable points $\mathcal{S}_p$ for an image i.

4 Experiments

We use PyTorch 1.3.1 [56] and an NVIDIA 1080 Ti GPU, as well as the official implementation of FusionNet [2] as our sparse-to-dense architecture. The batch size is set to 4 and we use Adam [57] with a learning rate of 0.001. For the synthetic data, we train by randomly projecting to the left or right camera with the same probability. In the first step of training, we use only synthetic data (*i.e.*, $\lambda_S = 1.0$, $\lambda_R = 0.0$, $b_S = 4$ and $b_R = 0$) until performance plateaus in the synthetic validation set. In the second step, we mix real and synthetic data setting $\lambda_S = 1.0$, $\lambda_R = 1.0$, $b_S = 2$, $b_R = 2$, the filter's window size to $w_p = 16$ pixels, the filter's object thickness to $\theta = 0.5$ m, and train for 40,000 iterations.

To test our approach, data from a real LiDAR+RGB set-up is needed as we address the artifacts arising from projecting the LiDAR to the RGB camera. There are no standard real LiDAR+RGB indoor depth completion datasets available. In NYUv2 [58] the dense ground-truth is synthetically sparsified using Bernoulli sampling, while VOID [8] provides sparse depth from visual inertial odometry that contains no projection artifacts. Thus, the KITTI depth completion benchmark [7] is our real domain dataset, as it provides paired real noisy LiDAR

Table 1: Ablation study on the selected validation set. *BerHu* refers to using BerHu for real data supervision. All of *2nd Step* results use *LiDAR Mask + Proj + CycleGAN RGB*. We use Bernoulli with p_B=0.062 as the KITTI LiDAR density for the crop used is approximately 6.2%. RMSE and MAE are reported in mm, and iRMSE and iMAE in 1/km.

Model	RMSE	MAE	iRMSE	iMAE
1st Step: Only Synthetic Supervision				
Syn. Baseline 1: Bernoulli (p_B=0.062)	1735.59	392.81	7.68	1.73
+ Proj.	3617.98	1411.36	23.42	9.06
Syn. Baseline 2: LiDAR Mask	1608.32	386.49	7.13	1.76
+ Proj.	1335.00	342.16	5.41	1.55
+ Proj. + CycleGAN RGB	1247.53	308.08	4.54	1.34
2nd Step: Adding Real Data				
No Filter	1315.74	315.40	4.70	1.40
$\mathcal{S}_p$+BerHu	1328.76	320.23	4.25	1.33
Full Pipeline: $\mathcal{S}_p$	**1150.27**	**281.94**	**3.84**	**1.20**
Real GT Supervision	802.49	214.04	2.24	0.91

depth with RGB images, along with denser depth ground truth for testing. We evaluate our method in the selected validation set and test set, each containing 1,000 images. Following [2], we train using images of 1216x256 by cropping their top part. We evaluate on the full resolution images of 1216x356. The metrics used are Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), reported in mm, and inverse RMSE (iRMSE) and inverse MAE (iMAE), in 1/km.

Synthetic Data. We employ CARLA 0.84 [11] to generate synthetic data using the camera set-up in Figure 1. We collect images from 154 episodes resulting in 18,022 multicamera images for training and 3,800 for validation. An episode is defined as an expert agent placed at random in the map and driving around while collecting left and right depth+RGB images, as well as the virtual LiDAR depth. We use for the virtual LiDAR camera a regular dense depth camera instead of the provided LiDAR sensor in CARLA because the objects in the LiDAR view are simplified (*e.g.*, CARLA approximates the cars using cuboids). The resolution of the images is 1392x1392 with a Field Of View of 90°. To match the view and image resolution in KITTI, we first crop the center 1216x356 of the image and then the upper part of 1216x256. To adapt the synthetic RGB images, we train the original implementation of CycleGAN [12] for 180,000 iterations.

4.1 Ablation Study

We include an ablation study in Table 1 using the validation set. For the result of the whole pipeline, we average the results of three different runs to account for training variability. All of the proposed modules provide an increase in accuracy.

Table 2: Results in the validation set depending on the input type for the whole pipeline. RMSE and MAE are reported in mm, and iRMSE and iMAE in 1/km.

Input Data	RMSE	MAE	iRMSE	iMAE
Only Sparse Depth	1175.54	290.51	4.11	1.27
+ RGB	1167.83	289.86	3.87	1.28
+ Img. Transfer from [42]	1184.39	306.66	3.99	1.36
+ CycleGAN RGB	**1150.27**	**281.94**	**3.84**	**1.20**

CARLA Adaptation. Table 1 shows that projecting the sparse depth is as important as matching the LiDAR sampling pattern, decreasing the RMSE by 23.1% when used jointly. Table 1 also shows that using Bernoulli sampling and then projecting the sparse depth results in worse performance compared to training only with Bernoulli sampling, showing that it is the combination of using a LiDAR distribution of points and projection to another camera which reduces the domain gap. Even though CycleGAN mostly adapts the brightness, contrast and colors of the images as shown in Figure 3, using image translation further reduces the RMSE by 6.6% when used jointly with real LiDAR masks sampling and projections. Figure 4 includes some predictions for examples with projection artifacts, showing that simulating the see-through artifacts via projections in the synthetic images is crucial to deal with the noisy input in the real domain.

Introducing Real Domain Data. Using the reliable points $\mathcal{S}_p$ as supervision in the real domain alongside the MAE loss function increases the performance as Table 1 shows. If we use BerHu along with reliable points supervision, the method deteriorates as the noisy points are likely to dominate the loss. Using MAE without filtering also drops the performance due to the high noise rate η. These results show that using the noisy LiDAR points for supervision as in [4, 8] is detrimental to the performance. If we define a point to be noisy if its difference with the ground-truth is more than 0.3 meters, the noise rate η for the unfiltered depth is 5.8%, and with our filtering method is reduced to 1.7% while dropping 45.8% of input points. The results suggest that η in $\mathcal{S}_p$ is more important than the total amount of points used for supervision. Supervising with real filtered data (*Full Pipeline*) improves both synthetic baselines (*Syn. Baseline*) in Table 1.

Impact of RGB Modality. Contrary to self-supervised methods, which use RGB information to compute a photometric loss, we do not require the RGB image for good performance as shown in Table 2. Including RGB information reduces the error by 0.7% in RMSE, and by using the CycleGAN RGB images the RMSE is reduced by 2.1%. In a fully supervised manner the difference is 16.3% for FusionNet [2], showing that methods aiming to further reduce the RGB domain gap may increase the overall performance. Due to computational constraints, we train the CycleGAN model in a separate step. To test an end-to-end approach, we use the method in [42], which does not use cycle-consistency, but we obtained lower-quality translated images and larger error as Table 2 shows.

Table 3: Comparison of results in the KITTI selected validation set and the official online test set. *DA Base* is our Domain Adaptation baseline formed by CycleGAN [12] + LiDAR Masks. RMSE and MAE are reported in mm, and iRMSE and iMAE in 1/km.

		Validation Set				**Online Test Set**			
Model	Param.	RMSE	MAE	iRMSE	iMAE	RMSE	MAE	iRMSE	iMAE
Unsuperv.									
DDP [5]	18.8M	1325.79	355.86	-	-	1285.14	353.16	3.69	1.37
Self-Sup.									
SS-S2D [4]	27.8M	1384.85	358.92	4.32	1.60	1299.85	350.32	4.07	1.57
DDP+St. [5]	18.8M	1310.03	347.17	-	-	1263.19	343.46	3.58	1.32
VOICED [8]	9.7M	1239.06	305.06	3.71	1.21	1169.97	299.41	3.56	1.20
Dom. Ada.									
DA Base	2.6M	1630.31	423.70	6.64	1.98	-	-	-	-
+ D. Out.	2.6M	1636.89	390.59	6.78	1.78	-	-	-	-
+ D. Feat.	2.6M	1617.41	389.88	7.01	1.79	-	-	-	-
Ours	2.6M	1150.27	281.94	3.84	1.20	1095.26	280.42	3.53	1.19
Ours-S2D	16.0M	1211.97	296.19	4.24	1.33	-	-	-	-
+ Self-Sup.									
Ours+SS [4]	2.6M	**1112.83**	**268.79**	**3.27**	**1.12**	**1062.48**	**268.37**	**3.12**	**1.13**
Supervised									
S-S2D [4]	27.8M	878.56	260.90	3.25	1.34	814.73	249.95	2.80	1.21
FusionNet [2]	2.6M	802.49	214.04	2.24	0.91	772.87	215.02	2.19	0.93
DDP [5]	18.8M	-	-	-	-	836.00	205.40	2.12	0.86

4.2 Method Evaluation

Comparison to State-of-the-Art. In Table 3 we compare our method, *Ours*, with the real domain GT-free state-of-the-art. In the test set our method decreases the RMSE by 6.4%, the MAE by 6.3% and obtains better results for iRMSE and iMAE compared to VOICED [8]. Note that these improvements upon previous methods are obtained by using an architecture with fewer parameters. Table 1 and Table 3 show that we achieve similar results to [8] by training only with synthetic data, *i.e.*, in the first training step, which validates the observation that the main source of error to simulate are the see-through points. DDP [29] uses synthetic ground truth from Virtual KITTI [10] for training, however no adaptation is performed on the synthetic data, resulting in worse results compared to our method even when using stereo pairs (*DDP+St.*). Both VOICED [8] and SS-S2D [4] use, besides video self-supervision, the noisy sparse input as supervision with no filtering, reducing the achievable performance as shown in Table 1 in *No Filter*.

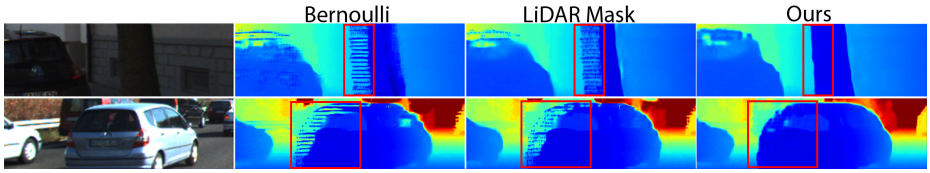


Fig. 4: Qualitative results with different training methodologies. *Bernoulli* refers to training using $x_D^{s_B}$, *LiDAR Mask* to training using $x_D^{s_M}$ and *Ours* to our full pipeline. Both rows show projection artifacts which we deal with correctly.

Table 4: Semi-supervised results in the selected validation set for different pretraining strategies before finetuning on available annotations. S and I are the number of annotated sequences and images respectively. For *Only supervised*, the weights are randomly initialized. RMSE and MAE are reported in mm, and iRMSE and iMAE in 1/km.

	S:1 / I:196		**S:3 / I:1508**		**S:5 / I:2690**	
Pretraining Strategy	RMSE	MAE	RMSE	MAE	RMSE	MAE
Only Supervised	2578.72	1175.78	1177.90	302.30	1042.75	295.73
DA Baseline	1130.79	310.68	1042.70	255.56	986.09	244.94
Ours	**1106.30**	**262.29**	**996.28**	**247.00**	**949.63**	**242.61**

Domain Adaptation Baselines. Following synthetic-to-real depth estimation methods [41, 42], we use as a domain adaptation baseline a CycleGAN [12] to adapt the images. To sparsify the synthetic depth, we use the real LiDAR masks [43], shown in Table 1 to perform better than Bernoulli sampling. The performance of this domain adaptation baseline is presented in Table 3 in *DA Base*. We explore the use of adversarial approaches to match synthetic and real distributions on top of the *DA Base*. *DA Base + D. Out.* in Table 3 uses an output discriminator using the architecture in [59], with an adversarial loss weight of 0.001 similarly to [60]. Following [42], we also tested a feature discriminator in the model bottleneck in *DA Base + D. Feat.* with weight 0.01. Table 3 shows that the use of discriminators has a small performance impact and that standard domain adaptation pipelines are not capable of bridging the domain gap.

Semi-Supervised Learning. In some settings, a subset of the real data may be annotated. Our full pipeline mimics the noise in the real sparse depth and takes advantage of the unannotated data by using the filtered reliable points $\mathcal{S}_p$ for supervision. This provides a good initialization for further finetuning with any available annotations as Table 4 shows. Compared to pretraining using the *DA Baseline*, our method achieves in all cases a better performance after finetuning.

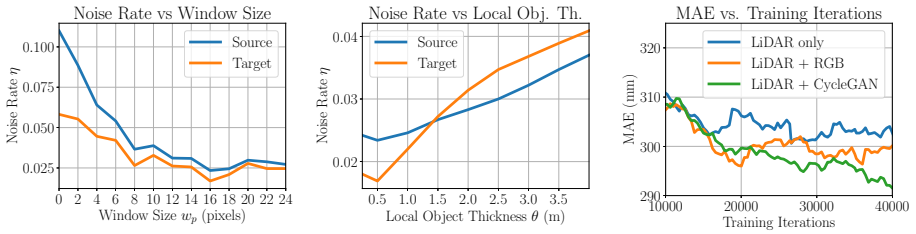


Fig. 5: Hyperparameter analysis. The two left images show the noise rate η vs. w_p ($\theta = 0.5$ m) and θ ($w_p = 16$ pixels). The right plot shows MAE vs. number of training iterations in the second step, where we evaluate every 400 iterations, use a moving average with window size 25 and average 3 runs to reduce the variance.

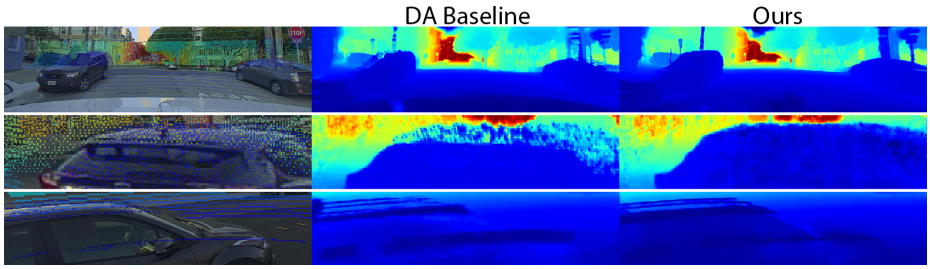


Fig. 6: Qualitative results in PandaSet [61] for our *DA Baseline* and full method (*Ours*) trained in CARLA and KITTI. RGB images also show sparse depth input.

Hyper-Parameter Selection. We do not tune the loss weights λ_S and λ_R. The projected points x_D^{sP} in the synthetic validation set are used to choose the filter window size w_p and the filter object thickness θ by employing the noise rate η in the reliable points $\mathcal{S}_p$ as the indicator for the filtering process performance. Figure 5 shows the noise percentage depending on w_p and θ, where we see that curves for the noise rate η follow a similar pattern in both the synthetic and real domain. We first select w_p and then θ as the gain in performance is lower for θ. The optimal values found are $w_p = 16$ pixels and $\theta = 0.5$ m. Figure 5 also shows the MAE depending on the number of iterations in the second step. After 40,000 training iterations, we did not see any improvement.

Adding Self-Supervision. When real domain video data is available, our approach can be combined with self-supervised methods [4, 8]. *Ours+SS* in Table 3 adds the photometric loss $\lambda_{ph}\mathcal{L}_{ph}$ from [4] to our pipeline during the second step of training for the real data, with $\lambda_{ph} = 10$ to have similar loss values as $\mathcal{L}_S + \mathcal{L}_R$. *Ours+SS* further reduces the error in the test set and achieves, compared to VOICED [8], a lower RMSE by 9.2% and a lower MAE by 10.4%.

Model Agnosticism. We chose FusionNet [2] as our main architecture, but we test our approach with the 18-layers architecture from [4] to show our method

Fig. 7: Failure cases of our method in KITTI, which cannot correct all types of noise. The left side example shows a set of noisy inputs on the wall. The right side example shows dropping of points due to low-reflectance black surfaces.

is robust to changes of architecture. Due to memory constraints we use the 18-layers architecture instead of the 34-layers model from [4], which accounts for the different parameter count in Table 3 between *Ours-S2D* and *SS-S2D*. We set the batch size to 2, increase the number of iterations in the second step to 90,000 (the last 20,000 iterations use a lower learning rate of 10^{-4}), and freeze the batch normalization statistics in the second step. The result is given in Table 3 in *Ours w/S2D arch.*, which achieves state-of-the-art RMSE and MAE.

Qualitative Results in PandaSet [61] are shown in Figure 6 for our full method compared to the DA Baseline trained for CARLA and KITTI without further tuning. PandaSet contains a different camera set-up and physical distances compared to the one used in training, *e.g.*, top row in Figure 6 corresponds to a back camera not present in KITTI. Our method is still capable of better correcting projection artifacts (top row and middle row) and completing the missing data (bottom row) compared to the DA Baseline. PandaSet does not provide depth completion ground-truth, thus no quantitative results can be computed.

Limitations. While we addressed see-through artifacts, other types of noise can be present in the real sparse depth as Figure 7 shows. The left side example shows a set of noisy inputs on the wall that is not corrected. The right side example shows missing points in the prediction due to the lack of data in the black hood surface. The fully supervised model deals properly with these cases, suggesting that approaches focused on other types of noise could further decrease the error.

5 Conclusions

We proposed a domain adaptation method for sparse depth completion using data-driven masking and projections to imitate real noisy and sparse depth in synthetic data. The main source of noise in a joint RGB + LiDAR set-up was assumed to be the see-through artifacts due to projection from the LiDAR to the RGB reference frame. We also found a set of reliable points in the real data that are used for additional supervision, which helped to reduce the domain gap and to improve the performance of our model. A promising direction is to investigate the use of orthogonal domain adaptation techniques capable of leveraging the RGB inputs even more to correct also other types of error in the LiDAR co-modality.

Acknowledgements. This research was supported by UK EPSRC IPALM project EP/S032398/1.

References

1. Mal, F., Karaman, S.: Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE (2018) 1–8
2. Van Gansbeke, W., Neven, D., De Brabandere, B., Van Gool, L.: Sparse and noisy lidar completion with rgb guidance and uncertainty. In: International Conference on Machine Vision Applications (MVA), IEEE (2019) 1–6
3. Qiu, J., Cui, Z., Zhang, Y., Zhang, X., Liu, S., Zeng, B., Pollefeys, M.: Deeplidar: Deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2019) 3313–3322
4. Ma, F., Cavalheiro, G.V., Karaman, S.: Self-supervised sparse-to-dense: self-supervised depth completion from lidar and monocular camera. In: 2019 International Conference on Robotics and Automation (ICRA), IEEE (2019) 3288–3295
5. Yang, Y., Wong, A., Soatto, S.: Dense depth posterior (ddp) from single image and sparse range. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2019) 3353–3362
6. Xu, Y., Zhu, X., Shi, J., Zhang, G., Bao, H., Li, H.: Depth completion from sparse lidar data with depth-normal constraints. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). (2019) 2811–2820
7. Uhrig, J., Schneider, N., Schneider, L., Franke, U., Brox, T., Geiger, A.: Sparsity invariant cnns. In: Proceedings of the International Conference on 3D Vision (3DV)), IEEE (2017) 11–20
8. Wong, A., Fei, X., Tsuei, S., Soatto, S.: Unsupervised depth completion from visual inertial odometry. IEEE Robotics and Automation Letters **5** (2020) 1899–1906
9. Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.M.: The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2016) 3234–3243
10. Gaidon, A., Wang, Q., Cabon, Y., Vig, E.: Virtual worlds as proxy for multi-object tracking analysis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2016)
11. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: Carla: An open urban driving simulator. In: Proceedings of the Conference on Robot Learning (CoRL). (2017) 1–16
12. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). (2017) 2223–2232
13. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. International Journal of Computer Vision (IJCV) **47** (2002) 7–42
14. Lazaros, N., Sirakoulis, G.C., Gasteratos, A.: Review of stereo vision algorithms: from software to hardware. International Journal of Optomechatronics **2** (2008) 435–462
15. Tippetts, B., Lee, D.J., Lillywhite, K., Archibald, J.: Review of stereo vision algorithms and their suitability for resource-limited systems. Journal of Real-Time Image Processing (JRTIP) **11** (2016) 5–25
16. Faugeras, O.D., Lustman, F.: Motion and structure from motion in a piecewise planar environment. International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI) **2** (1988) 485–508

17. Huang, T.S., Netravali, A.N.: Motion and structure from feature correspondences: A review. In: Advances In Image Processing And Understanding. World Scientific (2002) 331–347
18. Handa, A., Whelan, T., McDonald, J., Davison, A.J.: A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), IEEE (2014) 1524–1531
19. Mur-Artal, R., Tardós, J.D.: Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. IEEE Transactions on Robotics (T-RO) **33** (2017) 1255–1262
20. Engel, J., Koltun, V., Cremers, D.: Direct sparse odometry. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **40** (2018) 611–625
21. Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., Navab, N.: Deeper depth prediction with fully convolutional residual networks. In: Proceedings of the International Conference on 3D Vision (3DV), IEEE (2016) 239–248
22. Godard, C., Mac, O., Gabriel, A., Brostow, J.: UCL_Unsupervised Monocular Depth Estimation with Left-Right Consistency. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2017) 7
23. Guo, X., Li, H., Yi, S., Ren, J., Wang, X.: Learning monocular depth by distilling cross-domain stereo networks. In: Proceedings of the European Conference on Computer Vision (ECCV), Springer (2018) 484–500
24. Li, Z., Snavely, N.: MegaDepth: Learning Single-View Depth Prediction from Internet Photos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2018) 2041–2050
25. Godard, C., Aodha, O.M., Firman, M., Brostow, G.J.: Digging into self-supervised monocular depth estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). (2019) 3828–3838
26. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: Advances in Neural Information Processing Systems (NIPS). (2014) 2366–2374
27. Poggi, M., Tosi, F., Mattoccia, S.: Learning monocular depth estimation with unsupervised trinocular assumptions. In: Proceedings of the International Conference on 3D Vision (3DV)). (2018)
28. Klodt, M., Vedaldi, A.: Supervising the new with the old: learning sfm from sfm. In: Proceedings of the European Conference on Computer Vision (ECCV), Springer (2018) 698–713
29. Yang, N., Wang, R., Stuckler, J., Cremers, D.: Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. In: Proceedings of the European Conference on Computer Vision (ECCV), Springer (2018) 817–833
30. Watson, J., Firman, M., Brostow, G.J., Turmukhambetov, D.: Self-supervised monocular depth hints. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). (2019) 2162–2171
31. Voynov, O., Artemov, A., Egiazarian, V., Notchenko, A., Bobrovskikh, G., Burnaev, E., Zorin, D.: Perceptual deep depth super-resolution. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). (2019) 5653–5663
32. Lutio, R.d., D'Aronco, S., Wegner, J.D., Schindler, K.: Guided super-resolution as pixel-to-pixel transformation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). (2019) 8829–8837
33. Riegler, G., Rüther, M., Bischof, H.: Atgv-net: Accurate depth super-resolution. In: Proceedings of the European Conference on Computer Vision (ECCV), Springer (2016) 268–284

34. Ku, J., Harakeh, A., Waslander, S.L.: In defense of classical image processing: Fast depth completion on the cpu. In: Proceedings of the Conference on Computer and Robot Vision (CRV), IEEE (2018) 16–22
35. Jaritz, M., De Charette, R., Wirbel, E., Perrotton, X., Nashashibi, F.: Sparse and dense data with cnns: Depth completion and semantic segmentation. In: Proceedings of the International Conference on 3D Vision (3DV), IEEE (2018) 52–60
36. Chodosh, N., Wang, C., Lucey, S.: Deep convolutional compressed sensing for lidar depth completion. In: Proceedings of the Asian Conference on Computer Vision (ACCV), Springer (2018) 499–513
37. Eldesokey, A., Felsberg, M., Khan, F.S.: Confidence propagation through cnns for guided sparse depth regression. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2019)
38. Lee, B.U., Jeon, H.G., Im, S., Kweon, I.S.: Depth completion with deep geometry and context guidance. In: Proceedings of the International Conference on Robotics and Automation (ICRA), IEEE (2019) 3281–3287
39. Chen, Y., Yang, B., Liang, M., Urtasun, R.: Learning joint 2d-3d representations for depth completion. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). (2019)
40. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2012) 3354–3361
41. Atapour-Abarghouei, A., Breckon, T.P.: Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2018) 2800–2810
42. Zheng, C., Cham, T.J., Cai, J.: T2net: Synthetic-to-realistic translation for solving single-image depth estimation tasks. In: Proceedings of the European Conference on Computer Vision (ECCV), Springer (2018) 767–783
43. Atapour-Abarghouei, A., Breckon, T.P.: To complete or to estimate, that is the question: A multi-task approach to depth completion and monocular depth estimation. In: Proceedings of the International Conference on 3D Vision (3DV), IEEE (2019) 183–193
44. Mayer, N., Ilg, E., Fischer, P., Hazirbas, C., Cremers, D., Dosovitskiy, A., Brox, T.: What makes good synthetic training data for learning disparity and optical flow estimation? International Journal of Computer Vision (IJCV) **126** (2018) 942–960
45. Manivasagam, S., Wang, S., Wong, K., Zeng, W., Sazanovich, M., Tan, S., Yang, B., Ma, W.C., 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). (2020) 11167–11176
46. Yue, X., Wu, B., Seshia, S.A., Keutzer, K., Sangiovanni-Vincentelli, A.L.: A lidar point cloud generator: from a virtual world to autonomous driving. In: Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval (ICMR), ACM (2018) 458–464
47. Huang, Z., Fan, J., Yi, S., Wang, X., Li, H.: Hms-net: Hierarchical multi-scale sparsity-invariant network for sparse depth completion. arXiv preprint arXiv:1808.08685 (2018)
48. Cheng, X., Zhong, Y., Dai, Y., Ji, P., Li, H.: Noise-aware unsupervised deep lidar-stereo fusion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2019) 6339–6348

49. Zhao, S., Fu, H., Gong, M., Tao, D.: Geometry-aware symmetric domain adaptation for monocular depth estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2019) 9788–9798
50. Li, J., Wong, Y., Zhao, Q., Kankanhalli, M.S.: Learning to learn from noisy labeled data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2019) 5051–5059
51. Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., Sugiyama, M.: Co-teaching: Robust training of deep neural networks with extremely noisy labels. In: Advances in Neural Information Processing Systems (NIPS). (2018) 8527–8537
52. Zhang, Z., Sabuncu, M.: Generalized cross entropy loss for training deep neural networks with noisy labels. In: Advances in Neural Information Processing Systems (NIPS). (2018) 8778–8788
53. Zwald, L., Lambert-Lacroix, S.: The berhu penalty and the grouped effect. arXiv preprint arXiv:1207.6868 (2012)
54. Zou, Y., Yu, Z., Vijaya Kumar, B., Wang, J.: Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In: Proceedings of the European Conference on Computer Vision (ECCV), Springer (2018) 289–305
55. Tang, K., Ramanathan, V., Fei-Fei, L., Koller, D.: Shifting weights: Adapting object detectors from image to video. In: Advances in Neural Information Processing Systems (NIPS). (2012) 638–646
56. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in PyTorch. In: Advances in Neural Information Processing Systems (NIPS), Autodiff Workshop. (2017)
57. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proceedings of the International Conference on Learning Representations (ICLR). (2015)
58. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: Proceedings of the European Conference on Computer Vision (ECCV), Springer (2012) 746–760
59. Pilzer, A., Xu, D., Puscas, M., Ricci, E., Sebe, N.: Unsupervised adversarial depth estimation using cycled generative networks. In: Proceedings of the International Conference on 3D Vision (3DV), IEEE (2018) 587–595
60. Tsai, Y.H., Hung, W.C., Schulter, S., Sohn, K., Yang, M.H., Chandraker, M.: Learning to adapt structured output space for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). (2018) 7472–7481
61. Scale AI: Pandaset. https://scale.com/open-datasets/pandaset (2020)